

1 Low-resource GAN-stack for high-resolution floor plan 2 generation with enhanced evaluation and contextual 3 validation

4 Michael Sahl Lystbæk^{a,*}, Michail J. Beliatas^a, Archontis Giannakidis^{b,c}

*^aDepartment of Business Development and Technology, Aarhus University, Birk
Centerpark 15, Herning, 7400, Denmark*

*^bSchool of Science and Technology, Nottingham Trent University, Nottingham, NG11
8NS, UK*

*^cArchimedes Unit in Artificial Intelligence, Data Science and Algorithms, Athena
Research Center, Marousi, 15125, Greece*

5 Abstract

The fast and reliable prototyping of floor plan layouts is a crucial element in the early stage of the building construction cycle. The purpose of this study is to ameliorate the major difficulties associated with the use of generative adversarial networks (GANs) for automating high-resolution floor plan generation: vast computational and data requirements, training instability, and problematic result evaluation. A stable resource-efficient multi-module GAN-stack framework is proposed comprising pre-processing (denoising and $4\times$ down-sampling), floor plan image generation, and up-sampling modules. Each module is individually optimized. An innovative holistic evaluation framework of the generated building floor plans is presented covering image quality, diversity, truthfulness, overall training time, energy spent during training and associated carbon emissions. A novel validation framework is introduced, involving contextual building functionality, data privacy, capability to manipulate design generation to suit the designer's desires, usability

*Corresponding author. Email: msl@btech.au.dk
Preprint submitted to Journal of Building Engineering

in BIM downstream tasks, and inference speed. Results demonstrate that the apropos network architecture choices allow for significantly cutting down the wall clock training time ($<60\text{h}$) while maintaining superior generated image quality and contextual meaningfulness, given only sub-thousand 1024×1024 training images of single-story residential Danish homes and a limited computing resource (a single RTX-3090 GPU). The proposed computer-aided pipeline may support decisions between architects and their clients. It may broaden access to the GAN-based research on the automation of building design. The presented automated tools could find application in other industries which have the same driving needs and resource constraints for adopting GANs, and that also lack ways of validating their end product.

6 *Keywords:* Architectural design automation, Generative machine learning,
7 Generative adversarial network, Artificial intelligence, Resource efficiency,
8 Deep learning, Floor plan layout

9 **1. Introduction**

10 *1.1. Background*

11 A floor plan is a horizontal cross-section representing the spatial archi-
12 tecture of a building and its functionalities. The floor plan design is a crucial
13 early phase in the architectural design process, since many fundamental de-
14 cisions need to be made at this stage; decisions which are both difficult and
15 expensive to change later on [1]. It is also a complex iterative task, which
16 largely depends on the designer’s experience, skills, intuition, and creativity
17 [1]. In order for the architectural consultants to properly communicate the
18 design solutions with clients and reach final agreement, the fast iterative pro-

19 totyping of architectural building layouts is of vital importance. However,
20 the automated generation of floor plans is challenging since it requires ex-
21 pertise due to building functionalities [2]. In addition, floor plan drawings
22 are significantly different from other natural images [3].

23 Generative machine learning (GML) is a widely discussed branch of arti-
24 ficial intelligence (AI), mainly because of its capacity for performing creative
25 tasks, which had always been associated with a unique ability possessed
26 by humans [4–11]. NVIDIA impressed the world in 2018 by introducing
27 StyleGAN, which is a style-based generative adversarial network (GAN) [12]
28 with the ability to generate high-quality images; for instance, artificial hu-
29 man faces which have been introduced in a variety of different cases (i.e.,
30 “thispersondoesnotexist.com”). StyleGAN has been further improved ever
31 since, lastly as the StyleGAN3 implementation [13]. Other variants such as
32 StyleGAN-XL, which has a deeper artificial neural network (ANN) struc-
33 ture, do exist [14]. The research community finds the use of GANs relevant
34 because of their relatively fast image generation process compared to the
35 highly credited diffusion models. The latter have recently been the focus of
36 attention due to their ability to generate high-quality images by relying on
37 large datasets and training setups [15]. Tests by Sauer et al. [15] showed
38 that GANs have the power to perform image generations much faster than
39 other GML techniques and still maintain a high image quality [15, 16]. Uti-
40 lizing StyleGAN with adaptive discriminator augmentation (ADA) has been
41 shown to improve a model’s performance on smaller datasets [17]. Besides,
42 the integration of the CLIP dual text and image encoder into the pipeline
43 has pointed out GANs’ capability to perform text to image generation [18].

Recent research has focused on improving the training process by modifying the StyleGAN discriminator and generator architecture and by adapting weights from pre-trained models [19–21]. These techniques have shown superior results in comparison to the original model proposed by NVIDIA, both in the sense of faster training and training on smaller datasets [19–21]. GML is expected to have a growing impact on the development of construction applications.

1.2. The challenge

In a recent review study, Chai et al. [22] identified the four main challenges concerning the use of GANs in construction applications as:

1. **Questionable applicability due to problematic evaluation of results:** As GANs are implicit models and do not return likelihood values, it is inherently difficult to evaluate these models using general quality metrics [23]. Fréchet inception distance (FID) is a popular quality evaluation metric, having dominated image quality evaluation also in construction applications. However, it has been criticized for being too unilateral due to its single-dimensional assessment values and also for providing some unrealistically low quality scores owing to leftover bias [16]. In addition, precision and recall have been found to be too sensitive to outliers [24]. The above factors call for a new and more exhaustive evaluation framework for GAN applications to construction.
2. **Requirement for substantial computational resources:** GAN training requires vast amounts of compute power [21]. Tech giants

such as OpenAI, Alphabet, MidJourney or NVIDIA have been keen to enhance their computing infrastructure to meet the demands for training large generic GML models [9]. However, access to such large-scale GPU clusters is a limitation in the technology’s applicability to the construction industry.

3. **Provision of a large training dataset:** The dataset quantity, as well as quality, are of paramount importance for developing and training GAN models [17, 25, 26]. The proper amount of data needed for training GANs can be difficult to define. Previous studies indicated several thousand, or hundreds of thousands, of images to be a cohesive dataset [17, 27]. However, the collection of a large training dataset in the range of a hundred thousand or millions of images is costly or even not feasible [17, 22]. It’s worth noting that the investigation of limited datasets starts from 2k images [17]. The small datasets can be supported with data augmentation techniques which can have an important impact on the training [17, 28].
4. **Training instability:** This challenge relates to the generated image quality. It refers to vanishing gradient problems, mode collapse in the generator (low diversity), and other discrepancies [22, 26].

The above difficulties in generating high quality floor plan images with GANs have also been described by other studies [29].

1.3. Related work

In the GML-based synthetic floor plan generation domain, several studies have been performed recently, focusing on various aspects of the problem. A

92 variety of methodologies have been presented, reflecting the growing inter-
93 est in using advanced computational techniques to automate and enhance
94 architectural design processes.

95 Graph-based GAN methods are a prominent approach for the conditional
96 generation of 2D architectural space layouts. The conditions are user-defined,
97 offering flexibility and control over the floor plan design output. Recent
98 works have highlighted how these graph-based techniques can be employed
99 to model the spatial relationships and architectural constraints in a way that
100 aligns with traditional design principles while also pushing the boundaries of
101 automated design [30–32].

102 The pixel-based generation of architectural floor plan layouts with GAN
103 techniques has also been widely explored due to its ability to produce de-
104 tailed and visually coherent layouts [33–41]. Image-to-image conditional
105 GANs (cGANs) such as CycleGAN [33] offer flexibility in working with data
106 classes. On the other hand, pix2pix [34–36] needs to be trained on data pairs.
107 Diffusion-based image inpainting is proposed for conditional floor plan anal-
108 ysis and generation tasks [42, 43]. However, diffusion models are notoriously
109 data-hungry and computationally expensive to train. A coupled generative
110 adversarial network (CoGAN)-based framework [41] for the rapid generation
111 of an unlimited number of floor plan design solutions has also been proposed.

112 For construction applications in particular, GAN-chains have also been
113 proposed in several studies so that the multi-module collaborative system
114 can benefit the processing of multi-property tasks [38, 44, 45].

115 The studies discussed above highlight the promise of employing GAN
116 techniques in construction applications. However, to the best of our knowl-

117 edge, no literature exists to date that has attempted to collectively address
118 all four challenges identified in the review study [22].

119 1.4. Aim and contribution

120 The aim of this study is to make important headway in dealing with the
121 challenges discussed in Section 1.2. The contributions are summarized as:

- 122 • A novel resource-efficient multi-module GAN-stack framework is pro-
123 posed (Fig. 1) for the autonomous generation of unique and contex-
124 tually meaningful high-resolution (HR) (1024×1024) floor plan layouts
125 aimed at assisting in the architectural design workflow. The framework
126 comprises pre-processing (i.e., denoising and $4 \times$ image down-sampling),
127 floor plan image generation, and image up-sampling modules. The pro-
128 posed implementations are the AREA down-sampling method [46] and
129 a transformer-based denoising approach [47] (pre-processing module),
130 the Projected GAN [21] with FastGAN generator [19] (module 1), and
131 the Real-ESRGAN [48] (module 2). The configuration of each module
132 is individually optimized.
- 133 • To address gaps in previous related studies, an innovative holistic eval-
134 uation framework of the generated floor plan images is presented that
135 covers (apart from generated image quality) also fidelity, diversity, and
136 truthfulness. In addition, the framework involves assessment of the
137 overall training time, energy spent during training, and associated car-
138 bon emissions. The latter are important due to deep learning’s terrible
139 environmental cost [49].

- 140 • A novel validation framework is presented to ensure that the computer-
141 ized framework performs the way it was intended and it can be used to
142 achieve business objectives. The framework includes elements such as
143 contextual functionality, data privacy, capability to manipulate design
144 generations to suit the designer’s desires, usability in a downstream
145 task, namely a building information modeling (BIM) software-based
146 implementation task, and inference speed.

- 147 • Results demonstrate that the apropos framework architecture choices
148 allow for advanced computational resource and data efficiency and, at
149 the same time, mitigate the training instability in the GML-based floor
150 plan generation. Notably, the proposed framework is shown to signifi-
151 cantly cut down the wall clock training time (<60h) while maintaining
152 superior generated image quality, given only sub-thousand high-fidelity
153 training images of single-story residential Danish homes and a limited
154 computing resource (i.e., a single RTX-3090 GPU).

- 155 • The performed experiments provide insight that could form a knowl-
156 edge base about the future applicability of the proposed tools (for image
157 generation, evaluation, and validation) to other industries for which re-
158 liability and fast iterative prototyping are some of the driving needs for
159 adopting GML, but they do not have access to large-scale data and
160 computational resources and that also lack ways of ensuring that their
161 end product is consistent with the intended application.

162 This study introduces down-sampling (pre-processing) and up-sampling
163 (post-processing) in the floor plan generation process (in contrast to train-

ing models on HR datasets) as a means to boost computational resource efficiency. This decision was driven by the fact that image resolution has been shown to have a large impact on the training performance of GANs [19, 50]. The proposed multi-module GAN-stack framework for generating HR architectural floor plan layouts is therefore in a sense analogous to mobile telecom voice/video data processing methodologies, where the original data signals are down-sampled and compressed with multi-rate signal processing so that the time and cost required to transmit the data across the channel is reduced, and then the opposite process is applied on the receiver side [51]. It has been found that down-sampling the data resolution from 1024×1024 pixels to 256×256 pixels reduces the training processing time (s/king) by threefold and it also lowers the required VRAM [19, 50].

2. Methodology and implementation

A novel multi-module GAN-based framework (Fig. 1) is proposed for generating HR synthetic floor plans with low data and computational resources. In the pre-processing module, the floor plan images are rendered without noise (i.e., unwanted text content, dimensions, and logos) and downsized for the streamlined training of the subsequent floor plan generator. The purpose of the next module is to train a resource-efficient style-based GAN model to generate new synthetic floor plan designs. In the final module, a GAN-based up-sampler, trained on HR and low-resolution (LR) dataset pairs provided by the pre-processing module, increases the image resolution and quality of the floor plan generated in the previous module. To boost the overall framework performance, each module configuration is optimized. The framework

188 performance is evaluated and validated using a combination of established
189 and innovative metrics. A definition of technical terms is presented in Table
190 1.

191 2.1. Dataset

192 The experiments are based on an initial dataset of 1,233 HR architectural
193 floor plan images of one-story, single-family homes in Denmark, scraped from
194 a real estate web database [52].

195 2.2. Pre-processing module

196 The original floor plan image dataset contains noise in the form of logos,
197 text descriptions, and dimensions. To suppress this noise, a multi-head at-
198 tention method [47] is proposed in the pre-processing module. The denoised
199 HR dataset is further down-sampled to an LR dataset to achieve a cost-
200 effective and faster training process with low computational resources in the
201 subsequent GAN modeling. It is proposed to optimally downscale the clean
202 dataset using the AREA method (OpenCV INTER_AREA) [46, 53]. The
203 impact of removing unwanted text content from the initial training dataset
204 is evaluated on a downstream StyleGAN3-based floor plan image genera-
205 tion task. In particular, the quality of images generated by the GAN model
206 trained on a LR dataset, with and without text content, is compared. A qual-
207 itative visual assessment is found appropriate for this comparison. A visual
208 comparison is then conducted between the AREA down-sampling method
209 and three baselines: bilinear, bicubic, and nearest neighbor.

Table 1: Definition of technical terms.

Term	Definition
GAN	A GML model consisting of a generator and a discriminator architecture, trained adversarially to produce realistic synthetic data.
SRGAN	A GAN-based model for upsampling low-resolution image inputs to high-resolution outputs.
StyleGAN	This GAN architecture introduces style-based modulation at different layers of the generator, enabling fine control over image features and structure to produce highly realistic images.
Latent space	A compressed numerical representation space used by generative models to encode features.
FID	A metric used to evaluate the quality of images generated by generative models by measuring the distance between the feature distributions of real and generated images.
Density & coverage	Metrics that assess how well generated data matches or covers the real data distribution.
CNN classifier	A CNN-model that can assess image content, including whether generated images can be distinguished from real images based on visual features.
TL	Transfer learning enables model training on one task to improve performance on a related task with limited training data.
End-to-end training	This training setup enables the full training of a model, where all components are optimized from input to output.
Carbontracker	A tool for estimating energy use and CO ₂ emissions in ML experiments.
Image hashing	A technique that maps data to fixed-size representations, often used for fast similarity search and retrieval in large datasets.
Morphing	A technique used to smoothly transform one image into another by interpolating shapes and appearances, often applied in data augmentation or style blending.

210 2.2.1. Dataset denoising

211 The dataset is pre-processed with the image text remover and label ex-
 212 tractor (ITRLE) model [47] in an automated process based on the text spot-
 213 ting transformer (TESTR) architecture [54]. In the encoder, the model relies
 214 on ResNet-50 [55] to extract features from the input images [47, 54], whereas
 215 the detection of small text instances is enhanced by applying multi-scale de-
 216 formable attention to the feature maps. To train the dual (i.e., character
 217 recognition and location) decoder, the following loss function (L_{dec}) is used:

$$L_{\text{dec}} = \sum_j \left(\lambda_{\text{cls}} L_{\text{cls}}^{(j)} + \lambda_{\text{coord}} L_{\text{coord}}^{(j)} + \lambda_{\text{char}} L_{\text{char}}^{(j)} \right) \quad (1)$$

218 where λ_{cls} , L_{cls} represent the classification loss weight and classification loss
 219 of the text instances, respectively, λ_{coord} is the control point coordinate re-
 220 gression loss weight, L_{coord} represents the control point loss, λ_{char} and L_{char}
 221 are the classification loss weight and character classification loss, respectively,
 222 and j represents the query number.

223 2.2.2. Dataset down-sampling

224 The denoised images are resized from the resolution of 1024×1024 pix-
 225 els to 256×256 pixels. The AREA method (OpenCV INTER_AREA) [53],
 226 which relies on pixel area relations and resizes images using adaptive aver-
 227 age pooling, is utilized. It is a highly endorsed method to shrink an image
 228 because the results are free from aliasing-based Moiré patterns.

229 2.3. Floor plan generation module

230 The floor plan generation module is applied to the LR dataset, obtained
 231 after the pre-processing described in Section 2.2. To achieve the objective

232 of a resource-efficient and stable framework, this module is implemented
 233 using Projected GAN with FastGAN architecture for the generator [19, 21].
 234 A holistic comparison of the method is conducted with two state-of-the-art
 235 (SOTA) approaches, namely end-to-end StyleGAN3 training and StyleGAN3
 236 transfer learning (TL), in terms of generated floor plan image quality, fidelity,
 237 diversity, and truthfulness, as well as VRAM requirements, overall training
 238 time, energy consumption and the associated environmental footprint. The
 239 majority of these criteria have not been used before to assess GAN-based
 240 generation in construction applications.

241 *2.3.1. StyleGAN3 end-to-end training*

242 The end-to-end trained floor plan generator is based on the StyleGAN3
 243 architecture, which is the latest release of the iconic style-based GAN by
 244 NVIDIA [13]. While the discriminator of StyleGAN2 is left unchanged, the
 245 StyleGAN3 generator has an improved architecture by upgrading the quality
 246 of the up-sampling filters to reduce aliasing issues that have been one of the
 247 main challenges that StyleGAN2 faces. As opposed to StyleGAN2, Style-
 248 GAN3 relies on Fourier features, the per-pixel noise inputs are discarded,
 249 the mapping network depth is reduced, the mixing and path length regu-
 250 larizations are disabled, the output skip connections are disposed of, the
 251 boundaries and up-sampling are handled with enhanced filtering, the nonlin-
 252 earities are treated with a custom CUDA kernel leading to a tenfold decrease
 253 in training time, oversampling is used in all layers except for the highest-
 254 resolution ones, flexible layer specifications are allowed, 1×1 convolutions
 255 are used, an improved jinc-based down-sampling filter is used in all layers
 256 except for the two critically sampled ones, and Gaussian blurring is applied

early in the training. Even though StyleGAN3 matches FID-wise the StyleGAN2 generation quality, their internal representations are noticeably different. StyleGAN3 is developed to perform a natural transformation hierarchy, ensuring that the sub-pixel positioning of each feature is accurately inherited from the underlying coarse features, as described by [13].

The objective function (Eq. 2), $V(D, G)$, for GANs can in general be expressed as [56]:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]. \quad (2)$$

G represents the generator, in which V is aimed to be minimized, and D represents the discriminator, where V is aimed to be maximized. Hence, there is a competition between G and D driven by V , with G trying to generate images that D is not able to classify as fake. $\mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)]$ encourages D to rightly predict high probabilities when classifying real data samples from the data distribution. $\mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$ helps D to correctly classify fake data generated by G .

The above adversarial training process for GANs iterates for a large number of epochs. This iterative backpropagation process requires high computational resources. Over time, G becomes better at creating realistic images, while D improves its capability to identify fakes over real images. Convergence is achieved when G generates realistic-looking data.

