UDC 004.93:004.92
DOI https://doi.org/10.32782/2663-5941/2026.3.1/02

Bondarenko V.S.
https://orcid.org/0009-0009-1117-4501
Kharkiv National University of Radio Electronics

Barkovska O.Yu.
https://orcid.org/0000-0001-7496-4353
Kharkiv National University of Radio Electronics

Trunov V.O.
https://orcid.org/0009-0009-0758-5228
Kharkiv National University of Radio Electronics

Havrashenko A.O.
https://orcid.org/0000-0002-8802-0529
Kharkiv National University of Radio Electronics

A METHOD FOR DATASET AUGMENTATION IN IMAGE SEGMENTATION TASKS BASED ON IMAGE SYNTHESIS

This work considers the problem of expanding training datasets for image segmentation tasks, which is particularly relevant given the high cost and labour-intensive nature of manually annotating pixel data. Traditional augmentation approaches based on geometric, photometric and occlusion transformations are compared with generative methods and image synthesis. Although traditional methods remain useful, they do not generate fundamentally new information and therefore have limited potential for increasing the diversity of the training sample. Generative methods and approaches based on image synthesis are analysed separately, and the feasibility of using three-dimensional scenes as a controlled source of realistic data with the possibility of automatically obtaining accurate annotations is justified. A method is proposed for the automated generation of annotated images for segmentation based on three-dimensional scenes, physically based rendering (PBR) of materials and lighting, as well as information accumulated in G-buffers during deferred rendering. Unlike the traditional approach using a custom stencil, the proposed method does not require direct modification of a particular scene and can be applied to scenes that have already been built, which significantly enhances its practical value. The implementation of the method is based on identifying subsets of pixels according to the values of the roughness, metallic and specular parameters, followed by their combination into a final segmentation mask. This approach allows the use of internal scene characteristics that are inaccessible when working solely with a ready-made two-dimensional image, and ensures a high level of control over the data generation process. The paper also presents a comparison of the contents of G-buffers from various engines, including Unreal Engine, Unity, CryEngine, Frostbite and RAGE, which confirms the universality of the general idea whilst highlighting the need to adapt the implementation to a specific environment. The results of experimental studies have shown that the implemented shader-based approach provides segmentation accuracy at a level close to that of the manual method, but significantly outperforms it in terms of processing speed. The segmentation time per frame was also significantly lower than that of the comparable YOLOv8-based approach. At the same time, it has been established that the main drawback of the method remains the lengthy preparatory stage involved in setting up the scene, materials and segmentation parameters. It is concluded that the proposed approach is a promising means of augmenting datasets for neural network segmentation systems, particularly in cases where real-world images are difficult to obtain, whilst three-dimensional models of objects can be created or utilised from existing libraries.

Keywords: dataset, augmentation, neural network, 3D-scene, deferred rendering, PBR.

Formulation of the problem. In computer vision tasks, the quality and size of the training dataset directly influence the model's performance. This is particularly true for image segmentation tasks, where each pixel must be accurately annotated. Creating such datasets is a labour-intensive and costly process, as it involves more than just acquiring images for training. It also requires significant human resources for manual data annotation. Consequently, there is a need for dataset augmentation methods that allow the number of training examples to be increased without a proportional rise in annotation costs.

The study is motivated by the growing need for high-quality, scalable methods for expanding training datasets for image segmentation tasks, which can reduce the costs of manual annotation, improve the generalisation ability of models, and ensure their effectiveness when annotated data is limited.

Analysis of recent research and publications. There are a significant number of approaches to expanding datasets. The most common are classical augmentation methods based on geometric and photometric image transformations, as well as modern methods for synthesising new data using generative models or procedural modelling. Each of these approaches has its own advantages, limitations and areas of application, which must be taken into account when constructing training samples for segmentation tasks (Figure 1).

Furthermore, different methods of dataset augmentation affect the statistical properties of the training data in different ways. Some approaches merely modify existing images whilst preserving their semantic structure, whereas others enable the generation of entirely new scenes and objects. This leads to varying degrees of data diversity and, consequently, differing levels of effectiveness when training segmentation models. Therefore, a systematic comparison of existing dataset augmentation methods is warranted.

