

Received 15.06.2024	Research Article	JOTS
Accepted 14.07.2024		8/2
Published 20.07.2024		2024: 278-301

# Augmenting parametric data synthesis with 3D simulation for OCR on Old Turkic runiform inscriptions: A case study of the Kül Tegin inscription

*Eski Türk Runik Yazıtlarında OCR için Parametrik Veri Sentezini 3D Simülasyon ile Arttırma: Kül Tegin Yazıtı Üzerinde Bir Örnek*

*Светой памяти Д. Д. Васильева (1946-2021)*

Mehmet Oguz DERİN\*

(Seoul/Republic of Korea)

E-mail: mehmetoguzderin@mehmetoguzderin.com

Erdem UÇAR\*

Jena University (Jena/Germany)

E-mail: erdem.ucar@uni-jena.de

Optical character recognition for historical scripts like Old Turkic runiform script poses significant challenges due to the need for abundant annotated data and varying writing styles, materials, and degradations. The paper proposes a novel data synthesis pipeline that augments parametric generation with 3D rendering to build realistic and diverse training data for Old Turkic runiform script grapheme classification. Our approach synthesizes distance field variations of graphemes, applies parametric randomization, and renders them in simulated 3D scenes with varying textures, lighting, and environments. We train a Vision Transformer model on the synthesized data and evaluate its performance on the Kül Tegin inscription photographs. Experimental results demonstrate the effectiveness of our approach, with the model achieving high accuracy without seeing any real-world data during training. We finally discuss avenues for future research. Our work provides a promising direction to overcome data scarcity in Old Turkic runiform script.

Key Words: optical character recognition, Old Turkic runiform script, data synthesis, vision transformer model.

---

\* ORCID ID: 0000-0002-6264-3509.

\* ORCID ID: 0000-0002-0039-9619.

## 1. Introduction

The earliest Turkic writings in the Old Turkic runiform script appear in two variants: Orkhon and Yenisei. The Orkhon script, dating from the 720s, was found along the Orkhon River in Mongolia, while the Yenisei script appeared in eighth-century inscriptions in Khakasia, Tuva, and South Siberia. These scripts, used for memorials, grave stelae, border signs, and graffiti, have been discovered in regions such as Talas, Kazakhstan, the Altai region, and Xinjiang (Johanson, 2021: 402-411).

The Khaganate inscriptions are an important part of the Old Turkic runiform corpus, and among them, the K l Tegin inscription is a stele that, alongside Chinese, contains Old Turkic script text, which has been of essential importance to understanding Old Turkic script and context surrounding the Khaganate inscriptions.

Optical character recognition (OCR) has made significant strides in recent years, enabling the digitization and analysis of vast amounts of textual information (AlKendi et al., 2024). However, OCR for historical scripts remains an open challenge due to the scarcity of labeled data, including the Old Turkic runiform script (Poncelas et al., 2020). Annotated datasets for training OCR models are scarce, as manual transcription of texts with Old Turkic runiform script is a time-consuming and expertise-intensive process, and the visual complexity of the script, with its variations, diverse layouts, and background noise, poses difficulties for robust classification of graphemes.

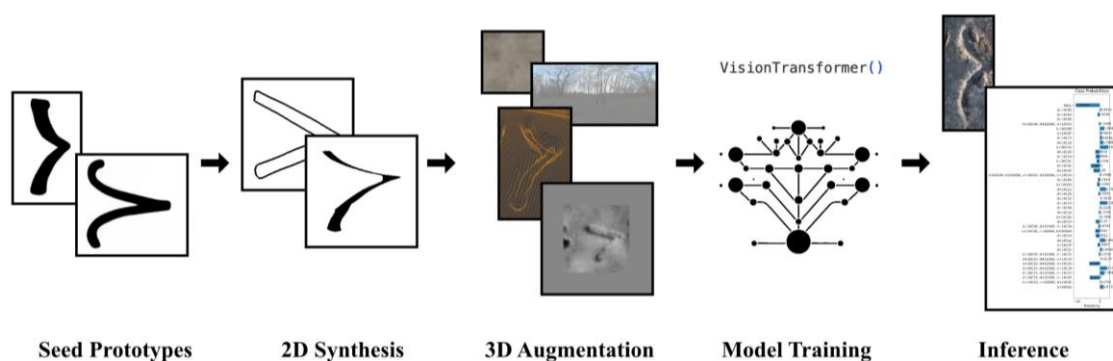


Figure 1. End-to-end pipeline.

To address these challenges, we propose a synthetic data generation approach for training machine learning models for Old Turkic runiform script. We verify the plausibility by evaluating a custom-trained classification model, establishing the first application of neural classification models with the attention mechanism to Old Turkic runiform script. Our key idea is to leverage 3D rendering techniques to create realistic images of Old Turkic runiform script with a wide range of variations in appearance, including lighting, texture, environment, noise, and artifacts, and then to test a Vision Transformer (ViT) model (Dosovitskiy et al., 2020) on real-world data with a focus on the Kül Tegin inscription.

Our main contributions are as follows:

1. A parametric model for synthesizing realistic Old Turkic glyphs based on a small set of prototype distance fields.
2. A 3D simulation and rendering pipeline that augments the parametric data with realistic geometry, textures, and lighting.
3. An application validating the effectiveness of the proposed approach through a ViT-based classification model that achieves high accuracy on photographs of the Kül Tegin inscription when trained purely on synthesized data.

