UDC 004.946.2
DOI https://doi.org/10.32782/tnv-tech.2024.5.3

IMPROVEMENT OF 3D GRAPHICS IMAGE OPTIMIZATION TECHNOLOGY

Hrabovskyi Ye. M. *– PhD, Associate Professor at the Department of Multimedia Systems and Technologies of the Simon Kuznets Kharkiv National University of Economics*
ORCID ID: 0000-0001-7799-7249
Scopus-Author ID: 57201773546

Kobzev I. V. *– PhD, Associate Professor at the Department of Multimedia Systems and Technologies of the Simon Kuznets Kharkiv National University of Economics*
ORCID ID: 0000-0002-7182-5814
Scopus-Author ID: 57226248104

The article proposes the improvement of the technology of image optimization of 3D graphics. The object of research in this article is the process of using a software component to improve the technology of image optimization of 3D graphics. The practical significance of this study lies in the development of recommendations for improving the technology of image optimization of 3D graphics. The article provides a justification for the need to optimize graphic objects in such a way that they look realistic, but at the same time do not overload the hardware. The problems of having a high number of polygons, textures and complex lighting effects, which can create a significant load on the graphics processor, are considered. The importance of optimizing 3D graphics images to reduce the demands on computer computing resources, such as RAM and video card memory, is considered. The article describes the impact of optimizing 3D graphics images on reducing rendering time, which is especially important in industries that require high data processing speed, such as architectural modeling, simulations, and virtual reality. New rendering approaches such as real-time ray tracing and voxel-based rendering are covered. The characteristic features of the polygonal model for representing 3D objects are systematized. The need to use compression and texture optimization as important elements in 3D graphics that directly affect rendering performance and memory usage is substantiated. The article proposes a calculation of the effectiveness of using a texture atlas. In this study, mathematical modeling of polygons representing 3D objects was carried out. The article provides a lighting precalculation technique that is used for complex scenes, which allows to reduce the load on the system during the execution of the program. In this work, mathematical modeling of ray tracing processes is presented, which allows to describe the interaction of light with the surfaces of objects, taking into account reflection and refraction. The article presents a technology for optimizing polygonal grids, which is directly related to the number of polygons that make up the object. The improved technology of image optimization of 3D graphics acts as a scientific result of the conducted research.

Key words: *3D graphics, modeling, texture atlas, optimization, lighting, rendering, complex scenes.*

Грабовський Є. М., Кобзев І. В. Вдосконалення технології оптимізації зображень 3D-графіки

У статті запропоновано вдосконалення технології оптимізації зображень 3D-графіки. У якості об'єкта дослідження в даній статті виступає процес застосування програмної складової для вдосконалення технології оптимізації зображень 3D-графіки. Практична значущість даного дослідження полягає в розробленні рекомендацій стосовно вдосконалення технології оптимізації зображень 3D-графіки. У статті наведено обгрунтування необхідності оптимізувати графічні об'єкти таким чином, щоб вони виглядали реалістично, але при цьому не перевантажували апаратне забезпечення. Розглянуто проблеми наявності високої кількості полігонів, текстур та складних світлових ефектів, які можуть створювати значне навантаження на графічний процесор. Розглянуто значення оптимізації 3D-графічних зображень для зниження вимог до обчислювальних ресурсів комп’ютера, таких як оперативна пам'ять та пам'ять відеокарти. У статті описаний вплив оптимізації 3D-графічних зображень на зниження часу рендерингу, що є особливо

