Spatial Layout Generation via Generative Adversarial Networks

Yue Yang[a], Yueyue Hu[b], Hanchi Ren*[a], Yunying Wang[b], Jingjing Deng[b], and Xianghua Xie[a]

[a]Swansea University, Swansea, UK
[b]Durham University, Durham, UK

ABSTRACT

The process of architectural floor plan generation is a complex task traditionally performed by architects, requiring a deep understanding of spatial relationships, structural constraints, and aesthetic principles. In recent years, advances in computational design and Artificial Intelligence (AI) have enabled the automation of floor plan generation, significantly enhancing efficiency and creativity in the architectural workflow. In this paper, we explored the integration of traditional architectural design methods with advanced technology, focusing on the transformative role of Generative Adversarial Networks (GAN) in floor plan generation. In this work, we created a new dataset containing more than 1200 carefully processed images for the automatic generation of floor plans. These samples come from different platforms and are processed to become algorithm-friendly types. We will make the dataset public available. The algorithm we used is the pix2pix network, which is enhanced with a self-attention mechanism for better spatial understanding and spectral normalization for improved output quality. We demonstrate the versatility of the GAN model in generating complex floor plans for various architectural needs based on our dataset. It also addresses challenges such as model stability, detail refinement, and generating non-standard room shapes, offering insights for future advancements in the field.

Keywords: Deep Learning, GAN, Building Floor Plan Generation, Self-Attention, Spectral Normalization, Multi-unit Building.

1. INTRODUCTION

Architectural floor plans have a rich history that dates back to ancient civilizations. The representation of architectural design through floor plans can be traced to the Egyptians and Mesopotamians, who depicted simple layouts of buildings on materials like papyrus or clay tablets. These early floor plans mainly focused on temples,[1] palaces, and residential structures. During the Renaissance,[2] there was a significant revival of interest in classical architecture, and more detailed floor plans began to be used in architectural design.[3] The use of formal architectural drawings became more prevalent, and the representation of buildings on paper became more sophisticated.[4] However, it wasn't until the emergence of Computer-Aided Design (CAD)[5,6] that the way architects and designers worked underwent a transformative shift. With the introduction of CAD technology, hand-drawn plans gradually gave way to digital drafting methods.[7] Architects and designers could now use computer software to create precise and detailed representations of architectural plans. This shift greatly improved efficiency, accuracy, and the ability to make revisions to designs. Throughout the design process, the dynamic and malleable nature of floor plans necessitates a degree of adaptability. Unforeseen modifications to spatial boundaries may arise, dictated by shifting needs, changing regulations, or evolving aesthetic considerations. These unexpected changes necessitate extensive reassessment and, in some cases, a complete revision of the entire interior layout. Consequently, architects and designers are compelled to invest considerable time and effort into iterative sketching. This iterative and resource-intensive cycle poses a considerable challenge to designers. Not only does it demand substantial manpower and intellectual investment, but it can also result in significant time inefficiencies. Ultimately, this challenge calls for a more efficient and dynamic approach to architectural design—one that can more readily adapt to changing constraints and evolving project parameters. This is not merely a matter of improving the design process; it is about fundamentally changing how architects and designers approach their work, enabling them to create more effective and responsive designs. With the rapid development of Artificial Intelligence (AI), many methods have been proposed for generating floor plans. House-Generative Adversarial Network (GAN)[8] presents a novel approach to house layout generation using GAN.[9] The authors proposed to generate a diverse set of realistic house layouts under the constraints of a bubble diagram, which is a graph where nodes encode rooms with their types and edges encode their spatial adjacency. Another work[10] presents a novel approach to architectural design. The authors proposed a method to automate the generation of detailed interior design layouts in the early design stage, thereby reducing the workload of architects. The methodology is centered around the use of a GAN. The work[11] introduces an approach for the automatic generation of rectangular floor plans based on existing legacy floor plans with the capability of further improvement and customization.

