# Generation of camouflage based on the use of generative-competitive neural networks

Victor Sineglazov[1,*,†], Michael Zgurovsky [2,†], Dmytro Nikulin[3] , Kyrylo Lesohorskyi[2]

[1] *Institute of Cybernetics of the National Academy of Sciences of Ukraine, Kyiv, Ukraine*
[2] *National Technical University of Ukraine "Ihor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine*
[3] *Faculty of Air Navigation, Electronics and Telecommunications, National Aviation University, Kyiv, Ukraine*

**Abstract**

This work is devoted to camouflage pattern generation using generative adversarial neural networks (GANs). The problems of creating camouflage are identified: limited adaptability, the complexity of development, the subjectivity of evaluation, and lack of individualization. It has been shown that GANs are an effective means of creating visual camouflage despite the instability of their behavior during training associated with the fall of the gradient and the appearance of "mode collapse". To solve this problem, the following GAN architectures were analyzed: Progressive Growing GAN, StyleGAN/StyleGAN2, CycleGAN, Pix2Pix, Deep Convolutional GAN, Wasserstein GAN, Conditional GAN, Self-Attention GAN, Hybrid GAN. As a result, a new GAN architecture is proposed, based on the use of U-Net as a generator, a simplified discriminator architecture, the use of a combination of cross-entropy and MSE as a cost function, multiple angles, concatenation of features. The proposed GAN architecture demonstrates the potential to create effective camouflage patterns tailored to specific landscapes.

**Keywords**

camouflage, generative adversarial neural networks. structural-parametrical synthesis , generator, discriminator



## 1. Introduction

Camouflage is a collection of methods and means designed to disguise objects, such as people, equipment, or structures, by visually merging them with the environment. The main purpose of camouflage is to make an object difficult to see or recognize for an observer, which can be achieved by using colors, patterns, shapes, and materials that mimic the natural environment or create optical illusions [1]. Camouflage is an important tool used in various fields of human activity to achieve a variety of purposes, from ensuring safety and survival to creating aesthetic effects and entertainment. The development of new technologies and

[1] **ITTAP'2024: 4th International Workshop on Information Technologies: Theoretical and Applied Problems, October 23-25, 2024, Ternopil, Ukraine, Opole, Poland*

[1*] Corresponding author.
[†] These authors contributed equally.
svm@nau.edu.ua (V. Sineglazov); mzz@kpi.ua (M. Zgurovsky); 6350583@stud.nau.edu.ua (D. Nikulin); lesogor.kirill@gmail.com (K. Lesohorskyi);
0000-0002-3297-9060 (V. Sineglazov); 0000-0001-5896-7466 (M. Zgurovsky); 0000-0003-2773-7398 (K. Lesohorskyi);

materials opens up new opportunities for creating more effective and versatile camouflage, which can be used in an even wider range of applications.

Camouflage is used not only in the military sphere but also in many other fields of human activity: cinema and television, art and design, fashion, architecture and interior design.

## 2. Related Works

Creating effective camouflage is a complex task that requires a comprehensive approach and consideration of many factors. Among the main problems faced by camouflage developers, the following can be distinguished [2]:

- limited adaptability: Traditional camouflage patterns are developed for specific types of terrain and lighting. When the conditions (time of day, weather, season) change, their effectiveness can significantly decrease. This makes them less effective in the dynamic environment of the modern battlefield;
- design complexity: Creating an effective camouflage pattern requires a deep understanding of the principles of camouflage, taking into account the peculiarities of human and animal vision, as well as analyzing a large amount of environmental data. This is a labor-intensive and time-consuming process that requires the involvement of highly qualified specialists;
- subjectivity of assessment: Evaluation of the effectiveness of camouflage often depends on the subjective opinion of experts, which can lead to ambiguous results and make it difficult to choose the optimal solution;
- lack of individualization: Traditional camouflage patterns, as a rule, are universal and do not take into account the individual characteristics of military personnel and equipment. This can reduce their effectiveness in specific situations.