## 2. Background and related work

Optical character recognition (OCR) has made significant progress in digitizing textual information, enabling large-scale analysis and preservation of historical documents (Mori et al., 1992; Liang et al., 2005). However, applying OCR to historical scripts (Ma et al., 2024; Martínek Jiří et al., 2020) like Old Turkic runiform remains challenging due to the scarcity of labeled training data and the visual complexity of the script (Uçar, 2024). Old Turkic runiform script was used across a wide geographical area in Eurasia (Robbeets & Savelyev, 2020). The script exhibits significant variations in character shapes, writing styles, and materials used, including stone<sup>1</sup>, paper, and wood, as well as degradations from

---

<sup>1</sup> In 2017, 30 Old Turkic runiform inscriptions in the Altai Mountains were digitalized using 3D technology to document their carvings in remote areas. Digital photogrammetry, utilizing affordable hardware and free software, provided high-quality results (Nevskaya et al., 2018: 194-216).

aging and exposure to the elements (Tekin, 1968). These factors make obtaining large annotated datasets needed to train robust OCR models difficult (Derin & Harada, 2021).

Data synthesis has emerged as a promising approach to overcome the data scarcity problem in many domains (de Melo et al., 2022). The key idea is to generate realistic synthetic data that captures real-world data's essential characteristics and variations, which can be used to train machine learning models. For OCR, this typically involves generating images of text with different fonts, styles, backgrounds, and degradations (Jaderberg et al., 2014). However, most existing OCR data synthesis approaches rely on simple 2D rendering techniques and fail to capture the full complexity of historical scripts like Old Turkic runiform script, which are often inscribed on 3D surfaces with intricate textures and lighting conditions.

3D rendering and simulation techniques from computer graphics offer a powerful way to generate realistic synthetic data that accounts for the 3D nature of historical texts. By modeling the text, materials, lighting, and environment in 3D and simulating the imaging process, one can generate synthetic data that more closely resembles real-world images. Such techniques have been successfully applied to generate training data for various computer vision tasks, including object detection, pose estimation, and segmentation (Tremblay et al. 2018). However, their application to OCR and historical script recognition has been limited.

Recent advances in generative models, such as Generative Adversarial Networks (GANs) (Goodfellow et al., 2020) and diffusion models (Ho et al., 2020), have shown impressive results in generating realistic images, including text images (Karras et al. 2019). These models learn to generate new images similar to a given training set, capturing complex styles and variations. However, training these models requires a large dataset of photographs and glyph-level annotations, often unavailable for historical scripts. Moreover, these models do not provide explicit control over the content and layout of the generated images, making it challenging to generate annotated data for downstream tasks. In contrast, 3D rendering and simulation approaches can generate realistic images

with precise control over the content, style, and degradations, making them well-suited for low-resource OCR data synthesis.

Regarding model architectures for OCR and classification, convolutional neural networks (CNNs) have been the dominant approach for the last decade, achieving state-of-the-art results on many benchmarks (Shi et al. 2016). However, CNNs are known to be sensitive to variations in the input image, such as changes in scale, rotation, and distortion. More recently, Vision Transformers (ViTs) have emerged as a promising alternative to CNNs for many computer vision tasks, including OCR (Ströbel et al. 2023). ViTs are based on the self-attention mechanism and can capture long-range dependencies in the input image, making them more robust to variations and distortions when enough training data is available.

### **3. Methodology**

#### **3.1. Overview of the proposed approach**

Our proposed approach consists of two main components: 1. A data synthesis pipeline; 2. A grapheme classification model for OCR. The data synthesis pipeline takes a set of glyph prototypes and generates a large dataset of realistic 3D renderings. The OCR system then trains on grapheme boxes extracted from this synthesized dataset and evaluated on a glyph-level image dataset from photographs.

#### **3.2. Data synthesis pipeline**

##### **3.2.1. Signed-distance field representation**

We represent each Old Turkic glyph as a signed distance field (SDF), which encodes the distance of each point in a 2D grid to the nearest point on the boundary (Osher et al. 2004). SDFs have several advantages over binary masks or outlines for representing shapes:

##### **Smooth interpolation**

SDFs allow smooth interpolation between shapes, enabling continuous glyph style and weight variations.

### **Efficient rendering**

SDFs can render efficiently using sphere tracing, a ray marching technique that avoids the need for explicit triangulation of the shape boundary (Hart, 1996).

### **Compact representation**

SDFs can be stored as low-resolution grids or compressed using truncated signed distance fields (TSDFs), reducing memory requirements (Curless & Levoy, 1996).

To create the SDF for each Old Turkic glyph, we start with a high-resolution glyph outline from upscaled vectorization of an impressional table of Old Turkic script grapheme variants. We then compute the signed distance of each pixel to the outline. The resulting SDF becomes a 2D grid of floating-point values, where negative values indicate points inside the glyph and positive values indicate outside.

#### **3.2.2. Parametric variations**

To increase the diversity of the generated text images, we apply parametric variations to the base SDFs of each glyph. Some of the variations we consider include:

##### **Weight**

We vary the weight of the glyph by offsetting the SDF values by a constant amount, effectively growing or shrinking the glyph boundary.