In the rapidly evolving landscape of AI-Generated Content (AIGC) and AI, this work was conceived with a clear and ambitious vision.[12] The overarching aim was to instigate a transformative shift in the architectural design process, a domain that has long been characterized by iterative and resource-intensive cycles. Traditional design methodologies, while integral to the evolution of architectural concepts, often demand substantial manpower, intellectual investment, and time inefficiencies. Recognizing these challenges, the work sought to harness the power of innovative Deep Learning (DL) techniques. Specifically, the focus was on the creation of a dataset and development of a tool that would equip architects with efficient and creative design solutions right from the project's inception. The journey of this work unveiled the untapped potential of GAN in revolutionizing the architectural design domain. By leveraging GAN, a novel method for floor plan generation was introduced, holding the promise to redefine design toolkit. The GAN model, through its intricate algorithms and learning mechanisms, showcased the capability to transform rudimentary input images, which depicted design boundaries and facade openings, into detailed, functional, and aesthetically pleasing interior designs. We also created a new dataset containing more than 1200 carefully processed images for the automatic generation of floor plans. These samples come from different platforms and are processed to become algorithm-friendly types. We will make the dataset public available. The results, as evidenced by the model's outputs, were indicative of a future where computers could autonomously generate designs that were both functional and met design, structural, environmental and other architectural constraints.

2. RELATED WORKS

Since the inception of GAN,[9] there have been significant transformations in various research domains. In the field of image processing, there are several open-source GAN framework models such as, Conditional GANs (CGAN),[13] Deep Convolutional GANs (DCGAN),[14] Wasserstein GANs (WGAN),[15] WGAN-Gradient Penalty (WGAN-GP),[16] DiscoGAN,[17] DTN[18] and pix2pix[19]). This article major provides pix2pix. It is based on the CGAN model, where the generator adopts the U-NET architecture, and the discriminator utilizes the PatchGAN classifier. A notable feature of pix2pix is its ability to achieve pixel-level image transfer, producing highly realistic images. Additionally, due to a significant reduction in parameters, this model excels in training speed and efficiency. However, it has stringent dataset requirements, necessitating paired one-to-one datasets. The model is primarily used for style transfer between paired images, such as style conversion of maps and generation of real objects with contour diagrams.

House-GAN[8] uses a dataset of 117,587 house layouts, which they divide into five groups based on the number of rooms. For the generation of layouts in each group, they exclude samples in the same group from the training to ensure that the model cannot simply memorize layouts. After the train was finished, they evaluated House-GAN using three metrics: realism, diversity, and compatibility. Realism is measured by a user study with graduate students and professional architects, who compare generated layouts with ground truth layouts. Diversity is measured by the Fréchet Inception Distance (FID)[20] score, which quantifies the difference between the distributions of generated layouts and real layouts. Compatibility is measured by the graph edit distance, which quantifies the difference between the adjacency graphs of generated layouts and the input bubble diagrams. House-GAN outperforms competing methods and baselines in all metrics, except for compatibility against a baseline method with a small margin. This work[11] is quite comprehensive and innovative. The authors have implemented their system, termed GADG. For an input floor plan with dozens of rooms, GADG generates various alternative layouts within milliseconds. This is demonstrated in a detailed step-by-step case study, where the input floor plan file is generated using Autodesk Revit. Customization and generation of floor plans based on graph transformations have enhanced the shape grammar interpreter by allowing users to import shapes and images and to use colors, among other enhanced capabilities. The graphical user interface facilitates users in importing sources and setting rule parameters. In this enhanced interpreter, a dual graph can be generated from any input according to the adjacency relations of rooms and with transformation rules applied to the graph. If the graph is a point-to-point graph, GADG maps rooms onto the base layout according to the improved rectangular dual-finding algorithm. The interpreter is able to accept any parameters for customization and thus generate layouts to suit a wide range of design requirements. Both GADG and House-GAN have employed GAN using pix2pix for generating building floor plans. Frequency features refer to large-scale patterns or changes occurring at a slower rate, capturing overall trends, global structure, or smooth transitions in the input data. These features are common in diverse data types, such as images, audio signals, and time-series data. For instance, in image processing, low-frequency components encompass the overall brightness, gradient, or large-scale texture of an image. Their research efforts provide valuable insights into leveraging GAN in architectural design and lay the groundwork for further advancements in this domain.

3. DATASET CONSTRUCTION

In this section, we delve into the process of dataset selection. We provide a detailed explanation of constructing the dataset. The potential challenges and issues that might be encountered during this process are also thoroughly discussed.