Solving these problems requires an integrated approach that combines knowledge from different fields. Generative-competitive neural networks (GANs) are promising ways to create visual camouflage[3]: GANs are a powerful machine learning tool that can be used to create high-quality and adaptive visual camouflage. They consist of two neural networks: a generator that creates images and a discriminator that evaluates their realism. These two networks are trained in a competitive process where the generator tries to produce images that the discriminator cannot distinguish from real photos [4, 5, 6, 7].

There are many ways to classify GANs, depending on their architecture, the type of data they generate, and their specific applications [8], some of the most popular architectures are:

1. Progressive Growing GAN (PGGAN) [9]: This architecture allows you to gradually increase the resolution of the generated images, starting with a low resolution and adding new layers during the training process. This avoids problems with learning instability and provides high-quality images. PGGAN can be particularly useful for generating detailed camouflage patterns that must accurately represent terrain [28] textures and colors;
2. StyleGAN/StyleGAN2 [10,11]: This architecture allows you to control the style of generated images using latent vectors. This can be used to create camouflage patterns

with different styles to suit different lighting conditions or types of terrain. StyleGAN2 is an improved version of StyleGAN that provides even more control over style and image quality;

3. CycleGAN [12]: This architecture allows you to transform images from one domain to another (for example, from a photo of a summer forest to a photo of a winter forest). CycleGAN can be used to create camouflage patterns that adapt to different seasons or weather conditions;
4. Pix2Pix [13]: This architecture is designed to convert images from one type to another while preserving the content of the original. Pix2Pix can be used to create camouflage patterns that closely match the shape and contours of the object being camouflaged;
5. Deep Convolutional GAN (DCGAN) [14]: This architecture is one of the simplest and most efficient for image generation. It uses convolutional layers for image processing and can be easily adapted to generate camouflage patterns;
6. Wasserstein BY (WGAN) [15,16,17]: WGAN solves the instability problem of GAN training by using the Wasserstein metric to measure the distance between the distributions of real and generated data. This avoids the problem of vanishing gradients and provides more stable training;
7. Conditional GAN [18,19,20]: This type of GAN allows you to control the generation process with additional conditional data. In the context of camouflage generation, this can be data about the type of landscape, lighting conditions, or the type of object being camouflaged;
8. Self-Attention GAN (SAGAN) [21]: SAGAN uses a self-attention mechanism to model dependencies between different parts of an image. This allows for more detailed and realistic images to be generated, which can be useful for creating high-resolution camouflage patterns;
9. Hybrid generative adversarial network (HGAN): provides a way to avoid the mode collapse problem. Using the hybridization approach both in the development of a new topology and in training methods can significantly increase the efficiency of the neural network [22,23,24]. In HGAN [25] provides data density estimation using an autoregressive model and supports both adversarial and likelihood structures in the form of joint learning, which diversifies the estimated density to cover different modes;Drag and drop all .otf files to the Font Book window.

# 3. Method

## 3.1. GAN Architecture

During the development of the camouflage pattern generator, several neural network architectures were considered, including traditional convolutional networks (CNNs) and various variations of generative adversarial networks (GANs).[26]. After careful analysis, the U-Net architecture was chosen as the most suitable for solving the given task. The U-Net architecture was first proposed by Olaf Ronneberger, Philipp Fischer, and Thomas Brocks in 2015 [27] in the article "U-Net: Convolutional Networks for Biomedical Image Segmentation". It was originally developed to solve biomedical image segmentation problems but quickly found applications in many other areas of computer vision, including image generation. U-Net has a number of advantages that make it an attractive choice for image generation:

- efficiency in image generation tasks: U-Net is widely used for image generation, especially in medical imaging, where it exhibits high quality reconstruction of details and structures. This feature is important for creating realistic camouflage patterns that must accurately match the features of the landscape;
- skip connections: U-Net uses skip connections between the encoder and decoder layers. These connections allow information from the early encoder layers to be passed directly to the corresponding decoder layers. This helps preserve image detail, especially important for rendering fine textures and contours to generate more accurate and realistic images, which is critical for effective camouflage;
- symmetrical structure: The U-shaped structure of the network (Fig. 1) ensures the symmetry of the encoding and decoding processes, which contributes to a better understanding and interpretation of the model. Such a structure allows you to easily modify and adjust the model to the specific requirements of the task;
- adaptability to different image sizes: U-Net can be easily adapted to work with images of different sizes, which is important for creating camouflage patterns that can be applied to objects of different sizes and shapes;
- effectiveness of learning: U-Net can be effectively trained even on limited data sets, which is relevant for the task of military camouflage generation, where collecting a large amount of representative data can be difficult;
- considering these advantages, the U-Net architecture was chosen as the basis for the camouflage pattern generator in this work. It allows you to effectively use information from multiple landscape images, preserve detail and structure, and create realistic and adaptive camouflage patterns;