##### **Outline**

We vary between drawing only the outline or with fill.

A set of random variables drawn from predefined distributions controls the parametric variations, allowing us to generate a wide range of glyph styles and shapes.



**Figure 2.** This figure displays the parametric variations generated using the data synthesis pipeline. From left to right: the glyph <b> variant with angular features, the glyph <A> variant showcasing a curved structure, the intricate glyph <G> variant, the bold and glyph <W> variant, and the distinct angular glyph <p> variant.

### 3.2.3. Random text generation

To generate completely random Old Turkic script text strings for rendering, the algorithm starts with an empty string and iteratively adds characters without any preconditions. We apply additional constraints and heuristics during sampling to ensure that the generated text covers various characters and ligatures. For example, we may require that each generated string contains at least one instance of each Old Turkic script grapheme.

We do not use lexicons or language models to generate Old Turkic script text strings, aiming to avoid time-period or region-specific biases. This prioritizes recognizing visual features, as research shows that the Old Turkic script was used across a wide geographical area over a long period of time, attesting to the development of the language, its dialects, and the script (Erdal, 1979).

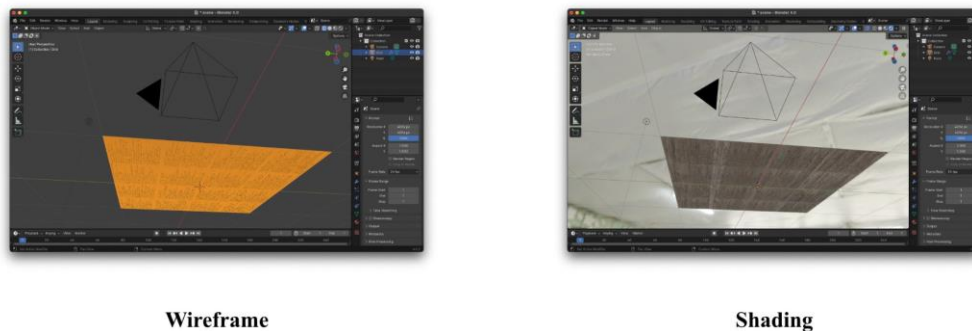
### 3.2.4. Texture mapping

To render the generated Old Turkic text strings in a realistic 3D environment, we first map them onto 3D surfaces using texture mapping (Heckbert, 1986). The surfaces mimic the characteristics of flat faces of stele inscriptions.

To map the text strings onto the 3D surfaces, we use UV mapping, which associates each point on the surface with a 2D texture coordinate. We generate

the UV coordinates using a parametric mapping function that considers the dimensions and orientation of the text string relative to the surface. The mapping function may also include random distortions and perspective effects to simulate the imperfections of inscriptions.

### 3.2.5. Lighting and environment setup



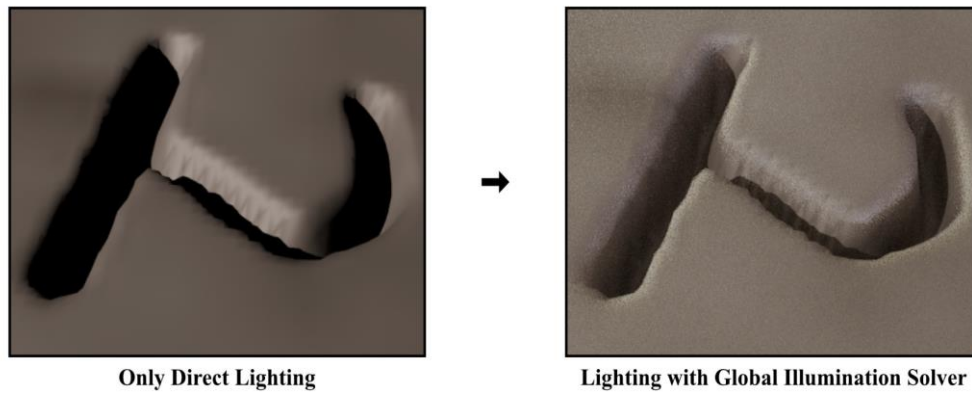
**Figure 3.** The figure shows a 3D simulation of an Old Turkic surface using Blender. The image on the left displays the wireframe view of the scene, highlighting the geometric structure and mesh of the text elements. The image on the right shows the shading view, where textures and lighting effects are applied to simulate the realistic appearance of the inscription in a natural environment.

We set up a virtual environment with one or more light sources and a background scene to create realistic lighting and shading effects. The design of the light sources mimics natural illumination conditions, such as sunlight or torchlight, with parameters by their position, intensity, and color (Akenine-Moller et al., 2019).

The design of the background scene provides visual context and realism to the rendered images (Debevec, 1998). We use a combination of 3D models and 2D textures to create backgrounds resembling Old Turkic inscriptions' natural environments, such as deserts, steppes, or mountains.