The manual method, which involves acquiring real-world images and segmenting them manually by experts, remains a reliable and historically significant approach because it enables the creation of reference datasets for training and evaluating segmentation algorithms, although its high labour intensity and significant time expenditure substantially limit dataset scale. Image modification, including photometric, occlusion [1], noise augmentation [2-3], and related methods, remains reliable because the image content is essentially preserved and re-labelling is usually unnecessary; however, such methods do not generate fundamentally new information, remain limited by the initial dataset, and may lead to unrealistic images or overfitting when used excessively. Generative augmentation, including GANs (Generative Adversarial Networks) [4], can produce new images, but it does not inherently provide pixel-level class information, which makes additional annotation or complex joint generation of images and masks necessary, while also tending to produce artifacts for rare or uncommon objects. In contrast, generating synthetic images based on three-dimensional scenes [5] is one of the most controllable methods of data synthesis for segmentation tasks, since it allows the construction of scenes with object models, light sources, and a camera, followed by rendering not only the final image but also segmentation maps, depth maps, normal maps, and other auxiliary data, while providing full control over scene parameters and enabling the creation of large and diverse datasets with automatically generated annotations.

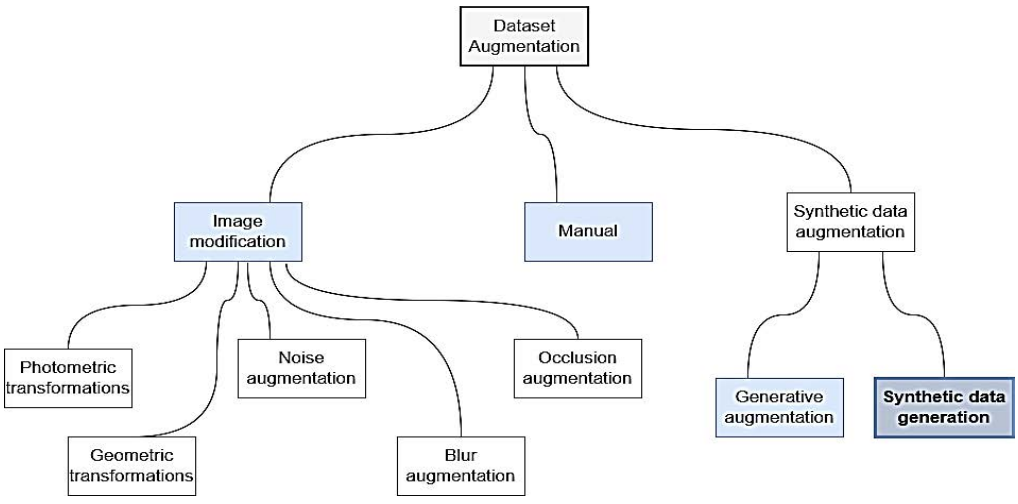

Fig. 1. Classification of methods for datasets augmentation

At the same time, building three-dimensional scenes requires a significant initial investment of effort. Creating high-quality models and scenes often necessitates the involvement of specialists from related fields, such as 3D artists or computer graphics engineers. It is also important to pay close attention to the accuracy of the scene construction: the configuration of materials, lighting, object scales and camera behaviour. An insufficient level of realism can lead to discrepancies between synthetic and real-world data.

Once a basic set of 3D scenes has been created, this approach proves highly efficient in terms of scaling data generation. Significant initial time is required for modelling objects and configuring materials, lighting and scene parameters. However, once this stage is complete, it becomes possible to automatically generate a large number of images with different scene configurations, object positions and camera parameters. At the same time, the corresponding segmentation masks can be generated automatically during the rendering process. Thus, the initial costs of preparing the three-dimensional environment are offset by the ability to rapidly and scalably generate significant volumes of annotated data.

Based on the calculations and estimates reported in [1-5], a comparison was performed. The results of this comparison are presented in Table 1.