U-Net consists of two main parts – encoder and decoder. Contracting path ,or encoder, is a sequence of convolutional layers (Conv2D) that reduce the spatial dimensions (height and width) of the input image and increase the number of channels (depth). Each convolutional layer is usually accompanied by an activation layer (for example, ReLU or LeakyReLU) and a batch normalization layer (Batch Normalization). Encoder responds for removing features from the image.

Expansive path, or decoder, is a sequence of transposed convolutional layers (Conv2D Transpose), which increase the spatial dimensions of the image and reduce the number of channels. Each transposed convolutional layer is also usually accompanied by an activation layer and a batch normalization layer. The decoder is responsible for generating the image based on the extracted features (Fig. 1).

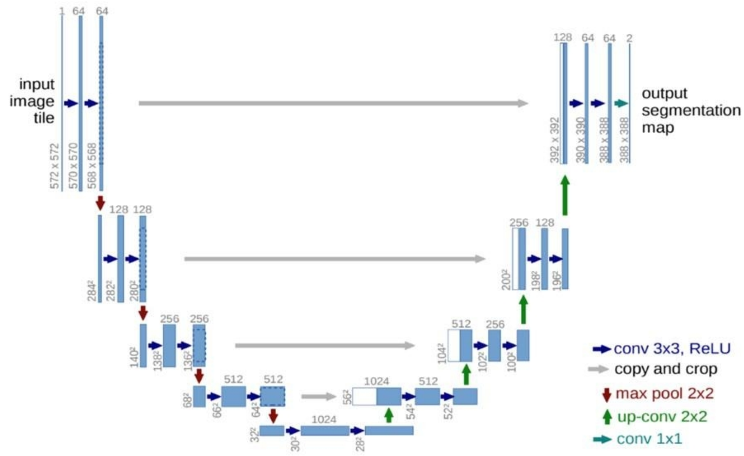


**Figure 1:** U-net generator architecture.

In our implementation of the camouflage generator, a modified U-Net architecture that takes into account the specifics of the task is proposed. In particular, we feed 10 images of the same landscape taken from different angles to the input of the generator.

Each image is processed by a separate encoder, which extracts a set of features from it. These features are then combined using the Concatenate layer, creating a single tensor that contains information about all angles of the landscape. This tensor is then passed to a decoder that generates an image of the camouflage pattern. Input data consists of a batch of 10 landscape images of $256 \times 256 \times 3$ dimensons. Encoder path gradually reduces dimensionality and extracts features. Encoder consists of blocks, namely 2D convolutional layer (Conv2d), followed by a ReLu activation function that introduces non-lineriality, and then followed by BatchNorm to normalize layer outputs to enable more stable learning. After the feature extraction, the whole batch is concatenated together into a single tensor to combine multi-angle information. After concatenation, the tensor is passed into the decoder pathway that consists of 2D transposed convolutions that extends the dimensions to the original input size. Last layer is the output layer that produces the $256 \times 256 \times 3$ camouflage pattern patch. Advantages of such approach are:

- taking into account the variety of visual characteristics of the landscape: The landscape can significantly change its appearance depending on the viewing angle, lighting and other factors. Using 10 photos from different angles allows the generator to take this diversity into account and create a camouflage pattern that will be effective when viewed from different positions;
- creating more realistic and adaptive patterns - each photo contains unique information about the landscape, such as textures, colors, shapes, and shadows. Combining this information allows the generator to create more realistic and adaptive camouflage patterns that blend in better with the environment.
- prevention of overfitting - using 10 photos instead of one helps prevent generator overfitting. The model learns not just to reproduce specific details from a single image, but to detect general patterns and structures inherent in a given landscape.