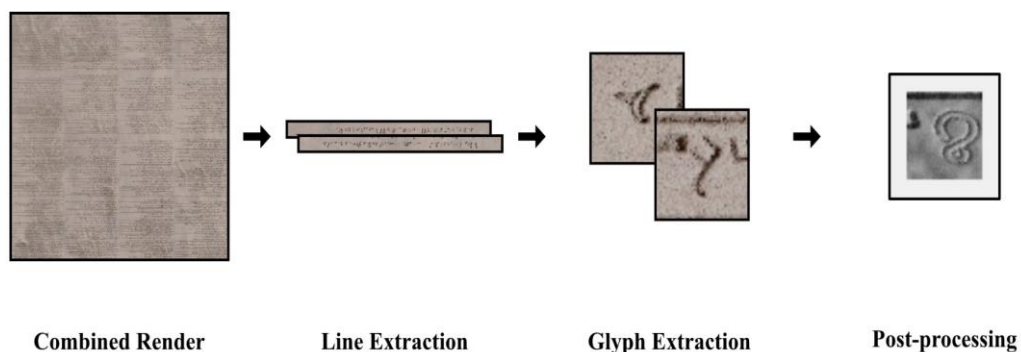
### 3.2.6. 3D Simulation



**Figure 4.** Comparison of rendering results using only direct lighting (left) versus a global illumination solver (right). The left image shows a flatter and less detailed rendering. In contrast, the right image demonstrates enhanced depth and detail.

We render the 3D scenes using a global illumination solver (Pharr et al. 2023). The renderer simulates the interaction of light with the scene elements using physically based shading models and ray tracing techniques. We set the camera parameters, such as focal length and sensor size, to match the full-frame digital single-lens reflex (DSLR) cameras used to document Old Turkic inscriptions.

### 3.2.7. Post-processing



**Figure 5.** Post-processing starts with a combined render of synthesized text images (Combined Render), followed by extracting individual lines of text (Line Extraction). Each line is further decomposed into

individual glyphs (Glyph Extraction), which are then processed to prepare them for classification (Post-processing). This pipeline facilitates the creation of realistic training data by simulating various effects.

The rendered images are then post-processed to introduce various artifacts and degradations that mimic the challenges of original Old Turkic inscription images. Some of the post-processing steps we apply include:

### Color distortions

We apply random color balance, saturation, and contrast adjustments to the images to simulate lighting and camera settings variations.

### Compression artifacts

We compress the images using JPEG or other lossy compression algorithms to simulate the degradation caused by image storage and transmission (Xia et al., 2009).

Random variables drawn from predefined distributions control the type and severity of the post-processing effects, allowing us to generate a wide range of realistic degradations (Buslaev et al., 2020).

## 3.3. Recognition system



**Real-time Recognition with Remote Camera Stream**

**Figure 6.** Real-time recognition of Old Turkic runiform script using a Vision Transformer model trained on synthetic data. The image shows the model accurately predicting a character from a remote camera stream, utilizing OpenCV for video capture and processing. The green bounding box highlights the detected character, and the prediction label displays the identified grapheme, demonstrating the model’s robustness in dynamic environments.

We employ a Vision Transformer (ViT) architecture for the classification model, which has shown promising results in various computer vision tasks. The ViT model inputs character images and predicts the corresponding class. We train the model using a cross-entropy loss function and the Adaptive Moment Estimation with Weight Decay (AdamW) (Loshchilov & Hutter 2019) Stochastic Gradient Descent (SGD) optimizer. The model architecture and hyperparameters are selected based on empirical evaluation and validation from a held-out subset of synthesized data.

## 4. Experiments

### 4.1. Implementation details


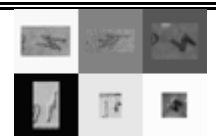

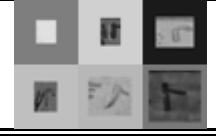
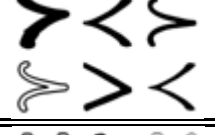
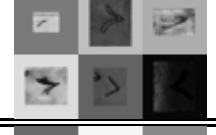

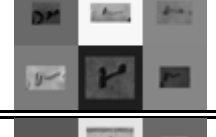

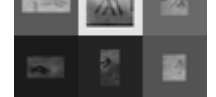
The 2D synthetic data generation pipeline is implemented in Python, using the NumPy (Harris et al., 2020) and SciPy libraries for numerical computations and the OpenCV library (Bradski et al., 2000) for image processing. This pipeline takes the prototype graphemes, converts them to distance field representation, and the positions as lines within 512 x 512 images. These images are then combined into 4096x4096 larger images to utilize scarce physically-based scan materials better. The pipeline then parallelizes using the multiprocessing module to take advantage of multiple CPU cores.

To generate the final training images through simulation, we render the 3D scene using the photorealistic global illumination renderer Cycles of Blender through the Python scripting API (Blender Foundation, 2024). We use a physics-based shading model with randomized material properties. The rendering parameters are sampled randomly within predefined ranges to generate diverse images. We use image-based input for both environment lighting and material configuration. The ground-truth label for each rendered image derives from keeping track of the 2D character bounding boxes in perspective projection. The simulation experiments execute on a system with NVIDIA RTX 4060 Laptop GPU and 32 GB of RAM.