важливим у галузях, де потрібна висока швидкість обробки даних, таких як архітектурне моделювання, симуляції та віртуальна реальність. Розглянуто новітні підходи до рендерингу, такі як трасування променів у реальному часі та рендеринг на основі вокселів. Систематизовано характерні особливості полігональної моделі для представлення 3D-об'єктів. Обґрунтовано необхідність використання компресії та оптимізації текстур як важливих елементів у 3D-графіці, що безпосередньо впливають на продуктивність рендерингу та використання пам'яті. У статті запропоновано розрахунок ефективності використання текстурного атласу. В даному дослідженні проведено математичне моделювання полігонів представлення 3D-об'єктів. У статті наведено техніку попереднього розрахунку освітлення, яка використовується для складних сцен, що дозволяє знизити навантаження на систему під час виконання програми. В даній роботі подано математичне моделювання процесів трасування променів, що дозволяє здійснити опис взаємодії світла з поверхнями об'єктів, враховуючи відбиття та заломлення. В статті подана технологія оптимізації полігональних сіток, яка прямо пов'язана з кількістю полігонів, що складають об'єкт. У якості наукового результату проведеного дослідження виступає вдосконалена технологія оптимізації зображень 3D-графіки.

Ключові слова: *3D-графіка, моделювання, текстурний атлас, оптимізація, освітлення, рендеринг, складні сцени.*

Formulation of the problem. 3D graphics are an integral part of many industries today, including gaming, film, virtual reality, and architectural modeling. However, increasing requirements for image quality and rendering speed create new challenges for developers and designers. One of the key aspects of achieving high performance is optimizing 3D graphics images.

3D graphics have a huge impact on many industries in the modern world, including computer games, virtual reality (VR), augmented reality (AR), cinematography and architectural modeling. Optimizing 3D images is critical to achieving realistic scenes without sacrificing performance.

3D graphics are actively used in architectural modeling, medicine and scientific research, where visualization is of key importance. For these areas, it is important to provide the most realistic images without losing quality during optimization. Modern optimization algorithms, in particular texture and polygon compression methods, allow you to achieve this balance. This significantly improves the user experience and increases the efficiency of the work of specialists in various fields.

Artificial intelligence and machine learning also play an important role in improving 3D graphics optimization technologies . For example, neural networks are able to automatically identify the least important elements of an image and optimize them without affecting the overall quality. This provides new opportunities for creating high-quality graphic content with minimal resource consumption. 3D graphics image optimization technologies have great potential for development, which makes them an important direction in modern research and development.

Therefore, the relevance of improving the technology of image optimization of 3D graphics is due to the growing requirements for the quality of visualization and the speed of data processing in many areas.

Analysis of recent research and publications. In studies [1–3], approaches are proposed to create an adaptive interface of web-based tools for optimizing graphic images. A description of specific algorithms, on the basis of which certain elements of 3D graphics optimization technology can be designed, is presented in works [4, 5]. Studies [6, 7] provide recommendations on the use of 3D graphics to create the interface of web applications and real-time tools for navigating in a virtual environment. Scientific works [8, 9] contain methodological recommendations for substantiating an innovative strategy for the development of information technologies as a basis for further improvement

of 3D graphics optimization technology . Practical recommendations for the use of Workflow, which can be used to optimize 3D graphic images, are given in studies [10, 11].

The analysis of literary sources shows that in the specialized literature there are no methodological recommendations for improving the technology of image optimization of 3D graphics.

Purpose and task statement. The aim of the work is to improve the image optimization technology of 3D graphics.

The object of the study is the process of applying the software component to improve the technology of image optimization of 3D graphics.

The subject of research is the technology of image optimization of 3D graphics.

Presentation of the main research material. Optimization in 3D graphics is a key aspect of developing any application that uses 3D models, as it directly affects performance and user experience. High polygon counts, textures, and complex lighting effects can put a heavy load on the graphics processing unit (GPU), resulting in slower framerates and lag. To ensure a smooth display of the scene, especially in real time, it is necessary to optimize the objects so that they look realistic, but at the same time do not overload the hardware.

In addition to ensuring performance, optimization is important to reduce the demands on computing resources such as RAM and graphics card memory. This allows for better cross-platform compatibility, including resource-constrained mobile devices. Texture compression, LOD reduction, and polygon model simplification can be used to maintain a high level of visual quality even on weaker devices. This makes optimization critical for developers looking to strike a balance between image quality and accessibility to a wide audience.