Table 1

Comparison of different types of data augmentation methods

Method(s) type	Improving the accuracy of the dataset	What additional calculations are required
Image modification	0,3-1%	Minor computations that can be parallelised. Segmentation is performed on the input image
Generative methods	4-7,1%	Time taken to generate the image, time taken for segmentation
Synthetic images	3,9-6,5%	Segmentation and image generation are carried out in parallel
Manual	0-100%	Significant human effort is required for image acquisition and segmentation.

It follows that the unresolved aspects of the overall problem remain the simultaneous achievement of high accuracy, speed and scalability in the generation of annotated data for segmentation tasks. Manual annotation yields high-quality results but is overly labour-intensive and expensive; classical augmentation methods do not generate fundamentally new information and are limited by the source dataset; generative methods do not always produce accurate pixel masks and may generate artefacts. At the same time, data synthesis based on 3D scenes is promising, but requires significant initial expenditure on constructing a realistic environment and configuring materials, lighting and scene parameters. Consequently, the task of developing a controllable method for the automated generation of realistic images with accurate annotations, suitable for the effective expansion of segmentation datasets, remains unresolved.

Task statement. The aim of this research is to develop a method for the automated generation of annotated images for segmentation based on 3D scenes and deferred rendering G-buffers, and to evaluate its effectiveness in terms of segmentation accuracy, processing speed and practical suitability for expanding training datasets.

Such a method should not only demonstrate high quantitative performance in terms of accuracy and performance, but also overcome the practical limitations of existing approaches to generating annotated data. In particular, its application should involve not only data specialists, whose demand continues to grow, but also specialists in constructing 3D-scenes and working with 3D engines. Furthermore, the advantages of three-dimensional scenes over real-world images must be taken into account, as they allow for flexible control over the framing, the ability to change the viewpoint - including positioning it at a considerable height, below the surface or in other positions inaccessible to conventional filming, as well as providing access to all elements of the scene and their parameters in the system's memory. Unlike a real image, which, once captured, contains only the final visual result, a three-dimensional scene provides full control over its components, creating additional opportunities for the automated generation of accurate annotations.

Outline of the main material of the study.

Technological components of the 3D scene-based augmentation method. 3D scene-based augmentation is founded on three key technological foundations:

– physically based rendering (PBR);

– deferred rendering architecture;

– the availability of a large number of pre-existing realistic 3D scenes.

Such scenes are currently available both as commercial libraries and as open-source resources created by professional artists and computer graphics enthusiasts.

PBR enables the reproduction of physically consistent material properties and the interaction of light with surfaces, resulting in highly photorealistic images [6]. Deferred rendering, in turn, involves

storing a set of pixel characteristics in special buffers (G-buffers) during the first rendering pass [7]. These characteristics may include surface normals, material properties, scene depth, object identifiers and other parameters. The availability of this information makes it possible to generate additional representations of the scene, such as segmentation maps or other types of annotations. Since PBR relies on a unified approach to material definition, the buffer data associated with similar objects (e.g., plants) do not differ significantly.

Another key factor is the vast number of 3D scenes and models that have been accumulated over many years within the computer graphics, game development and visualisation industries. This allows existing resources to be utilised without the need to create an environment from scratch, which significantly lowers the barrier to entry for generating synthetic datasets. To illustrate this point, Figure 2 shows a bamboo grove scene created by Leo Torres using Unreal Engine 5.2, whilst Figure 3 shows a similar photograph taken by photographer Scott Kelby for comparison.

Fig. 2. Virtual bamboo grove

Fig. 3. Photo of a real bamboo grove

Given that these technologies have been actively developed for over a decade, the proposed approach can be regarded as an example of exaptation. According to Rich Geldreich [8], game developers recognised the potential of this technology as early as the early 2000s, although the technology itself is not new and was described by Michael Frank Dering in [7] as far back as 1988. In this case, tools originally developed for photorealistic visualisation tasks in computer graphics are being repurposed for a different purpose - the automated creation of annotated data for computer vision tasks. This approach allows us to build upon an existing technological foundation and utilise its potential in a related field.

