Efficient and user-friendly visualization of neural relightable images for cultural heritage applications

LEONARDO RIGHETTO, University of Verona, Italy
MOHAMMAD KHADEMIZADEH, University of Verona, Italy
ANDREA GIACHETTI, University of Verona, Italy
FEDERICO PONCHIO, ISTI-CNR, Italy
DAVIT GIGILASHVILI, NTNU, Norway
FABIO BETTIO, CRS4, Italy
ENRICO GOBBETTI, CRS4, Italy

We introduce an innovative multiresolution framework for encoding and interactively visualizing large relightable images using a neural reflectance model derived from a state-of-the-art technique. The framework is seamlessly integrated into a scalable multi-platform framework that supports adaptive streaming and exploration of multi-layered relightable models in web settings. To enhance efficiency, we optimized the neural model, simplified decoding, and implemented a custom WebGL shader specific to the task, eliminating the need for deep-learning library integration in the code. Additionally, we introduce an efficient level-of-detail management system supporting fine-grained adaptive rendering through on-the-fly resampling in latent feature space. The resulting viewer facilitates interactive neural relighting of large images. Its modular design allows the incorporation of functionalities for Cultural Heritage analysis, such as loading and simultaneous visualization of multiple relightable layers with arbitrary rotations.

CCS Concepts: • **Computing methodologies** → **Reflectance modeling**; **Graphics systems and interfaces**; • **Applied computing** → **Arts and humanities**.

Additional Key Words and Phrases: Multi-light image collections, RTI, relighting, neural representations

1 INTRODUCTION

The interactive examination of an object's shape and appearance is crucial across various application fields [41]. In the Cultural Heritage (CH) domain, interactive virtual inspection routinely substitutes, enhances, or supplements the examination of real objects for the general public and experts like scholars and conservators [49]. Due to their flexibility, relighting approaches, notably popularized by Reflectance Transformation Imaging (RTI) viewers [7], have emerged as one of the prevalent and effective methods in this context [41].

Relighting viewers operate on a 2D view of the targeted scene. Their defining characteristic is that, in addition to providing camera motion functionalities such as panning and zooming and employing stratigraphic rendering techniques to present multiple facets of the scene, they enable the manipulation of a virtual light source to

Authors' addresses: Leonardo Righetto, leonardo.righetto@univr.it, University of Verona, Strada Le Grazie 15, Verona, Italy, 37134; Mohammad Khademizadeh, mohammad.khademizadeh@studenti.univr.it, University of Verona, Strada Le Grazie 15, Verona, Italy, 37134; Andrea Giachetti, andrea.giachetti@univr.it, University of Verona, Strada Le Grazie 15, Verona, Italy, 37134; Federico Ponchio, federico.ponchio@isti.cnr.it, ISTI-CNR, , Pisa, Italy; Davit Gigilashvili, davit.gigilashvili@ntnu.no, NTNU, , Gyøvik, Norway; Fabio Bettio, fabio.bettio@crs4.it, CRS4, Cagliari, Italy; Enrico Gobbetti, enrico.gobbetti@crs4.it, CRS4, Cagliari, Italy.

ACM 1556-4711/2024/8-ART
https://doi.org/10.1145/3690390

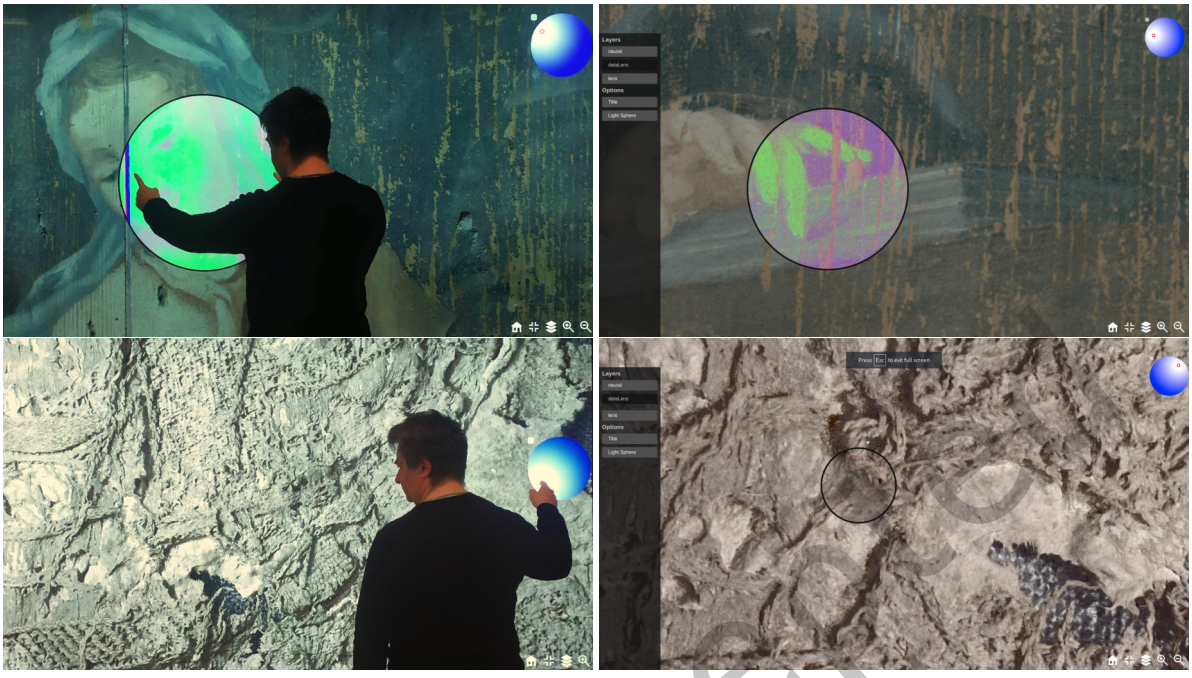


Fig. 1. **Interactive neural relighting.** Left: interactive exploration on a 98-inch multi-touch display connected to a desktop PC (4K, NVIDIA RTX 2080 Ti graphics). Right: interactive exploration on a desktop monitor connected to a laptop with integrated graphics (Full HD, Intel Coffee Lake GT2 graphics); Both applications use the same code executed in a web browser and allow the user to display multiple layers, change the illumination, and explore and visualize simultaneously multiple facets of the dataset using a visualization lens. The model in the first row is derived from a high-resolution MLIC capture of textile artifacts from the Oseberg Find from a Viking Age burial mound at Oseberg in South Norway. The model in the second row is of an organ door with an announcing angel, 17th century, belonging to the Diocesan Museum of Vicenza and reconstructed from a handheld-light MLIC capture in collaboration with the Accademia delle Belle Arti of Verona.