We implement our model and training pipeline using the PyTorch deep learning framework (Paszke et al., 2019) and PyTorch Lightning (Falcon & The PyTorch Lightning team, 2019). Our ViT model takes a 64x64 grayscale image as input and predicts a 40-dimensional output vector corresponding to the 40 Old Turkic character classes (39 graphemes and one null class). The number of

output classes is a fixed hyperparameter of the model, but adding more classes through transfer learning is possible. The image is first split into 4 x 4 patches. Then, each patch linearly projects to a 128-dimensional embedding. The patch embeddings are concatenated with a learnable class embedding and fed into a stack of transformer encoder layers. Each encoder layer consists of multi-head self-attention followed by a feed-forward network. The average of slightly rotated final encoder outputs corresponding to the class embedding becomes the source to predict the character class. We train the model using the AdamW optimizer with a batch size 1536 and a learning rate 0.001. We use cross-entropy loss as the training objective. The model trains for 16384 steps. During training, we monitor the character accuracy on a held-out validation set to avoid overfitting. The classifier experiments run on a system with an NVIDIA A100 GPU and 64 GB of RAM.

#### 4.2. Dataset preparation<sup>2</sup>

ID	Seed Examples	Render Examples	Unicode Decomposition	Transliteration
r00	∅	∅	∅	∅
r03			U+10C00	A
r04			U+10C03	I
r05			U+10C06	W
r07			U+10C06, U+0200D, U+10C03	w
r11			U+10C0B	b

<sup>2</sup> D. D. Vasilyev (1946-2021) was a significant figure in the field of Old Turkic runiform script palaeography. In his 1983 work, based on his doctoral thesis, he compiled a comprehensive and descriptive list of runiform glyphs, indicating the location of the inscriptions where each glyph was attested (95-148). We have also greatly benefited from his work in this section.

r12			U+10C0F	g
r13			U+10C13	d
r14			U+10C18	y
r15			U+10C1A	k
r16			U+10C20	l
r17			U+10C24	n
r18			U+10C3C	r
r19			U+10C3E	s
r20			U+10C45	t
r22			U+10C06, U+0200D, U+10C03, U+0200D, U+10C1A	wk
r26			U+10C09	B
r27			U+10C0D	G

r28			U+10C11	D
r29			U+10C16	Y
r30			U+10C1E	L
r31			U+10C23	N
r32			U+10C34	Q
r33			U+10C3A	R
r34			U+10C3D	S
r35			U+10C43	T
r37			U+10C06, U+0200D, U+10C34	<sup>w</sup> Q
r38			U+10C3D, U+0200D, U+002D9	§
r42			U+10C14	z
r43			U+10C22	m

r44			U+10C2F	p
r45			U+10C32	ç
r47			U+10C03, U+0200D, U+10C32	<sup>i</sup> ç
r48			U+10C03, U+0200D, U+10C34	<sup>i</sup> Q
r49			U+10C1E, U+0200D, U+10C43	<u>L</u> T
r50			U+10C23, U+0200D, U+10C16	Ñ
r51			U+10C23, U+0200D, U+10C32	<u>N</u> Ç
r52			U+10C23, U+0200D, U+10C43	<u>N</u> T
r53			U+10C24, U+0200D, U+10C0F	η
r57			U+0003A	:

**Table.** This table enumerates synthesized graphemes, detailing their ID, seed examples, render examples, Unicode decomposition, and transliteration. Although primarily extensional to work with larger OCR systems, the null class represents the case where the model is explicitly confident that it lacks a glyph it can classify as one of the Old Turkic runiform glyphs.

To evaluate, we prepare a dataset of 100 graphemes from photographs of Kül Tegin inscription captured under different lighting conditions and camera

angles. We manually annotate the images with character-level bounding boxes. All images in this dataset consist of the test set.

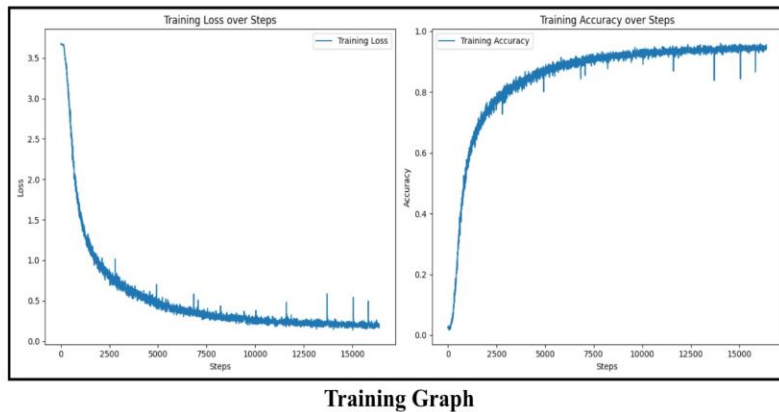
We generate a synthetic dataset of around 1 million grapheme images to train our models using the proposed 3D scene synthesis pipeline. The synthetic images cover all 39 distinct Old Turkic runiform script character classes in Kül Tegin inscription. The synthetic images are split into 95% for the training set and 5% for the validation set.

### 4.3. Evaluation metrics

We use character-level accuracy as the primary evaluation metric. It is defined as the percentage of characters that the system correctly classifies. We report the average accuracy across all test images.