The significance of controllable parameters in architectural design cannot be overstated. These parameters, while technical in nature, have profound implications for the aesthetic and functional outcomes of a design. For instance, the building boundaries, while primarily serving a functional purpose, also influence the visual appeal and integration of the structure within its environment. Similarly, the positioning of facade openings can dramatically alter the energy consumption of a

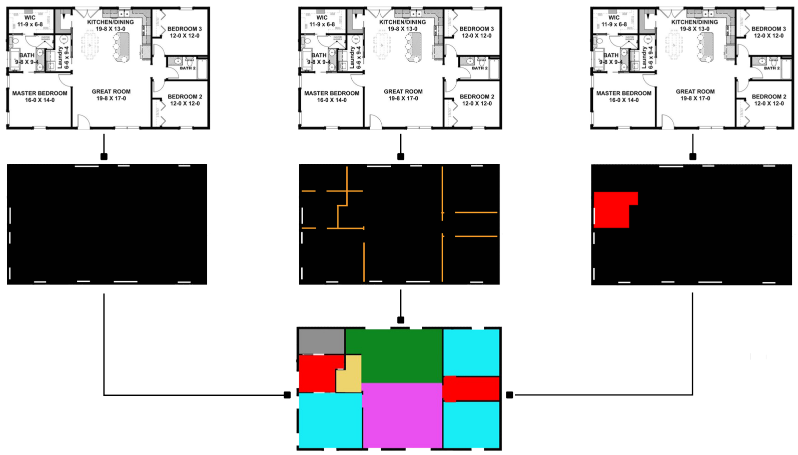

Figure 1. The left column of images represents the locations of building facade exits, the middle column indicates the positions of fixed interior walls, and the right column corresponds to specific functional area locations. In the lower right corner of the images, it is a color key for various functional zones.

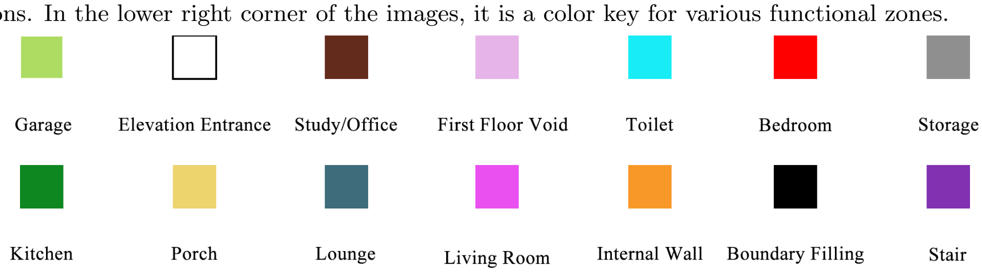

Figure 2. A legend showing the different functional zoning and color comparisons of building plans

building, impacting its sustainability and environmental footprint. The controllable parameters in this work are segmented into four critical aspects, each playing a vital role in the architectural design, as shown in Figure 2. The intricate nature of these parameters necessitated a comprehensive search for suitable existing datasets. However, the specialized requirements rendered existing datasets inadequate, leading to the need for a custom solution. These controllable parameters include:

- **Building Boundaries** articulate the overall geometry and footprint of the structure. The boundaries must be carefully considered to align with zoning regulations, aesthetic considerations, and functional requirements, forming the foundational architectural silhouette.
- **Facade Openings** are pivotal in regulating natural lighting and ventilation. The positioning and size of windows, doors, and other openings must be optimized to contribute to the building's energy efficiency, comfort, and visual appeal.
- **Internal Wall Locations** facilitate functional zoning and optimal space utilization. The thoughtful placement of internal walls ensures a harmonious and practical layout, allowing

for efficient movement and usage within the building.

- **Specific Functional Areas** encompass the strategic alignment of essential zones such as bedrooms, bathrooms, and staircases. The Facade Openings coordination of these areas is a crucial aspect of the coherent design of internal plumbing systems, electrical wiring, and overall flow within the building.