reilluminate the scene during inspection, thereby exposing subtle visual and geometric features of the inspected objects [4, 22]. The success of this approach is due to the ease with which such scenes can be acquired using a fixed camera and a variety of setups [41], as well as to several practical reasons that facilitate the deployment and usage of such viewers. First, many cultural objects (e.g., paintings, coins, bas-reliefs, or manuscripts) have a dominant viewing direction, and many others are naturally inspected from one or a few preferential views (e.g., many statues). Second, restricting camera control to 2D panning and zooming removes the complexity of 3D camera control, which is one of the main difficulties of 3D exploration applications, reducing learning curves [21]. Finally, a relighting interface supports a type of visualization that is very appropriate to inspect fine surface and appearance details and resembles the classical physical inspection with raking light sources to reveal surface detail of actual objects under study. For this reason, the interactive inspection of relightable models has been applied to a wide range of items and has shown to be very appropriate both for expert and casual users [41].

Such viewers can be implemented either through the rendering of a reconstructed, physically-based model of the object's shape and material properties (e.g., incorporating depth, normal, and BRDF) or by rendering novel images from a reconstructed reflectance field. The first approach, while in principle more powerful, is, however, practically limited to situations where an accurate model of the imaged object is not only available but can also

be reconstructed with the available measurements, thus restricting the applicability to only certain classes of objects and capture setups [41].

Moreover, even when such a model is available, producing realistic images necessitates non-trivial run-time illumination computation during rendering to consider non-local global illumination lighting effects and/or transparency and translucency. For this practical reason, the vast majority of relighting viewers employ Reflectance Transformation Imaging (RTI) methods that simply reproduce the acquired reflectance field without any separation into shape and reflectance components. Starting from a Multi-Light Image Collection (MLIC), i.e., a set of photographs acquired with a fixed camera under varying lighting conditions, such methods strive to compactly encode and interpolate acquired data using view-dependent per-pixel reflectance functions, allowing efficient transmission, storage, and generation of new images using any light direction in the hemisphere around the camera place.

While not as complete as a full, but hard to obtain, physics-based separate reconstruction of the shape and optical properties of the object, this solution guarantees a sufficiently realistic relighting independently on the material behavior and directly supports a variety of inspection processes. In particular, letting users control the direction and intensity of a single illumination source becomes natural and is the de-facto standard in current relighting viewers. The per-pixel representation of the reflectance function makes it, however, possible to support more complex illumination settings, including spotlights and local lights, by controlling per-pixel intensities and directions, as well as multiple lights, by summing individual contributions [41].

Many solutions have been proposed for modeling reflectance functions and fitting them to the input data generated by the acquisition methods that are mostly used in practice, such as fixed domes and free-form manual acquisitions [7]. Classic interactive solutions, especially in Web settings, still rely on low-frequency analytical representations, such as Polynomial texture mapping (PTM) or Hemi-Spherical Harmonics (HSH), due to their ease of integration in standard pipelines and good compression ratio [41]. However, the low-frequency approximation nature of these methods often fails to suitably represent the subtle illumination effects generated by the intertwining of complex local geometric and appearance characteristics [45].

In recent years, reflectance functions based on artificial neural networks have been shown to produce higher-quality images at similar storage costs, than classical analytical formulations (see section 2). However, their integration into interactive tools has so far been limited for several practical reasons. First of all, current approaches still have a higher computational cost, making the per-pixel evaluation on massive models and large pixel-count displays incompatible with interactivity constraints, primarily on commodity and mobile platforms. Moreover, their deployment in viewers typically requires the integration of libraries and frameworks for neural inference that require non-trivial programming efforts and complicate data communication with the rest of the graphics infrastructure (see section 2). This is particularly true for current web-based viewers relying on standard WebGL, HTML5, and JavaScript features to ensure portability.

In this work, we introduce our solution for integrating complex relightable representations into modern, scalable systems for interactively inspecting and reilluminating stratigraphic models. Following a brief overview of the related work (section 2), we discuss our approach for the efficient integration into a GPU-accelerated web-rendering architecture of a state-of-the-art neural reflectance model based on an asymmetric auto-encoder design [10]. The methods support the direct evaluation of the network into fragment shaders using standard WebGL features without relying on external libraries (section 3). Then, we discuss the integration of this solution into a multiresolution framework designed to manage multi-layered relightable models in web environments. Notably, this integration involves leveraging latent representation interpolation to create level-of-detail (LOD) approximations of the neural representation. This process is applied both offline to establish a discrete LOD hierarchy and at run-time to facilitate fine-grained adaptive time-critical rendering through resampling (section 4). The method's performance and capabilities are analyzed using both synthetic and cultural heritage models

(section 5), and its practical application is discussed in the context of the cultural heritage case study (section 6). We conclude by summarizing our findings and discussing future works (section 7).

This article extends our contribution to the *2023 Eurographics Workshop on Graphics for Cultural Heritage* [48]. It provides a more comprehensive exposition and includes significant new material on network design, comparisons of different loss functions, empirical validation of the latent space interpolation approach, qualitative and quantitative evaluations using real and synthetic data, and discussion of a real-world case study in cultural heritage.

The solutions introduced in this paper will be released as open-source software within the *OpenLIME* framework [37], a new open-source initiative focused on the web-based exploration of stratigraphic relightable models.

2 RELATED WORK

Adaptive web-based transmission and interactive exploration of relightable images are well-researched subjects. Due to the breadth of the literature, we discuss here only the approaches closely related to ours, referring the reader to established surveys[41, 52] for broader coverage.

2.1 Relightable image models and neural representations

Image-based relighting models store a description at each image location that is used in real-time to render image areas under a novel illumination while panning, zooming, and controlling light parameters. The approaches that disentangle shape and material information can generate physically based representations that can be integrated into high-quality renderers.