The optimization also helps reduce rendering times, which is especially important in industries that require high data processing speeds, such as architectural modeling, simulations, and virtual reality (VR). By reducing the time required to render complex scenes, the optimization allows you to focus on improving detail and interactivity, creating a more realistic and immersive experience for users. Thus, optimization in 3D graphics provides performance, availability and quality, which makes it an integral part of the process of developing visual content.

Optimizing images in 3D graphics consists in reducing the load on hardware resources, while ensuring maximum image quality. This is especially important for real-time, where every delay can negatively affect the user experience. The main aspects to optimize are polygons, textures, lighting and shadows, and shader-based rendering.

Optimization of polygons, or «LOD» (Level of Detail), consists in reducing the number of polygons in objects located at a great distance from the camera. This allows you to significantly reduce the load on the graphics processor (GPU), which is especially important for games and applications with an open world. Modern methods use dynamic displacement algorithms that automatically adapt the level of detail depending on the position of the camera.

Textures take up a significant portion of the memory used when rendering a scene. Using compression methods such as DXT (S3TC), ASTC or the latest formats based on machine learning allows you to reduce the size of textures without losing quality. In addition, the use of texture atlases allows you to avoid additional loading and switching of textures, which speeds up rendering.

Lighting and shaders are important components of 3D scenes that affect the final image quality. Traditional lighting is often replaced by approaches that use global

lighting and physically correct rendering techniques (PBR). Using such techniques as baking lighting or Light Probes, allows you to preserve the quality of lighting effects with minimal computational costs. Shader optimization consists in simplifying complex operations and using a computer buffer (G – buffer) to calculate lighting in real time.

Rendering approaches, such as real-time ray tracing and voxel-based rendering, allow for significant improvements in image quality, but require significant computing resources. To reduce the load on the system, hybrid approaches combining traditional rendering and ray tracing are used. Artificial intelligence algorithms, such as DLSS (Deep Learning Great Sampling) from NVIDIA, which allow you to render images in a lower resolution and increase them without a significant loss of quality.

Machine learning and neural networks are becoming increasingly popular for optimizing 3D images. They allow you to automate processes that previously required significant human involvement, such as creating LOD models, generating normalized maps, and predicting complex lighting effects. Thanks to machine learning, it is possible to achieve optimization of image quality without significantly increasing the load on the hardware.

Although modern optimization methods have greatly improved the quality and performance of 3D images, there are still many challenges. The development of new algorithms and the integration of machine learning into the rendering process require significant computing resources and time to train models. However, with the development of cloud computing and more powerful graphics processors, there are new opportunities to further improve the optimization process.

The main goal of image optimization in 3D graphics is to reduce the load on the graphics processor (GPU) and the central processor (CPU), while maintaining maximum image quality. This allows you to achieve smooth animation, avoid delays and ensure high quality graphics in real time.

A polygonal model is the basis for representing 3D objects. Polygons are usually represented by triangles that are joined to create surfaces. Optimization consists in reducing the number of polygons without losing the detail of objects.

The formula for the area of a triangle based on the coordinates of the three vertices should look like this:

$$S = \frac{1}{2}|(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)|$$

where $(x_1; y_1)$, $(x_2; y_2)$ and (x_3, y_3) are the coordinates of the vertices of the triangle .

Algorithms for reducing the number of polygons (mesh simplification), such as Quadric Error Metrics (QEM), which calculate the error between the original and simplified models.

Textures are used to add detail to 3D objects. Reducing the size of textures is an important optimization step, as it allows you to reduce the amount of video memory (VRAM) used.

Texture compression and optimization are important elements in 3D graphics that directly affect rendering performance and memory usage. Textures are two-dimensional images superimposed on the surface of 3D objects to give them a more realistic appearance. They are used to convey characteristics such as color, glossiness, normals, and reflection maps. Because textures are often high-resolution, their volume can significantly exceed the volume of the geometry of the model itself. This creates high requirements for video memory (VRAM), especially when working with large scenes or in real time. Hence the need for texture compression, which allows you to reduce their size without a significant loss of quality.