- increasing the effectiveness of masking - a camouflage pattern created on the basis of 10 photos will be more effective in masking an object on a given landscape because it takes into account its visual characteristics from different angles. This makes it possible to reduce the probability of detecting the object when observing from different positions.
- Versatility - generator model trained on 10 images of one landscape can be easily adapted to create camouflage for other landscapes. To do this, it is enough to replace the input images with photos of a new landscape and retrain the model.;

Thus, using 10 photos at the input of the generator is an effective approach to create a versatile and adaptable military camouflage. It allows you to take into account the variety of visual characteristics of the landscape, prevent overtraining of the model and increase the effectiveness of camouflage.

Discriminator architecture is not significantly changed in comparison to classic image GANs. A standard CNN architecture, derived from the encoder pathway with binary classification head is used.

## 3.2. Loss Function

Loss functions play a key role in training generative adversarial networks (GANs) because they determine exactly how the model evaluates its performance and adjusts its parameters [30, 31]. In the context of GANs, two main loss functions are used: one for the generator and one for the discriminator.

We start by outlining the loss functions that are commonly used in the generator.

Binary Cross Entropy (BCE) is widely used in GANs to evaluate how well the generator can fool the discriminator. It calculates the difference between the probability distributions predicted by the discriminator for real and generated images.

Mean Squared Error (MSE) measures the root mean square difference between generated and real images at the pixel level. This lossy function can be useful for improving the visual quality of the generated images, ensuring that they resemble the original photographs.

Wasserstein Loss is used in Wasserstein GAN (WGAN) and provides more stable training compared to BCE. It measures the distance between the distributions of real and generated data, which makes it less sensitive to the problems of vanishing gradients and mode collapse. This is especially important when there is a limited amount of training data, as in our case.

Perceptual Loss: This loss function estimates the difference between the generated image and the real image based on high-level features extracted from a pre-trained neural network (eg VGG). It helps the generator create images that are not only pixel-like, but also in content and style, which can be useful for creating realistic camouflage patterns:

$$PL=\sum_{i=0}^{N} \lambda_i \cdot \left\| \varphi_i(y) - \varphi_i(y') \right\|_2^2, \tag{1}$$

where $N$ is the number of layers in the network, which are used to calculate losses, i are weighting factors for layers $\lambda_i$; $\varphi_i(y)$ is a feature function that extracts features from a layer $i$ for the target image $y$; $\|\|_2^2$is the L2 norm.

Discriminator can use some of the loss functions outlined above, however usually it uses either of the following options. Binary Cross Entropy - as with the generator, BCE is used to evaluate the ability of the discriminator to distinguish between real and generated images. Wasserstein Loss - in WGAN, the discriminator also uses Wasserstein loss, but with the opposite sign compared to the generator.

In the proposed approach, mean squared loss (MAE) is used as a generator loss. This loss function is used to compare the generated image with the concatenated image from 10 angles. This helps the generator to create a pattern that will be similar to all input images, taking into account different angles of the landscape:

$$MSE=\frac{1}{N}\sum_{i=0}^{N}(y_i-y_i^{'})^2, \tag{2}$$

where $N$ is the number of samples; $y_i$ is the actual value for the sample i, $y_i^{'}$ is the predicted value for the sample *i*.

For discriminator, binary cross entropy loss is used. It causes the discriminator to correctly classify real images as genuine and generated ones as fakes. This helps the discriminator become better at its task, which in turn forces the generator to produce more realistic images:

$$BCE=\frac{-1}{N}\sum_{i=0}^{N} ¿¿, \tag{3}$$

where $N$ is the number of samples, $y_i$ is the actual label (0 or 1) for the sample *i*, $p_i$ is the predicted probability for the sample *i*.

## 3.3. Hyper-parameter tuning

Setting hyperparameters is an important step in the process of developing and training a GAN, as they significantly affect the quality and stability of the model. In the case of our GAN, the key hyperparameters are:

Batch size (batch_size): Defines the number of images processed per training iteration. Increasing the batch size can speed up training, but requires more memory. In our case, batch_size = 10 (number of angles).