In MLIC settings, the most common decoupled representations combine normal and/or depth maps with per-pixel Spatially-Varying Bidirectional Reflectance Distribution Functions (SV-BRDF). The recovery of such a representation from MLIC through photometric stereo and BRDF fitting is non-trivial since a typical single-view multi-illumination capture only sparsely measures a single slice of the BRDF, forcing either multi-view capture, or the application of heavy priors [18, 40]. In recent years, the emergence of deep-learning solutions that recover hidden relations from large amounts of data has also led to effective methods that aim to extract 3D models, materials, and lights from single- or multi-view image-based acquisitions (e.g., [6, 13, 23, 27, 36, 58]). These learned priors are more flexible than the imposition of analytical constraints. However, the methods that perform an explicit decomposition into 3D geometry and analytical materials (e.g., [58]) blend well with traditional rendering engines but are restricted in the kind of objects that can be modeled (e.g., opaque ones), while more general models that directly synthesize new reilluminated views from a neural representation are more costly to evaluate than simple reflectance field approximations.

In general, despite their efficacy when applicable, the methods that decouple shape and material representations are challenging to generate from the commonly available sampled data, exhibit limitations concerning the range of object classes and material behaviors they can effectively capture, and require non-trivial rendering efforts to emulate the global illumination effects needed to replicate input images [18, 40].

For these reasons, most methods approximate the reflectance field by directly mapping lighting parameters to final renderable values, without explicitly separating shape and material information [41, 57]. Polynomial Texture Mapping (PTM) [31, 57], Hemi-Spherical Harmonics (HSH) [14], and Discrete Modal Decomposition (DMD) [43] are compactness and low complexity formulations of widespread use for fast interactive relighting in local and remote visualization. Without extra information, however, these methods are limited to model only low-frequency behavior [9]. Direct Radial Basis Function (RBF) interpolation of the original data has been proposed as an alternative to simple parametric functions [15], but the method requires run-time access to the original massive image stack and is not suitable for interactive relighting. The approach was later combined with Principal Component Analysis (PCA) compression of the image stack and RBF interpolation in light space to

improve efficiency at the cost of a slight reduction in quality [45]. In recent years, neural networks have emerged as a viable technique for compression and nonlinear approximation and interpolation tasks from large amounts of data and have also been applied to rendering settings [52]. Representative examples are light-transport-matrix interpolation [47] and deep relighting [56]. These approaches, however, were not directly applied in a MLIC setting. The NeuralRTI approach [10, 42] uses a fully connected asymmetric autoencoder to encode the original per-pixel information into a low-dimensional vector and to decode it to reconstruct pixel values from the pixel encoding and a novel light direction.

The image quality is higher than that of classic solutions at equal storage cost but the computational cost of inference makes interactive relighting difficult for large models and/or screen sizes. In this work, we tune the network and its training to improve the quality of the relighting while simplifying the decoding. To that end, we performed extensive tests to optimize the number and the size of the encoder/decoder layers as well as the loss used for the training. Moreover, we exploit the continuity of the latent representation to produce a multiresolution structure that can be efficiently encoded in web- and GPU-friendly formats and supports continuous levels of detail.

2.2 Integration of neural models in web viewers

Running deep-learning-powered Web applications in browsers has been the target of many recent efforts since the web environment promises to simplify cross-platform portability and can streamline cross-platform portability and cater to a diverse user base [29]. For this reason, several JavaScript-based deep-learning frameworks and libraries have been introduced recently. These include *ConvNetJS* [24], *WebDNN* [20], *Mind* [34] and *TensorFlow.js* [50]. However, these solutions strive to offer general-purpose ways to build entire applications or embed complex networks. While interoperability with existing rendering systems is feasible, it demands specific attention, especially concerning data processing and post-processing. Since relighting networks are typically tiny and simple in their structure (e.g., reduced feed-forward decoders), realizing them directly within the shader infrastructure simplifies the deployment of applications by reducing dependencies and makes it possible to streamline the data encoding and to efficiently integrate pre- and post-processing within a single shader pass. In this work, we present such an integration and shows how we can seamlessly integrate neural relighting within a general web framework supporting adaptive multi-resolution inspection and reillumination of large models. We also illustrate how such an approach simplifies the application of pre- and post-filtering without resorting to multipass rendering.

2.3 Interactive relighting tools

Many interactive exploration tools for flat but visually and geometrically rich models have been proposed and, sometimes, deployed for open, public use [41]. While some of these methods target static exploration of image data (e.g., multispectral or stratigraphic data [35] or multi-light image collections [30, 53]), the vast majority exploit specific compact reflectance models for supporting dynamic exploration through virtual relighting. Tools that support shape and material models, e.g., normals and BRDFs [22], make it possible to emulate realistic local lighting at negligible costs but are applicable only when such a representation exists or is reasonable to generate [1]. For this reason, the vast variety of tools exploits specific relightable image formats, such as PTM [31] and HSH [14]. While early software tools were designed for desktop use and locally resident data [7], web-based solutions have now emerged as a flexible means to support multi-platform exploration [41]. These tools typically use JavaScript, HTML5, and WebGL to interactively display the models and tailor the exploration experience to a variety of setups and displays [3, 46]. Web-based tools for image-based relighting, e.g., *WebRTIViewer* [39], *PLDWebviewer* [26], *Digital Materiality* [8, 12], and *Relight* [44, 45], typically support only specific parametric formulations of relightable images (in particular, PTM and/or HSH), and use them to provide interactive relighting

and some enhancement capabilities. In this work, we extend the open-source *OpenLIME* framework [37] to flexibly support neural relighting in addition to PTM, HSH, Normal and BRDF, and RBF formats. In addition to showing for the first time an effective integration of neural models through simple WebGL shaders into a multiscale tiled renderer, we also introduce adaptive resampling techniques that could be employed to ensure fine-grained adaptivity and time-critical results for other computationally costly techniques.