Texture compression is done using special algorithms such as DXT (S3TC), ASTC and the latest methods that use machine learning to reduce file sizes. For example, the DXT method uses block compression, which reduces texture size while preserving color and alpha channel data.

The texture compression formula looks like this:

$$R = \frac{S_{original}}{S_{compressed}}$$

where R is the compression ratio, $S_{original}$ is the size of the initial texture, $S_{compressed}$ is the size of the compressed texture.

Common compression methods include DXT (S3TC) and ASTC, which use specialized compression algorithms to preserve color data with minimal loss of quality.

For example, if the original size of the texture is 16 MB and the compressed size is 4 MB, the compression ratio is 4, which means a fourfold reduction in size. This significantly reduces memory usage and allows you to store more textures in video memory, improving rendering speed.

Texture optimization also includes the use of texture atlases – large images that combine multiple textures into a single image. This allows you to reduce the number of texture switches during rendering, which has a positive effect on performance, especially in large scenes. An important aspect is the correct placement of textures on the atlas and the minimization of empty space, which can be determined using the texture atlas efficiency formula:

The formula for the effectiveness of using a texture atlas:

$$E_{atlas} = \frac{\sum_{i=1}^{n} A_{texture\ i}}{A_{atlas}}$$

where E_{atlas} is the efficiency of the atlas, $A_{texture\ i}$ is the area and textures, A_{atlas} is the area of the entire texture atlas.

High efficiency means that more of the atlas space is used for useful information, minimizing resource wastage.

Compression and optimization are closely related to the mathematical modeling of polygons and the importance of optimization in 3D graphics. Because texture compression reduces the memory load, it frees up resources to handle more complex geometry. Optimized polygons and textures work together to achieve the balance between quality and performance that is critical for applications with high real-time demands, such as gaming and virtual reality (VR). Compression techniques help preserve detail and provide fast access to data, allowing you to achieve realistic rendering of complex scenes without overloading the GPU and maintaining high frame rates.

Lighting significantly affects the display quality of 3D scenes. Realistic lighting is achieved using global lighting (Global Illumination, GI), which takes into account the reflection of light from various surfaces. Physically correct rendering (Physically – Based Rendering, PBR) allows you to model materials that react to light as closely as possible to real ones.

Lighting and shaders are key elements in creating realistic images in 3D graphics. They are responsible for the interaction of light with the surfaces of objects, determining how colors, shadows, reflections and other lighting effects will look. Lighting is based on physical principles such as reflection and refraction of light to create visually believable scenes. At the same time, shaders are used to program these effects on GPU, providing detailed lighting processing for each pixel or vertex. This allows you to maintain a high level of image quality, even in complex scenes.

Lighting should be calculated using the Lambert model (for diffuse lighting):

$$I_{diff} = I_{light} \cdot \max(0, n \cdot l)$$

where I_{diff} – diffuse light intensity, I_{light} is the intensity of the light source, n is the normal vector to the surface, l is the direction vector to the light source.

For complex scenes, the lighting precalculation technique is used (light baking), which allows you to reduce the load on the system during program execution.

Lighting and shaders are closely related to the optimization of polygons and textures, as the correct use of shaders can significantly reduce the computational complexity of a scene, while still keeping it visually appealing. By using methods like normal maps, developers can reduce the number of polygons without sacrificing detail, which is important for high rendering performance. At the same time, texture compression preserves image quality while reducing memory usage, allowing more efficient use of computing resources for complex lighting effects. Ultimately, optimizing lighting and textures allows you to create more realistic and productive visual scenes, which is critical for applications with high demands on graphics quality and performance.