The learning performed by GAN models is captured in a vector space denoted as the latent space. StyleGAN was constructed with three latent space levels to gradually develop the style features [57]. Z is the first latent space, which is the random (typically Gaussian) input noise vectors that provide the starting point for the image generation process [13, 57]. Next, the interme-

281 diate latent space W is formed by the learned, fully-connected layer output
 282 mapping of the Z -noise vectors. W is a higher-level representation of the
 283 images [13, 57]. $W+$ is an improved (extended) intermediate latent space, in
 284 which a distinct 512-dimensional w vector is assigned to each layer of the gen-
 285 erator [57, 58]. The intermediate latent space W ($W+$) contains information
 286 about “**style**” **attributes** of the generated image such as texture, color, and
 287 object shape [58]. Last, the Style Space S represents the high-dimensional
 288 space of channel-wise style parameters, derived through distinct affine trans-
 289 formations applied to each layer of the generator [57, 59, 60]. Compared to
 290 the other intermediate latent spaces, S demonstrates a significantly higher
 291 degree of disentanglement [57]. Therefore, it is the most suited latent space
 292 to represent styles. After training is completed, the model is then capable of
 293 generating data in a sampling process; that is, by mapping into the learned
 294 latent space [57]. Style vectors within this space can manipulate aspects of
 295 the generated images, such as lighting, color schemes, textures, and object
 296 appearance, while maintaining the overall structure defined by the W -latent
 297 space.

298 ADA is a mechanism which allows StyleGAN to make adjusted data aug-
 299 mentation “on the fly” during training. This technique helps to improve the
 300 quality and diversity of the generated images by avoiding overfitting of the
 301 discriminator when training on limited datasets [17].

302 2.3.2. *StyleGAN3 transfer learning*

303 Former research has shown that TL can have a positive effect on the
 304 generative models’ ability to adapt features from the real image to the gen-
 305 erated outputs earlier in the training process, especially when the subject of

the data used to train the previous model and the new data subject share semantic similarities [61]. Thus, TL has been the go-to method for cases with limited availability of training data. However, it might be difficult to find highly relatable pre-trained models for domain-specific tasks. For the TL experiments, the StyleGAN3 model, pre-trained on the LHQ-256 dataset [62], is employed.

2.3.3. Projected GAN with FastGAN generator

Projected GAN utilizes pre-trained representations to expedite and stabilize training [21]. Following guidance from [21], EfficientNet-Lite1 [63] is employed for the pre-trained feature network, an image classification network trained on ImageNet [64], which has been shown to produce compact representations that achieve lower FIDs and at the same time reduce computational cost and training time. In order for the discriminator to fully exploit features from deeper layers of the pre-trained model, Projected GAN mixes features across channels and scales using differentiable random projections that are fixed [21]. The use of multiple discriminators allows for the provision of multi-scale feedback. Differentiable augmentation methods [65] are leveraged to prevent overfitting in the discriminator. The Projected GAN training objective is expressed by [21]:

$$\min_{\{G\}} \max_{\{D_l\}} \sum_{l \in L} (\mathbb{E}_x [\log D_l(P_l(x))] + \mathbb{E}_z [\log(1 - D_l(P_l(G(z))))]) \quad (3)$$

where $\{D_l\}$ is a set of independent discriminators that operate on different feature projections. D_l is associated with features from layer l of the pre-trained feature network where L is the total number of layers. The feature projectors P_l , introduced for mapping real and generated images to the input

space of the discriminators, are fixed and only the parameters of $\{G\}$ and $\{D_l\}$ are optimized [21]. The consistency theorem of [66] ensures that G matches the generated and true feature space distributions at convergence (i.e. after solving the optimization problem described by Eq. 3).

For the generator, Projected GAN uses the minimalistic FastGAN architecture [19], comprising several up-sampling and skip-layer-excitation (SLE) blocks. The SLE block, inspired by residual blocks [55], provides a short-cut to gradient flow by applying channel-wise multiplications between the activations and by performing skip connections between resolutions of much longer range [19]. The SLE module is formally defined as:

$$y = F(x_{\text{low}}, \{W_i\}) \cdot x_{\text{high}} \quad (4)$$

where x_{low} , x_{high} are the 8×8 and 128×128 input feature maps, respectively, y is the output feature map, F are the operations on x_{low} and W_i are the weights to be learnt. The structure of an SLE module and the overall generator can be seen in Fig. 2.

The Projected GAN with FastGAN generator has been shown to outperform StyleGAN2-ADA, FastGAN, and Projected StyleGAN2-ADA in a variety of domains in terms of convergence speed, data efficiency, and image quality [21]. It has further shown promising results for training on small datasets and achieving low computational cost [19, 21].

2.4. Up-sampling module

The up-sampling module is applied to the images obtained by the floor plan generation module, described in Section 2.3. The up-sampling module is implemented using the Real-ESRGAN architecture [48]. Comparisons

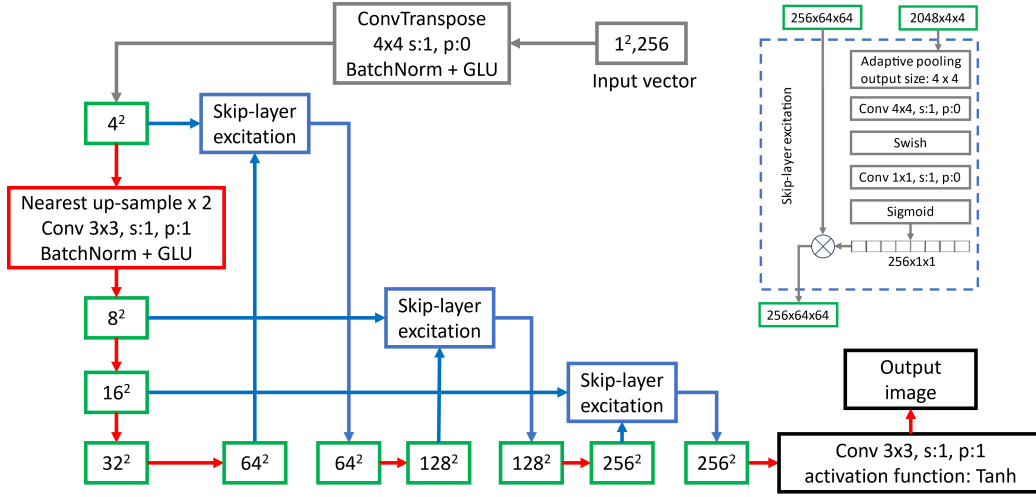


Figure 2: The generator network, inspired by the original FastGAN generator [19]. Grey represents the initial mapping structure, green boxes indicate the spatial size of the feature maps, red boxes and arrows illustrate the up-sampling structure, and blue boxes contain the skip-layer excitation (SLE) modules. Dashed blue line illustrates the detailed structure on an SLE module.

are drawn between the proposed approach and a baseline method, namely SRGAN [67], in terms of up-sampled image quality. Given that the framework proposes the up-sampling of generated (synthetic) floor plan samples for which it is inherently impossible to have the ground through (GT) HR images, only a qualitative evaluation of the investigated SR models is conducted. All SR models are fed with synthetic images obtained from the best performing floor plan generation model.

2.4.1. SRGAN

SRGAN [67] is the first single image super-resolution (SISR) framework, proposed for high ($4\times$) upscaling factors, that is not solely based on the mean squared reconstruction error (MSE) content loss to obtain the solution [67, 68]. The generator comprises 16 identical residual blocks, whereas the differentiable discriminator, trained to distinguish real HR images from generated HR samples, consists of eight convolutional layers followed by two dense layers and a sigmoid activation function [67].

To optimize the SR image quality, the perceptual loss, L^{SR} , was proposed. L^{SR} minimizes errors in perceptually relevant characteristics and is given by the sum of two terms: the content loss, L_X^{SR} , and adversarial loss, L_{Gen}^{SR} [67]:

$$L^{SR} = \underbrace{L_X^{SR} + 10^{-3}L_{Gen}^{SR}}_{\text{perceptual loss}}. \quad (5)$$

The content loss is defined as the Euclidean distance between the high-level visual geometry group (VGG19) [69] feature maps of the reconstructed and reference images, whereas the adversarial loss component pushes solutions towards the natural photo-realistic image manifold through the discriminator network.

375 2.4.2. Real-ESRGAN

376 The Real-ESRGAN [48] is built on the enhanced SRGAN (ESRGAN)
 377 [70]. ESRGAN has shown great results by adopting a more complex architec-
 378 ture, which generates more realistic and natural textures in the up-sampling
 379 process. The three main improvements of the ESRGAN architecture over
 380 SRGAN are:

- 381 1. Higher capacity and easier training by replacing the original basic block
 382 structure of the generator with 23 residual-in-residual dense blocks
 383 (RRDBs). Furthermore, all batch normalization (BN) layers have been
 384 removed to enhance performance and generalization ability, stabilize
 385 training, and lower the memory usage and computational complexity
 386 for both SR and deblurring tasks. In addition, residual scaling and
 387 smaller initialization are leveraged to improve training.
- 388 2. The discriminator has also been modified to a relativistic discriminator
 389 utilizing the relativistic average GAN (RaGAN) structure [71], enabling
 390 it to predict relative realism between real and generated images rather
 391 than the absolute realism.
- 392 3. The perceptual loss (L_{percep}) is improved by using the VGG19 [69]
 393 network feature space before the activation layers, instead of after as
 394 employed in the SRGAN architecture [67], which strengthens the super-
 395 vision for brightness consistency and texture recovery [70]. The total
 396 loss for the ESRGAN generator (total loss $_{G_{\text{ESRGAN}}}$) is expressed as:

$$\text{total loss}_{G_{\text{ESRGAN}}} = L_{\text{percep}} + \lambda_{\text{bal}} L_G^{\text{Ra}} + \eta_{\text{bal}} L_1 \quad (6)$$

where L_{percep} denotes the VGG loss, L_G^{Ra} denotes the adversarial loss, $L_1 = \mathbb{E}_{x_i} \|G(x_i) - y\|_1$ is the content loss, and λ_{bal} , η_{bal} are the balancing coefficients.

The Real-ESRGAN [48] further extends the above ESRGAN [70] capabilities, by enabling blind SR capable of reconstructing from unknown and complex real-world degradations. While the generator stays the same, the VGG-style discriminator has been modified to a U-Net design with skip connections for handling a significantly larger degradation space than the one that the ESRGAN is capable of [48]. Spectral normalization (SN) is employed as a regularization technique to improve restored textures and stabilize the training of the discriminator [48].

For ESRGAN, it is assumed that degradation noises are applied to images once, while real-world images can, in fact, undergo a series of degradations. To model more practical real-world degradations (such as blur, general noise (Gaussian, Poisson, Color, Gray), and JPEG compression), a high-order degradation process is proposed for Real-ESRGAN [48]. Also, a 2D-sinc filter is applied to represent ringing and overshoot artifacts. Training Real-ESRGAN on pure synthetic data has resulted in better visual performance in restoring real-world images, making it more practical than ESRGAN [48].

2.5. Floor plan generation evaluation framework

2.5.1. Fréchet inception distance

The FID score is commonly used as an evaluation metric for the quantitative assessment of the quality of the images generated by a model [16, 23]. It computes the distance between the distributions of real and generated images by embedding them to a vision-relevant feature space [16, 72]. FID is typically based on an ImageNet [64] pre-trained Inception-V3 classifier network

422 [16, 73]. The FID metric is expressed as:

$$\text{FID} = \|\mu_X - \mu_Y\|_2^2 + \text{Tr}(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y}) \quad (7)$$

423 where X, Y represent the real and synthetic image feature vectors, respec-
424 tively, $\|\mu_X - \mu_Y\|_2^2$ denotes the squared Euclidean distance between the mean
425 vectors, and $\text{Tr}(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y})$ represents the trace of the sum of the
426 two covariance matrices minus twice the matrix square root of their product.
427 A low FID score indicates high quality between the 50k generated images
428 and the real image dataset [13].

429 2.5.2. Density and coverage metrics

430 The FID score is a one-value metric encompassing both the fidelity and
431 diversity aspects of the generated image quality. Therefore, previous studies
432 have focused on separating FID into two distinct metrics, namely precision
433 and recall, that are representative of these two aspects of generative model
434 performance [74–76].

435 However, precision and recall fail when the real and fake distributions are
436 identical, and also in the presence of outliers and certain mode droppings.
437 They also do not allow for systematic selection of hyper-parameter values
438 [24]. In this study, the density and coverage metrics are employed as a more
439 reliable method to tackle the above matters [3]. Density is representative of
440 fidelity, defining how closely the model resembles the real data. Coverage,
441 on the other hand, represents diversity, measuring how well the generated
442 samples cover the variety of the real dataset [24]. On top of this, the density
443 and coverage metrics tackle the introduced bias by the ImageNet pre-trained
444 VGG16 model [76], which may limit fair evaluation. In cases where the

dataset distribution is distinct from ImageNet, random embedding is introduced as a better choice [24].

The density function is an improvement of the precision metric [76], where the overestimation of the manifold around real outliers has been fixed by counting how many real-sample neighborhood spheres contain fake samples Y_f . The density is calculated as:

$$\text{density} = \frac{1}{kM_{fake}} \sum_{f=1}^{M_{fake}} \sum_{r=1}^{M_{real}} I_{Y_f \in B(X_r, \text{NND}_k(X_r))} \quad (8)$$

where X_r are the real samples, k denotes the number of nearest neighbors, $\text{NND}_k(X_r)$ denotes the distance from X_r to the k^{th} nearest neighbor of the remaining X_r , $B(X_r, \text{NND}_k(X_r))$ represents the neighborhood spheres around each real sample X_r , M_{fake} and M_{real} are the numbers of fake and real samples, respectively, and I is the indicator function.

Coverage enhances the recall metric [76] by computing the nearest neighbor manifolds around real samples, which are expected to have fewer outliers than fake samples. It is defined as [24]:

$$\text{coverage} = \frac{1}{M_{real}} \sum_{r=1}^{M_{real}} I_{\exists f \text{ where } Y_f \in B(X_r, \text{NND}_k(X_r))} \quad (9)$$

where the function $I_{\exists f}$ returns 1 if there exists at least one fake sample Y_f within the neighborhood sphere of real sample X_r .

2.5.3. CNN detection score

The generated floor plan images should be indistinguishable from the real data. A convolutional neural network (CNN)-based detection score is employed for assess the floor plan generation module’s ability to fool a SOTA

465 classifier from a visual perception perspective. Previous literature has also
 466 relied on CNN classification for distinguishing synthetic samples generated
 467 by GANs from real images [77–79].

468 In this study, a binary CNN classifier based on Inception V3 [73] is em-
 469 ployed. The GoogLeNet [80] Inception V3 [73]-based CNN architecture is
 470 widely used for visual machine learning tasks [73]. The model is trained and
 471 tested on equal proportions of synthetically-generated (fake) and real data.
 472 The accuracy is reported, calculated as follows:

$$\text{accuracy} = \frac{\text{number of correct predictions}}{\text{total number of predictions}}. \quad (10)$$

473 Low accuracy values would indicate that the classification model is confused
 474 about the truthfulness of fake data, and thereby the GAN floor plan generator
 475 performs well.

476 2.5.4. Carbontracker

477 The Carbontracker tool is a method for evaluating the energy consump-
 478 tion and the associated carbon dioxide emissions in deep learning experiments
 479 [81]. It has been utilized by previous work on developing low emission deep
 480 learning models [13, 82, 83]. The total energy consumption (EC) is calculated
 481 based on the formula:

$$\text{EC} = \text{PUE} \sum_{e \in E} \sum_{c \in C} P_{\text{avg},de} T_e \quad (11)$$

482 where PUE stands for power usage effectiveness and is a ratio parameter
 483 characterizing a data center’s efficiency and the energy overhead of computing
 484 equipment, $P_{\text{avg},de}$ denotes the average power consumed by device $c \in C$ in

epoch $e \in E$, and T_e is the duration of epoch e .

The carbon footprint (CF) for training a model is then calculated by converting the monitored EC using the formula:

$$CF = EC \times CI \quad (12)$$

where CI is the carbon intensity. There is some variation in the CI calculation depending on the source or region where the electricity comes from. To better appreciate the CF generated during training, the equivalent distance traveled by an average car is calculated and reported, using a rate of 107.5 g of CO₂eq/km [81].

2.6. Validation framework

After assessing the HR synthetic floor plan designs produced by the up-sampling module, a validation of the overall proposed framework would be beneficial. To ensure that the computerized framework performs as intended and can be used to achieve business objectives, validation is performed in terms of contextual functionality and data privacy. To showcase the capability of the framework to be manipulated to suit the designer’s desires, and its usability in a downstream task, namely a software-based building information modeling (BIM) implementation task is conducted. Lastly, driven by the intended use of the proposed framework, a validation test is performed to assess the fast delivery of the generated HR floor plan images. Another goal of the proposed validation framework is to identify potential errors. The validation is retrospective and is applied to floor plan images produced by the best-performing floor plan generation and image up-sampling GAN model

507 stack. It is also independent, meaning that the input images for validation
508 are different from the ones that had been used for model evaluation.

509 *2.6.1. Contextual functionality*

510 The main objective of this study is to generate floor plan images for ar-
511 chitectural building design processes where the generated content is intended
512 to support: (i) the early design stage, and (ii) design identification between
513 clients and architects. Hence, quality standards should reflect the “diversity
514 of human demands” in a context-based manner. Driven by the above ob-
515 jective and the intended use of the framework, a manual validation test is
516 performed on the generated samples concerning contextual functionality.

517 Such a validation test also reflects the proposed framework’s practical
518 usability in the context of architectural floor plan design, allowing one to
519 assess how well each generated sample has adopted good design practices
520 from the input data. These practices are influenced by local building codes
521 and regulations, as well as structural, energy, and sun- and wind-exposure
522 considerations. To make this contextual functionality validation possible,
523 the authors conducted semi-structured interviews with experts in Danish
524 building architecture.

525 Inspired by the work of Park et al. [2], where over 80,000 real and syn-
526 thetic floor plans were qualitatively assessed based on architectural rules-
527 of-thumb, the key elements of that method are adapted for the contextual
528 functionality validation of the generated floor plans. The contextual func-
529 tionality and applicability of each analyzed floor plan image were gauged
530 by all authors of this study, taking into account the holistic synthetic floor
531 plan assessment guidance by Weber et al. [29], national guidance [84], and

532 domain-specific knowledge obtained from interviews with experts in Danish
533 architectural building design. The contextual functionality criteria that are
534 considered for validation are presented in Table 2. The number of misplaced
535 or missing doors and the number of windows with defects in each analyzed
536 floor plan were also counted. Fig. 3 shows examples of good and bad door
537 and window placement, with reference to a real floor plan.