3 IMPROVED NEURAL RELIGHTABLE REPRESENTATION

Our relightable image representation follows the basic idea of NeuralRTI [10]. Independently for each pixel, the data captured in a multi-light image collection is used to train a fully connected asymmetric autoencoder mapping the original light-dependent per-pixel information into a low- dimensional vector and a decoder able to reconstruct pixel values from the pixel encoding and a novel light direction. In this section, we illustrate how we optimized the network for fast decoding (subsection 3.1) and how we efficiently encode the network and its per-pixel parameters for execution in a shader that computes per-pixel colors as a function of light direction (subsection 3.2). In the next section, we will detail how we build a tiled multiresolution format on top of the single-resolution representation described here and how we exploit the representation for adaptive time-critical rendering during interaction.

3.1 Network structure, decoding, and training

NeuralRTI [10] uses an asymmetric encoder-decoder structure. The encoder processes all the observations of a single pixel (N RGB tuples associated with the N sampled input light directions) with fully connected layers and ELU activation functions (Figure 2 top) and produces $K \ll 3N$ latent-space features. The decoder takes latent-space features, concatenates them with a given light direction, and produces the single associated RGB value for the pixel. The network is trained end-to-end on all the pixels (or a random subset of) the original MLIC by minimizing a pixel-based loss measuring the difference between predicted and ground truth pixel color. As a result, the network learns, per pixel, how to produce the color associated with light directions. We can approximate continuous relighting functions by suitably distributing input lights over the visible hemisphere.

Once the training phase is finished, it is possible to use the encoder to produce a final version of the latent features associated with the observations of each pixel and encode them in a map of per-pixel latent features. Afterward, the encoder is discarded, and relighted images can be computed just from the learned decoder's parameters (weights and biases of the decoder network), which are common for the entire image, and the per-pixel latent features, associating them to interactively-set light directions to produce the input of the decoder (Figure 2 bottom).

The major problem in the original NeuralRTI proposal is related to the computational complexity of the decoding step, making it not suitable for the interactive relighting of large images, typically required in Cultural Heritage applications.

In the NeuralRTI model, the total number of decoder weights (W) and biases (B) is calculated using the following expressions:

$$\begin{aligned} W &= (K+2) \times S + (S \times S) * (L-1) + S \times 3 \\ B &= L * S + 3 \end{aligned} \tag{1}$$

where K is the latent feature vector size, S is the size of the decoding layers (147 in the original model), L is the number of decoding layers (3 in the original model).

This means that the number of weights and related multiplications required to render the pixel color in the original model is larger than 45,000 and the number of biases and related additions is $B = 297$.

To simplify the decoding while not adversely affecting the quality of generated images, we performed extensive tests to reduce this complexity, testing the effects of changing both the number of fully connected layers and their

size. The outcome of these experiments is reported in section 5 and shows that a reasonable tradeoff is to use only two decoding layers of size $S = 50$. With this choice, with a latent feature vector of size $K = 9$, the number of weights and related multiplications is $W = 3{,}200$, and the number of biases and related additions is $B = 103$. The relighting software stores these parameters globally, which need to be loaded by the rendering system.

The $K = 9$ tent code values are, instead, stored per pixel and can be distributed as image layers after a quantization step. The number of latent features and the quantization method thus define the compression rate. For the results in this article, we used $K = 9$ as it has been shown that this very compact number of features is sufficient to provide a relighting with better quality than $3^r d$-order HSH encoding using 48 per-pixel parameters (see section 5). We currently perform feature quantization after training to store each latent code in 8 bits. In the future, we plan to include quantization directly in the training process to further improve accuracy.

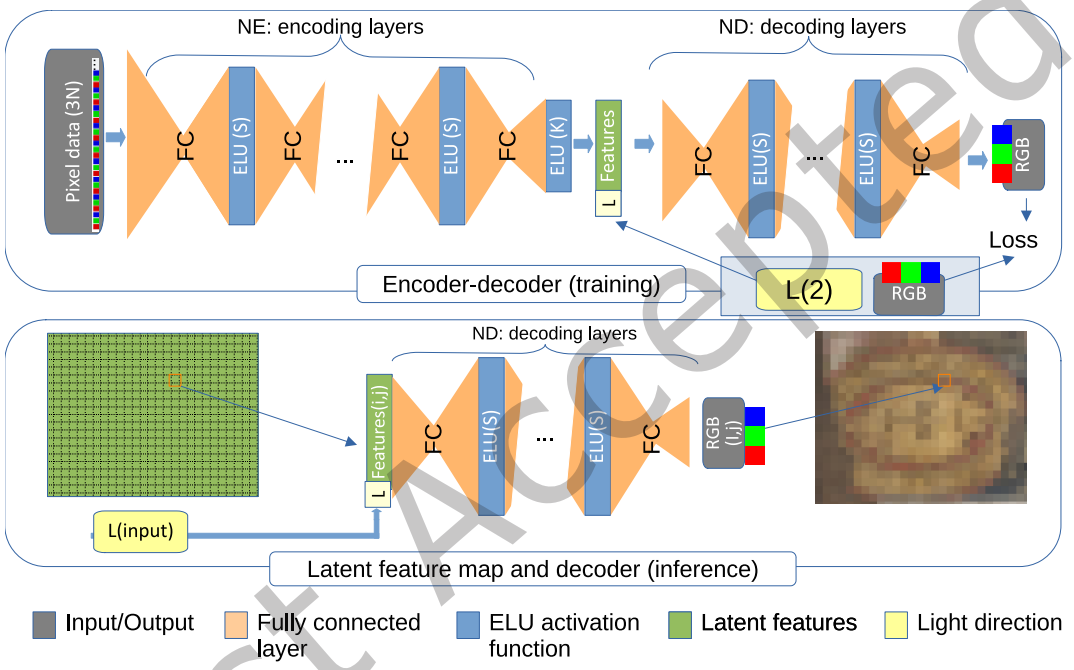


Fig. 2. NeuralRTI scheme. In the original implementation, the number of encoder layers ND was 3, and the decoder (top image, on the right) had NE=4 layers. We found that an optimal tradeoff for decoding size and quality is to set ND=4 and NE=3. The latent features array K is concatenated with the light directions vector L = $[L_x, L_y]$. At inference time, the encoder is replaced by a pre-computed latent feature map, and only the decoder needs to be executed.