The improvement of rendering technologies is an important stage in the development of 3D graphics, as it directly affects the quality of scene display and the performance of graphic applications. With the development of hardware and image processing algorithms, new rendering methods appear that allow creating more realistic visual effects. One of these methods is real – time ray tracing ray tracing), which allows you to accurately simulate the behavior of light, including reflections, refractions and shadows. Unlike traditional rasterization, which renders objects using triangles, ray tracing models the path of each light ray through the scene, providing high realism.

Ray tracing allows you to model the interaction of light with the surfaces of objects, taking into account reflection and refraction. This allows for realistic effects such as reflections and refraction.

The basic ray tracing equation:

$$P(t) = O + tD,$$

where $P(t)$ is a point on the beam, O is the starting point (camera), D is the direction of the beam, t – parameter that determines the distance from the starting point.

For each pixel on the screen, the path of the ray passing through the scene is calculated, and its interaction with objects is determined, taking into account reflection and refraction. This allows you to create realistic effects such as transparent materials, mirror surfaces and soft shadows.

Hybrid rendering methods combine ray tracing with traditional rasterization methods, allowing you to maintain performance when rendering complex scenes. For example, rasterization can be used to display major objects in a scene, while ray tracing is only used to calculate the effects of global lighting, reflections, or refraction in narrow areas of the scene. This allows you to avoid fully modeling complex light interactions for each pixel and only calculate them when you really need to.

Another modern approach to rendering is voxel – based methods rendering and using deep learning rendering like DLSS (Deep Learning Great Sampling). Voxels, or volumetric pixels, represent 3D space in the form of a regular grid, which simplifies the calculation of lighting and collisions between objects. Voxel rendering can be used to simulate volumetric effects such as fog or smoke. At the same time, machine learning-based technologies such as DLSS enable higher rendering resolution by reproducing high-detail images based on lower-resolution input frames. The formula for estimating the resolution increase in DLSS is as follows:

$$S_{output} = f(S_{low}, \theta),$$

where S_{output} is the original high-resolution image, S_{low} – input image with low resolution, θ – parameters of the trained model.

This allows you to reduce the load on the GPU while maintaining a high frame rate and image quality.

The improvement of rendering technologies is directly related to the optimization of polygons, textures and lighting, as new methods allow more efficient use of computing resources. For example, thanks to ray tracing and hybrid methods, less detailed polygon meshes and optimized textures can be used, focusing calculations on lighting and reflection. Using machine learning to improve image quality allows you to achieve high resolution without significant memory and power costs. Thus, the latest rendering technologies provide maximum image quality with minimum resource consumption, which is critical for applications with high performance requirements, such as video games and virtual reality.

The use of machine learning (ML) to optimize 3D graphics is becoming more and more common, as it allows automating complex processes and reducing the load on hardware resources. In particular, neural networks help in the processes of image resolution enhancement, texture compression, lighting optimization, and automatic creation of LODs (levels of detail). This allows for high performance and realism without significantly increasing the number of polygons or memory needed to store textures. For example, NVIDIA's Deep Learning Super Sampling (DLSS) technology uses neural networks to reconstruct high-resolution images from low-resolution input frames.

The basic equation for neural network training used in DLSS can be represented as the minimization of the loss function:

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} (y_i - f(x_i, \theta))^2,$$

where $L(\theta)$ is the cost function, y_i are the real values, $f(x_i, \theta)$ are the predictions of the neural network, x_i are the input data, θ and are the parameters of the model.

With this approach, the model learns to transform low-resolution images into more detailed versions, allowing it to maintain a high frame rate even in demanding scenes.

Machine learning is also heavily used for texture compression, which is related to the topic of texture and polygon optimization. For example, the use of autoencoders allows you to reduce the size of textures without losing a significant amount of visual information. An autoencoder consists of two parts: an encoder that compresses the input texture to a reduced representation, and a decoder that reconstructs the texture.

Neural networks can be used to predict complex lighting effects or generate high-quality textures from low-resolution samples.

Modern cloud technologies allow performing complex calculations for rendering and optimization on remote servers. This greatly expands the possibilities for developers, providing access to powerful resources without the need to purchase expensive hardware.