## 5. Results

### 5.1. Quantitative results

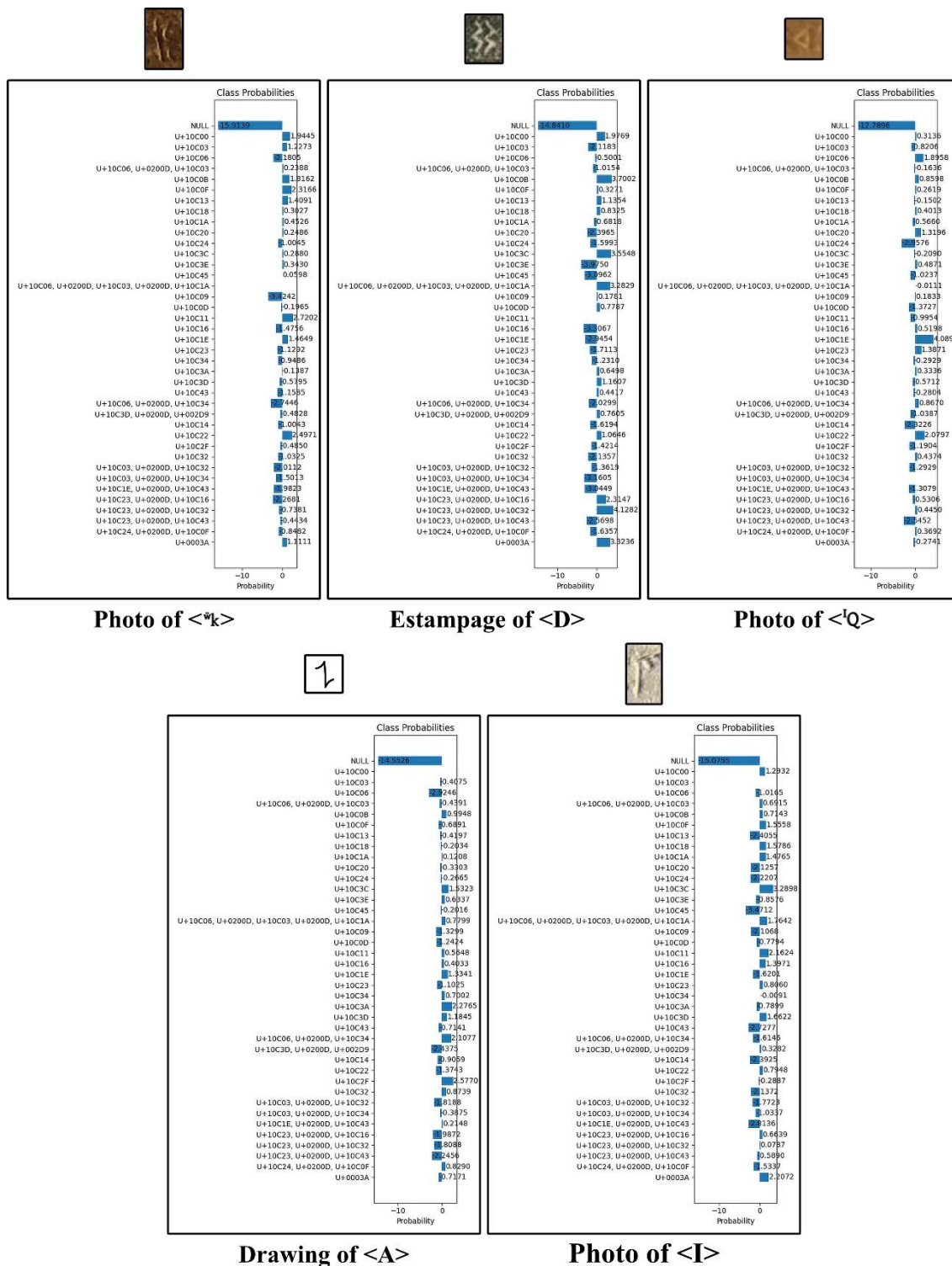


**Figure 7.** The figure shows the training progress of the Vision Transformer model used for Old Turkic script recognition. The left graph illustrates the training loss over the steps, while the right graph depicts the training accuracy over the same steps. The training loss decreases steadily, indicating that the model is learning effectively. Simultaneously, the training accuracy improves and stabilizes at a high value, demonstrating the model’s proficiency in recognizing graphemes from the synthesized data.

The figure in this section shows the accuracy of our approach as we train the Vision Transformer model. Our model, trained purely on synthesized data, achieves an accuracy of 95.6% in synthetic validation and 82% in photographs.



## 5.2. Qualitative analysis



**Figure 8.** Qualitative analysis of the grapheme classification model on photographs of the Kül Tegin inscription. The images depict various styles and conditions: photo of glyph <w>k>, estampage of glyph <D>, photo of glyph <Q>, drawing of <A>, and photo of <I>. The bar charts beside each image show the class probabilities predicted by the Vision Transformer model for each grapheme.

The figure in this section shows some qualitative examples of our model's predictions on photographs. The model accurately recognizes characters with various styles, lighting, backgrounds, variations, resolutions, and degradations. The qualitative results highlight the robustness and generalization ability of our approach. However, the model occasionally struggles with photographs that are significantly occluded, rotated, distorted, or taken from angles that do not directly face the camera.

## 6. Discussion

### 6.1. Key Findings and insights

The novel data synthesis pipeline proposed in this study, combining parametric generation of Old Turkic runiform glyphs with 3D scene rendering, effectively addresses the challenge of data scarcity in OCR on historical documents, focusing on the specific case of Old Turkic script. By generating realistic and diverse training data without manual annotation, this approach opens up new possibilities for digitizing and analyzing historical texts for which annotated data is scarce or unavailable.