Our tests revealed that we can safely remove a fully connected layer in the decoder without a relevant decay of the relight quality. We also verified that the size of inner layers can be set to the number of lights of a typical dome, e.g., $S = 50$, while a further reduction results in a non-negligible reduction of the

We maintained the small size of the latent representation $K = 9$ proposed in the original work, as it has been shown that this number of features is sufficient to provide a relighting with better quality than $3^r d$-order HSH encoding using 48 per-pixel parameters [10].

At the same time, we looked for improvements in the training procedure not affecting the decoding complexity. We found that the addition of an additional encoding layer provides a relevant improvement. We also tested the

use of different per-pixel loss functions. A possible choice is to adopt difference distance in the RGB space (L1 norm, L2 norm, Euclidean distance) or robust distances to avoid issues with outliers [2]. However, we limited the tests with linear or squared distances, as we need in our case to weight outliers to avoid losing accuracy for specular reflections. We also tested the use of Euclidean distances in a perceptually uniform space as CIE Lab. In subsection 5.1, we will show the outcomes of all the tests performed, suggesting that the use of the L1 norm provides better relighting results.

3.2 Practical decoding and custom fragment shader

While the NeuralRTI relighting has improved quality over standard relighting methods at similar or minor storage costs, this comes at the cost of a more complex online rendering procedure. A NeuralRTI real-time decoder must, in fact, perform the following calculations for each pixel: internal product between the layer's input and weights matrix, addition with biases vector, and application of the activation function.

We implemented a custom WebGL 2 shader to perform these tasks, avoiding external library dependencies and easing the integration with a rendering subsystem. The network sequence of operations is the following: sample the latent space feature using the pixel's coordinate, combine it with a given light direction, and perform forward propagation of these values in the network until we reach the final stage with the output color, each layer computing its output as a dot product of input and weights, a sum with biases, and an application of the activation function (ELU for inner layers, identity for the final one).

Fig. 3. Data is courtesy of Tomasz Łojewski (AGH University of Science and Technology, Kraków). Top row: samples from the MLIC (50 images in total). Bottom row: the 9 per-pixel latent features are quantized and stored in three parallel RGB textures. Image resolution: 1,900x1,900 px.

The latent space features are generated as a matrix of size $H \times W \times K$, where $H \times W$ is the resolution of the input images and K is the number of features saved for each pixel, that we set equal to 9 in the experiments. To

facilitate sampling this representation, we encode the features in a series of RGB textures. hus, the matrix is split into sub-matrices with dimensions $H \times W \times 3$, and values quantized to 8 bits, with associated per-channel slope/intercept values for floating point conversion. We will see in section 4 that we further split these textures into tiles, and build a multiresolution hierarchy, to enable adaptive rendering. Figure 3 shows the texture encoding of the latent feature map of a 256x256 portion of the dataset explored in Figure 1 left.

At the beginning of the shader, all the textures are sampled at the target pixel coordinates, and the result is stored in a vector of size $K + 2$, with the last two components filled with the light direction communicated to the shader in a uniform variable. This vector constitutes the input of the first network layer. Also, note that it would be possible to control per-pixel light direction through a light map.

The storage cost of the decoding network weights and biases, which must be transmitted once per image, is less than 3.5K floating point values for typical network configurations (Equation 1). We have thus decided to store it in uniform arrays of 4-component vectors (`vec4`). This grouping makes it possible to fit the full network within the limits on array size for uniform arrays and, most importantly, allows us to exploit alignment and vectorized dot products and sums to compute the input to the activation function. By padding arrays of size not multiples of four with zeros, we can handle any network configuration. The same padding to a multiple of four is performed on the $K + 2$ input vector (features+light direction), leading to three *vec*4 variables when $K = 9$.

Since all the pixels compute the same sequence of vectorized operations to produce the output pixel, and all dot products and sums are aligned on 16-byte boundaries, there is no divergence in the shading threads.

Even though the network is customizable with different numbers of features and layer sizes, a size-specific shader is produced at compile time, transforming all network configuration parameters to constants in the shader source. This permits the use of constant-size uniform arrays and fixed-size loops that can be effectively unrolled.

Moreover, we exploit the dataflow design of *OpenLIME* shaders [37] to combine NeuralRTI color computation with a sequence of post-processing operations, without the need for multipass rendering. In this design, the shading functions associated with each representation are not implemented in the `main()` function of the fragment shader, but as a shader function that produces a fragment color given their input. Filtering functions can then be appended to the shader, each one modifying the input fragment color to produce another one. The system generates a suitable `main()` function for the fragment shader that calls all the individual shading functions in the correct sequence. We use this, in particular, to perform gamma, brightness, color-remapping, and contrast correction on neural renderings using the same tools the framework makes available to all other rendering representations (Normal+BRDF, PTM, HSH, RBF).

4 SCALABLE EXPLORATION OF MULTI-LAYERED MODELS

Interactive exploration of large datasets on web platforms requires the usage of adaptive techniques to ensure sufficient low-latency and high-frequency feedback while coping with limitations in end-to-end bandwidth, memory available on client devices, and computation power. Multi-layered relightable models require additional resources both for the number of layers and the complexity and cost of the mathematical models used for rendering. At the core of these methods is the precomputation of levels-of-detail (subsection 4.1), their efficient storage and transmission (subsection 4.2), and their exploitation in adaptive rendering methods that maintain and render an adaptive working set (subsection 4.3). After discussing our approach's general concepts, we present its efficient implementation within the OpenLime framework (subsection 4.4).

4.1 Precomputed level-of-detail structure

Scalability on large planar or 2.5D datasets is usually achieved with pre-computation and storage of the source model in a pyramidal format, typically a quadtree [38]. To construct the pyramidal representation, the dataset is iteratively filtered and scaled by a factor of two, and each level of detail (LOD) is cut into small tiles. This allows

the exploration of arbitrarily large datasets with limited resources. By matching the tile resolution in the working set with the screen resolution, rendering can be achieved with just one sample per pixel at all scales, and the amount of resources (the number of tiles) required to be transmitted, locally stored, and processed is determined only by the size of the screen.

Originally developed for large image exploration, this approach is routinely employed for relighting models. The construction of the various levels of detail, however, requires repeated filtering of higher-resolution data to construct the coarser levels, as well as to interpolate among them to avoid restricting adaptivity to power-of-two resolutions. To avoid aliasing, this filtering cannot be a simple sub-sampling but has to suitably combine the values in the filter's kernel.