Solution of the stated problem**.** This paper proposes a method for generating annotated images for segmentation tasks based on the use of three-dimensional scenes and information generated during deferred rendering (Figure 4). The method is based on using material parameters and other scene characteristics stored in G-buffers to automatically identify subsets of pixels that can be interpreted as image segments. It should be noted that this approach differs from the classical one, in which the target object is assigned a custom stencil pass and segmentation of such an object is performed automatically. However, the classical approach has a number of drawbacks. It requires access to the scene development process, whereas our approach, based on deferred rendering, can be applied to compiled scenes, as in [3]. Secondly, as it is based on PBR, the pre-configuration is performed not for a single specific scene, but for a single object across different scenes.

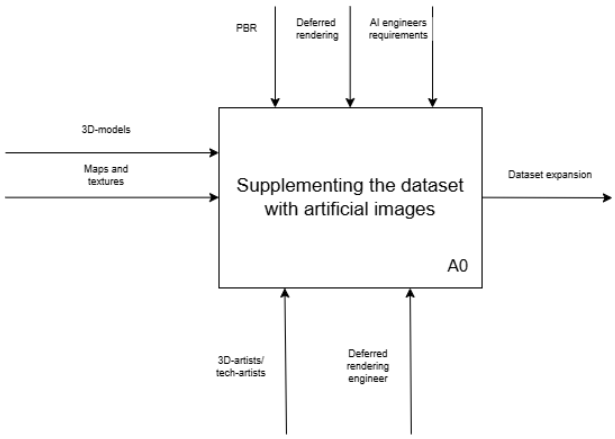

Fig. 4. Context diagram of our method

The research methodology consists of several sequential stages:

– in the first stage, a 3D scene containing the necessary objects, materials and lighting is constructed or selected;

– next, during the scene rendering process, a set of G-buffers is generated, containing material parameters and the geometric characteristics of surfaces. Based on the values of these buffers, subsets of pixels satisfying the specified material parameter ranges are

identified, after which they are combined in accordance with the described mathematical model;

– the final stage involves the software implementation of the proposed approach. This includes the development of appropriate shaders and the configuration of the rendering pipeline, which provide access to the necessary G-buffers and perform operations on their values.

A formal representation of the functional decomposition of the proposed method is shown in Figure 5.

The implementation of G-buffers - specifically, the information they contain - may vary across different engines. It is also worth noting that the term 'G-buffer' is not always used in the documentation, so as not to mislead the user when referring to an engine-specific implementation. For example, the RAGE engine distinguishes between the Velocity buffer and G-buffers. However, for simplicity, we will use the term 'G-buffer' to refer to the pixel information collected during the first pass of rendering.

Table 2 contains information on the differences in the contents of G-buffers across various engines, such as Unreal Engine, Cry Engine, Unity, RAGE and Frostbite, as described in [9-15]. The proprietary RAGE and Frostbite engines do not have public documentation, so we can only rely on presentations and reverse engineering.

Next, we consider the implementation of the proposed method in Unreal Engine. Let us take a closer look at the methodology represented by A2 in the diagram in Figure 5.

Let P be the set of all pixels in the image.

r(p) denote the value of the roughness buffer at pixel p ;

m(p) the value of the metallic buffer at pixel p ;

s(p) the value of the specular buffer at pixel p .

hen define the subsets of pixels whose values belong to the prescribed intervals. The subset of pixels whose roughness values belong to the interval $\left[r(a_i);r(b_i)\right]$ is defined by

Table 2

Differences in G-buffers across different engines

Engine	Unique information in G-buffers	Missing information in G-buffers
Unreal Engine	Per-object GBuffer data. Selective output mask	Classical world-space XYZ coordinates are not stored directly; instead, depth and transformation matrices are used
Frostbite	Smoothness. Material id	Alpha and material ID are packed into a single channel; radiosity and emissive color are also packed together
CryEngine	Baked AO. Subsurface scattering. Gloss/Smoothness	Depth and transformation matrices are used instead of explicit XYZ coordinates
Unity	MaterialFlags	Emission, lighting, lightmaps, and reflection maps are accumulated in a single channel; no explicit metalness
RAGE	Velocity buffer; includes several channels not assigned to fixed purposes and usable, for example, separately for hair	Does not always store the classical PBR parameter set in separate channels