Task distribution formula:

$$T_{total} = \frac{T_{task}}{N} + T_{overhead},$$

where T_{total} is the total execution time, T_{task} is the execution time of one task, N is the number of servers, $T_{overhead}$ is the overhead of task distribution.

Distributed computing allows you to significantly reduce the rendering time of complex scenes.

Mathematical modeling of polygons is the basis for creating three-dimensional objects in 3D graphics. Polygonal meshes consist of a large number of triangles that are combined to form the surfaces of objects. Each triangle is described by the coordinates of its vertices in three-dimensional space, which allows forming complex geometric figures. The mathematical models describing these grids use mathematics to represent space, calculating the normals, areas, and volumes of objects that are necessary for the correct interaction of light with the surface. An important part of this process is understanding how calculations affect system performance, as complex models can significantly slow down rendering.

The optimization of polygon meshes is directly related to the number of polygons that make up the object. The more polygons a model has, the more detailed it looks, but the more computing resources are needed to display it. Therefore, to reduce the load on the graphics processor and reduce the rendering time, developers use optimization techniques, such as reducing the level of detail (Level of Detail, LOD). These methods allow you to automatically change the number of polygons depending on the distance of the object to the camera, leaving a highly detailed model only when it is close to the viewer. Mathematical modeling in this case includes the calculation of curves and simplified geometries that maintain visual integrity with a minimum number of polygons.

Besides LOD, another method is to optimize meshes using simplification algorithms such as Quadric Error Metrics (QEM). QEM allows you to calculate the error that occurs when removing or replacing a triangle vertex with another, and based on this, remove insignificant polygons that do not affect the overall appearance of the object. The formula for calculating the quadratic error allows you to reduce the number of polygons, while maintaining the maximum approximation to the original shape. This approach is particularly useful for objects with a large amount of detail, such as video game characters or architectural elements, where it is important to preserve the aesthetic appeal of the object.

Mathematical modeling of polygons is directly related to the importance of optimization in 3D graphics, since the effective use of resources depends on the correct calculation and application of geometric models. Optimized models allow for high image quality without significant performance loss, which is especially important for applications with high real-time requirements, such as video games and VR/AR applications. The use of mathematical modeling methods in combination with optimization techniques provides a balance between detail and speed, creating the most comfortable and realistic experience for users, while reducing the load on the hardware.

The optimization of polygon meshes is directly related to the number of polygons that make up the object. The more polygons a model has, the more detailed it looks, but the more computing resources are required to display it. Therefore, to reduce the load on the graphics processor and reduce the rendering time, developers use optimization techniques, such as reducing the level of detail (Level of Detail, LOD). These methods allow you to automatically change the number of polygons depending on the distance of the object to the camera, leaving a highly detailed model only when it is close to the viewer.

An important aspect of such algorithms is the calculation of the visible area of the object:

$$A_{visible} = A_{model} \cdot \cos(\theta),$$

where A_{model} is the area of the object, θ is the angle between the normal to the surface of the object and the direction to the camera.

Besides LOD, another method is to optimize meshes using simplification algorithms such as Quadric Error Metrics (QEM). QEM allows you to calculate the error that

occurs when removing or replacing a triangle vertex with another, and based on this, remove insignificant polygons that do not affect the overall appearance of the object. The quadratic error formula for the vertex v has the form:

The formula for the quadratic error for a vertex is as follows:

$$Q(v) = v^T Q_v v,$$

where $Q(v)$ is the error matrix describing the surface distortion when simplified, and are v the coordinates of the vertex.

By calculating this error for each vertex, it is possible to determine which vertices can be removed or moved with minimal loss of geometric accuracy.