We can produce this representation either by zooming the images first and then performing the per-layer neural encoding or by producing neural representations only at the higher resolution and averaging nearby latent space features. In this work, we are directly performing resampling and filtering in latent space to simplify the implementation, using the same tools as for other representations (e.g., PTM or HSH). This choice allowed us to streamline the entire building operation, which uses the same code base for the entire construction of the tiled pyramid, only changing the per-pixel decoder. Even though filtering the resulting images or the source coefficients is mathematically equivalent for linear representations, the results slightly differ for the nonlinear NeuralRTI. As discussed in subsection 3.2, we empirically verified that this simplified approach does not negatively affect the quality of relighting. Moreover, we also exploit latent-space resampling and interpolation to construct continuous levels of detail from the original power-of-two pyramid, to achieve fine-grained adaptivity (subsection 3.2).

It should be noted that the coefficient interpolation approach has also been used for creating mipmaps for nonlinear representations, such as BRDFs (e.g, [22], which builds a pyramid over Ward parameters). While the results are already acceptable for the datasets we have used, we plan, for the future, to explicitly introduce constraints in the network to ensure the interpolability conditions at training time, as done, e.g., in generative networks for image interpolation [5, 28]

4.2 Web-friendly compression and decompression

In remote visualization, data compression is essential to cope with bandwidth constraints. In our case, the transmission cost is almost entirely due to the latent feature maps storing per-pixel data since the network structure, weights, and biases are shared for the entire dataset and amount to a few kilobytes (see section 3). For compressing the latent maps, the execution on a web platform encourages the usage of PNG, JPEG, or WebP, which are the formats natively supported in all common browsers [32]. As seen in Figure 3, the images resulting from the coefficient planes exhibit a structure similar to that of RGB photographic images, making JPEG a suitable choice for efficient compression of the image tiles that encode per-pixel latent-space features. However, the default parameters for JPEG compression are optimized for perceptually-aware quantization. For example, the compressor handles chroma and luminance at different sampling rates and strives to take advantage of the correlation among RGB planes. However, neural coefficients do not exhibit a direct or strong correlation with these characteristics in the final image (unlike, e.g., PTM). As a result, we have chosen to disable the default conversion to the YUV color space, deactivate chroma subsampling, and utilize non-biased quantization tables.

4.3 Adaptive rendering

The computational power required for interactive neural network relighting can easily become a bottleneck depending on the hardware available, especially over the web, where a wide variety of devices needs to be supported. Therefore, we optimized our server to meet time constraints. Since the computation of the final color, given the latent space features and a light direction, is the most costly operation, we can improve performance

by controlling the number of relighted pixels. All the optimizations exploit auxiliary memory (framebuffer) to render offline the tiles before presenting them on screen.

The first optimization exploits the decoupling between camera motion and shading under direct illumination. Since the final color is independent of camera motion, we can cache the relighted result and reuse already available shaded tiles when the camera moves. In particular, only a few newly entering tiles per frame need to be computed under panning.

The second optimization introduces time-critical features, in which we adaptively control the number of relighted pixels per frame. We first measure the time required to relight a few tiles and use this data to compute the number of pixels that can be relighted per second. By dividing this number by the preset frame rate (e.g., 30 fps), we obtain the relighted pixel budget. At each frame, we then dynamically adapt the resolution of intermediate textures to the available budget by setting the virtual rendering resolution to be the minimum between the viewport resolution and the resolution dictated by the available pixel budget. The input coefficients for each target pixel are then resampled from the closest available LOD. This LOD is the first one with a resolution higher or equal to the required resolution, if available, or the highest resolution parent, if not available. Bilinear filtering is used for sampling.

Image processing filters on PTM and HSH models have been shown to produce reasonable results, which is justified by the linearity in the per-pixel coefficients of the mathematical model. While the NeuralRTI representation is not linear, we have also empirically verified, in practice, that averaging nearby latent space features also tend to produce a pixel whose relighting behavior is similar to that of the averaged pixels (i.e., close to averaging the reflected colors at similar incident angles).

In subsection 5.2, we illustrate the results of an extensive benchmark, showing that the error obtained by resampling the latent features at the output resolution and then performing relighting instead of relighting at the original resolution and resampling the results is small, and can be considered negligible in real-world setting if compared with the errors stemming from the acquisition of the ground truth images and the effects of emulating light motion from a fixed number of light directions. The results obtained with the two relighting procedures are visually similar with no artifacts at the materials' edges.

4.4 Integration in a rendering framework

An experimental software library supporting the methods described in this work has been implemented using JavaScript and WebGL2 and integrated within the *OpenLIME* framework [37]. The basic features for neural rendering have been implemented in a specific shader class, while the features describing interactive adaptive rendering have been realized by modifying the overall rendering framework and are, therefore, available also to other rendering tools. The preprocessing features for the offline creation of image pyramids containing the neural coefficients were, instead, realized using Python and Keras for the fitting part and the creation of the full resolution representation, and *vips* [55] for the conversion to the multiresolution *deepzoom* format [33]. We employed a tile size of 256×256 with no overlap. Using the *tarzoom* utility provided by *OpenLime*, the directory tree containing all the tiles is then sequentially concatenated into a single file, augmented with an index that contains the start offset of each tile (and thus implicitly also its size). Having a single data file makes it possible to move the entire representation quickly among different machines and file systems, and supports very efficiently the extraction of individual tiles with simple range queries on a locally stored file (through *mmap*) or remotely (through any modern HTTP server).

4.5 Additional visualization options

As mentioned in section 6, *OpenLime* is well suited for visualization applications and the implementation of case studies. The original interface allows users to zoom and rotate the RTI image. This simple feature is

appreciated by cultural heritage users. We also introduced the possibility of visualizing multiple relightable layers superimposed (blended) or side-by-side on the same canvas. This leads to different applications. For instance, it is now possible to compare the same object processed with different RTI encoding or with different image enhancement algorithms. Another possibility is to visualize many objects together, where each one can be rotated and translated independently. This is particularly interesting as it helps experts to interactively align captured fragments and relight them simultaneously and coherently.