The floor plans sourced from platforms such as ArchDaily and HousePlan are not inherently compatible with direct application in DL algorithms. To address this limitation, we have implemented a series of modifications to the floor plans. These alterations involve the utilization of distinct color schemes to symbolize various functional partitions within the layout. The corresponding color swatches, representing each unique functional area, are delineated in Figure 2. To achieve the desired outcomes in the different generation modes, it is essential to produce different distinct data types:

- **Building Boundaries and Facade Entrances** employ a black fill to delineate the building's footprint. Additionally, white lines are utilized to indicate the positions of the building's internal openings. The dataset for this generation mode consists of 600 distinct image sets.
- **Locations of Interior Walls** are built based on the foundational data of building boundaries and facade openings, and orange lines are introduced to specify the desired placement of the building's interior walls as determined by the user. The dataset for this generation mode consists of 600 distinct image sets.
- **Position of Functional Zones** incorporate color-coded blocks that correspond to distinct functional zones. The primary objective of this type of data is to facilitate the generation of floor plans, with an emphasis on the predetermined positioning of specific functional areas. The dataset for this generation mode consists of 600 distinct image sets.
- **Multi-unit Building Plan** is derived from the methodologies used in the first three data types (building boundary, interior and specific position function area). However, while the initial three focus on single-unit building plans, this dataset encompasses 20 groups of multi-unit data.
- **Multi-storey Building Plan** is founded on the first three data types. In this case, the Ground Floor Plan is presented on the left, while the First Floor Plan is on the right. This dataset also consists of 20 groups.

4. METHOD

In this section, we first focus on an in-depth introduction and analysis of the algorithmic framework. Then, we compare the pros and cons of various algorithmic frameworks, thereby clarifying for the reader why our work performs better. We also present the data obtained in our experiment, laying the groundwork for subsequent analysis and discussion.

In this paper, the GAN, which amalgamates both convolutional and anti-convolutional kernels, has been employed. This model, as delineated by Goodfellow et al.,[9] is lauded for its precise correspondence between the input and the resultant planar graph, a feature indispensable in intricate image processing tasks where accuracy is paramount. For the practical implementation of our work, we have opted for the open-source framework, pix2pix.[19] This selection was predicated on the framework's demonstrated efficacy in image-to-image translation endeavors and its inherent adaptability to a myriad of datasets. The foundational architecture of the pix2pix framework is characterized by its distinct components. The generator is intricately designed, encompassing convolutional, residual, and inverse convolutional layers. These layers collaboratively function to metamorphose the input image, ensuring the preservation of salient features whilst introducing requisite alterations, and continually generating images that resemble the original functional zoning map of the building. Concurrently, the discriminator serves as an arbiter of quality, juxtaposing the synthesized image against its original counterpart and furnishing iterative feedback to the generator for distinguishing between the generated image as well as the original functional zoning map. This interdependent relationship engenders a cycle of refinement, with the generator perpetually enhancing its outputs and the discriminator honing its evaluative acumen.

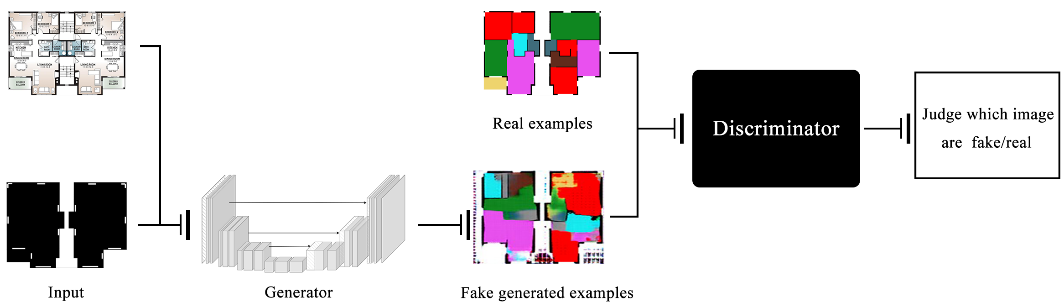

Figure 3. Training a conditional GAN involves mapping edges to photorealistic images. The discriminator D is designed to distinguish between real images and synthetic ones generated by G. Meanwhile, the generator G continuously improves to produce increasingly realistic images, aiming to deceive the discriminator.