Number of epochs: Specifies the number of passes over the entire training data set. More epochs can lead to better learning, but can also lead to overtraining. In our case, epochs = 1000.

Learning rate (learning_rate): Determines how much the model weights change after each iteration. Too high a learning rate can lead to instability, and too low a slow learning rate. In our case, learning_rate = 0.005 for the generator and learning_rate = 0.0001 for the discriminator.

Adam optimizer beta parameters (beta_1, beta_2): Defines how the Adam optimizer takes first-order and second-order moments into account. The values of beta_1 = 0.5 and beta_2 = 0.5 are used.

## 3.4. Metrics for evaluating the quality of generated images

Both qualitative and quantitative metrics are used to evaluate the quality of the generated camouflage patterns. Quantitative metrics allow you to objectively assess the variety and realism of images, while qualitative metrics are based on the subjective assessment of experts. Inception Score (IS) and Fréchet Inception Distance (FID) metrics were used to quantify the quality of generated camouflage patterns. The Inception Score (IS) is a metric used to evaluate the quality and diversity of images generated by models such as Generative Adversarial

Networks (GANs). It evaluates both the information diversity and the plausibility of the generated images.

To calculate the IS, a pre-trained Inception model is first used, which classifies images into different classes. The probabilities of the images belonging to each class are obtained. For each generated image $x$, the Inception model gives the probability $(y|x)$ as well $y$ – class.

Inception Score is calculated through the ratio between $(y|x)$ and $(y)$:

The Inception Score (IS) is a metric used to evaluate the quality and diversity of images generated by models such as Generative Adversarial Networks (GANs). It evaluates both the information diversity and the plausibility of the generated images.

To calculate the IS, a pre-trained Inception model is first used, which classifies images into different classes. The probabilities of the images belonging to each class are obtained. For each generated image $x$, the Inception model gives the probability $(y|x)$ as well $y$ – class.

Inception Score is calculated through the ratio between $(y|x)$ and $(y)$:

$$IS = e^{E_x\left[D_{KL}(P(y \vee x) \vee \text{¿} P(y))\right]}, \tag{4}$$

where $D_{KL}$ stands for Kullback-Leibler divergence, which calculates the distance between two distributions:

$$D_{KL}(P(x) \vee |P(y)) = \sum_{y} P(x) \log_2\left(\frac{P(x)}{P(y)}\right). \tag{5}$$

Fréchet Inception Distance (FID) is a metric used to evaluate the quality of generated images compared to real images. FID takes into account not only the diversity of images, but also how well their distribution approximates the distribution of real images.

FID calculation is based on the comparison of the statistical characteristics of the images extracted using the pre-trained Inception model. In particular, FID compares the mean vectors and covariances of two sets of images in feature spaces that are obtained from a certain intermediate layer of the Inception model. FID is calculated by the formula:

$$FID = \text{¿} \vee \mu_r - \mu_g \vee \text{¿}^2 + \sqrt{Tr\left(\Sigma_r + \Sigma_g - 2\Sigma_r \Sigma_g\right)}, \tag{6}$$

where, $N(\mu_r, \Sigma_r)$, $N(\mu_g, \Sigma_g)$are multivariate normal distributions for real and generated images, respectively, where $\mu_r$and $\Sigma_r$ are average and covariance for real images, and $\mu_g$ and $\Sigma_g$ is the mean and covariance for the generated images, $Tr$is the trace of the matrix (the sum of its diagonal elements).

## 4. Results

As an example, consider the problem of generating camouflage patterns (Fig. 2) based on realistic images (Fig. 3). Table 1 shows that the value of the Inception Score (IS) increases with each learning epoch. This indicates that the generated images are becoming more diverse and clear. At the beginning of training (100 epochs), the IS is 1.28, which indicates low image quality. However, as the number of epochs increases, the IS increases to 2.23 per 1000 epochs, indicating a significant improvement in the quality of the generated images. Fréchet Inception Distance (FID) values decrease with each training epoch. This means that the feature distribution of generated images becomes closer to the feature distribution of real images. At the beginning of training (100 epochs), the FID is 148.3, indicating a significant difference between the generated and real images. However, as the number of epochs increases, the FID decreases to 56.2 per 1000 epochs, indicating a significant improvement in the quality and realism of the generated images.