A global rotation can be applied to all objects, too. We also introduced the possibility of applying a mask to the object, for background removal, yet the mask has to be pre-computed. Figure 4 shows an example of multiple objects view (here different regions of the same textile fragment are shown), with different rotations. The interface button controls the global rotation, and each layer has an individual controller, which acts when the mouse is superimposed on the layer. The image is translated by clicking and dragging one layer, the image is translated, while if the mouse wheel is scrolled, the single image is rotated. The light control (light rays) controls the global light direction, and the relighting is adapted to be performed coherently in the reference system of the visualization canvas.

5 SYSTEM VALIDATION

To evaluate the effects of our design choices, we conducted extensive testing aimed at (i) verifying the quality of the relighting provided by the improved neural representation, (ii) assessing the error introduced by the linear interpolation of the neural coefficients in the adaptive rendering solution, (iii) evaluate the efficiency of the interactive relighting on different platforms.

	HSH 3rd ord.	**original NeuralRTI**	**NeuralRTI v2**	**NeuralRTI v3**	**NeuralRTI v4**	**NeuralRTI v5**
Texture Bytes/pixel	48	9	9	9	9	9
Encoder/decoder layers	-	3/4	4/3	4/3	4/3	4/2
Layers'size	-	150	150	50	25	150
PSNR on SynthRTI Single	33.33	31.00	**34.60**	34.28	33.64	31.92
PSNR on SynthRTI Multi	26.46	27.39	28.80	**28.96**	28.24	25.76

Table 1. Comparison of average PSNR obtained on the SyhthRTI benchmark [10]. We report results obtained with a 3rd-order HSH (48 coefficients), the original NeuralRTI (9 coefficients), and four modified versions of NeuralRTI obtained changing number and size of fully connected layers.

5.1 Network improvements and relighting quality

The improved quality of data-driven neural relighting with respect to methods using fixed bases with a similar or moderately higher number of coefficients has been assessed in previous works [10], and we do not perform an extensive analysis here. We just report that our modified version of NeuralRTI, with a reduced decoder and an improved encoder, provides results that are better than the original code. Table 1 summarizes the results obtained on the SyhthRTI benchmark [10]. We report results obtained with a 3rd order HSH (16 coefficients per color channel and a total of 48 bytes per pixel), the original NeuralRTI (9 bytes per pixel), and four modified versions of NeuralRTI with a different number of encoder/decoder layers and different layers' size. As we can see (Table 1 column NeuralRTIv2), the additional encoding layer allowed us to obtain a higher PSNR not only with respect to the original NeuralRTI even removing a decoding layer, at the same storage cost. The quality is also higher with respect to the relight obtained with third-order HSH coefficients featuring five times the storage and bandwidth requirements. The PSNR remains high when we reduce the size of the layers to a value approximately equal to the

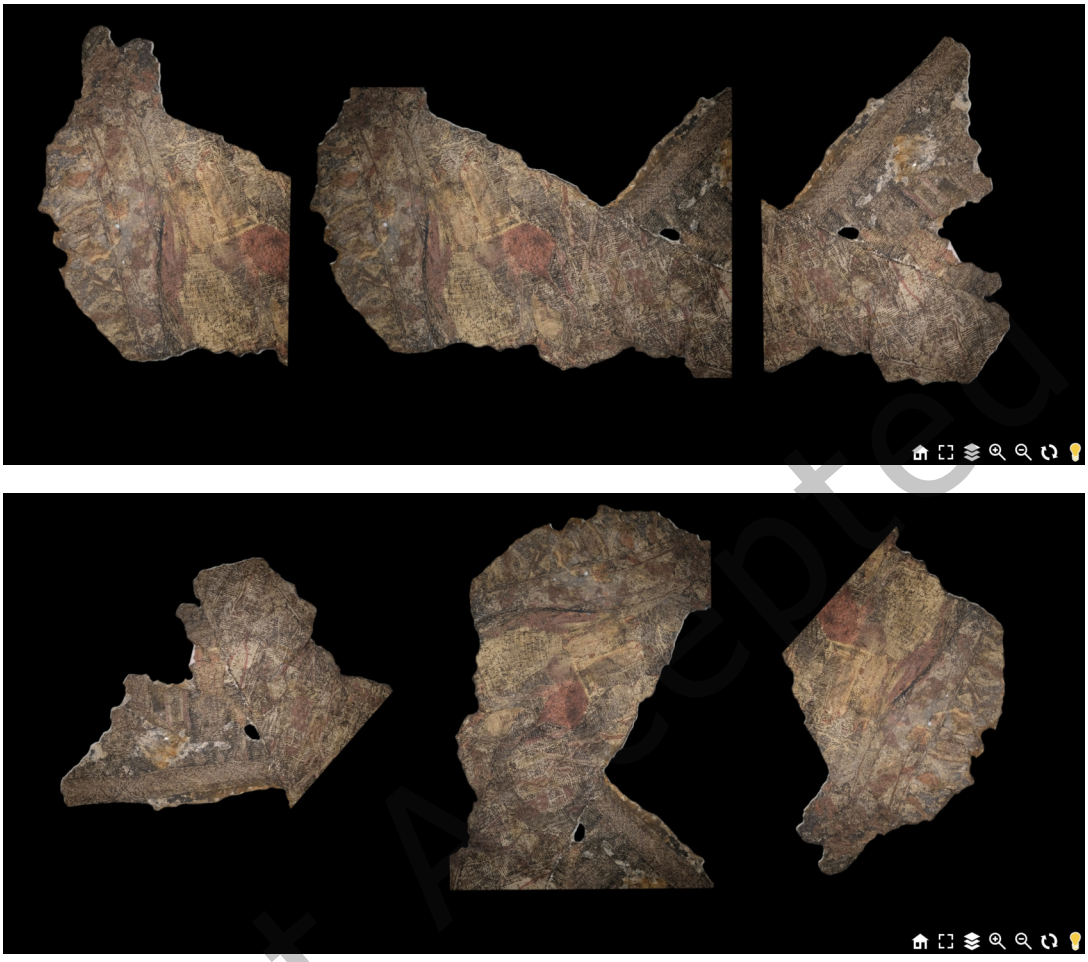


Fig. 4. Example of multiple objects view. Different regions of the same textile fragment from the Oseberg Find. Top: aligned fragments. Bottom: independently rotated fragments. Resolution of source images: 6,240x4,160 px.

typical dome light distribution (Table 1 column NeuralRTIv3, S=50), while there is a more relevant degradation when we further reduce the layer size or we remove another layer (NeuralRTI v4 and v5).