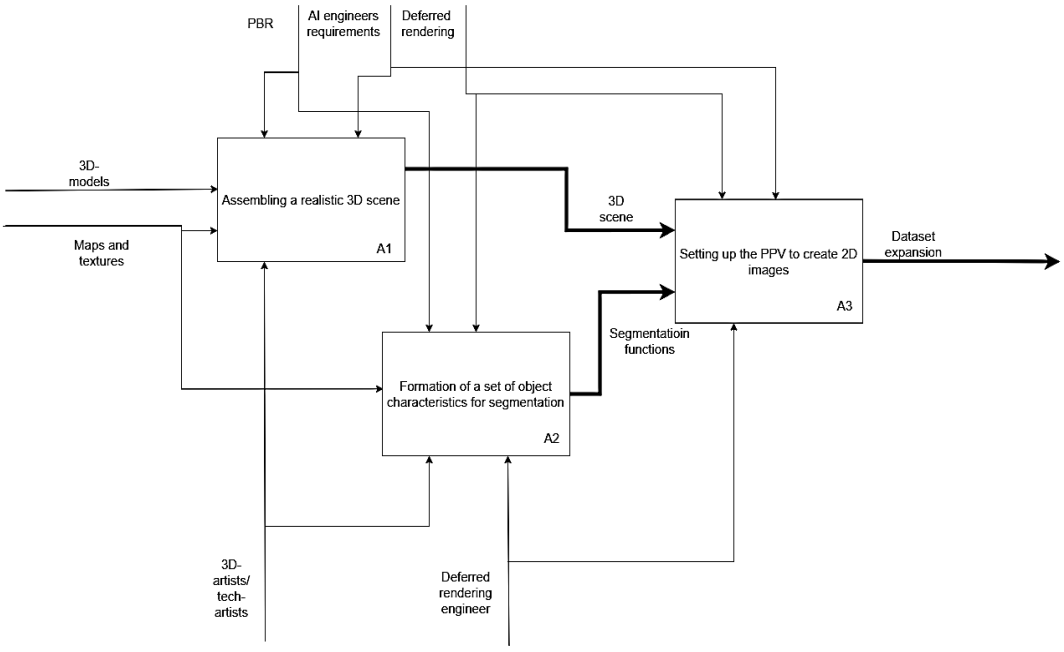

Fig. 5. Functional decomposition of our method

$$R_i = \{ p \in P | r(a_i) \le r(p) \le r(b_i) \}.$$

The subset of pixels whose metallic values belong to the interval $[m(c_i); m(d_i)]$ is defined by

$$M_i = \{ p \in P | m(c_i) \le m(p) \le m(d_i) \}.$$

The subset of pixels whose specular values belong to the interval $[s(e_i); s(g_i)]$ is defined by

$$S_i = \{ p \in P | s(e_i) \le s(p) \le s(g_i) \}.$$

It should be noted that deferred rendering uses a significantly larger number of G-buffers, including normals, depth, albedo, and others. However, for the experiments considered in this paper, it is sufficient to use only the three aforementioned buffers: roughness, metallic, and specular:

$$T_i = R_i \cap M_i \cap S_i,$$

where T_i is the subset of pixels for which all three conditions are satisfied simultaneously.

This set is intended to represent the set of pixels belonging to the particular object under study and to be segmented. At this stage, it is not necessary for T_i to cover all pixels of the given object. The essential requirement is that it should not contain pixels belonging to other objects. Next, we construct the family of sets $\{T_i\}_{i=1}^{n}$. Define the set $T = \bigcup_{i=1}^{n} T_i$.

In this case, the number n of subsets may be large in order to satisfy the stated conditions while covering all pixels belonging to the object. Increasing the number of such subsets often, though not necessarily, improves the accuracy of the proposed method.