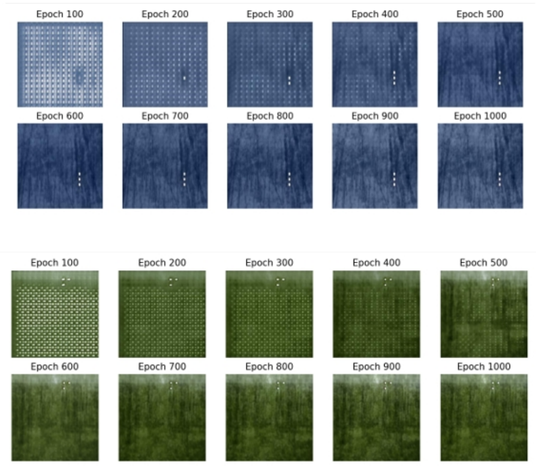


**Figure 2:** Generated images.

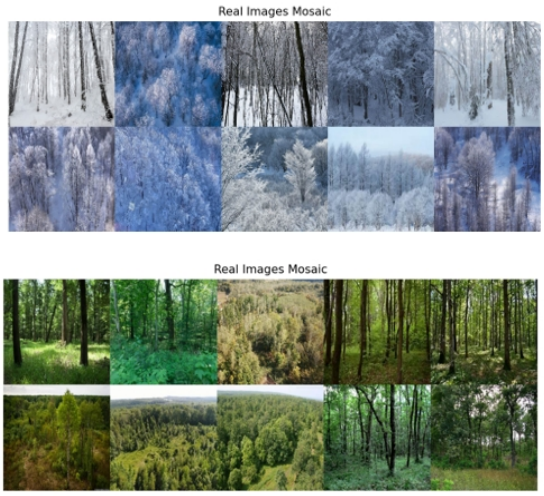


**Figure 3:** Real images from the training dataset.

**Table 1**
Quality assessment results

| Epoch | Inception Score (IS) | Fréchet Inception Distance (FID) |
|---|---|---|
| 100 | 1.28 | 148.3 |
| 200 | 1.42 | 132.1 |
| 300 | 1.58 | 116.9 |

| | | |
|---|---|---|
| 400 | 1.71 | 102.4 |
| 500 | 1.85 | 88.7 |
| 600 | 1.92 | 79.5 |
| 700 | 1.98 | 71.2 |
| 800 | 2.05 | 63.8 |
| 900 | 2.12 | 57.1 |
| 1000 | 2.23 | 56.2 |

Quality evaluation results based on IS and FID metrics confirm that the proposed GAN model is capable of learning and improving the quality of generated camouflage patterns over time. An increase in IS and a decrease in FID mean that images become more diverse, sharp and realistic.

## 5. Conclusions

The proposed GAN architecture demonstrates the potential to generate effective camouflage patterns adapted to specific landscapes. It takes into account different terrain angles, uses effective training methods and can be easily adapted for different types of data. Further research and experimentation may lead to even more impressive results in camouflage generation.

Batches of 10 images of the same landscape from different angles to the input of the generator allow us to take into account the diversity of its visual characteristics. This helps create more realistic and adaptive camouflage patterns that effectively mask objects when viewed from different angles. Combining the features extracted from each image provides the generator with more complete information about the landscape, allowing it to create more complex and varied patterns that better match the characteristics of the terrain.

Using a simplified discriminator architecture with fewer layers and neurons helps prevent overtraining and maintains a balance between the generator and the discriminator. This is especially important when there is a limited amount of training data. The combination of these loss functions allows to simultaneously improve the quality of the generated images (MSE) and the ability of the generator to deceive the discriminator (BCE).

The proposed GAN architecture can be easily adapted to generate camouflage patterns for different types of landscapes. To achieve this, it is enough to replace the input images with photos of a new landscape and retrain the model.

Future research involves using more complex architectures (e.g. StyleGAN2-ADA), additional loss functions (e.g. perceptual loss), as well as regularization methods (e.g. spectral normalization).