The Vision Transformer model trained solely on the synthesized data achieves a high accuracy of 82% in classifying Old Turkic graphemes from real-world photographs of the Kül Tegin inscription. This result demonstrates the effectiveness of the proposed data synthesis approach in capturing the essential characteristics and variations of the historical script, enabling the model to generalize well to original inscription data. It highlights the potential of this approach.

The qualitative analysis of the model's predictions on photographs showcases its robustness to variations in character styles, lighting conditions, backgrounds, and image degradations. This suggests that the data synthesis pipeline successfully generates training data representative of the diversity and complexity of historical inscriptions. However, the model's performance on extreme cases of occlusion, rotation, distortion, or non-frontal camera angles indicates room for further improvement in the realism and comprehensiveness of the synthesized data.

While the proposed approach achieves promising results, it has several limitations that motivate future research directions. These include simplifying assumptions made in the data synthesis pipeline, ensuring greater efficiency and scalability, extending scene understanding tasks beyond character recognition, and evaluating and interpreting the model more comprehensively. Addressing these limitations could further enhance the practicality and generalizability of the approach for historical manuscript OCR.

## **6.2. Limitations and future work**

While our proposed approach demonstrates promising results for Old Turkic runiform script recognition using synthesized training data, there are several limitations to the current work that motivate future research directions.

### **Data synthesis pipeline assumptions and realism**

Our data synthesis pipeline makes several simplifying assumptions that may impact the realism and generalizability of the generated images. For example, the current system only produces flat surfaces with extrusion in glyphs without modeling surface-level degradations or diversities that are common in historical inscriptions. Incorporating more sophisticated approximations of real-world phenomena, such as modeling second-order distortions over edges or simulating weathering effects, could improve the realism of the synthesized data and the robustness of models trained on it. The current synthesis system also post-processes images to finalize them as monochrome grayscale images. Although this simplification is sufficient for the current classification task, future work should ensure that the model can handle cases where color is the only perceivable distinguishing factor between glyphs.

### **Scalability and efficiency**

Future work could explore techniques to optimize the data synthesis pipeline, such as using end-to-end GPU execution with ahead-of-time compilation (Lattner et al., 2021) through a type-safe language such as Rust with emerging compute-centric vector engines (Levien & Uguray, 2024) or leveraging transfer learning and domain adaptation techniques to reduce the amount of synthesized data needed. The recognition model's efficiency could be improved

through techniques like model compression, quantization, or architecture search (Choudhary et al., 2020).

### **Extending to scene understanding**

Our current approach focuses on character-level recognition. However, a complete OCR system for historical scripts like Old Turkic would also need to handle scene understanding tasks such as layout analysis and text line segmentation. The recognition model could also be extended to handle these additional tasks, potentially using multi-task learning or end-to-end architectures (Yousef & Bishop, 2020).

### **Evaluation and interpretability**

While our experimental results demonstrate the effectiveness of the proposed approach, there are several areas where the evaluation could be strengthened. Future work should include confusion matrices to analyze the types of errors made by the model and ablation studies to assess the impact of different components of the data synthesis pipeline and recognition model. Ablation and interpretability studies, including attention maps, could provide insights into the model’s decision-making process and help identify potential biases or failure modes (Chefer et al., 2021).

## **7. Conclusion**

In this paper, we proposed a data synthesis approach based on 3D rendering for training OCR models for Old Turkic script recognition. Our pipeline generates realistic and diverse training data by decomposing characters, applying variations, and rendering them in 3D scenes with simulation of global illumination phenomena. We trained a Vision Transformer model on the synthesized data. We evaluated it on photographs of Old Turkic inscriptions, achieving reasonable accuracy in classifying glyphs without using real-world data during training. Our results demonstrate the effectiveness of data synthesis for low-resource historical scripts such as Old Turkic runiform and open up avenues for future research in this direction.

## **8. Acknowledgments**

We are grateful to the teams creating and developing Blender, PyTorch, PyTorch Lightning, NumPy, OpenCV, SciPy, Poly Haven, Mermaid, and Penrose

for maintaining open-access and open-source tools and resources with permissive licenses that have been instrumental in our research.

We are also grateful to the first author's mother, Nazmiye Derin, for manually rebooting render server after unexpected power outages with remote instructions.

## References

Akenine-Moller, T. et al. (2019). *Real-time rendering*. London-New York: AK Peters/CRC Press.

AlKendi, W. et al. (2024). Advancements and Challenges in Handwritten Text Recognition: A Comprehensive Survey. *Journal of Imaging*, 10(1), 18.

Blender Foundation. (2024). *Blender - A 3D modelling and rendering package*. (Retrieved from [www.blender.org](http://www.blender.org))

Bradski, G. et al. (2000). OpenCV. *Dr. Dobb's Journal of Software Tools*, 3(2).

Buslaev, A. et al. (2020). Albumentations: fast and flexible image augmentations. *Information*, 11(2), 125.

Celso M. de Melo et al. (2022). Next-generation deep learning based on simulators and synthetic data. *Trends in Cognitive Sciences*, 26(2), 174–187.

Chefer, H. et al. (2021). Transformer interpretability beyond attention visualization. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 782–791.