Through an exhaustive analysis of the operational patterns of designers, we introduce a workflow that more closely aligns with the routine practices and proclivities of designers as shown in Figure 4. Recognizing the challenges designers may encounter in practical implementations, we have incorporated the GAN technology, aiming to guide and streamline their work. Once the design outcomes are generated, it is imperative for designers to conduct thorough evaluations, rectify elements that deviate from architectural standards or exhibit logical inconsistencies, and further refine the functional zoning.

4.1 Objective

The objective of the pix2pix can be represented as:

$$\mathcal{L}_{\text{pix2pix}}(G, D) = \mathbb{E}_{x,y}[logD(x, y)] + \mathbb{E}_{x,z}[log(1 - D(x, G(x, z)))] \tag{1}$$

where the generator G aims to minimize this objective while the discriminator D seeks to maximize it, thus, the optimal G^* can be formulated as:

$$G^* = \arg\min_G \max_D \mathcal{L}_{\text{pix2pix}}(G, D) \tag{2}$$

To evaluate the impact of conditioning on the discriminator, we also test an unconditional version, where D does not receive x as input:

$$\mathcal{L}_{\text{GAN}}(G, D) = \mathbb{E}_y[\log D(y)] + \mathbb{E}_{x,z}[\log(1 - D(G(x, z)))] \tag{3}$$

Prior research suggests that combining the GAN objective with a traditional loss, like L2 distance,[21] can be beneficial. While the discriminator's role remains focused on classification, the generator is encouraged not only to deceive D but also to approximate the true output, reducing error in an L2 sense. In our work, we employ L1 distance instead, as it tends to produce sharper results:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1] \tag{4}$$

Our final objective becomes:

$$G^* = \arg \min_G \max_D \mathcal{L}_{\text{cGAN}}(G, D) + \lambda \mathcal{L}_{L1}(G) \tag{5}$$

Without z, the network could map x to y but would yield deterministic outputs, thus failing to represent diverse outputs. Previous conditional GANs addressed this by introducing Gaussian noise z alongside x as an input to G.[22] However, in our initial tests, this approach proved ineffective, with the generator learning to disregard the noise—a finding consistent with Mathieu et al.[23] Instead, we inject noise through dropout across multiple layers in the generator, applied both during training and testing. Despite this, we observe only limited randomness in the outputs.

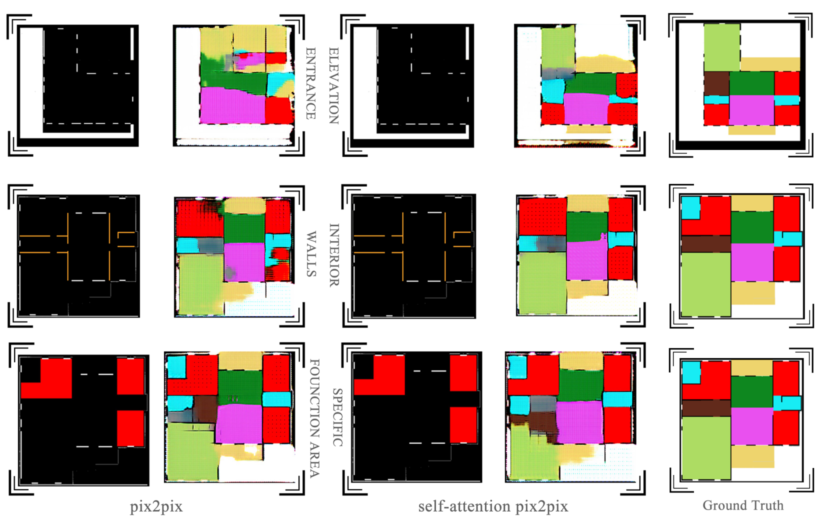

Figure 4. The comparison of generated images using pix2pix with and without optimization.

5. EXPERIMENTS

In this section, we first compare the pix2pix model without the introduction of the self-attention module and spectral normalization with the model that incorporates these techniques in terms of architectural floor plan generation. Through experiments, we verify whether these techniques can optimize the generation of architectural floor plans and explore the areas they have improved. Then, we delve into the generation of multi-unit architectural floor plans, examining whether GAN models can generate multi-unit or multi-layer architectural floor plans.