538 Although the dataset does not distinguish between building age, the lay-
539 out planning in a Danish context can generally be divided into three design
540 paradigms from the period of 1950 to today. Houses in the period from
541 1950-1960 were designed with rooms centered around the building’s chimney,
542 which was the dominant heat source of that time, in a cruciform deformation
543 to exploit the heat in all the rooms of the building. Later, in 1960-1980,
544 the floor plan layouts became larger and rooms were designed to benefit
545 from the central heating system, where radiators were placed under the win-
546 dows in almost every room. The rooms were typically connected through
547 a pistol-shaped corridor, a characteristic of the building plans at the time.
548 Contemporary plan layouts are characterized by an open area kitchen and
549 dining room, with minimized corridor areas to optimize space, and the floor
550 heating systems (which most modern houses are equipped with) offer more
551 freedom in the room design shapes. However, the size of buildings has been
552 constantly increasing over time [85].

553 Since the proposed framework aims to support the early design stage
554 with automatically-generated HR floor plan layouts, on which further design
555 development can be based, it is important that the generated layouts support
556 the floor plan development in the right direction to be useful in that process.

Table 2: Contextual functionality validation criteria.

Structural design

- Minimization of the open area spans to reduce structural design issues.
- Presence of load-bearing and shear walls to bring stability to the construction.

Energy

- Minimization of the building’s energy consumption by reducing outer walls and placing fewer conditioned zones with buffer zones in the conditioned rooms.
- Balancing the access to daylight throughout the building layout with optimal space allocation.

Space optimization

- Minimization of the wasted space in layout planning, which is compatible with adaptable and modular layouts.

Layout relations

- Ensuring proper room connections.
-

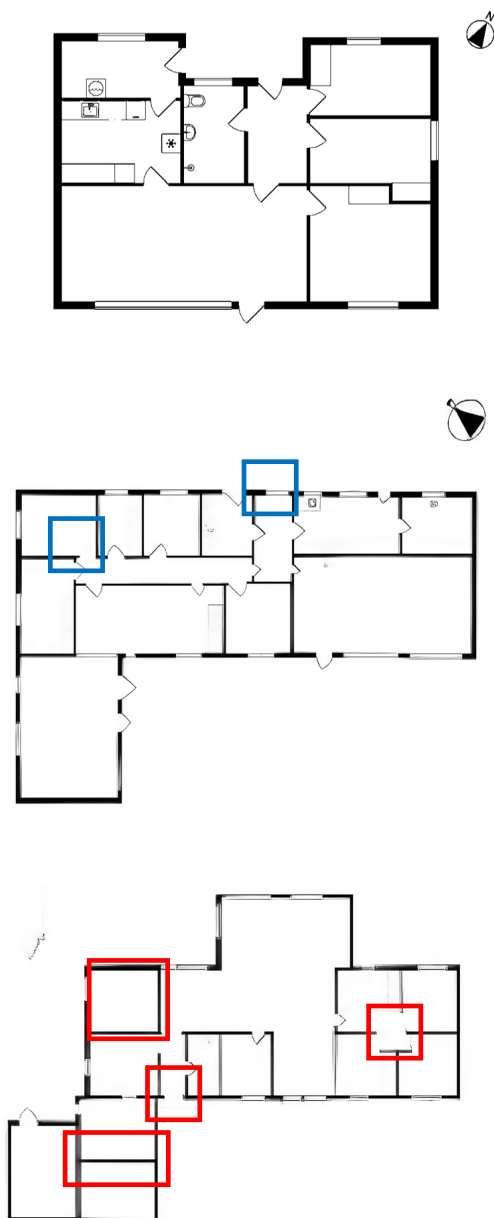


Figure 3: Top: A real floor plan image. Center: An example of a relatively good, generated floor plan with very few door and window misalignments, marked with blue boxes. Bottom: An example of a bad generation with several door, window (and even wall) missing details, marked with red boxes.

557 The average contextual functionality quality (Q_{context}) for each validation
 558 criterion is calculated by visual human evaluation of a generated sample of
 559 plan layouts using the formula:

$$Q_{\text{context}} = \frac{1}{N_{\text{Gen}}} \sum_{i=1}^{N_{\text{Gen}}} Q_i \quad (13)$$

560 where N_{Gen} is the number of generated images, and Q_i is the quality assess-
 561 ment of an individual floor plan image for the specific validation aspect (with
 562 permissible Q_i values: 2 = Excellent, 1 = Good, 0 = Insufficient).

563 2.6.2. Data privacy with perceptual image hashing

564 The existing GAN-based autonomous design generation literature is not
 565 aimed at detecting overfitting that could lead to violations of intellectual
 566 properties (IPs). However, copying data from the training set is a highly
 567 important challenge for generative models. It is necessary to ensure that
 568 these models do not obtain high-quality scores by merely outputting data
 569 similar to the input data [86]. To demonstrate the proposed framework’s
 570 capability to preserve privacy in floor plan generation, robust image hashing
 571 is utilized, where the generated floor plans are checked against the input
 572 data [87]. Image hashing is the most common data-copy detection method
 573 [88–90] and has been previously used to investigate if the outputs generated
 574 by GANs suffer from overfitting [91].

575 Image hashing allows the mapping of the perceptual image content into
 576 bit strings for content-based indexing [90, 92] by extracting content-based
 577 features. For perceptual hashing, a popular discrete cosine transform (DCT)-
 578 based technique (pHash) [87, 89] is utilized to transform images into their
 579 frequency components [93].

580 For evaluating image similarities, the Hamming distance (HD) is em-
 581 ployed, which has been shown to be efficient [90, 91, 94]. The normalized
 582 HD (NHD) [92] enables the use of different hash types. After hashing reduces
 583 the images to bitstrings, HD can be computed as follows [90]:

$$\text{HD}(h, h') = \sum_{i=1}^S I(h_i - h'_i) \quad (14)$$

584 where h_i, h'_i are the real and generated image bitstrings, respectively,

$$I(h_i - h'_i) = \begin{cases} 0, & h_i = h'_i \\ 1, & h_i \neq h'_i \end{cases}$$

585 denotes the indicator function, and S represents the total number of samples.

586 2.6.3. Latent space manipulation with image morphing

587 Another aim is to confirm that the framework can be manipulated to suit
 588 the designer’s needs. To this end, the approach leant on image morphing.
 589 Image morphing is an established technique within computer graphics and
 590 computer vision for image manipulation [95]. To drift the style, appearance,
 591 and size of floor plans, latent code interpolation is performed between the
 592 respective latent vectors of embedded images [95, 96]. By adding a linear
 593 combination of the latent vectors w_1 and w_2 of two embedded images, a
 594 subsequent image can be generated as a new outcome w . Cross-domain
 595 image morphing is also possible [20].

596 2.6.4. Downstream BIM implementation

597 The framework’s usability is showcased in a downstream task, namely
 598 a BIM software-based implementation task. Resketching the generated HR

599 floor plan using a BIM software environment offers the opportunity to make
600 the necessary design adjustments and to launch the layout planning of room
601 placement. The BIM design could benefit from the automated generated
602 floor plan with a fast and straightforward manual transformation of the raster
603 image into the spatial BIM format.

604 *2.6.5. Framework inference speed*

605 The intended use of the proposed framework is to be able to provide high-
606 quality products in a rapid way even in low-resource regimes. Therefore, the
607 average latent space sampling time is calculated for the two GAN-based mod-
608 ules of the framework for a sequence of 10,000 generated floor plan images.

609 *2.7. Implementation*

610 In this study, all implementations were run on a commercial NVIDIA
611 RXT-3090 GPU in a Linux-based Anaconda operating system.

612 *2.7.1. Pre-processing module*

613 The denoising ITRLE model is a local running Python script for pro-
614 cessing the full dataset automatically. To optimize the trade-off between
615 intended and unintended content removal from the floor plan dataset, an
616 inference threshold value of 0.4 was chosen [36]. After the denoising process,
617 the data were manually examined for errors. Any damaged data, in which
618 the automated pre-processing had produced either text leftovers or missing
619 image content, was scraped or reworked.

620 Next, any non-square images were squared up (1440×1440 pixels) by cen-
621 tering them into a square white background. These were then converted to
622 black and white images using the OpenCV library’s “COLOR_BGR2GRAY”

623 function [97], where a threshold value of 180 was set using “cv2.threshold”
624 and the type “cv2.THRESH_BINARY”, with 255 assigned to pixels exceed-
625 ing the threshold. The next step was to resize the images to 1024×1024 pixels
626 for processing purposes. Finally, all ($4 \times$) down-sampling methods were im-
627 plemented by running a local Python script based on OpenCV libraries [53].

628 2.7.2. Floor plan generation module

629 The StyleGAN3 implementation was based on the official PyTorch GitHub
630 release [13]. The dataset preparation was carried out using the “dataset_tool.py”
631 module [13]. The floor plan generation was processed by the built-in Style-
632 GAN3 script “gen_images.py”, which was customized to generate batches of
633 randomly generated floor plan layouts and to plot the sampling time. Dif-
634 ferent hyperparameter value combinations were explored with reference to
635 the proposed StyleGAN3 training configuration values [50]. A series of six
636 experiments was conducted for the end-to-end training of the StyleGAN3
637 floor plan generator on a set of different fixed and variable hyperparameters
638 (Tables 3 and 4). For training of models 1-6, ADA was activated with the
639 default target value of 0.6 [17]. Every model was trained for 10,000 king of
640 iterations, resulting in approximately one week of training per model. This
641 training duration was chosen to ensure a progressive and comprehensive flow
642 of the experiments. All six models were evaluated based on the FID score
643 and training time. The two best performing models were selected for a more
644 comprehensive evaluation covering generated floor plan image fidelity, diver-
645 sity, and truthfulness, as well as resource consumption and environmental
646 impact.

647 To assess the benefits of training with TL on pre-trained models for the

648 floor plan generator, a further experiment (model 7) was run, based on the hy-
649 perparameter set of one of the best-performing StyleGAN3 end-to-end mod-
650 els. The StyleGAN3 LHQ-256 model was trained with the first four layers of
651 the discriminator frozen. The generated floor plan image outputs of the TL
652 model were comprehensively assessed as described above.

Table 3: Hyperparameters for the end-to-end StyleGAN3 training experimental setup.

Type	Hyperparameters
Fixed	--cfg=stylegan3-t, --gpus=1, --aug=ada (0.6), --cmax=256, --kimg=10000
Variable	--batch, --batch-gpu, --gamma

Table 4: Tunable hyperparameter values for the six end-to-end trained models.

Model #	1	2	3	4	5	6
--gamma	2	2	32	32	32	32
--batch size	32	32	32	32	12	16
--batch gpu	16	4	16	4	4	16

653 To evaluate the impact of using Projected GAN with the FastGAN gener-
654 ator for floor plan synthesis, three additional models were trained. The first
655 model (model 8) was trained using the proposed hyperparameter settings
656 from the original implementation (batch=64, batch-GPU=8) [27]. In the
657 second model (model 9), both the batch and batch-GPU sizes were halved
658 (batch=32, batch-GPU=4). The third model (model 10) used the hyperpa-

parameter values of the best-performing StyleGAN3 end-to-end training experiment. The performance of all three models was thoroughly evaluated.

2.7.3. Up-sampling module

The Real-ESRGAN was fine-tuned on the domain-specific paired datasets comprising 984 LR and HR image pairs. The former comprises the down-sampled images by using the AREA method (pre-processing module), whereas the latter are the pre-processed images just before the down-sampling occurred. One model was trained for 30.5k iterations and another for 372.7k iterations. The implementation parameters are given in Table 5.

An initial comparison was made between the pre-trained Real-ESRGAN model, proposed in the original paper [48], and the custom-trained (fine-tuned) Real-ESRGAN. The pre-trained Real-ESRGAN model adapted many of ESRGAN’s [70] settings. It had been trained with four NVIDIA V100 GPUs on data from the DIV2K, Flickr2K, and Outdoor Scene Training (OST) datasets for 400K iterations with $lr = 1 \times 10^{-4}$, using a combination of L1 loss, perceptual loss [70, 98] and GAN loss [48]. The Adam optimizer with the settings $\beta_1 = 0.9$, $\beta_2 = 0.999$ had been used.

The baseline SRGAN up-sampling model was fine-tuned on the same custom data as Real-ESRGAN. A $4\times$ upscaling factor was attempted to output a final resolution of 1024×1024 pixels for the generated architectural floor plans. The pre-trained SRGAN had been trained with an NVIDIA Tesla M40 GPU, on a sample of 350 thousand random images from the ImageNet database [64]. The LR dataset had been obtained by using the bicubic kernel for down-sampling on the HR dataset with a factor of 4. The model had been further trained with an $lr = 10^{-4}$ for 10^5 update iterations and an $lr = 10^{-5}$

Table 5: Real-ESRGAN fine-tuning hyper-parameter values.

Parameter	Value
Optimizer	Adam
Batch size	8
Lr	1e-4
Scale	4
Network_g	RRDBNet
Beta 1, 2	[0.9, 0.99]
Network_d	UNetDiscriminatorSN
GAN loss	Vanilla
Use_shuffle	True
Patch_size	24

684 for another 10^5 iterations, where the Adam optimizer had been utilized with
685 $\beta_1 = 0.9$ [67]. The training hyper-parameter values are shown in Table 6.

686 2.7.4. Floor plan generation evaluation framework

687 For reporting FID values, the FID50k metric is utilized; that is, 50k gen-
688 erated test images are compared with the real dataset. To realize the metric,
689 the “calc_metrics.py” module was used, which is available in both the Style-
690 GAN3 and the Projected GAN with FastGAN generator implementations.

691 The density and coverage metrics were implemented using code from [99],
692 which builds on the original implementation by [24]. For the calculations, the
693 real LR dataset was compared to a dataset of 10k synthetic images generated
694 for each evaluated model. The nearest neighbor parameter values $k = 2$ for

Table 6: SRGAN fine-tuning hyper-parameter values.

Parameter	Value
Optimizer	Adam
Batch size	16
lr	1e-3
Scale	4
Patch size	24

the density metric and $k = 1$ for the coverage metric were utilized.

To estimate the detection score of each generative model, the Inception-v3 [73] CNN was trained for 50 epochs utilizing early stopping. For the training, a custom dataset containing 1,000 generated images and 988 real images is utilized. An 80/20 data split was made for the training/testing dataset, and accuracy was reported on the testing set.

Carbontracker was implemented using code from the GitHub repository [100]. The tracker was embedded to the training code of both the floor plan generation and up-sampling networks. Ten iterations were measured and the total energy consumption for the full training time was calculated based on the average CPU and GPU power values. Following a CI value of 149.75 CO₂/kWh was employed, corresponding to Danish data from 2020 [81].

2.7.5. Validation framework

For the contextual functionality validation, a sample of 25 out of 10,000 randomly generated and up-sampled floor plans by the best performing GAN-stack pipeline was selected. The reported values for all criteria are the mean

711 opinion scores assigned by the list of authors.

712 For computing and comparing the image-hashes, the method by [87, 101]
713 was employed. To detect any similarities between the two datasets (i.e., HR-
714 real and SR-fake), 984 real HR floor plans and 25 generated floor plans were
715 used. Additionally, a function was applied to obtain 12 positions in each
716 real image by rotating the images by 90°, 180°, and 270°, and by flipping the
717 image positions horizontally and vertically to make the implementation more
718 robust. This process provided 295,200 data rows for analyzing the diversity
719 in real and fake images, aiming to verify the framework’s capability to gen-
720 erate unique floor plan designs. The pHash relied on the DCT technique for
721 generating 6-bit hash strings for each real and fake image, which were then
722 compared using the HD metric [101]. It was implemented using code from
723 the Imagehashing library [101].

724 3. Results

725 3.1. Pre-processing module

726 3.1.1. Dataset denoising

727 After scraping the damaged data (i.e., images with text leftovers or miss-
728 ing image content) of the denoising process, the dataset size was reduced
729 from 1,233 to 984. Fig. 4 illustrates the LR generated floor plans when
730 training the StyleGAN3 generator with and without text denoising. A visual
731 assessment of Fig. 4 clearly emphasizes the need for removing text content,
732 since the generated image text in the floor plan (A) is unrecognizable and
733 appears as noise that will most likely impact further post-processing through
734 the framework. Thereby this result strongly advocates for removing text

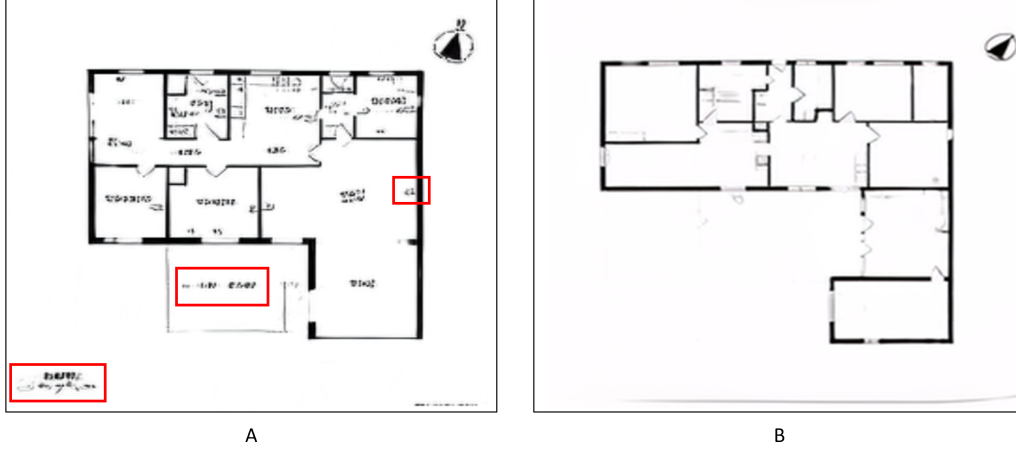


Figure 4: Generated floor plans when StyleGAN3 was trained (A) on the LR version of the original dataset at batch size 12, and (B) on the denoised and downsampled dataset at batch size 12. Red boxes indicate locations of generated unrecognizable noise.

735 from the input dataset due to the GAN model’s poor ability to imitate text.

736 3.1.2. Dataset down-sampling

737 Fig. 5 shows the results of the four tested four-fold down-sampling meth-
 738 ods, as well as the HR input image with a resolution of 1024×1024 pixels.
 739 Down-sampling the images to 256×256 pixels with the AREA module min-
 740 imized the loss of details compared to the other three methods, and the
 741 results appear less blurred and have a better representation of the geometry
 742 compared to the other methods. The bilinear and bicubic down-sampling
 743 methods tend to have a relatively high loss of details such that the image
 744 details appear very crude.