Mathematical modeling of polygons is directly related to the importance of optimization in 3D graphics, since the effective use of resources depends on the correct calculation and application of geometric models. Optimized models allow for high image quality without significant performance loss, which is especially important for applications with high real-time requirements, such as video games and VR/AR applications. The use of mathematical modeling methods, such as the calculation of the area of triangles and the use of error metrics, in combination with optimization techniques provides a balance between detail and speed, creating the most comfortable and realistic experience for users, while reducing the load on the hardware.

Distributed computing and cloud technologies play an important role in modern 3D graphics, as they allow complex calculations to be performed for rendering and processing of large volumes of data on remote servers, reducing local hardware requirements. This is especially true for resource-intensive tasks, such as rendering photorealistic scenes with ray tracing, simulating global lighting, and processing high-resolution textures. Distributed systems allow you to divide such tasks into several nodes that perform their parts of work in parallel, significantly speeding up the rendering process. This enables high performance and scalability, which is important for large animation studios, game companies and architectural renderings.

Cloud technologies such as Amazon Web Services (AWS), Google Cloud Platform (GCP) and Microsoft Azure, allow you to use powerful GPUs for distributed rendering in a remote environment. This allows small teams or individual developers to have access to powerful computing resources without investing in expensive hardware. For example, you can run physically correct rendering simulations or perform ray tracing on GPU – enabled virtual machines.

The high computational demands of ray tracing and machine learning methods can be reduced by distributed computing, which allows processing large amounts of data in parallel. For example, creating LODs (levels of detail) for a large number of models or generating highly detailed textures can be performed on several servers at the same time, which significantly speeds up the process. Cloud technologies also allow textures to be stored in high resolution and synchronized with local machines, ensuring optimal access to resources when rendering complex scenes. This allows developers to achieve high rendering quality by efficiently using available resources.

Conclusions. Thus, image optimization in 3D graphics is an important aspect of the modern digital industry. The use of advanced methods, such as texture compression, dynamic change of detail levels, hybrid rendering methods and machine learning, allows you to achieve high quality images with reduced hardware load. Further improvements in these technologies will help create even more realistic and immersive visual worlds for users.

The scientific result of the research is an improved technology for image optimization of 3D graphics.

The practical result of the work is recommendations regarding the optimization of 3D graphics images.

A further direction of research can be the assessment of the effectiveness of using software tools for optimizing images of 3D graphics.

BIBLIOGRAPHY:

1. Gu C., Lu X., Zhang C. Example-based color transfer with Gaussian mixture modeling. *Pattern Recognition* . Vol. 129. R r . 771-774, 2022. DOI: https://doi.org/10.1016/j.patcog.2022.108716

2. Starkova O., Bondarenko D., Hrabovskyi Y. Providing software support for economic analysis. *Technology Audit and Production Reserves,* 2023, No. 5 (2 (73)), pp. 34–39.

3. Hrabovskyi Y., Bondarenko D., Ushakova I. Usage of adaptive design technologies for the design of a web application for analysis of the efficiency of solar panels. Academic notes of TNU named after V.I. Vernadskyi. Series: Technical Sciences, 2024, Vol. 35 (74), No. 1, pp. 118-126.

4. Joern B., Peter C. Foresight and Design: New Support for Strategic Decision Making, She Ji. *The Journal of Design, Economics, and Innovation* . No. 6(3). pp. 408-432, 2020. DOI: https://doi.org/10.1016/j.sheji.2020.07.002

5. Martin R. Twenty challenges for innovation studies . *Science and Public Policy,* 2016, No. 43(3), pp. 432–450.

6. Hrabovskyi Y., Bondarenko D., Kobzev I. Improving the technology for constructing a software tool to determine the similarity of raster graphic images. *Eastern-European Journal of Enterprise Technologies* . 2024. No. 1(2 (127). pp. 16–25. DOI : https://doi.org/10.15587/1729-4061.2024.298744

7. Khoroshevska I., Khoroshevskyi O., Hrabovskyi Y., Lukyanova V., Zhytlova I. Development of a multimedia training course for user self– development. *Eastern-European Journal of Enterprise Technologies* . 2024. No. 2(2 (128). pp . 48–63.