Results of empirical studies. Discussion of the results. Table 3 presents the parameter values that define the intervals for constructing the sets introduced above.

By implementing a shader based on the defined sets, we obtain a frame with the segmentation already applied, as shown in Figure 6. This image also displays the G-buffers used, where 0 represents a black pixel and 1 represents a white pixel.

Once the method has been implemented, we obtain a large number of images to expand the dataset. Figure 7 shows some of them.

A comparative analysis of the performance of our method is presented in Table 4, based on the following criteria:

– segmentation accuracy;
– segmentation speed;
– preparation time.

Segmentation speed is measured in seconds. For other methods, it is assumed that the frame is already

Table

The intervals for constructing the sets

i	$r(a_i)$	$r(b_i)$	$m(c_i)$	$m(d_i)$	$s(e_i)$	$s(g_i)$
1	0	1	0,01	1	0	1
2	0,3	1	0	1	0,45	0,55

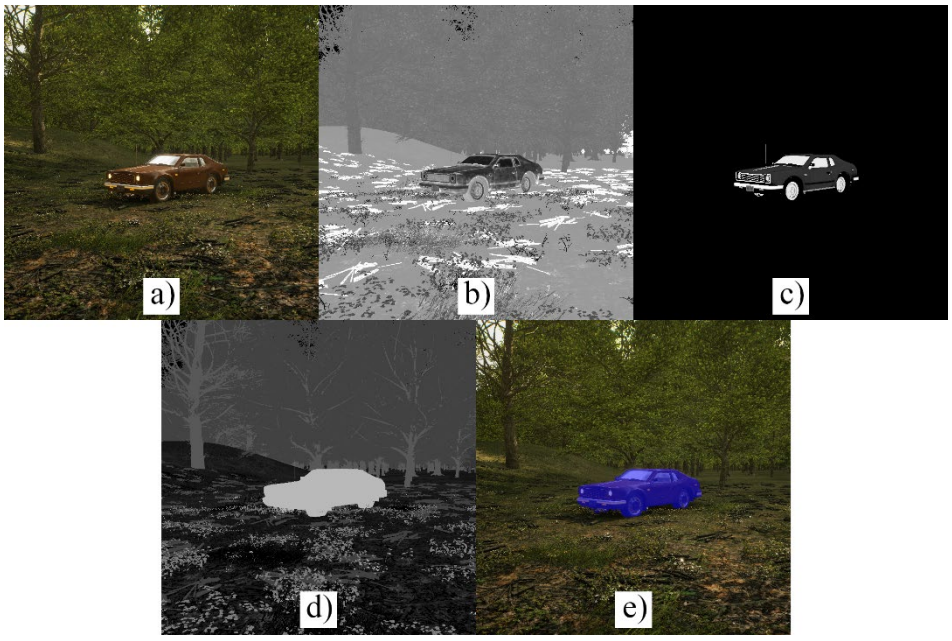

Fig. 6. Frame pipeline: (a) original frame, (b) roughness buffer, (c) metallic buffer, (d) specular buffer, (e) segmented frame

Fig. 7. Generated frame with segmentation

prepared for segmentation, whereas in our method described in this paper, the time taken to generate the frame is also taken into account, given that segmentation and generation are performed in parallel.

Preparation speed, described in terms of fuzzy logic, includes only configuration and parameter selection and does not account for the time spent on software implementation. The measurements take the average value from 10 experiments conducted.

Table 4

Comparison of numerical parameters for dataset augmentation methods

	Our method	YOLOv8	Manual way
Segmentation accuracy	98,85%	88,53%	99.9%
Segmentation speed	0,067 c	1,52 c	40 c
Preparation time	Long	Very fast	Not required

As shown, our method achieves high segmentation accuracy, comparable to that of the manual approach, whilst significantly outperforming both the manual approach and YOLOv8 in terms of processing speed. This makes it effective for tasks where speed and result quality are critical. At the same time, the main drawback is the considerable length of the preparatory stage, which may complicate its practical implementation. Despite this, it can be regarded as a useful addition to existing dataset augmentation methods.