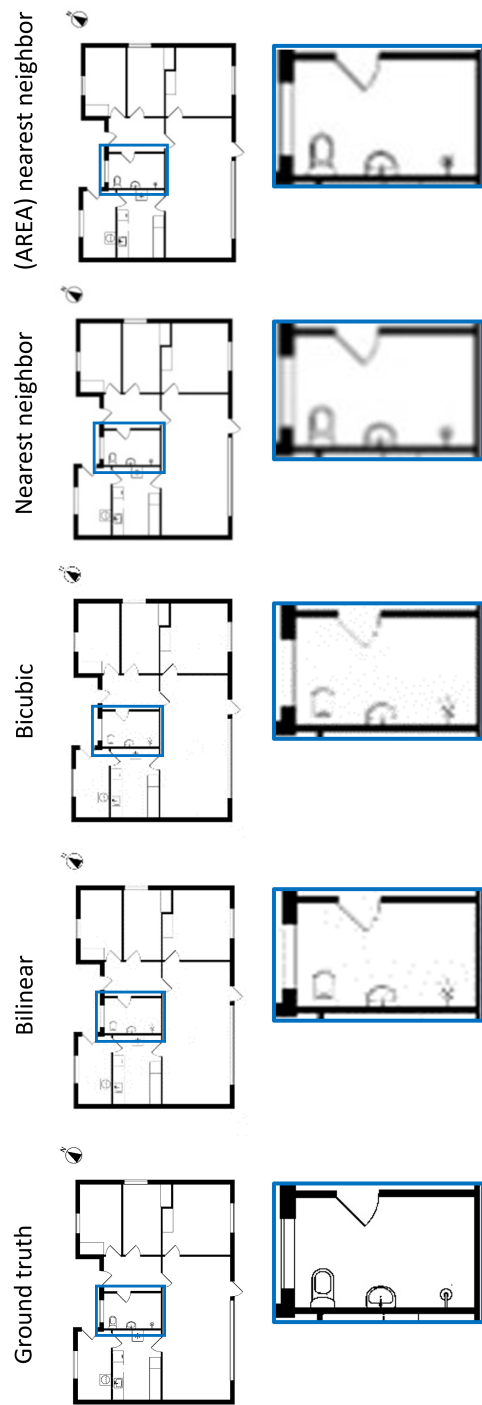


Figure 5: Results of the down-sampling methods in the dataset pre-processing. The images are resized from 1024×1024 pixels to 256×256 pixels. Top: The complete floor plans. Bottom: Zoomed-in view of the areas inside the blue boxes.

Table 7: Overview of results (FID, training time, iterations) for end-to-end StyleGAN3 training. Boldface indicates best performance.

Model #	FID	Time [h/m]	Iterations [king]
1	88.9	146h 58m	10,000
2	101.7	112h 14m	7,008
3	30.4	103h 40m	7,008
4	55.4	160h 00m	10,000
5	19.1	112h 13m	7,005
6	19.3	147h 53m	10,000

745 3.2. Floor plan generation module

746 3.2.1. StyleGAN3 end-to-end training

747 The results of the end-to-end training are shown in Table 7. Model 5
748 achieved the lowest FID score (19.1) with the shortest training time (≈ 112 h).
749 Model 6 ranked second, with an FID score of 19.3 after ≈ 148 hours of train-
750 ing. Although the lower batch size and batch GPU in experiment 5 resulted
751 in a longer training time per king compared to model 6, the image quality
752 generated by model 5 improved much faster than that of model 6.

753 Fig. 6 shows a plot of the model performance (calculated FID score)
754 as a function of king (total number of training iterations) for the end-to-
755 end StyleGAN3 training experiments. The graph is based on four FID data
756 points for each experiment, computed at 1k, 4k, 7k, and 10k of king. It can
757 be seen that increasing the value of the regularization weight gamma from 2.0
758 (models 1 and 2) to 32.0 (models 3-6) made the performance improve rapidly.

759 In addition, using a batch size smaller than 32 exhibited a lower FID score
 760 after 1k king and showed a relatively flat learning curve over the 10k king
 761 iterations. Also shown in Fig. 6 (right) are representative floor plan images
 762 generated at the minimum FID timepoint for the six models. The visual
 763 assessment is in agreement with the calculated FID scores. Models with
 764 higher gamma combined with a lower batch size tend to generate output
 765 data that is more representative of the input data. In summary, models 5
 766 and 6 have minor differences in performance. Model 5 has a higher processing
 767 time per iteration, but it achieves a lower FID score much earlier than model
 768 6. Both models 5 and 6 appear to have reached their optimum level, and
 769 they would most likely have started collapsing like model 3 if the training
 770 process had continued.

771 3.2.2. *StyleGAN3 transfer learning*

772 The results in Table 8 show that utilizing TL for StyleGAN3 (with hyper-
 773 parameter values: batch size=16, batch GPU=16, gamma=32) helped the
 774 model achieve faster training progress by adapting the weights from the pre-
 775 trained model. TL based on LHQ-256 reached an FID level of 20.8 after
 776 1k king iterations (22h of training). However, even though TL provided
 777 faster FID development compared to the end-to-end training of the Style-
 778 GAN model, a model collapse was observed, resulting in a decrease in model
 779 performance. The training adaptation process of the StyleGAN3 LHQ-256
 780 pre-trained model is shown in Fig. 7, where it can be clearly seen how the
 781 model transitions from the pre-trained landscape domain to the style domain
 782 of floor plans.

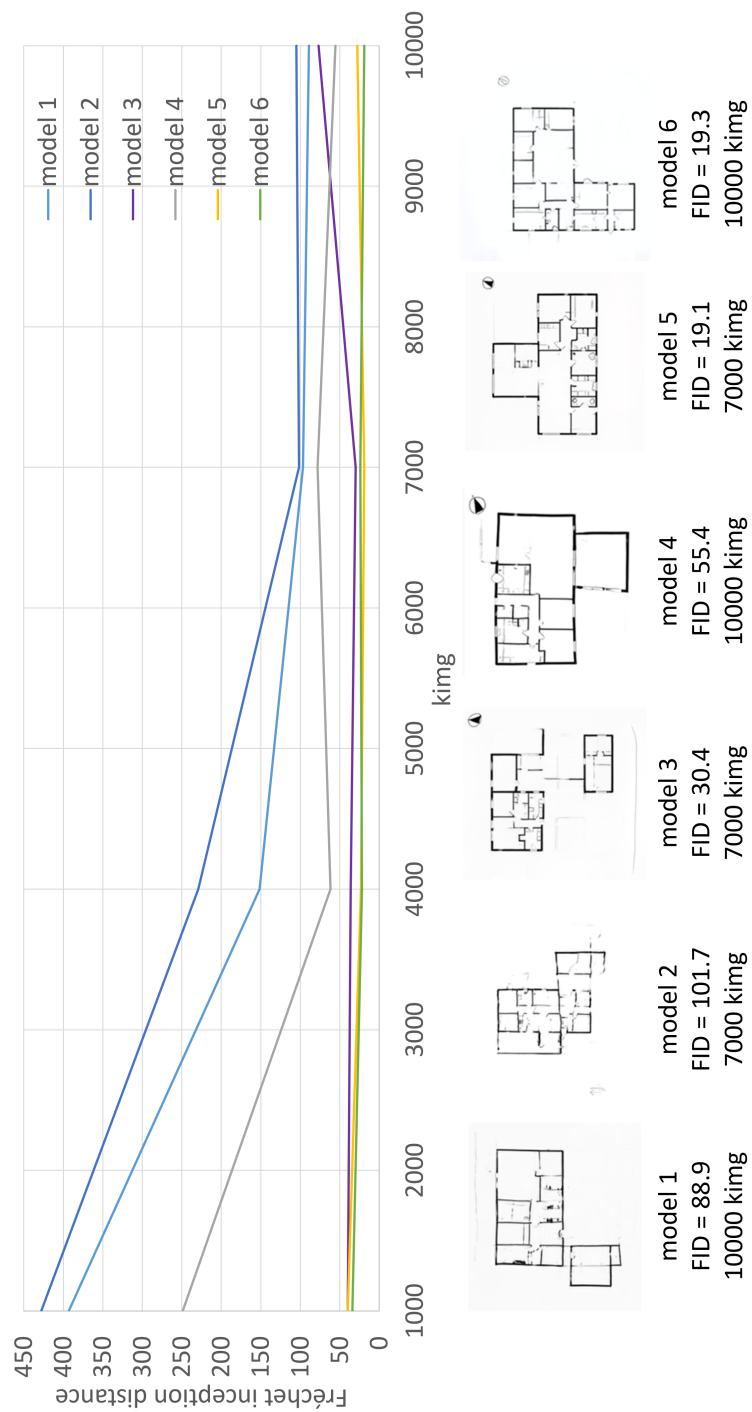


Figure 6: End-to-end StyleGAN3 training results (left), and representative floor plan images generated at the minimum FID timepoint (right) for the six models.

Table 8: Overview of results (FID, training time) for StyleGAN3 transfer learning (model 7).

King	0.2k	1k	4k	7k	10k
FID	22.2	20.8	56.8	62.5	75.1
Time	4h	21h	86h	151h	215h
	30m	44m	13m	03m	32m



Figure 7: A representative example of a generated floor plan image with StyleGAN3 transfer learning. Left is the initial result of the LHQ-256 model, center is the result after 16 king, and right is the result after 1,000 king (FID=20.8).

Table 9: Overview of results (FID, training time, iterations) for Projected GAN with FastGAN generator training (models 8-10). Boldface indicates best performance.

Model #	FID	Time [h/m]	Iterations [king]
8	9.8	85h 59m	9,475
9	9.4	58h 30m	5,400
10	9.3	106h 54m	8,416

3.2.3. Projected GAN with FastGAN generator

The results of the Projected GAN with FastGAN generator training are outlined in Table 9. It was observed that reducing the batch and batch-GPU sizes by half slightly improved the FID performance. Model 10, which used the hyperparameter values of model 5, achieved an FID of 9.3 after 106h 54m of training time. Fig. 8 displays both the FID vs. king graph (left) and the generated images in the best performing iterations (right). Even though the generated image quality performance of the three implemented models is quite similar, model 9 converged faster in relation to FID. Both models 9 and 10 have a significantly faster learning curve compared to model 8.

3.2.4. Comprehensive comparative analysis of floor plan image generation models

The graph in Fig. 9 (top) aggregates the best-performing models (based on FID vs. training time) of the three utilized training approaches. It can be seen that the Projected GAN with FastGAN generator-based models have superior performance compared to the other models. In comparison to the

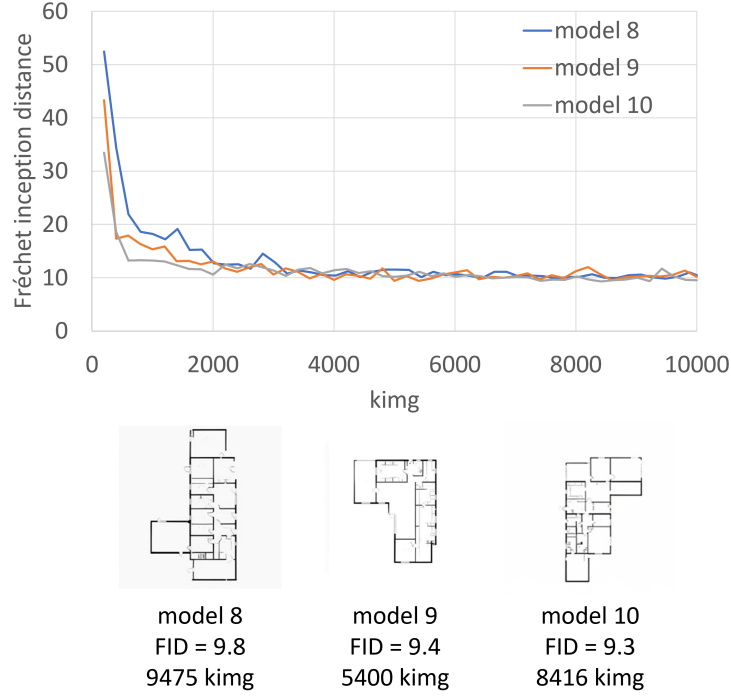


Figure 8: Projected GAN with FastGAN generator training results (left), and representative floor plan images generated at the minimum FID timepoint (right) for the three models.

799 StyleGAN3 end-to-end trained models, the FID score was improved from
800 19.1 in model 5 after 112h of training to 9.4 after 59h of training in model 9.

801 In reference to the StyleGAN TL training, the TL model achieved its
802 minimum FID score of 21 after 22h of training, whereas model 9 achieved an
803 FID score of 17 after only 4h of training. Hence, a primary strength of the
804 Projected GAN with FastGAN generator is the low FID score, which is not
805 attained by other methods, even when extending their training time.

806 From a visual inspection point of view, the anticipated quality gap in
807 the generated floor plan images by the various models (Fig. 9, bottom), as

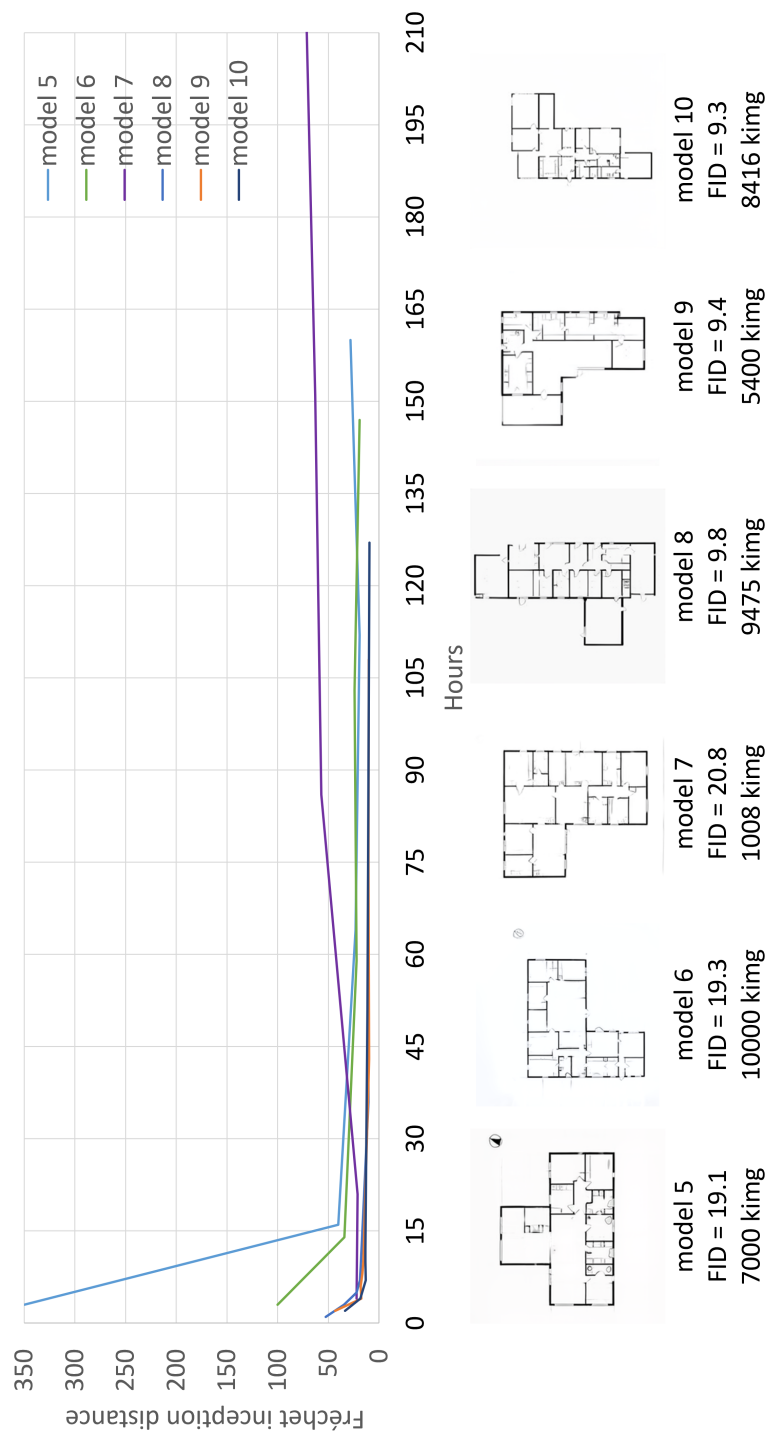


Figure 9: Comparative analysis of training performance (top), and representative floor plan images generated at the minimum FID timepoint (bottom) for all explored models.

808 indicated by the FID differences, was not observed.

809 Table 10 gives the holistic quantitative comparison results of the investi-
810 gated floor plan image generation methods. From the results, a connection
811 between GPU-batch size and duration time per training iteration is seen,
812 where the models with a high GPU-batch seem to achieve a lower FID ear-
813 lier in the training process. There is also a relatively high difference in the
814 GPU memory requirements during training among the various models. How-
815 ever, all those GPU memory values are nowadays easily available.

816 The CNN classifier detection scores show that the Projected GAN with
817 FastGAN implementations (models 8-10) achieved by far the lowest accuracy
818 scores, indicating that these models are the best at generating fake outputs
819 with high truthfulness.

820 The density and coverage results show that the generated floor plan im-
821 ages by the Projected GAN with FastGAN generator-based model 9 achieved
822 the best image fidelity and diversity performance. Moreover, the performance
823 of the CNN classifier shows that model 9 has the highest number of false neg-
824 ative (false) instances and the lowest number of true positives (real) image
825 instances. This demonstrates that the generated floor plan images by model 9
826 are relatively indistinguishable from the real image data. It also achieved the
827 second-best FID score. The density and coverage metrics have nevertheless
828 been shown to provide a more exhaustive and itemized image quality perfor-
829 mance evaluation than the single-number FID, which has received criticism
830 [24].

831 Furthermore, model 9 notably emitted the lowest carbon dioxide emis-
832 sions during training in comparison to the other models that were in the

Table 10: Holistic floor plan generation model performance evaluation. Boldface indicates best performance.

Model	FID	Density/Coverage (ImageNet)	CNN detection score (GoogLeNet)		Training resource consumption		Energy consumption					
#		Density	Coverage	TP/FP	FN/TN	Acc.	Iterations [king]	Time [h/m]	VRAM [GB]	Energy [kWh]	CO ₂ eq [g]	Travel [km]
5	19.1	0.75	0.88	170/22	30/178	0.87	7,005	112h 13m	1.9	48.31	7,235	67.3
6	19.3	0.92	0.95	174/28	26/172	0.87	10,000	147h 53m	7.0	63.54	9,516	88.5
7	20.8	0.69	0.90	164/29	36/171	0.84	1,008	21h 44m	10.4	9.12	1,365	12.7
8	9.8	0.94	0.95	146/48	54/152	0.75	9,475	85h 59m	7.3	35.67	5,341	49.7
9	9.4	1.01	0.97	137/34	63/166	0.76	5,400	58h 30m	5.2	23.45	3,512	32.7
10	9.3	0.92	0.96	150/54	50/146	0.74	8,416	106h 54m	5.3	38.18	6,491	60.4

833 same FID range, causing pollution that is equivalent to traveling a distance
834 of 32.7 km by car [81]. This was nearly half the emissions of model 10, which
835 achieved a slightly better FID score, and 63% lower than the emissions of
836 the worst polluter (model 6). Model 9 also required significantly less training
837 time than the other models that achieved an FID score below 10. Therefore, it
838 is concluded that the Projected GAN with FastGAN generator-based Model
839 9 is the most effective model for floor plan image generation.