Choudhary, T. et al. (2020). A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review*, 53, 5113–5155.

Curless, B. & Levoy, M. (1996). A volumetric method for building complex models from range images. *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (pp. 303–312). New York: Association for Computing Machinery.

Debevec, P. (1998). Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (pp. 189–198). New York: Association for Computing Machinery.

Derin, M. O. & Harada, T. (2021). Universal Dependencies for Old Turkish. *Proceedings of the Fifth Workshop on Universal Dependencies (UDW, SyntaxFest 2021)* (pp. 129–141). Sofia: Association for Computational Linguistics.

Dosovitskiy, A. et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. ArXiv Preprint ArXiv:2010.11929.

Erdal, M. (1979). The Chronological Classification of Old Turkish Texts. *Central Asiatic Journal*, 23(3), 151-175.

Falcon, W. & The PyTorch Lightning team. (2019). *PyTorch Lightning* (Version 1.4).

Goodfellow, I. et al. (2020). Generative adversarial networks. *Communications of the ACM*, 63(11), 139-144.

Harris, C. R. et al. (2020). Array programming with NumPy. *Nature*, 585(7825), 357-362.

Hart, J. C. (1996). Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10), 527-545.

Heckbert, P. S. (1986). Survey of texture mapping. *IEEE Computer Graphics and Applications*, 6(11), 56-67.

Ho, J. et al. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33, 6840-6851.

Jaderberg, M. et al. (2014). Synthetic data and artificial neural networks for natural scene text recognition. ArXiv Preprint ArXiv: 1406.2227.

Johanson, L. (2021). *Turkic*. Cambridge: Cambridge University Press.

Karras, T. et al. (2019). A style-based generator architecture for generative adversarial networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4401-4410). Long Beach, CA, USA.

Lattner, C. et al. (2021). MLIR: Scaling compiler infrastructure for domain specific computation. *2021 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)* (pp. 2-14). Curran Associates.

Levien, R. & Uguray, A. (2024). GPU-friendly Stroke Expansion (v2). ArXiv Preprint ArXiv: 2405.00127v2.

Liang, J. et al. (2005). Camera-based analysis of text and documents: a survey. *International Journal of Document Analysis and Recognition (IJ DAR)*, 7, 84-104.

Loshchilov, I. & Hutter, F. (2019). Decoupled Weight Decay Regularization. ArXiv Preprint ArXiv: 1711.05101.

Ma, H.-Y. et al. (2024). Reading between the Lines: Image-Based Order Detection in OCR for Chinese Historical Documents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(21), 23808–23810.

Martínek, J. et al. (2020). Building an efficient OCR system for historical documents with little training data. *Neural Comput. Appl.*, 32(23), 17209–17227.

Mori, S. et al. (1992). Historical review of OCR research and development. *Proceedings of the IEEE*, 80(7), 1029–1058.

Nevskaya, I. et al. (2018). 3D documentation of Old Turkic Altai runiform inscriptions and revised readings of the inscriptions Tuekta-V and Bichiktu-Boom-III. *Turkic Languages*, 22(2), 194–216.

Osher, S. et al. (2004). Level set methods and dynamic implicit surfaces. *Appl. Mech. Rev.*, 57(3), B15–B15.

Paszke, A. et al. (2019). PyTorch: An imperative style, high-performance deep learning library. ArXiv Preprint ArXiv: 1912.01703.

Pharr, M. et al. (2023). *Physically based rendering: From theory to implementation*. San Francisco: Morgan Kaufmann.

Poncelas, A. et al. (2020). A Tool for Facilitating OCR Postediting in Historical Documents. *Proceedings of LT4HALA 2020 - 1st Workshop on Language Technologies for Historical and Ancient Languages* (pp. 47–51). Marseille: European Language Resources Association (ELRA).

Robbeets, M. & Savelyev, A. (2020). *The Oxford guide to the Transeurasian languages*. Oxford: Oxford University Press.

Shi, B. et al. (2016). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11), 2298–2304.

Ströbel, P. B. et al. (2023). The Adaptability of a Transformer-Based OCR Model for Historical Documents. In M. Coustaty & A. Fornés (Eds.), *Document Analysis and Recognition - ICDAR 2023 Workshops* (pp. 34–48). Springer Nature Switzerland.

Tekin, T. (1968). *A Grammar of Orkhon Turkic*. Bloomington: Indiana University.

Tremblay, J. et al. (2018). Deep object pose estimation for semantic robotic grasping of household objects. ArXiv Preprint ArXiv: 1809.10790.

Uçar, E. (2024). A New Interpretation of Line 17 (I/South 10) of the Tuñuquq Inscriptions. *Zeitschrift Der Deutschen Morgenländischen Gesellschaft*, 174(1), 161–172.

Vasilyev, D. D. (1983). *Grafičeskiy fond pamyatnikov Tyurkskoy runičeskoj pis'mennosti Aziatskogo areala (opit sistematizatsii)*. Moskva: Izdatel'stvo "Nauka" Glavnaya Redaktsiya Vostočnoy Literaturı.

Xia, J. et al. (2009). Perceivable artifacts in compressed video and their relation to video quality. *Signal Processing: Image Communication*, 24(7), 548–556.

Yousef, M. & Bishop, T. E. (2020). OrigamiNet: weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 14710–14719). Seattle.