8. Hood N. *Quality in MOOCs: Surveying the terrain.* Burnaby: Commonwealth of Learning, 2016, 40 p.

9. Hrabovskyi Y., Kots H., Szymczyk K. Justification of the innovative strategy of information technology implementation for the implementation of multimedia publishing business projects. *Proceedings on Engineering Sciences ,* 2022. No. 4(4). pp. 467–480. DOI: https://doi.org/ 10.24874/PES04.04.008

10. Ushakova I., Hrabovskyi Ye. Methodology for developing an information site with Workflow support for publishing articles. *Development management.* 2022. No. 20(3). Pr. 20–28. DOI: 10.57111/devt.20(3).2022.20-28

11. Ushakova I., Hrabovskyi Y., Bondarenko D. Modeling and selection of a distance learning system for a higher education institution based on the method of hierarchy analysis using the DSS. *Academic notes of TNU named after V.I. Vernadskyi. Series: Technical sciences* . 2023. Vol. 34(73). No. 2. P. 246-253.

REFERENCES:

1. Gu, C., Lu, X., Zhang, C. (2022) Example-based color transfer with Gaussian mixture modeling. *Pattern Recognition,* 129, 771-774. DOI: https://doi.org/10.1016/j.patcog.2022.108716

2. Starkova, O., Bondarenko, D., Hrabovskyi, Y. (2023) Providing software support for economic analysis. Technology Audit and Production Reserves, 5 (2 (73)), 34–39.

3. Hrabovskyi, Y., Bondarenko, D., Ushakova, I. (2024) Usage of adaptive design technologies for the design of a web application for analysis of the efficiency of solar panels. *Vcheni zapiski TNU named after V.I. Vernadskogo. Series: Tekhnichni nauki – Academic notes of V.I. Vernadsky TNU. Series: Technical sciences,* 35 (74), 1, 118–126 [in English].

4. Joern, B., Peter, C. (2020) Foresight and Design: New Support for Strategic Decision Making, She Ji. *The Journal of Design, Economics, and Innovation,* 6(3), 408–432 . DOI: https://doi.org/10.1016/j.sheji.2020.07.002

5. Martin, R. (2016) Twenty challenges for innovation studies. *Science and Public Policy,* 43(3), 432-450.

6. Hrabovskyi, Y., Bondarenko, D., Kobzev, I. (2024). Improving the technology for constructing a software tool to determine the similarity of raster graphic images. *Eastern-European Journal of Enterprise Technologies,* 1(2 (127), 16–25. DOI: https://doi.org/10.15587/1729-4061.2024.298744

7. Khoroshevska, I., Khoroshevskyi, O., Hrabovskyi, Y., Lukyanova, V., Zhytlova, I. (2024) Development of a multimedia training course for user self-development. *Eastern-European Journal of Enterprise Technologies*, 2(2 (128), 48–63

8. Hood, N. (2016) *Quality in MOOCs: Surveying the terrain.* Burnaby: Commonwealth of Learning, 40.

9. Hrabovskyi, Y., Kots, H., Szymczyk, K. (2022) Justification of the innovative strategy of information technology implementation for the implementation of multimedia publishing business projects. *Proceedings on Engineering Sciences ,* 4(4), 467–480. DOI: https://doi.org/ 10.24874/PES04.04.008

10. Ushakova, I., Hrabovskyi, Ye. (2022) Methodology for developing an information site with Workflow support for publishing articles. *Development management,* 20(3), 20–28. DOI: 10.57111/devt.20(3).2022.20-28

11. Ushakova, I., Hrabovskyi, Y., Bondarenko, D. (2023) Modeling and selection of a distance learning system for a higher education institution based on the method of hierarchy analysis using the DSS. *Vcheni zapiski TNU named after V.I. Vernadskogo. Series: Tekhnichni nauki – Academic notes of V.I. Vernadsky TNU. Series: Technical sciences,* 34(73), 2, 246–253 [in English].