Conclusions. This paper proposes a method for the automated generation of annotated images for segmentation based on 3D scenes and deferred rendering G-buffers. Within the scope of the research, a segmentation method was developed using the roughness, metallic, and specular parameters stored in G-buffers and the construction of pixel subsets with their subsequent merging into a segmentation mask. Based on this model, a shader was implemented that ensures the automatic generation of a segmented image during rendering.

The results confirmed the effectiveness of the approach, demonstrating high segmentation accuracy and high processing speed, since computations are performed on the graphics processing unit by default. The approach also provides full control over scene parameters, which distinguishes it from methods based only on captured real-world images. Its main drawback remains the lengthy preparatory stage associated with scene setup and segmentation parameter adjustment. Despite this limitation, the method is practically suitable for expanding training datasets and may be especially useful for objects whose 3D models are relatively easy to create, while real-world images are difficult to obtain or are not publicly available.

Further research will focus on analyzing the improvement in neural network accuracy achieved when training uses images generated by the proposed method, comparing the results with those obtained using other methods, and assessing the complexity of implementing this method for more complex and realistic scenes.

Bibliography:

1. Cubuk E. D., Zoph B., Mane D., Vasudevan V., Le Q. V. Autoaugment: Learning augmentation strategies from data. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2019. DOI: https://doi.org/10.1109/CVPR.2019.00020
2. Cubuk E. D., Zoph B., Shlens J., Le Q. V. Randaugment: Practical automated data augmentation with a reduced search space. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops.* 2020. DOI: https://doi.org/10.1109/CVPRW50498.2020.00359
3. DeVries T., Taylor G. W. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv*:1708.04552. 2017. DOI: https://doi.org/10.48550/arXiv.1708.04552
4. Frid-Adar M., Klang E., Amitai M., Goldberger J., Greenspan H. Synthetic data augmentation using GAN for improved liver lesion classification. *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018).* 2018. DOI: https://doi.org/10.1109/ISBI.2018.8363576
5. Richter S. R., Vineet V., Roth S., Koltun V. Playing for data: Ground truth from computer games. European Conference on Computer Vision. *Cham: Springer International Publishing,* 2016. DOI: https://doi.org/10.1007/978-3-319-46475-6_7
6. Cui J. Comparative Analysis of PBR and NPR: Technical Pathways, Resource Investment and Applicability. *Science and Technology of Engineering, Chemistry and Environmental Protection.* 2025. Vol. 1, No. 1. DOI: https://doi.org/10.61173/brg0ar49
7. Deering M. et al. The triangle processor and normal vector shader: a VLSI system for high performance graphics. *ACM SIGGRAPH Computer Graphics*. 1988. Vol. 22, No. 4. P. 21–30. DOI: https://doi.org/10.1145/378456.378468
8. Geldreich R. The Early History of Deferred Shading and Lighting. Rich Geldreich. URL: https://sites.google.com/site/richgel99/the-early-history-of-deferred-shading-and-lighting (Accessed: March 22, 2026).
9. Richter S. R., Abu AlHaija H., Koltun V. Enhancing photorealism enhancement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2022. Vol. 45, No. 2. P. 1700–1715. DOI: https://doi.org/10.1109/TPAMI.2022.3166687
10. New Shading Models and Changing the GBuffer. Epic Developer Community. 07.12.2022. URL: https://dev.epicgames.com/community/learning/tutorials/2R5x/unreal-engine-new-shading-models-and-changing-the-gbuffer (Accessed: March 22, 2026).
11. Deferred Rendering Path in URP. Unity Documentation. Version 12.0.0. URL: https://docs.unity3d.com/Packages/com.unity.render-pipelines.universal@12.0/manual/rendering/deferred-rendering-path.html (Accessed: March 22, 2026).
12. Courrèges A. GTA V - Graphics Study. Adrian Courrèges. 02.11.2015. URL: https://www.adriancourreges.com/blog/2015/11/02/gta-v-graphics-study/ (Accessed: March 22, 2026).
13. Hüseyin. Graphics Study: Red Dead Redemption 2. Dear World. 19.06.2020. URL: https://imgeself.github.io/posts/2020-06-19-graphics-study-rdr2/ (Accessed: March 22, 2026).
14. Lagarde S. SIGGRAPH 2014: Moving Frostbite to Physically Based Rendering V3. Sébastien Lagarde. 14.07.2015. URL: https://seblagarde.wordpress.com/2015/07/14/siggraph-2014-moving-frostbite-to-physically-based-rendering/ (Accessed: March 22, 2026).
15. Deferred Rendering Techniques. CRYENGINE Documentation. URL: https://www.cryengine.com/docs/static/engines/cryengine-3/categories/1638401/pages/1605647 (Accessed: March 22, 2026).