## References

[1] Talas L, Baddeley RJ, Cuthill IC. 2017 Data from: Cultural evolution of military camouflage. DataDryad Repository. (http://dx.doi.org/10.5061/dryad.n511h)

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in Neural Information Processing Systems, pages 2672–2680, 2014.

[3] [Salimans et al., 2016] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In Advances in Neural Information processing Systems 29, pages 2234–2242. 2016

[4] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in Neural Information Processing Systems, pages 6626–6637, 2017.

[5] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In Advances in Neural Information Processing Systems, 2020.

[6] V.M. Sineglazov, K.D. Riazanovskiy, O.I. Chumachenko, (2020). Multicriteria conditional optimization based on genetic algorithms. In: System research and information technologies, No. 3 (2020) . DOI: https://doi.org/10.20535/SRIT.2308-8893.2020.3.07 - 3

[7] Ming-Yu Liu, Xun Huang, Jiahui Yu, Ting-Chun Wang, and Arun Mallya. Generative adversarial networks for image and video synthesis: Algorithms and applications. arXiv preprint arXiv:2008.02793, 2020.

[8] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. "Progressive Growing of GANs for Improved Quality, Stability, and Variation". In: International Conference on Learning Representations. 2018

[9] Tero Karras, Samuli Laine, and Timo Aila. "A Style-Based Generator Architecture for Generative Adversarial Networks". In: Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019, pp. 4401–4410

[10] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. "Analyzing and Improving the Image Quality of StyleGAN". In: Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, June 2020

[11] Jun-Yan Zhu. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks URL: https://arxiv.org/pdf/1703.10593.pdf

[12] Joyce Henry, Terry Natalie, Den Madsen, "Pix2Pix GAN for Image-to-Image Translation", 2021

[13] Alec Radford & Luke Metz. Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks URL: https://arxiv.org/pdf/1511.06434.pdf

[14] M. Arjovsky, S. Chintala, and L. Bottou, Wasserstein generative adversarial networks, in International conference on machine learning, pp. 214–223, PMLR, 2017

[15] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, Improved training of wasserstein gans, arXiv preprint arXiv:1704.00028, 2017.

[16] H. Ni, L. Szpruch, M. Sabate-Vidales, B. Xiao, M. Wiese, and S. Liao, Sig-wasserstein gans for time series generation, arXiv preprint arXiv:2111.01207, 2021

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks URL: https://arxiv.org/pdf/1611.07004.pdf

[18] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In Proceedings of the 34th International Conference on Machine Learning, pages 2642– 2651, 2017
[19] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.
[20] Han Zhang, Ian Goodfellow, Dimitris Metaxas, Augustus Odena, "Self-Attention Generative Adversarial Networks", 2018
[21] Sineglazov , V., & Kot, A. (2021). Design of hybrid neural networks of the ensemble structure. Eastern-European Journal of Enterprise Technologies, 1(4 (109), 31–45. https://doi.org/10.15587/1729-4061.2021.225301
[22] Zgurovsky, M., Sineglazov, V., Chumachenko, E. (2021). Formation of Hybrid Artificial Neural Networks Topologies. In: Artificial Intelligence Systems Based on Hybrid Neural Networks. Studies in Computational Intelligence, vol 904. Springer, Cham. https://doi.org/10.1007/978-3-030-48453-8_3
[23] Zgurovsky, M., Sineglazov, V., Chumachenko, E. (2021). Classification and Analysis Topologies Known Artificial Neurons and Neural Networks. In: Artificial Intelligence Systems Based on Hybrid Neural Networks. Studies in Computational Intelligence, vol 904. Springer, Cham. https://doi.org/10.1007/978-3-030-48453-8_1
[24] Aditya Grover, Manik Dhar, and Stefano Ermon. Flow-gan:Combining maximum likelihood and adversarial learning ingenerative models. arXiv preprint arXiv:1705.08868, 2017.2, 4
[25] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4401–4410, 2019.
[26] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", 2015
[27] V. Sineglazov, "Main features of terrain-aided navigation systems," 2014 IEEE 3rd International Conference on Methods and Systems of Navigation and Motion Control (MSNMC), Kiev, Ukraine, 2014, pp. 49-52, doi:10.1109/MSNMC.2014.6979728.