Both the unmodified pix2pix and the pix2pix with the introduced self-attention module and spectral normalization were trained for 300 epochs. In this experiment, a batch size of 1 was utilized to facilitate effective instance normalization, optimizing the model's performance for small datasets. The learning rate was set at 0.0002 and applied throughout training with the Adam[24] optimizer, using momentum parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$ to stabilize convergence. The deep learning framework employed was PyTorch,[25] leveraging an NVIDIA RTX 3070 Ti GPU to accelerate computations and improve training efficiency. We also ensured that the data used for testing was not involved in training to guarantee the accuracy of the experiments. Figure 4.1 illustrates the experimental results for the same set of floor plan data. The left two columns show the generated results without the introduction of the self-attention module and spectral normalization. It can be observed that without these techniques, the quality of the generated image is low, especially in terms of data capture at architectural boundaries. Although there is some color mixing in the internal functional area division, the overall generation quality is still acceptable. The right two columns present the results after introducing the self-attention module and spectral normalization. Although the size of the generated balcony area does not fully match the original, the division of functional areas within the architectural boundary is satisfactory. This is of great significance for architects during the initial phase of functional area division. The first row presents a comparison of results generated with the constraint of elevation entrance locations, the second row shows a comparison based on the constraint of fixed interior wall positions, and the third row illustrates the comparison under the constraint of fixed specific functional zoning locations.

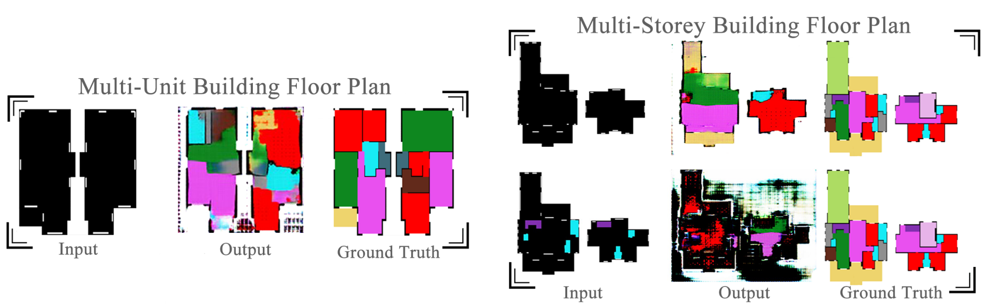

Figure 5. The generation results for multi-unit and multi-storey buildings are as follows: the left three columns displays the results generated with the elevation entrance as a condition for the multi-unit building, alongside the ground truth for comparison. The right three columns shows the multi-storey results generated with both the elevation entrance and specific functional areas as conditions, alongside their ground truth counterparts.

Our in-depth examination of the training mechanisms highlights the meticulous steps undertaken to optimize the GAN model. The model's progression, from initial hurdles to producing lucid and unequivocal floor plans, attests to its viability in real-world scenarios. Merging GANs with CAD tools could be a game-changer for architectural design. This study sets the stage for a paradigm where AI-driven tools like GANs collaborate with CAD software, infusing dynamism, efficiency, and innovation into the design process. In summary, this research's revelations could redefine the architectural design domain. By leveraging GAN, we introduce a pioneering method for floor plan generation, poised to revolutionize design toolkits and methodologies.

The single-unit building floor plan generation was assessed in the previous section. While the results were discernible to architects, some areas exhibited color blurring. For context, two works[8, 10] have previously attempted to output single-unit building floor plans using different methodologies. To determine if GAN can handle more intricate designs like multi-unit or multi-storey buildings, further experiments were conducted. Figure 5 displays multi-unit output. In the top images, while the building boundary is accurately captured, some internal functional areas exhibit color mixing. The generation test for such buildings is categorized into two modes: one without constraints and the other with fixed locations for stairs and bathrooms. The middle images depicts the generated plan with only building boundaries and facade openings. Without marked staircase positions in the input, the output lacks staircases, making it more akin to a multi-unit plan than a multi-storey one. The bottom images presents the results with marked staircase and bathroom locations. Unfortunately, the output is subpar and not usable for architectural design.