Бондаренко В.С., Барковська О.Ю., Трунов В.О., Гаврашенко А.О. МЕТОД РОЗШИРЕННЯ ДАТАСЕТІВ ДЛЯ ЗАДАЧІ СЕГМЕНТАЦІЇ НА ОСНОВІ СИНТЕЗУ ЗОБРАЖЕНЬ

У статті розглянуто проблему розширення навчальних датасетів для задач сегментації зображень, яка є особливо актуальною через високу вартість і трудомісткість ручного анотування піксельних даних. Порівняно традиційні підходи до аугментації, засновані на геометричних, фотометричних та оклюзійних перетвореннях, генеративні методи та синтез зображень. Традиційні методи хоч і залишаються корисними, не забезпечують формування принципово нової інформації, а тому мають обмежений потенціал щодо підвищення різноманітності навчальної вибірки. Окремо проаналізовано генеративні методи та підходи, що базуються на синтезі зображень, і обґрунтовано доцільність використання тривимірних сцен як керованого джерела реалістичних даних із можливістю автоматичного отримання точних анотацій. Запропоновано метод автоматизованого формування анотованих зображень для сегментації на основі тривимірних сцен, фізично коректного моделювання матеріалів і освітлення (PBR), а також інформації, що накопичується в G-буферах під час відкладеного рендерингу.

На відміну від класичного підходу з використанням custom stencil, запропонований метод не вимагає прямого втручання в процес побудови конкретної сцени та може бути застосований до вже створених або скомпільованих сцен, що суттєво підвищує його практичну цінність. Реалізація методу ґрунтується на виділенні підмножин пікселів за значеннями параметрів roughness, metallic і specular з подальшим їх об'єднанням у підсумкову сегментаційну маску. Такий підхід дозволяє використовувати внутрішні характеристики сцени, недоступні при роботі лише з готовим двовимірним зображенням, і забезпечує високий рівень керованості процесом формування даних. У роботі також наведено порівняння вмісту G-буферів різних рушіїв, зокрема Unreal Engine, Unity, CryEngine, Frostbite та RAGE, що підтверджує універсальність загальної ідеї та водночас вказує на необхідність адаптації реалізації до конкретного середовища. Результати експериментальних досліджень показали, що реалізований шейдерний підхід забезпечує точність сегментації на рівні, що є близьким до мануального способу, але суттєво перевершує його за швидкістю обробки. Час сегментації одного кадру становить також значно краще за показники порівнюваного підходу на основі YOLOv8. Водночас встановлено, що основним недоліком методу залишається тривалий підготовчий етап, пов'язаний із налаштуванням сцени, матеріалів і параметрів сегментації. Зроблено висновок, що запропонований підхід є перспективним засобом доповнення датасетів для нейромережевих систем сегментації, особливо в тих випадках, коли реальні зображення важко отримати, а тривимірні моделі об'єктів можна створити або використати з наявних бібліотек.

Ключові слова: *датасет, аугментація, нейромережа, 3D-сцена, відкладений рендеринг, PBR.*

Дата першого надходження статті до видання: 24.03.2026
Дата прийняття статті до друку після рецензування: 23.04.2026
Дата публікації (оприлюднення) статті: 19.05.2026