840 3.3. Up-sampling module

841 3.3.1. Real-ESRGAN

842 Fig. 10 shows the up-sampling comparison of the generated (synthetic)
843 floor plans between the pre-trained Real-ESRGAN (for blind-SR) and the
844 Real-ESRGAN fine-tuned on the custom dataset. The visual assessment of
845 the results corroborates that the fine-tuned Real-ESRGAN (for 30.5k itera-
846 tions) achieved superior results to the pre-trained Real-ESRGAN regarding
847 upscaling quality and restoration of detail errors. The up-sampling by the
848 custom trained Real-ESRGAN has improved features like walls, windows,
849 and door openings, and these features are better represented with fewer miss-
850 ing details.

851 Fig. 11 shows the results when continuing the Real-ESRGAN model
852 training from 30.5k to 372.7k iterations. Even though continuing training
853 improves the details and general quality of the generated images by sup-
854 pressing noise and repairing walls, doors, and windows, the model trained at
855 372.7k iterations begins to remove notable functional features from the floor
856 plans or to hallucinate windows instead of doors.

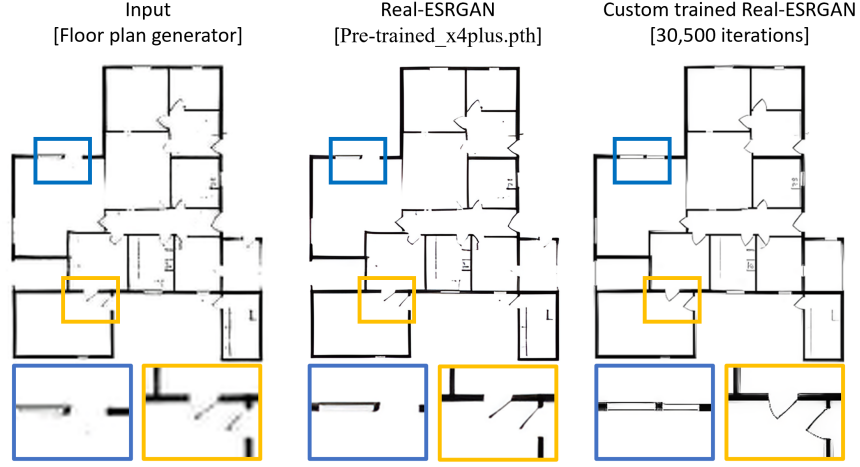


Figure 10: Comparison of the $4\times$ upscaling process between the pre-trained Real-ESRGAN (2nd column) and the custom fine-tuned Real-ESRGAN (3rd column), based on a generated input image of 256×256 pixels (1st column). Top: The complete floor plans. Bottom: Zoomed-in view of the areas inside the blue and yellow boxes.

857 3.3.2. Baseline comparison

858 Fig. 12 illustrates that the Real-ESRGAN has far superior results to the
859 baseline custom-trained SRGAN [67]. The outcome appears much sharper in
860 the Real-ESGAN-based results. Table 11 outlines the quantitative compari-
861 son between SRGAN and Real-ESRGAN (both custom-trained) in terms of
862 training duration, energy consumption, and associated carbon dioxide emis-
863 sions. Although SRGAN has lower energy requirements and environmental
864 cost, its performance was deemed insufficient due to poor generated image
865 quality compared to the more complex Real-ESRGAN architecture. There-
866 fore, it is concluded that the custom-trained Real-ESRGAN architecture is
867 the best option for the image up-sampling module.

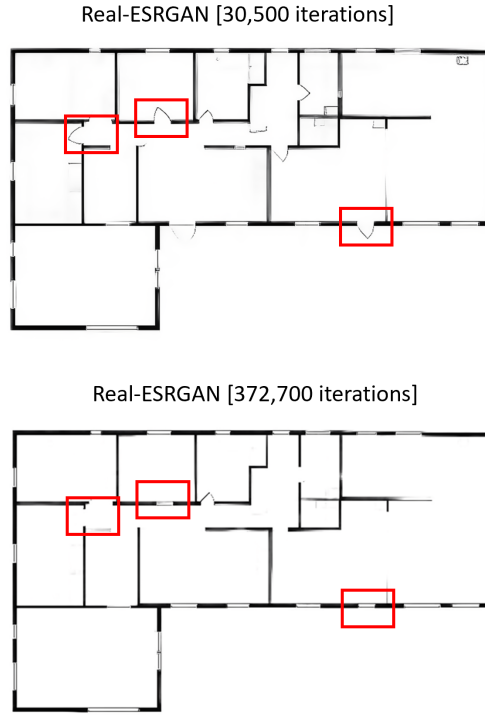


Figure 11: Real-ESRGAN comparison of training iterations. Bottom image shows overshoot artifacts (in red boxes) caused by the prolonged training.

Table 11: Comparison of SRGAN and Real-ESRGAN (both custom-trained) in terms of training duration, energy consumption and associated carbon dioxide emissions. Boldface indicates best performance.

Model	Iterations	Training hours	Total energy	Total CO ₂ eq	Travel
	[king]	[h]	[kWh]	[g]	[km]
Real-ESRGAN	30.5	4.96	2.05	307.64	2.9
SRGAN	31.2	0.78	0.19	27.86	0.3

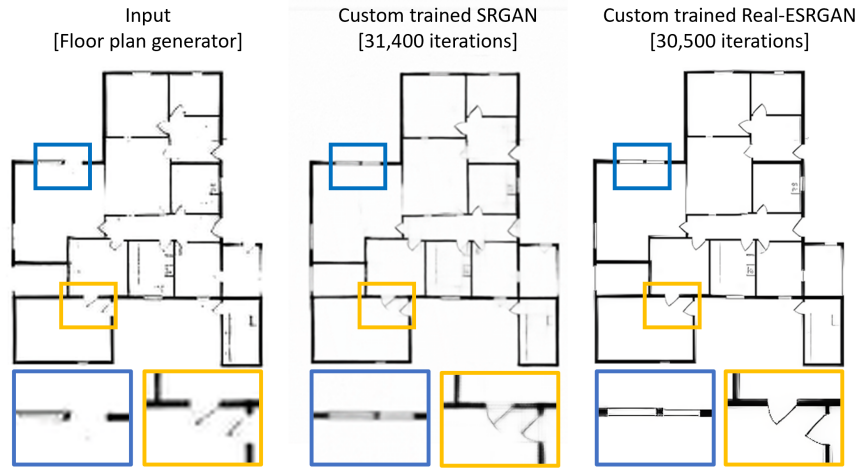


Figure 12: A comparison of the $4\times$ upscaling process by the pre-trained Real-ESRGAN (2nd column) and the custom fine-tuned Real-ESRGAN (3rd column), based on an input generated image of 256×256 pixels (1st column). Top: The complete floor plans. Bottom: Zoomed-in view of the areas inside the blue and yellow boxes.

3.4. Validation framework

3.4.1. Contextual functionality

Fig. 13 illustrates the 25 randomly selected floor plans for the framework validation. Table 12 lists the (individual and mean) numbers of misaligned doors and windows, as well as the contextual functionality scores obtained from the visual evaluation. It was found that the floor plans have an average of 1.32 misplaced or missing doors. The number of windows with defects was 1.82 per floor plan on average. However, it can be seen that the generated floor plans, in general, have performed well in placing windows and doors in contextually meaningful locations from an architectural perspective. For instance, windows are typically placed in outer walls and have a well-composed size relative to the room size, whereas doors are often placed so that they

880 open against a wall, which saves space in the layout design.

881 Structural area span, denoting the distance between bearing points in
882 the layout, was mainly affected by the large, squared rooms where the span
883 between two bearing walls is large. Relative to load and shear walls, the sta-
884 bility of the building relies on a well-composed layout that has both trans-
885 verse and longitudinal walls, which will help resist wind load. Of course,
886 other techniques do exist that provide wind bracing when spatial open de-
887 signs are required. In general, the 25 analyzed floor plan designs performed
888 exceptionally well on these two structural design functionalities.

889 Building energy has many regulating parameters, such as outer shape,
890 natural lighting, material selection etc. In fact, the early design phase is to
891 some degree disconnected from the detailed material selection. The main
892 selection criteria are based on the floor plan layout’s ability to minimize the
893 linear thermal bridging and optimize the natural light entrance, for which
894 the 25 analyzed designs showed a relatively good performance.

895 The waste of space in the floor plans gives an indication of the space used
896 in the connecting corridor. The placement of doors and windows also plays
897 a role if they take up unnecessary space by their functioning. The above
898 criterion together with the room connectivity optimization criterion were
899 found in general to underperform to some extent in the 25 analyzed synthetic
900 HR floor plans. In summary, regarding structural and energy considerations,
901 the proposed floor plan generation framework was verified to be consistent
902 with the intended application.

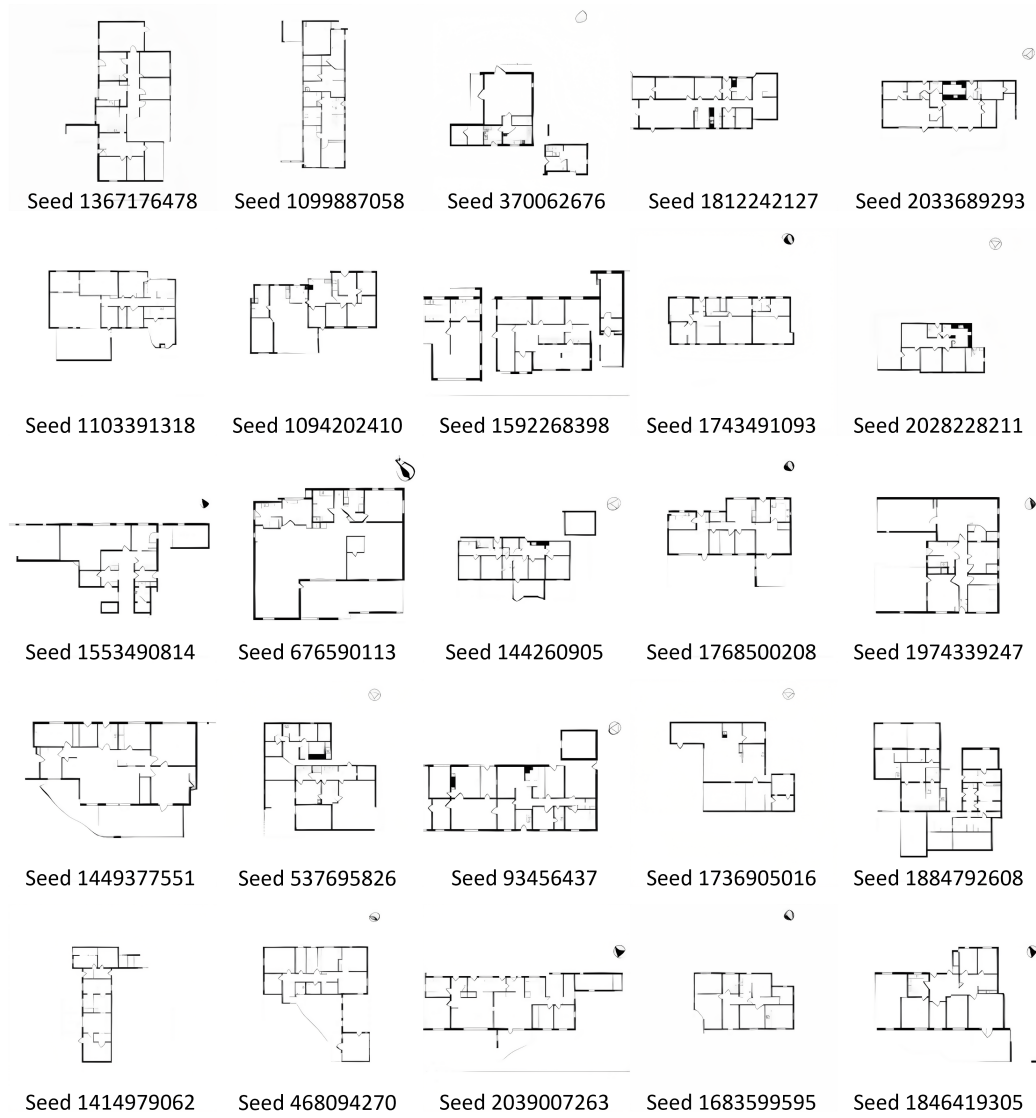


Figure 13: Image grid of the 25 floor plans used for the validation framework.

Table 12: Contextual functionality validation of the 25 analyzed floor plans, out of 10,000 randomly generations. Listed are the (individual and average) numbers of misaligned doors and windows, as well as the scores for each contextual functionality validation criterion. Reported are the mean opinion scores assigned by all co-authors.

Seed	Doors	Windows	Open-area minimization	Structural stability	Energy efficiency	Access to daylight	Wasted space minimization	Room connectivity
1099887058	3	2	2	2	1	2	1	0
1367176478	0	3	2	2	1	2	0	0
370062676	2	3	1	2	1	2	1	1
1812242127	1	3	2	2	2	2	0	2
2033689293	2	3	2	2	1	1	1	0
1103391318	2	1	2	2	2	0	1	1
1094202410	0	0	2	2	2	1	2	2
1592268398	0	2	2	2	0	1	1	2
1743491093	1	0	2	2	2	2	1	0
2028228211	1	3	2	2	2	2	1	2
1553490814	1	3	2	2	0	1	0	0
676590113	0	1	0	1	1	2	2	2
144260905	1	3	2	2	1	1	1	0
1768500208	0	1	2	2	2	1	1	2
1974339247	3	0	1	2	1	1	0	1
1449377551	0	0	2	2	1	2	0	2
537695826	3	3	2	2	1	0	1	0
93456437	1	3	2	2	1	1	0	0
1736905016	1	3	1	2	1	1	2	2
1884792608	3	3	2	2	0	0	0	0
1414979062	1	3	2	2	0	1	0	0
468094270	2	2	2	2	1	2	1	1
2039007263	1	0	2	2	1	2	0	0
1683599595	3	0	2	2	2	1	1	1
1846419305	1	0	2	2	1	1	1	1
Average	1.32	1.80	1.80	1.96	1.12	1.28	0.76	0.88

903 3.4.2. Data privacy with perceptual image hashing

904 Fig. 14 shows an example of a pair of floor plans with the lowest HD
 905 (pHash) and the corresponding 8×8 hash images. Some similarities due
 906 to the building shape can be observed, but there are no critical signs that
 907 indicate overfitting. This result verifies that no direct data copying takes
 908 place between the HR real dataset and the HR fake (after SR) dataset.

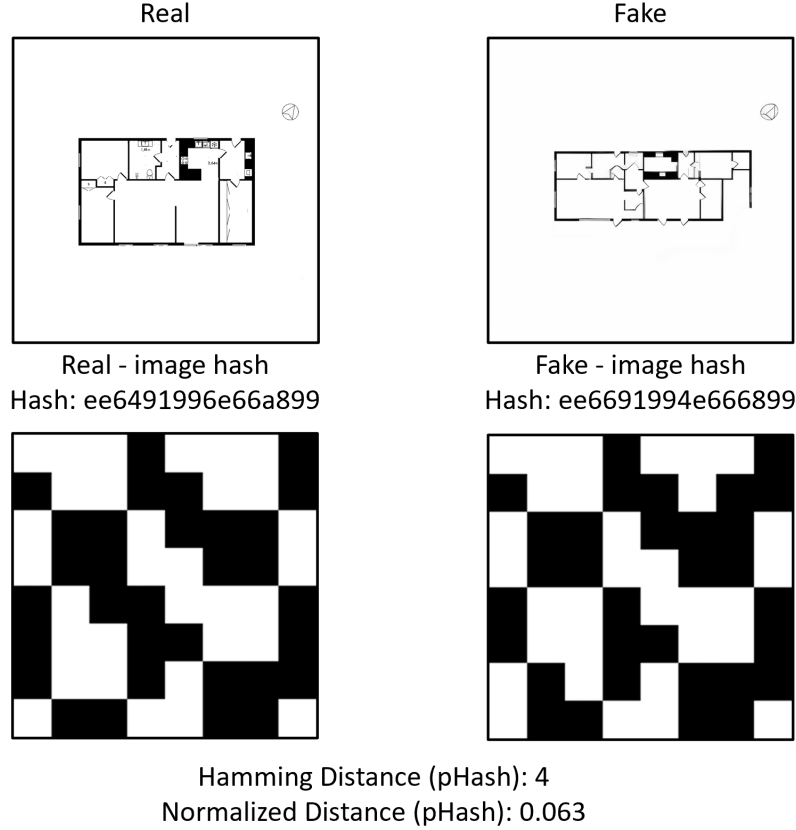


Figure 14: Overfitting evaluation based on model 9 (seed 2033689293) between a real floor plan image (left), and a synthetically generated floor plan sample (right). Top: The generated floor plans. Bottom: The 8×8 hash images. Also given are the Hamming and normalized Hamming distances.

909 3.4.3. *Latent space manipulation with image morphing*

910 Fig. 15 shows an example of the image morphing transformation tech-
911 nique, demonstrating the floor plan generator’s capability of controlling the
912 building layout size and shape styles by selecting seed versions from the
913 learned latent space. It illustrates the transformative power of the floor plan
914 generation module which gradually affects the style from one seed to another
915 (from left to right). The style drifting capability enables a deeper layer of
916 floor plan outputs.

917 3.4.4. *Downstream BIM implementation*

918 Fig. 16 shows an example of resketching a generated floor plan using a
919 BIM software environment. Seen is a building layout of 185 m² with maxi-
920 mum span of 5862 mm, two sheds with an option for transforming the large
921 one into a garage, two entrances, two bathrooms, a master bedroom with
922 connected bathroom, three rooms for children’s rooms/office, a connected
923 kitchen and living room, and a laundry room that connects the house with
924 the shed/garage. All rooms are connected with corridors (or entrances) which
925 can serve as buffer zones for heat differences between the rooms. The build-
926 ing design caters to all common construction principles such as light wall
927 timber frames etc. There is an option for placing shared walls and loading
928 walls in the floor plan design, due to the even distribution of walls. The
929 door openings seem to naturally optimize the space around them, also avoid-
930 ing collision with other openings. All living spaces (living room with open
931 kitchen) have access to daylight. For the living room, there is also an option
932 to add extra windows. The depth of all rooms relative to the window place-
933 ment also seems reasonable to ensure sufficient natural daylight. The only

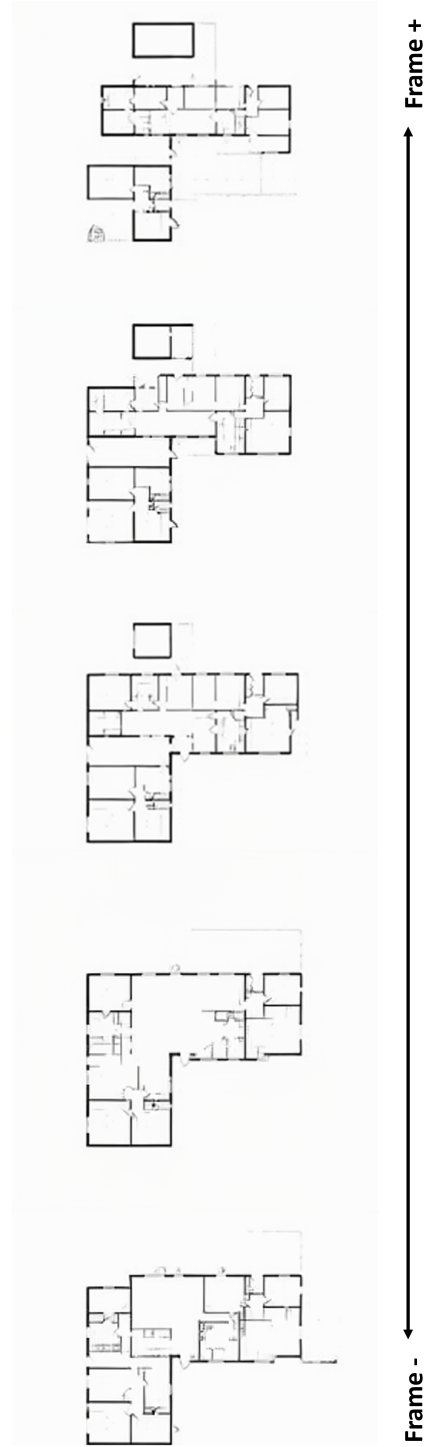


Figure 15: Floor plan generation with image morphing based on model 9.