6. CONCLUSION

This work underscores the transformative potential of GAN in producing high-caliber architectural floor plans. By automating this initial phase, architects can channel their energies into refining and personalizing designs, rather than building from the ground up. This streamlines the design journey, offering a swift visualization of multiple layouts. Furthermore, by embedding controllable parameters within the GAN model, architects gain a tool that balances functionality with aesthetics, ensuring the output is both practical and visually appealing. Our in-depth examination of the training mechanisms highlights the meticulous steps undertaken to optimize the GAN model. The model's progression, from initial hurdles to producing lucid and unequivocal floor plans, attests to its viability in real-world scenarios. Merging GAN with CAD could be a game-changer for architectural design. This work sets the stage for a paradigm where AI-driven tools like GAN collaborate with CAD, infusing dynamism, efficiency, and innovation into the design process. In summary, this work's revelations could redefine the architectural design domain. By leveraging GAN, we introduce a pioneering method for floor plan generation, poised to revolutionize design toolkit and methodology.

REFERENCES

[1] Payne, A., "Materiality, crafting, and scale in renaissance architecture," *Oxford art journal* **32**(3), 365–386 (2009).

[2] Jackson, T. G., [*The renaissance of Roman architecture*], vol. 1, CUP Archive (1921).

[3] Anderson, C., [*Renaissance architecture*], OUP Oxford (2013).

[4] Alberti, L. B., [*On the Art of Building in Ten Books*], The MIT Press (1991).

[5] Bandler, J. W., "Optimization methods for computer-aided design," *IEEE Transactions on Microwave Theory and Techniques* **17**(8), 533–552 (1969).

[6] Encarnacao, J. L., Lindner, R., and Schlechtendahl, E. G., [*Computer aided design: fundamentals and system architectures*], Springer Science & Business Media (2012).

[7] Carpo, M., [*The second digital turn: design beyond intelligence*], MIT press (2017).

[8] Nauata, N., Chang, K.-H., Cheng, C.-Y., Mori, G., and Furukawa, Y., "House-gan: Relational generative adversarial networks for graph-constrained house layout generation," in [*Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*], 162–177, Springer (2020).

[9] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., "Generative adversarial nets," *Advances in neural information processing systems* **27** (2014).

[10] Zheng, H., Keyao, A., Jingxuan, W., and Yue, R., "Apartment floor plans generation via generative adversarial networks," in [*25th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA 2020): RE: Anthropocene, Design in the Age of Humans*], 601–610, The Association for Computer-Aided Architectural Design Research in Asia ... (2020).

[11] Wang, X.-Y., Yang, Y., and Zhang, K., "Customization and generation of floor plans based on graph transformations," *Automation in Construction* **94**, 405–416 (2018).

[12] Li, C., Zhang, T., Du, X., Zhang, Y., and Xie, H., "Generative ai for architectural design: A literature review," *arXiv preprint arXiv:2404.01335* (2024).

[13] Mirza, M., "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784* (2014).

[14] Radford, A., "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434* (2015).

[15] Arjovsky, M., Chintala, S., and Bottou, L., "Wasserstein generative adversarial networks," in [*International conference on machine learning*], 214–223, PMLR (2017).

[16] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C., "Improved training of wasserstein gans," *Advances in neural information processing systems* **30** (2017).

[17] Kim, T., Cha, M., Kim, H., Lee, J. K., and Kim, J., "Learning to discover cross-domain relations with generative adversarial networks," in [*International conference on machine learning*], 1857–1865, PMLR (2017).

[18] Taigman, Y., Polyak, A., and Wolf, L., "Unsupervised cross-domain image generation," in [*International Conference on Learning Representations*], (2022).

[19] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A., "Image-to-image translation with conditional adversarial networks," in [*Proceedings of the IEEE conference on computer vision and pattern recognition*], 1125–1134 (2017).

[20] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S., "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems* **30** (2017).

[21] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A., "Context encoders: Feature learning by inpainting," in [*Proceedings of the IEEE conference on computer vision and pattern recognition*], 2536–2544 (2016).

[22] Wang, X. and Gupta, A., "Generative image modeling using style and structure adversarial networks," in [*European conference on computer vision*], 318–335, Springer (2016).

[23] Mathieu, M., Couprie, C., and LeCun, Y., "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440* (2015).

[24] Kingma, D. P., "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980* (2014).

[25] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al., "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems* **32** (2019).