Table 13: Framework sampling time based on 10,000 generated floor plan images.

Module	Sampling time [sec/image]
Floor plan generator (Projected GAN with FastGAN)	0.02
Up-sampling (Real-ESRGAN)	0.15

challenges of the specific floor plan design are mainly related to the share of outer walls that may reduce the energy performance and the indented areas which may cause limited light in the building.

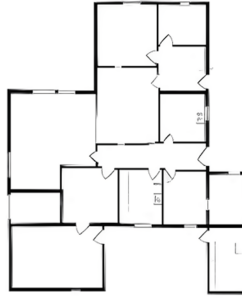
3.4.5. Framework inference speed

Table 13 outlines the average sampling times per image for the Projected GAN with FastGAN floor plan generator (model 9) and the custom-trained Real-ESRGAN up-sampling model. These values verify that the generated floor plan images by the framework are obtained in a swift manner while relying on limited computational resources.

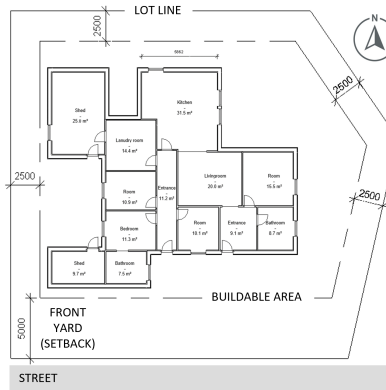
4. Discussion and future work

4.1. Main contribution of this study

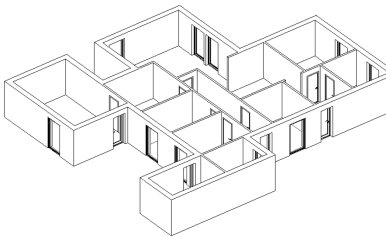
This study aimed to make progress on the four main challenges that concern the use of GML in construction applications, as they were pointed out in the recent review study by Chai et al. [22]. A novel stable multi-module GAN-stack was proposed for the autonomous generation of high-fidelity and



Synthetic floor plan



Resketched floor plan



3D floor plan

Figure 16: Architectural resketching (center) using BIM software of a generated output (top) by the framework, together with the corresponding 3D floor plan (bottom), also produced by BIM. The building area is 185 m^2 , and the max structural span is 5862 mm.

contextually meaningful floor plan layouts by relying on small-scale computational and data resources. The pipeline is made up of a pre-processing (i.e., denoising and $4\times$ down-sampling) module, a floor plan image generation module, and a $4\times$ image up-sampling module. Innovative frameworks were also introduced for: (i) holistic evaluation of the generated floor plans, and (ii) contextual validation of the proposed pipeline.

4.2. Experiments and main findings

All GAN experiments in this study were conducted using minimal training data and computational resources, namely a sub-thousand dataset of high-fidelity architectural floor plan images of single-story residential Danish homes and a single RTX-3090 GPU. To determine the best configuration of the individual modules, this work focused primarily on systematic hyperparameter tuning and model fine-tuning (custom training). In the pre-processing module, a transformer-based denoising approach [47] was deployed. The effect of cleaning datasets from unwanted text content was showcased in a downstream image generation task. Visual assessment was adopted, because using a quantitative metric such as the FID score would not have provided any meaningful insights. The experiments showed that removing text content from the training data significantly improved its practical utility. For down-sampling, a method based on OpenCV’s AREA module [46, 53] was utilized, which resulted in minimal degradation compared to other approaches. For the floor plan generation module, 10 different models were tested by varying the network architecture, training mode, and set of hyperparameter values. It was found that the best performing model was the one based on the Projected GAN with FastGAN generator [19, 21], utilizing

half the batch and batch GPU sizes compared to the original implementa-
 tion. The proposed model gained superior quality, reaching an FID score of
 9.4. The respective scores for the traditional StyleGAN3-based models were
 above 19.1. The comprehensive evaluation demonstrated that the proposed
 floor plan generation model also achieved the best density ($=1.01$) and cov-
 erage ($=0.97$) values, and the highest level of truthfulness of the generated
 images. In addition, it converged in less than 60 hours which was almost
 twice as fast as the other StyleGAN3 end-to-end trained models. Moreover,
 its energy consumption and carbon emissions were substantially lessened,
 highlighting its high resource efficiency and low environmental cost. Energy
 consumption and thereby CF should be relevant to GAN performance eval-
 uation given the growing concerns about the environmental impact of deep
 learning [13, 82, 83]. Extending training iterations alone would have been
 unlikely to yield major improvements due to stagnation in the model de-
 velopment, possibly caused by the limited dataset size. In the up-sampling
 module, a custom-trained Real-ESRGAN model [48] was selected as it was
 found to convincingly outperform the baseline method. The novel valida-
 tion framework was applied to 25 randomly selected floor plans, verifying
 that the proposed pipeline: (i) preserves data privacy, (ii) can be manipu-
 lated with image morphing to suit the designer’s needs, (iii) is exploitable
 in downstream vector-based BIM software implementation tasks, and (iv)
 can generate contextually meaningful 1024×1024 floor plan images in a swift
 manner by relying on small-scale hardware resources. Lastly, the contex-
 tual functionality validation corroborated that the proposed framework is
 consistent with the intended application regarding door/window placement,

999 structural design, and energy considerations, whereas there is some room for
1000 improvement in the space optimization and room connectivity aspects.

1001 *4.3. Why it works so well?*

1002 The outstanding efficiency and generated image quality performance that
1003 was achieved by the proposed multi-module GAN-stack is the result of adopt-
1004 ing best practices from a different context (i.e., natural images) for selecting
1005 the network architectures and set of techniques that suit the needs and goals
1006 of this study. Apart from historical precedent, several crucial network de-
1007 sign choices were based on empirical experimentations and intuition. GML
1008 architectures, which are able to rapidly sample training datasets and allow
1009 substantial control over the generation process, were chosen. The reduced
1010 computational cost and training time in the floor plan generation process
1011 without compromising image quality is most likely attributed to employing
1012 EfficientNet-Lite1 [63] as the pre-trained feature network. The stabilized
1013 training of the GAN-stack is in all probability the outcome of utilizing: (i)
1014 pre-trained representations in the floor plan image generation (Projected
1015 GAN with FastGAN generator) module, and (ii) U-Net design with skip
1016 connections together with spectral normalization in the discriminator of the
1017 image up-sampling (Real-ESRGAN) module. The lack of overfitting is pre-
1018 sumably due to leveraging differentiable augmentation methods [65] in the
1019 floor plan generation process.

1020 *4.4. Validation framework*

1021 A novel validation framework was introduced to confirm that the proposed
1022 computerized pipeline captures architectural business rules, attributes, and

relationships correctly. Ensuring high-quality validation is a troublesome task considering the strict regulations and the growing sophistication levels of related frameworks. However, validation is a pivotal step towards developing consistent, high-quality products. Domain-specific knowledge is essential for appreciating the validation criteria that matter the most in this type of research. Generating highly realistic-looking floor plan images is not an end in itself — contextual relevance is more critical. In this study, architects with experience in detached single-family home design were interviewed to provide the co-authors with domain-specific insight into the architectural context. Complementary literature was also adjusted [29, 84].

4.5. *Data privacy*

Fair usage, intellectual property issues, and ethical concerns regarding data sourcing have brought GML into question. There is no clear answer yet to these copyright issues [102]. To our knowledge, never before has data privacy been assessed in GAN-based floor plan generation papers. To validate data privacy and overfitting in this study, perceptual image hashing was introduced as a metric for detecting similarities between the real dataset and generated data. Image hashing, while computationally intensive on larger datasets, has the benefit that the generated hash of real images is stationary and can be reused in future evaluations. Rotations and flips were applied to the dataset for a more robust evaluation. Although GANs may avoid overfitting through their built-in classification mechanisms, image hashing provides a transparent and explainable method to ensure data privacy.

1046 4.6. Usefulness of the work

1047 The presented work could inspire new perspectives to the early architec-
1048 tural design phase. As the communication between professional consultants
1049 and their clients progresses in this essential phase, the proposed framework
1050 has the potential to enable the fast iterative prototyping of conceptual floor
1051 plan layouts. A large number of empirical experiments were conducted over
1052 the course of this study, which exceed the ones presented in this paper. These
1053 tests have provided the co-authors with valuable insights into the training
1054 process of various GANs in the context of architectural floor plan design.
1055 They have also contributed to a knowledge base for future research aimed
1056 at further improving the related training methods. This research advances
1057 GML resource efficiency for architectural applications. Efficient frameworks
1058 are certain to curb the computational and data requirements for generat-
1059 ing realistic floor plan images, which in turn may facilitate democratizing
1060 research in this area by enabling more companies to participate. Local train-
1061 ing of GML models will also contribute to a higher level of data control and
1062 thus data security for business and related data stakeholders in the process.
1063 Lastly, this work could lay the foundations for extending GML to other in-
1064 dustries that share driving needs and limitations for adopting GML with the
1065 architectural design industry [1, 2].

1066 4.7. Study limitations

1067 Hyperparameter tuning of GAN models is a rather time-consuming task
1068 with each adjustment requiring 10k king iterations (i.e. ~ 147 -160 hours).
1069 This has limited the ability to more finely search the hyperparameter space.
1070 However, GAN training is overly sensitive to hyperparameters, and further

1071 experiments with additional hyperparameter value combinations might have
1072 provided valuable insight into the optimal setting, especially with regard to
1073 the gamma level. The FID score is based on features obtained from the
1074 ImageNet pre-trained model. Therefore, caution should be exercised when
1075 using this metric for models such as Projected GAN which relies on Im-
1076 ageNet pre-trained representations [64]. In addition, it has been reported
1077 that there is some discrepancy between the FID metric and human assess-
1078 ment when evaluating non-ImageNet data [16]. However, this study offered a
1079 more reliable generated image quality evaluation across different GAN meth-
1080 ods by adopting a combination of three metrics (namely FID, density, CNN
1081 detection score) as well as human inspection. Even though downscaling the
1082 dataset from 1024×1024 to 256×256 pixels has sped up the training process,
1083 a degradation of the detail level of the floor plan images is inevitable. The
1084 up-sampling process can reconstruct some deficiencies, but this degradation
1085 remains a limitation worth investigating further. The validation framework
1086 of this study did not include criteria related to room modularity and natu-
1087 ral ventilation. These were nevertheless deemed too advanced for the early
1088 design phase that the proposed pipeline aims to support.

1089 4.8. Future work

1090 While StyleGAN is a promising network architecture for generating im-
1091 age content, it is worth testing and comparing additional GAN models to
1092 determine the optimal generator. Licensing aspects might also play a role in
1093 the future selection of models. Applying image-to-image style-transfer by us-
1094 ing models such as pix2pixHD [103], pix2pix [104], or CycleGAN [105] could
1095 enhance the appearance of the generated floor plans and add more details.

1096 Converting raster images to vector formats, suitable for BIM environments,
1097 is likely to improve the applicability of generated floor plans, as well as the
1098 demand to convert existing building layouts into BIM [25, 106]. Adding text-
1099 based conditions to the proposed framework may offer more control over the
1100 generated outputs [107]. Methods for text-to-image generation have been
1101 proposed in the literature [15, 107, 108], and incorporating these methods
1102 may further improve output specification. In the Danish architectural con-
1103 text, three main design paradigms influence floor plan layouts from 1950 to
1104 today. The style-based floor plan generator of the proposed framework should
1105 be able to reflect the above grouping. It may have created a hidden hierar-
1106 chy in the style space that represents these three design paradigms, offering
1107 potential for renovation and reconstruction projects through methods such
1108 as style transfer. This task remains to be elucidated. GAN-based pix2pix
1109 methods can replace costly computational fluid dynamics (CFD) analysis
1110 for layout performance as surrogate models with image-to-image translation
1111 to predict contour plots of layout performance to support designers in the
1112 early design stage both for 2D and spatial analysis [108]. Furthermore, re-
1113 cent studies have shown promising results using ControlNet-based diffusion
1114 model structures, which are capable of generating fast and highly detailed
1115 conceptual designs guided by control masks [108]. The modular structure
1116 of the framework offers flexibility in updating a specific module as future
1117 technological advances come to the fore.

1118 The model’s ability to autonomously generate highly realistic floor plans,
1119 combined with the control demonstrated by cGAN-based generation, presents
1120 a key future challenge in improving the usability of such systems for architec-

1121 tural purposes. A critical aspect of this challenge is enabling the generated
1122 floor plans to adapt to specific building plot conditions and meet relevant
1123 constraints such as zoning codes and national building regulations.

1124 5. Conclusion

1125 This study aimed at increasing the applicability of generative machine
1126 learning (GML) to the automation of early stage building engineering work-
1127 flows. It made notable strides in addressing the following key challenges on
1128 employing generative adversarial networks (GANs) towards this application:
1129 Substantial computational and data demands, training instability, and chal-
1130 lenges in evaluating results. A stable, multi-module GAN-stack was devel-
1131 oped for autonomously generating high-fidelity, contextually meaningful floor
1132 plan layouts, while operating on limited computational and data resources.
1133 The pipeline consisted of a pre-processing module comprising transformer-
1134 based text remover and AREA down-sampling, a floor plan image generation
1135 module based on the projected GAN with FastGAN generator network, and
1136 an image up-sampling module utilizing the Real-ESRGAN architecture. It
1137 was efficiently trained on real-world construction data. Innovative frame-
1138 works were also introduced for the holistic evaluation of the generated floor
1139 plans and the contextual validation of the proposed pipeline. The comprehen-
1140 sive evaluation results demonstrated that the proposed pipeline can expedite
1141 and stabilize the GML-based autonomous generation of 1024×1024 floor plan
1142 layouts while relying on minimal compute and data resources. The novel val-
1143 idation framework confirmed that the proposed computer-aided pipeline has
1144 contextual relevance to architectural design, ensuring adherence to architec-

1145 tural business rules, attributes, and relationships. The presented work could
1146 streamline the iterative prototyping of conceptual floor plan layouts in the
1147 early design phase, supporting the communication between architects (end
1148 users) and clients. The proposed framework has the capability to generate
1149 unique high-resolution (HR) contextually meaningful floor plan layouts in
1150 a total sampling time of 0.17 seconds. This research also has potential for
1151 broadening access to the GML-based automation in the building engineering
1152 research area. It may provide the foundation for extending GML applications
1153 to other industries with similar needs and resource constraints.

1154 **Acknowledgements**

1155 The authors would like to thank Home A/S for supporting the data col-
1156 lection process. Special thanks to Arkitec A/S, particularly the architect Ole
1157 Damgaard for inspiring discussions about perspectives on the professional use
1158 of AI-based design generation. We are also grateful to Professor Cand.arch.,
1159 PhD - Thomas Bo Jensen and Cand.arch., PhD fellow - Mette Boisen Lyhne
1160 for their valuable input on the design subject.

1161 **References**

- 1162 [1] R. C. Rodrigues, R. B. Duarte, Generating floor plans with deep
1163 learning: A cross-validation assessment over different dataset sizes,
1164 International Journal of Architectural Computing 20 (3) (2022)
1165 630–644. [doi:10.1177/14780771221120842](https://doi.org/10.1177/14780771221120842).
- 1166 [2] K. Park, S. Ergan, C. Feng, Quality assessment of residential layout
1167 designs generated by relational Generative Adversarial Networks

- (GANs), *Automation in Construction* 158 (2024).
[doi:10.1016/j.autcon.2023.105243](https://doi.org/10.1016/j.autcon.2023.105243).
- [3] Z. Luo, W. Huang, FloorplanGAN: Vector residential floorplan adversarial generation, *Automation in Construction* 142 (2022) 104470. [doi:10.1016/j.autcon.2022.104470](https://doi.org/10.1016/j.autcon.2022.104470).
- [4] J. McCormack, P. Hutchings, T. Gifford, M. Yee-King, M. T. Llano, M. D’Inverno, Design Considerations for Real-Time Collaboration with Creative Artificial Intelligence, *Organised Sound* 25 (1) (2020) 41–52. [doi:10.1017/S1355771819000451](https://doi.org/10.1017/S1355771819000451).
- [5] V. Paananen, J. Oppenlaender, A. Visuri, Using text-to-image generation for architectural design ideation, *International Journal of Architectural Computing* (2023). [doi:10.1177/14780771231222783](https://doi.org/10.1177/14780771231222783).
- [6] Z. Zhang, J. M. Fort, L. Giménez Mateu, Exploring the Potential of Artificial Intelligence as a Tool for Architectural Design: A Perception Study Using Gaudí’s Works, *Buildings* 13 (7) (2023) 1863. [doi:10.3390/buildings13071863](https://doi.org/10.3390/buildings13071863).
- [7] S. Zheng, StyleGAN-Canvas: Augmenting StyleGAN3 for Real-Time Human-AI Co-Creation, in: *CEUR Workshop Proceedings*, Vol. 3359, 2023, pp. 108–120, <https://ceur-ws.org/Vol-3359/paper12.pdf>. (Accessed July 31, 2025).
- [8] M. L. Castro Pena, A. Carballal, N. Rodríguez-Fernández, I. Santos, J. Romero, Artificial intelligence applied to conceptual design. A

- 1190 review of its use in architecture, *Automation in Construction* 124 (apr
1191 2021). [doi:10.1016/j.autcon.2021.103550](https://doi.org/10.1016/j.autcon.2021.103550).
- 1192 [9] P. Pouliou, A.-S. Horvath, G. Palamas, Speculative hybrids:
1193 Investigating the generation of conceptual architectural forms through
1194 the use of 3D generative adversarial networks, *International Journal*
1195 *of Architectural Computing* 21 (2) (2023) 315–336.
1196 [doi:10.1177/14780771231168229](https://doi.org/10.1177/14780771231168229).
- 1197 [10] J. Huang, M. Johanes, F. C. Kim, C. Doumptioti, G. C. Holz, On
1198 GANs, NLP and Architecture: Combining Human and Machine
1199 Intelligences for the Generation and Evaluation of Meaningful
1200 Designs, *Technology Architecture and Design* 5 (2) (2021) 207–224.
1201 [doi:10.1080/24751448.2021.1967060](https://doi.org/10.1080/24751448.2021.1967060).
- 1202 [11] H. Song, J. Ghaboussi, T.-H. Kwon, Architectural design of
1203 apartment buildings using the Implicit Redundant Representation
1204 Genetic Algorithm, *Automation in Construction* 72 (2016) 166–173.
1205 [doi:10.1016/j.autcon.2016.09.001](https://doi.org/10.1016/j.autcon.2016.09.001).
- 1206 [12] T. Karras, S. Laine, T. Aila, A style-based generator architecture for
1207 generative adversarial networks, in: 2019 IEEE/CVF Conference on
1208 Computer Vision and Pattern Recognition (CVPR), 2019, pp.
1209 4396–4405. [doi:10.1109/CVPR.2019.00453](https://doi.org/10.1109/CVPR.2019.00453).
- 1210 [13] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen,
1211 T. Aila, Alias-Free Generative Adversarial Networks, in: *Advances in*
1212 *Neural Information Processing Systems*, Vol. 34, 2021, pp. 852–863,

- 1213 [https://proceedings.neurips.cc/paper_files/paper/2021/](https://proceedings.neurips.cc/paper_files/paper/2021/file/076ccd93ad68be51f23707988e934906-Paper.pdf)
1214 [file/076ccd93ad68be51f23707988e934906-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/076ccd93ad68be51f23707988e934906-Paper.pdf). (Accessed
1215 July 31, 2025).
- 1216 [14] A. Sauer, K. Schwarz, A. Geiger, Stylegan-xl: Scaling stylegan to
1217 large diverse datasets, in: ACM SIGGRAPH 2022 conference
1218 proceedings, Association for Computing Machinery, 2022, pp. 1–10.
1219 [doi:10.1145/3528233.3530738](https://doi.org/10.1145/3528233.3530738).
- 1220 [15] A. Sauer, T. Karras, S. Laine, A. Geiger, T. Aila, StyleGAN-T:
1221 Unlocking the Power of GANs for Fast Large-Scale Text-to-Image
1222 Synthesis, in: Proceedings of the 40th International Conference on
1223 Machine Learning, Vol. 202 of Proceedings of Machine Learning
1224 Research, 2023, pp. 30105–30118,
1225 <https://proceedings.mlr.press/v202/sauer23a/sauer23a.pdf>.
1226 (Accessed July 31, 2025).
- 1227 [16] T. Kynkäänniemi, T. Karras, M. Aittala, T. Aila, J. Lehtinen, The
1228 Role of ImageNet Classes in Fréchet Inception Distance (2023).
1229 [doi:10.48550/arXiv.2203.06026](https://doi.org/10.48550/arXiv.2203.06026).
- 1230 [17] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, T. Aila,
1231 Training Generative Adversarial Networks with Limited Data, in:
1232 Advances in Neural Information Processing Systems, Vol. 33, 2020,
1233 pp. 12104–12114,
1234 [https://proceedings.neurips.cc/paper_files/paper/2020/](https://proceedings.neurips.cc/paper_files/paper/2020/file/8d30aa96e72440759f74bd2306c1fa3d-Paper.pdf)
1235 [file/8d30aa96e72440759f74bd2306c1fa3d-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/8d30aa96e72440759f74bd2306c1fa3d-Paper.pdf). (Accessed
1236 July 31, 2025).

- [18] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever, Learning Transferable Visual Models From Natural Language Supervision, in: Proceedings of the 38th International Conference on Machine Learning, Vol. 139 of Proceedings of Machine Learning Research, 2021, pp. 8748–8763, <http://proceedings.mlr.press/v139/radford21a/radford21a.pdf>. (Accessed July 31, 2025).
- [19] B. Liu, Y. Zhu, K. Song, A. Elgammal, Towards Faster and Stabilized GAN Training for High-fidelity Few-shot Image Synthesis (jan 2021).
- [20] A. Alanov, V. Titov, M. Nakhodnov, D. Vetrov, StyleDomain: Efficient and Lightweight Parameterizations of StyleGAN for One-shot and Few-shot Domain Adaptation, in: 2023 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE, 2023, pp. 2184–2194. [doi:10.1109/ICCV51070.2023.00208](https://doi.org/10.1109/ICCV51070.2023.00208).
- [21] A. Sauer, K. Chitta, J. Müller, A. Geiger, Projected GANs Converge Faster, in: Proceedings of the 35th International Conference on Neural Information Processing Systems, Vol. 34, Curran Associates, Inc., 2021, pp. 17480–17492, <https://proceedings.neurips.cc/paper/2021/file/9219adc5c42107c4911e249155320648-Paper.pdf>. (Accessed July 31, 2025).
- [22] P. Chai, L. Hou, G. Zhang, Q. Tushar, Y. Zou, Generative adversarial

- 1260 networks in construction applications, *Automation in Construction*
1261 159 (2024) 105265. [doi:10.1016/j.autcon.2024.105265](https://doi.org/10.1016/j.autcon.2024.105265).
- 1262 [23] E. Betzalel, C. Penso, A. Navon, E. Fetaya, A Study on the
1263 Evaluation of Generative Models (2022).
1264 [doi:10.48550/arXiv.2206.10935](https://doi.org/10.48550/arXiv.2206.10935).
- 1265 [24] M. F. Naeem, S. J. Oh, Y. Uh, Y. Choi, J. Yoo, Reliable fidelity and
1266 diversity metrics for generative models, in: *Proceedings of the 37th*
1267 *International Conference on Machine Learning*, Vol. 119, 2020, pp.
1268 7176–7185,
1269 <http://proceedings.mlr.press/v119/naeem20a/naeem20a.pdf>.
1270 (Accessed July 31, 2025).
- 1271 [25] J. Parente, E. Rodrigues, B. Rangel, J. Poças Martins, Integration of
1272 convolutional and adversarial networks into building design: A
1273 review, *Journal of Building Engineering* 76 (2023) 107155.
1274 [doi:10.1016/j.jobe.2023.107155](https://doi.org/10.1016/j.jobe.2023.107155).
- 1275 [26] S. Kookalani, E. Parn, I. Brilakis, S. Dirar, M. Theofanous,
1276 A. Faramarzi, M. A. Mahdavi-pour, Q. Feng, Trajectory of building
1277 and structural design automation from generative design towards the
1278 integration of deep generative models and optimization: A review,
1279 *Journal of Building Engineering* 97 (2024) 110972.
1280 [doi:10.1016/j.jobe.2024.110972](https://doi.org/10.1016/j.jobe.2024.110972).
- 1281 [27] Q. Feng, C. Guo, F. Benitez-Quiroz, A. Martinez, When do GANs
1282 replicate? On the choice of dataset size, in: *2021 IEEE/CVF*

- 1283 International Conference on Computer Vision (ICCV), IEEE, 2021,
1284 pp. 6681–6690. [doi:10.1109/ICCV48922.2021.00663](https://doi.org/10.1109/ICCV48922.2021.00663).
- 1285 [28] D. Newton, Deep Generative Learning for the Generation and
1286 Analysis of Architectural Plans with Small Datasets, in: Proceedings
1287 of the International Conference on Education and Research in
1288 Computer Aided Architectural Design in Europe, Vol. 2, 37th
1289 Conference on Education and Research in Computer Aided
1290 Architectural Design in Europe, 2019, pp. 21–28.
1291 [doi:10.5151/proceedings-ecaadesigradi2019_135](https://doi.org/10.5151/proceedings-ecaadesigradi2019_135).
- 1292 [29] R. E. Weber, C. Mueller, C. Reinhart, Automated floorplan
1293 generation in architectural design: A review of methods and
1294 applications, Automation in Construction 140 (2022) 13.
1295 [doi:10.1016/j.autcon.2022.104385](https://doi.org/10.1016/j.autcon.2022.104385).
- 1296 [30] L. Wang, J. Liu, Y. Zeng, G. Cheng, H. Hu, J. Hu, X. Huang,
1297 Automated building layout generation using deep learning and graph
1298 algorithms, Automation in Construction 154 (April) (2023) 105036.
1299 [doi:10.1016/j.autcon.2023.105036](https://doi.org/10.1016/j.autcon.2023.105036).
- 1300 [31] M. Aalaei, M. Saadi, M. Rahbar, A. Ekhlassi, Architectural layout
1301 generation using a graph-constrained conditional Generative
1302 Adversarial Network (GAN), Automation in Construction
1303 155 (November) (2023) 105053.
1304 [doi:10.1016/j.autcon.2023.105053](https://doi.org/10.1016/j.autcon.2023.105053).
- 1305 [32] J. Sun, W. Wu, L. Liu, W. Min, G. Zhang, L. Zheng, WallPlan:

- 1306 Synthesizing Floorplans by Learning to Generate Wall Graphs, ACM
1307 Transactions on Graphics 41 (4) (2022) 1–14.
1308 [doi:10.1145/3528223.3530135](https://doi.org/10.1145/3528223.3530135).
- 1309 [33] M. Rahbar, M. Mahdavinejad, M. Bemanian, A. Davaie Markazi,
1310 L. Hovestadt, Generating Synthetic Space Allocation Probability
1311 Layouts Based on Trained Conditional-GANs, Applied Artificial
1312 Intelligence 33 (8) (2019) 689–705.
1313 [doi:10.1080/08839514.2019.1592919](https://doi.org/10.1080/08839514.2019.1592919).
- 1314 [34] C.-W. Zhao, J. Yang, J. Li, Generation of hospital emergency
1315 department layouts based on generative adversarial networks, Journal
1316 of Building Engineering 43 (2021) 102539.
1317 [doi:10.1016/j.jobbe.2021.102539](https://doi.org/10.1016/j.jobbe.2021.102539).
- 1318 [35] H. Tanasra, T. Rott Shaham, T. Michaeli, G. Austern, S. Barath,
1319 Automation in Interior Space Planning: Utilizing Conditional
1320 Generative Adversarial Network Models to Create Furniture Layouts,
1321 Buildings 13 (7) (2023). [doi:10.3390/buildings13071793](https://doi.org/10.3390/buildings13071793).
- 1322 [36] I. Karadag, O. Z. Güzelci, S. Alaçam, EDU-AI: a twofold machine
1323 learning model to support classroom layout generation, Construction
1324 Innovation 23 (4) (2022) 898–914. [doi:10.1108/CI-02-2022-0034](https://doi.org/10.1108/CI-02-2022-0034).
- 1325 [37] M. Rahbar, M. Mahdavinejad, A. H. Markazi, M. Bemanian,
1326 Architectural layout design through deep learning and agent-based
1327 modeling: A hybrid approach, Journal of Building Engineering 47
1328 (2022). [doi:10.1016/j.jobbe.2021.103822](https://doi.org/10.1016/j.jobbe.2021.103822).

- 1329 [38] F. Jiang, J. Ma, C. J. Webster, X. Li, V. J. Gan, Building layout
1330 generation using site-embedded GAN model, Automation in
1331 Construction 151 (2023). [doi:10.1016/j.autcon.2023.104888](https://doi.org/10.1016/j.autcon.2023.104888).
- 1332 [39] S. Wang, W. Zeng, X. Chen, Y. Ye, Y. Qiao, C.-W. Fu,
1333 ActFloor-GAN: Activity-Guided Adversarial Networks for
1334 Human-Centric Floorplan Design, IEEE Transactions on
1335 Visualization and Computer Graphics 29 (3) (2023) 1610–1624.
1336 [doi:10.1109/TVCG.2021.3126478](https://doi.org/10.1109/TVCG.2021.3126478).
- 1337 [40] C. Zhao, J. Yang, W. Xiong, J. Li, Two Generative Design Methods
1338 of Hospital Operating Department Layouts Based on Healthcare
1339 Systematic Layout Planning and Generative Adversarial Network,
1340 Journal of Shanghai Jiaotong University (Science) 26 (1) (2021)
1341 103–115. [doi:10.1007/s12204-021-2265-9](https://doi.org/10.1007/s12204-021-2265-9).
- 1342 [41] P. Ghannad, Y.-C. Lee, Automated modular housing design using a
1343 module configuration algorithm and a coupled generative adversarial
1344 network (CoGAN), Automation in Construction 139 (2022).
1345 [doi:10.1016/j.autcon.2022.104234](https://doi.org/10.1016/j.autcon.2022.104234).
- 1346 [42] J. Shim, J. Moon, H. Kim, E. Hwang, FloorDiffusion: Diffusion
1347 model-based conditional floorplan image generation method using
1348 parameter-efficient fine-tuning and image inpainting, Journal of
1349 Building Engineering 95 (2024) 110320.
1350 [doi:10.1016/j.jobe.2024.110320](https://doi.org/10.1016/j.jobe.2024.110320).
- 1351 [43] H. Leng, Y. Gao, Y. Zhou, ArchiDiffusion: A novel diffusion model

- 1352 connecting architectural layout generation from sketches to Shear
 1353 Wall Design, *Journal of Building Engineering* 98 (2024) 111373.
 1354 [doi:10.1016/j.jobe.2024.111373](https://doi.org/10.1016/j.jobe.2024.111373).
- 1355 [44] S. Chaillou, ArchiGAN: Artificial Intelligence x Architecture, in:
 1356 P. F. Yuan, M. Xie, N. Leach, J. Yao, X. Wang (Eds.), *Architectural*
 1357 *Intelligence*, Springer Nature Singapore, 2020, pp. 117–127.
 1358 [doi:10.1007/978-981-15-6568-7](https://doi.org/10.1007/978-981-15-6568-7).
- 1359 [45] Z. Du, H. Shen, X. Li, M. Wang, 3D building fabrication with
 1360 geometry and texture coordination via hybrid GAN, *Journal of*
 1361 *Ambient Intelligence and Humanized Computing* 13 (11) (2022)
 1362 5177–5188. [doi:10.1007/s12652-020-02488-9](https://doi.org/10.1007/s12652-020-02488-9).
- 1363 [46] G. Rossi, M. Fontani, S. Milani, Neural network for denoising and
 1364 reading degraded license plates, in: *Pattern Recognition. ICPR*
 1365 *International Workshops and Challenges*, Vol. 12666, Springer
 1366 International Publishing, 2021, pp. 484–499.
 1367 [doi:10.1007/978-3-030-68780-9_39](https://doi.org/10.1007/978-3-030-68780-9_39).
- 1368 [47] M. Lystbæk, A. Giannakidis, M. J. Beliatas, M. Olsen, Removing
 1369 Unwanted Text from Architectural Images with Multi-Scale
 1370 Deformable Attention-Based Machine Learning, in: *2023 IEEE*
 1371 *International Conference on Imaging Systems and Techniques (IST)*,
 1372 IEEE, 2023, pp. 1–5. [doi:10.1109/IST59124.2023.10355658](https://doi.org/10.1109/IST59124.2023.10355658).
- 1373 [48] X. Wang, L. Xie, C. Dong, Y. Shan, Real-ESRGAN: Training
 1374 Real-World Blind Super-Resolution with Pure Synthetic Data, in:

- 1375 Proceedings of the IEEE International Conference on Computer
 1376 Vision, Vol. 2021-Octob, 2021, pp. 1905–1914.
 1377 [doi:10.1109/ICCVW54120.2021.00217](https://doi.org/10.1109/ICCVW54120.2021.00217).
- 1378 [49] R. Schwartz, J. Dodge, N. A. Smith, O. Etzioni, Green AI,
 1379 Communications of the ACM 63 (12) (2020) 54–63.
 1380 [doi:10.1145/3381831](https://doi.org/10.1145/3381831).
- 1381 [50] K. Tero, A. Miika, L. Samuli, H. Erik, H. Janne, L. Jaakko, A. Timo,
 1382 StyleGAN3 - Training configurations,
 1383 [https://github.com/NVlabs/stylegan3/blob/main/docs/](https://github.com/NVlabs/stylegan3/blob/main/docs/configs.md#recommended-configurations)
 1384 [configs.md#recommended-configurations](https://github.com/NVlabs/stylegan3/blob/main/docs/configs.md#recommended-configurations). (Accessed July 31,
 1385 2025) (2021).
- 1386 [51] E. C. Ifeachor, B. W. Jervis, Digital Signal Processing: A Practical
 1387 Approach, 2nd Edition, Prentice Hall, Harlow, England, 2002,
 1388 <https://books.google.com/books?id=slHJI09xmcEC&pgis=1>.
 1389 (Accessed July 31, 2025).
- 1390 [52] Home A/S, Home.dk, <https://home.dk>. (Accessed July 31, 2025).
- 1391 [53] Open Source Computer Vision (OpenCV), Geometric Image
 1392 Transformations - Image Processing, [https://docs.opencv.org/3.](https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html)
 1393 [4/da/d54/group__imgproc__transform.html](https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html). (Accessed July 31,
 1394 2025) (2024).
- 1395 [54] X. Zhang, Y. Su, S. Tripathi, Z. Tu, Text Spotting Transformers, in:
 1396 2022 IEEE/CVF Conference on Computer Vision and Pattern

- 1397 Recognition (CVPR), 2022, pp. 9509–9518.
1398 [doi:10.1109/CVPR52688.2022.00930](https://doi.org/10.1109/CVPR52688.2022.00930).
- 1399 [55] K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image
1400 Recognition, in: 2016 IEEE Conference on Computer Vision and
1401 Pattern Recognition (CVPR), IEEE, 2016, pp. 770–778.
1402 [doi:10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- 1403 [56] I. J. Goodfellow, J. Pouget-abadie, M. Mirza, B. Xu, D. Warde-farley,
1404 S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Nets, in:
1405 Advances in Neural Information Processing Systems 27 (NIPS),
1406 NeurIPS, 2014, pp. 1–9, [https://proceedings.neurips.cc/paper/](https://proceedings.neurips.cc/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf)
1407 [2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf](https://proceedings.neurips.cc/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf).
1408 (Accessed July 31, 2025).
- 1409 [57] Z. Wu, D. Lischinski, E. Shechtman, Stylespace analysis:
1410 Disentangled controls for stylegan image generation, in: Proceedings
1411 of the IEEE Computer Society Conference on Computer Vision and
1412 Pattern Recognition, CVPR, 2021, pp. 12858–12867.
1413 [doi:10.1109/CVPR46437.2021.01267](https://doi.org/10.1109/CVPR46437.2021.01267).
- 1414 [58] S. Park, How to edit images with GANs? Controlling the Latent
1415 Space of GANs, <https://medium.com/codex/afde630e53d1>.
1416 (Accessed July 31, 2025) (2021).
- 1417 [59] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, D. Lischinski,
1418 StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery, in:
1419 Proceedings of the IEEE International Conference on Computer

- 1420 Vision, ICCV, 2021, pp. 2065–2074.
1421 [doi:10.1109/ICCV48922.2021.00209](https://doi.org/10.1109/ICCV48922.2021.00209).
- 1422 [60] T. Oanda, Exploring and Exploiting the Latent Style Space,
1423 [https://blog.paperspace.com/
1424 exploring-and-exploiting-the-latent-style-space/](https://blog.paperspace.com/exploring-and-exploiting-the-latent-style-space/). (Accessed
1425 July 31, 2025) (2022).
- 1426 [61] M. K. Venturelli, P. H. Gomes, J. Wehrmann, Looks Like Magic:
1427 Transfer Learning in GANs to Generate New Card Illustrations, in:
1428 Proceedings of the International Joint Conference on Neural
1429 Networks, Vol. 2022-July, IJCNN, 2022, pp. 1–8.
1430 [doi:10.1109/IJCNN55064.2022.9892463](https://doi.org/10.1109/IJCNN55064.2022.9892463).
- 1431 [62] Justin Pinkney, StyleGAN3-t LHQ 256 dataset,
1432 <https://huggingface.co/justinpinkney/stylegan3-t-lhq-256>.
1433 (Accessed July 31, 2025) (2023).
- 1434 [63] M. Tan, Q. V. Le, EfficientNet: Rethinking Model Scaling for
1435 Convolutional Neural Networks, in: Proceedings of the 36th
1436 International Conference on Machine Learning, Vol. 97 of Proceedings
1437 of Machine Learning Research, 2019, pp. 6105–6114,
1438 <http://proceedings.mlr.press/v97/tan19a/tan19a.pdf>.
1439 (Accessed July 31, 2025).
- 1440 [64] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, Li Fei-Fei, ImageNet:
1441 A large-scale hierarchical image database, in: 2009 IEEE Conference

- 1442 on Computer Vision and Pattern Recognition, Vol. 20, IEEE, 2009,
1443 pp. 248–255. [doi:10.1109/CVPR.2009.5206848](https://doi.org/10.1109/CVPR.2009.5206848).
- 1444 [65] S. Zhao, Z. Liu, J. Lin, J.-Y. Zhu, S. Han, Differentiable
1445 Augmentation for Data-Efficient GAN Training, in: Proceedings of
1446 the 34th International Conference on Neural Information Processing
1447 Systems, Vol. 33, Red Hook, NY, USA, 2020, pp. 7559–7570,
1448 [https://proceedings.neurips.cc/paper/2020/file/
1449 55479c55ebd1efd3ff125f1337100388-Paper.pdf](https://proceedings.neurips.cc/paper/2020/file/55479c55ebd1efd3ff125f1337100388-Paper.pdf). (Accessed July
1450 31, 2025).
- 1451 [66] Neyshabur, Behnam and Bhojanapalli, Srinadh and Chakrabarti,
1452 Ayan, Stabilizing GAN training with multiple random projections,
1453 arXiv preprint arXiv:1705.07831 (2017).
1454 [doi:10.48550/arXiv.1705.07831](https://doi.org/10.48550/arXiv.1705.07831).
- 1455 [67] C. Ledig, L. Theis, F. Husz, J. Caballero, A. Cunningham, A. Acosta,
1456 A. Aitken, A. Tejani, J. Totz, Z. Wang, W. Shi, Photo-Realistic Single
1457 Image Super-Resolution Using a Generative Adversarial Network, in:
1458 IEEE Conference on Computer Vision and Pattern Recognition
1459 (CVPR), 2017, pp. 4681–4690. [doi:10.1109/CVPR.2017.19](https://doi.org/10.1109/CVPR.2017.19).
- 1460 [68] J. Guerreiro, P. Tomás, N. Garcia, H. Aidos, Super-resolution of
1461 magnetic resonance images using Generative Adversarial Networks,
1462 in: Computerized Medical Imaging and Graphics, Vol. 108, Elsevier
1463 Ltd, 2023, p. 102280. [doi:10.1016/j.compmedimag.2023.102280](https://doi.org/10.1016/j.compmedimag.2023.102280).
- 1464 [69] K. Simonyan, A. Zisserman, Very deep convolutional networks for

- 1465 large-scale image recognition, in: 3rd International Conference on
 1466 Learning Representations, ICLR 2015 - Conference Track
 1467 Proceedings, 2015, pp. 1–14. [doi:10.48550/arXiv.1409.1556](https://doi.org/10.48550/arXiv.1409.1556).
- 1468 [70] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, C. C. Loy,
 1469 ESRGAN: Enhanced super-resolution generative adversarial
 1470 networks, in: Proceedings of the European Conference on Computer
 1471 Vision (ECCV) Workshops, Vol. 11133 LNCS, 2019, pp. 63–79.
 1472 [doi:10.1007/978-3-030-11021-5_5](https://doi.org/10.1007/978-3-030-11021-5_5).
- 1473 [71] A. Jolicoeur-Martineau, The relativistic discriminator: a key element
 1474 missing from standard GAN (2018).
 1475 [doi:10.48550/arXiv.1807.00734](https://doi.org/10.48550/arXiv.1807.00734).
- 1476 [72] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, S. Hochreiter,
 1477 GANs Trained by a Two Time-Scale Update Rule Converge to a
 1478 Local Nash Equilibrium, [https://proceedings.neurips.cc/paper/](https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf)
 1479 [2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf](https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf).
 1480 (Accessed July 31, 2025) (2017).
- 1481 [73] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking
 1482 the Inception Architecture for Computer Vision, Proceedings of the
 1483 IEEE Computer Society Conference on Computer Vision and Pattern
 1484 Recognition 2016-Decem (2016) 2818–2826.
 1485 [doi:10.1109/CVPR.2016.308](https://doi.org/10.1109/CVPR.2016.308).
- 1486 [74] M. S. M. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, S. Gelly,
 1487 Assessing Generative Models via Precision and Recall, in: Advances

- 1488 in Neural Information Processing Systems, Vol. 31, 2018, pp.
 1489 5228–5237, [https://proceedings.neurips.cc/paper/2018/file/
 1490 f7696a9b362ac5a51c3dc8f098b73923-Paper.pdf](https://proceedings.neurips.cc/paper/2018/file/f7696a9b362ac5a51c3dc8f098b73923-Paper.pdf). (Accessed July
 1491 31, 2025).
- 1492 [75] L. Simon, R. Webster, J. Rabin, Revisiting Precision and Recall
 1493 Definition for Generative Model Evaluation, in: 36th International
 1494 Conference on Machine Learning, Vol. June, 2019, pp. 10174–10183,
 1495 <http://proceedings.mlr.press/v97/simon19a/simon19a.pdf>.
 1496 (Accessed July 31, 2025).
- 1497 [76] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, T. Aila, Improved
 1498 precision and recall metric for assessing generative models, in:
 1499 Proceedings of the 33rd International Conference on Neural
 1500 Information Processing Systems, Vol. 32, 2019, pp. 1–10,
 1501 [https://proceedings.neurips.cc/paper/2019/file/
 1502 0234c510bc6d908b28c70ff313743079-Paper.pdf](https://proceedings.neurips.cc/paper/2019/file/0234c510bc6d908b28c70ff313743079-Paper.pdf). (Accessed July
 1503 31, 2025).
- 1504 [77] L. Nataraj, T. M. Mohammed, B. S. Manjunath, S. Chandrasekaran,
 1505 A. Flenner, J. H. Bappy, A. K. Roy-Chowdhury, Detecting GAN
 1506 generated Fake Images using Co-occurrence Matrices (2019).
 1507 [doi:10.2352/ISSN.2470-1173.2019.5.MWSF-532](https://doi.org/10.2352/ISSN.2470-1173.2019.5.MWSF-532).
- 1508 [78] M. Barni, K. Kallas, E. Nowroozi, B. Tondi, CNN Detection of
 1509 GAN-Generated Face Images based on Cross-Band Co-occurrences
 1510 Analysis, in: 2020 IEEE International Workshop on Information

- 1511 Forensics and Security, WIFS, 2020, pp. 1–6.
 1512 [doi:10.1109/WIFS49906.2020.9360905](https://doi.org/10.1109/WIFS49906.2020.9360905).
- 1513 [79] S. Raj, J. Mathew, A. Mondal, Generalized and robust model for
 1514 GAN-generated image detection, Pattern Recognition Letters
 1515 182 (April) (2024) 104–110. [doi:10.1016/j.patrec.2024.04.018](https://doi.org/10.1016/j.patrec.2024.04.018).
- 1516 [80] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov,
 1517 D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with
 1518 convolutions, in: Proceedings of the IEEE Computer Society
 1519 Conference on Computer Vision and Pattern Recognition, 2015, pp.
 1520 1–9. [doi:10.1109/CVPR.2015.7298594](https://doi.org/10.1109/CVPR.2015.7298594).
- 1521 [81] Carbontracker: Tracking and Predicting the Carbon Footprint of
 1522 Training Deep Learning Models, iCML Workshop on Challenges in
 1523 Deploying and monitoring Machine Learning Systems (2020).
 1524 [doi:10.48550/arXiv.2007.03051](https://doi.org/10.48550/arXiv.2007.03051).
- 1525 [82] T. A. Bohoran, K. S. Parke, M. P. Graham-Brown, M. Meisuria,
 1526 A. Singh, J. Wormleighton, D. Adlam, D. Gopalan, M. J. Davies,
 1527 B. Williams, et al., Resource efficient aortic distensibility calculation
 1528 by end to end spatiotemporal learning of aortic lumen from
 1529 multicentre multivendor multidisease CMR images, Scientific Reports
 1530 13 (1) (2023) 21794. [doi:10.1038/s41598-023-48986-6](https://doi.org/10.1038/s41598-023-48986-6).
- 1531 [83] A. Siouras, S. Moustakidis, G. Chalatsis, T. A. Bohoran, M. Hantes,
 1532 M. Vlychou, S. Tasoulis, A. Giannakidis, D. Tsaopoulos, Economical
 1533 hybrid novelty detection leveraging global aleatoric semantic

- 1534 uncertainty for enhanced MRI-based ACL tear diagnosis,
 1535 Computerized Medical Imaging and Graphics 117 (2024) 102424.
 1536 [doi:10.1016/j.compmedimag.2024.102424](https://doi.org/10.1016/j.compmedimag.2024.102424).
- 1537 [84] E. de Place Hansen, Small Houses - design and function,
 1538 SBi-anvisning, Statens Byggeforskningsinstitut, Aalborg Universitet,
 1539 2018, [https://vbn.aau.dk/en/publications/](https://vbn.aau.dk/en/publications/small-houses-design-and-function)
 1540 [small-houses-design-and-function](https://vbn.aau.dk/en/publications/small-houses-design-and-function). (Accessed July 31, 2025).
- 1541 [85] Statistics Denmark, BOL106: Dwellings with registered population,
 1542 Tech. rep., Statistics Denmark, <https://www.statbank.dk/BOL106>.
 1543 (Accessed July 31, 2025).
- 1544 [86] C. Meehan, K. Chaudhuri, S. Dasgupta, A Non-Parametric Test to
 1545 Detect Data-Copying in Generative Models, in: Proceedings of the
 1546 Twenty Third International Conference on Artificial Intelligence and
 1547 Statistics, Vol. 108 of Proceedings of Machine Learning Research,
 1548 2020, pp. 3546–3556, [https:](https://proceedings.mlr.press/v108/meehan20a/meehan20a.pdf)
 1549 [//proceedings.mlr.press/v108/meehan20a/meehan20a.pdf](https://proceedings.mlr.press/v108/meehan20a/meehan20a.pdf).
 1550 (Accessed July 31, 2025).
- 1551 [87] J. Buchner, Github repo: Imagehash,
 1552 <https://github.com/JohannesBuchner/imagehash>. (Accessed July
 1553 31, 2025) (2021).
- 1554 [88] Z. Huang, S. Liu, Perceptual Image Hashing with Texture and
 1555 Invariant Vector Distance for Copy Detection, IEEE Transactions on
 1556 Multimedia 23 (2021) 1516–1529. [doi:10.1109/TMM.2020.2999188](https://doi.org/10.1109/TMM.2020.2999188).

- 1557 [89] S. Mckeown, W. J. Buchanan, Hamming Distributions of Popular
1558 Perceptual Hashing Techniques, *Forensic Science International:
1559 Digital Investigation* 44 (2023). doi:10.1016/j.fsidi.2023.301509.
- 1560 [90] Z. Huang, Z. Tang, X. Zhang, L. Ruan, X. Zhang, Perceptual Image
1561 Hashing With Locality Preserving Projection for Copy Detection,
1562 *IEEE Transactions on Dependable and Secure Computing* 20 (1)
1563 (2023) 463–477. doi:10.1109/TDSC.2021.3136163.
- 1564 [91] M. Tanaka, S. Shiota, H. Kiya, A detection method of operated
1565 fake-images using robust hashing, *Journal of Imaging* 7 (8) (2021).
1566 doi:10.3390/jimaging7080134.
- 1567 [92] Y. N. Li, P. Wang, Y. T. Su, Robust Image Hashing Based on
1568 Selective Quaternion Invariance, in: *IEEE Signal Processing Letters*,
1569 Vol. 22, IEEE, 2015, pp. 2396–2400.
1570 doi:10.1109/LSP.2015.2487824.
- 1571 [93] N. Krawetz, Looks Like It - Perceptual hash,
1572 [https://www.hackerfactor.com/blog/index.php/?archives/
1573 432-Looks-Like-It.html](https://www.hackerfactor.com/blog/index.php/?archives/432-Looks-Like-It.html). (Accessed July 31, 2025) (2011).
- 1574 [94] M. Tanaka, H. Kiya, Fake-image detection with Robust Hashing, in:
1575 *LifeTech 2021 - 2021 IEEE 3rd Global Conference on Life Sciences
1576 and Technologies*, IEEE, 2021, pp. 40–43.
1577 doi:10.1109/LifeTech52111.2021.9391842.
- 1578 [95] R. Abdal, Y. Qin, P. Wonka, Image2StyleGAN: How to embed images
1579 into the StyleGAN latent space?, *Proceedings of the IEEE*

- 1580 International Conference on Computer Vision 2019-Octob (2019)
1581 4431–4440. [doi:10.1109/ICCV.2019.00453](https://doi.org/10.1109/ICCV.2019.00453).
- 1582 [96] A. Bermano, R. Gal, Y. Alaluf, R. Mokady, Y. Nitzan, O. Tov,
1583 O. Patashnik, D. Cohen-Or, State-of-the-Art in the Architecture,
1584 Methods and Applications of StyleGAN, Computer Graphics Forum
1585 41 (2) (2022) 591–611. [doi:10.1111/cgf.14503](https://doi.org/10.1111/cgf.14503).
- 1586 [97] Open Source Computer Vision (OpenCV), Color conversions,
1587 [https://docs.opencv.org/3.4/de/d25/imgproc_color_](https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html#color_convert_rgb_gray/)
1588 [conversions.html#color_convert_rgb_gray/](https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html#color_convert_rgb_gray/). (Accessed July 31,
1589 2025) (2024).
- 1590 [98] J. Johnson, A. Alahi, L. Fei-Fei, Perceptual Losses for Real-Time
1591 Style Transfer and Super-Resolution, in: Computer Vision – ECCV
1592 2016, Springer International Publishing, 2016, pp. 694–711.
1593 [doi:10.1007/978-3-319-46475-6_43](https://doi.org/10.1007/978-3-319-46475-6_43).
- 1594 [99] Mahmood-Hussain, generative-evaluation-prdc, [https:](https://github.com/Mahmood-Hussain/generative-evaluation-prdc)
1595 [//github.com/Mahmood-Hussain/generative-evaluation-prdc](https://github.com/Mahmood-Hussain/generative-evaluation-prdc).
1596 (Accessed July 31, 2025) (2022).
- 1597 [100] L. F. W. Anthony, B. Kanding, R. Selvan, Github repo: Carbon
1598 Tracker, <https://github.com/lfwa/carbontracker>. (Accessed July
1599 31, 2025) (2023).
- 1600 [101] J. Buchner, Imagehashing, <https://pypi.org/project/ImageHash/>.
1601 (Accessed July 31, 2025) (2022).

- 1602 [102] R. E. Mitchell, Web scraping with Python: Data Extraction from the
1603 Modern Web, 3rd Edition, O'Reilly Media, Sebastopol, CA, 2024,
1604 [https://www.oreilly.com/library/view/web-scraping-with/](https://www.oreilly.com/library/view/web-scraping-with/9781098145347/)
1605 [9781098145347/](https://www.oreilly.com/library/view/web-scraping-with/9781098145347/). (Accessed July 31, 2025).
- 1606 [103] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, B. Catanzaro,
1607 High-Resolution Image Synthesis and Semantic Manipulation With
1608 Conditional GANs, in: 2018 IEEE/CVF Conference on Computer
1609 Vision and Pattern Recognition, 2018, pp. 8798–8807.
1610 [doi:10.1109/CVPR.2018.00917](https://doi.org/10.1109/CVPR.2018.00917).
- 1611 [104] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, Image-To-Image Translation
1612 With Conditional Adversarial Networks, in: 2017 IEEE Conference
1613 on Computer Vision and Pattern Recognition (CVPR), 2017, pp.
1614 5967–5976. [doi:10.1109/CVPR.2017.632](https://doi.org/10.1109/CVPR.2017.632).
- 1615 [105] J.-Y. Zhu, T. Park, P. Isola, A. A. Efros, Unpaired Image-to-Image
1616 Translation Using Cycle-Consistent Adversarial Networks, in: 2017
1617 IEEE International Conference on Computer Vision (ICCV), Vol.
1618 2017-Octob, IEEE, 2017, pp. 2242–2251.
1619 [doi:10.1109/ICCV.2017.244](https://doi.org/10.1109/ICCV.2017.244).
- 1620 [106] M. Urbietta, M. Urbietta, T. Laborde, G. Villarreal, G. Rossi,
1621 Generating BIM model from structural and architectural plans using
1622 Artificial Intelligence, Journal of Building Engineering 78 (2023)
1623 107672. [doi:10.1016/j.jobee.2023.107672](https://doi.org/10.1016/j.jobee.2023.107672).
- 1624 [107] M. S. Lystbæk, Text-to-image conditional gan-based floor plan

- 1625 generator, in: 2025 International Conference on Artificial Intelligence
1626 in Information and Communication (ICAIIC), 2025, pp. 0302–0307.
1627 [doi:10.1109/ICAIIC64266.2025.10920845](https://doi.org/10.1109/ICAIIC64266.2025.10920845).
- 1628 [108] M. S. Lystbæk, Machine learning-driven processes in architectural
1629 building design, *Automation in Construction* 178 (2025) 106379.
1630 [doi:10.1016/j.autcon.2025.106379](https://doi.org/10.1016/j.autcon.2025.106379).