We, therefore, adopted NeuralRTIv3 architecture as a reasonable tradeoff for our tests. It is worth noting that the parameters of the decoder can be changed

Visually, the improved quality is mostly perceivable in the reproduction of specular highlights and global illumination effects. An example can be seen in Figure 5.

The HSH relighting fails in reproducing the specular highlight, and the background is smoothed.

Concerning the loss used for the optimization, we did not obtain relevant improvements by changing the loss in relighting test images of SynthRTI Multi data. The results obtained with L1 and L2 computed in the RGB spaces are very similar, with L1 performing best on Object 1 The results obtained computing the distances in the CIE L*a*b* space are sensitively worse (see Table 2).

(a) Ground truth (b) 3rd-order HSH (c) Neural (d) HSH ꟻLIP difference (e) Neural ꟻLIP difference

Fig. 5. (a) Original image from a MLIC of modern carnelian beads, Indian origin (Angkur Gems society), courtesy of Modern Repository from F. Debrabant and Captured by Mercurio Imaging; (b) Image relighted with the same light direction using a 3rd order HSH model fitted on the original MLIC. (c) Image relighted with the same light direction using our neural relighting implementation trained on the original MLIC. (d) ꟻLIP difference map comparing the HSH relighting with the ground truth. High values are clearly visible in the regions with highlights and shadows. (d) ꟻLIP difference map comparing the neural relighting with the ground truth: perceptual differences are strongly decreased. Image resolution: 1,350×1,450 pixels.

Object	L2 RGB	L1 RGB	L2 L*a*b*	L1 L*a*b*
1	35.84	36.71	32.61	33.77
2	24.58	24.10	23.63	23.49
3	26.45	26.41	25.62	25.28
Average	28.96	**29.07**	27.28	27.51

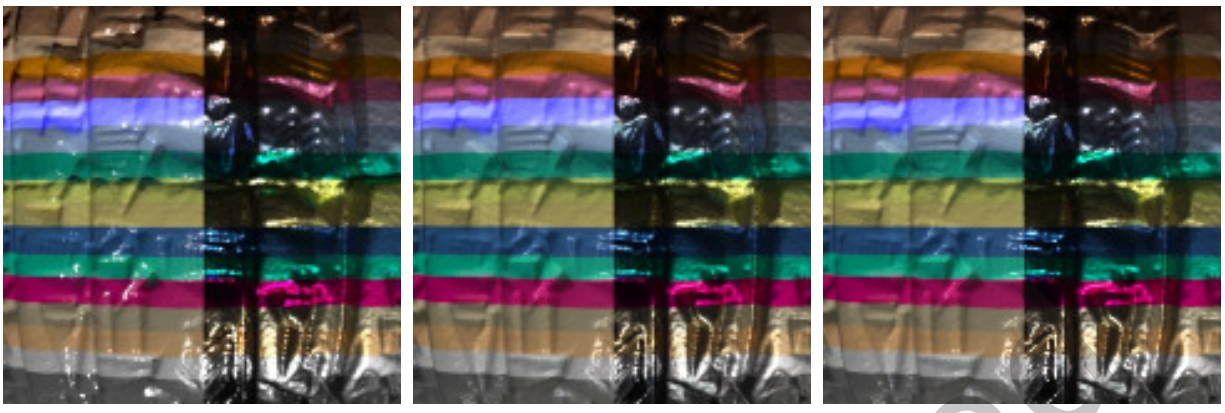
Table 2. PSNR score of NeuralRTI, trained on SynthRTI multi-material, using L1 and L2 loss functions, in RGB and CIE L*a*b* color spaces. The network is originally trained with L2 RGB. Improvements obtained using L1 loss are negligible.

5.2 Interpolation of latent representation

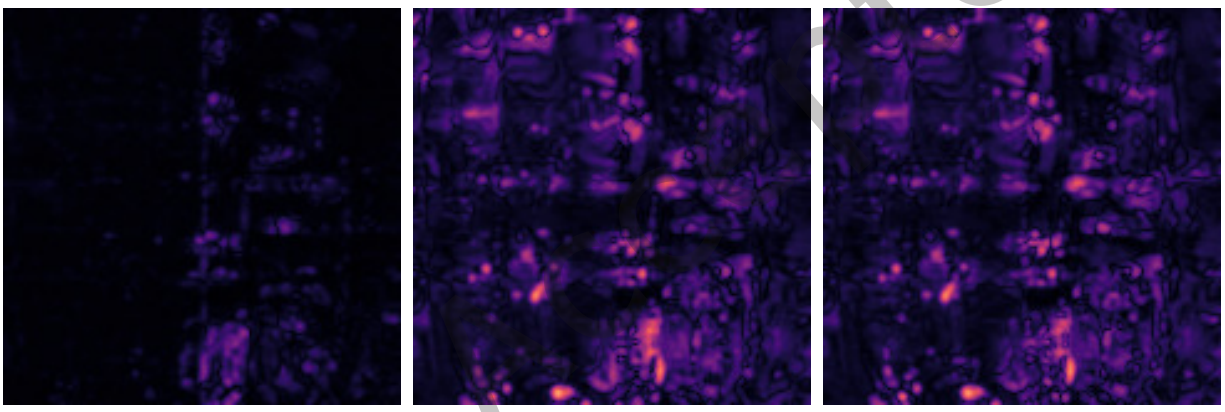
We tested the error introduced by the adaptive rendering scheme working in the latent space of the neural RTI representation by evaluating color, structural, and perceptive differences between images relighted from neural coefficient maps at full resolution and downscaled (50% of original size) with images relighted from neural coefficient maps resampled at 50% of the original size and also with the ground truth images resampled with the same scale factor. We performed these tests on the SynthRTI multi-material datasets as they feature varying shapes with multiple materials suddenly changing in known locations. In this dataset, three surfaces with different geometrical complexity have been textured with regions with different materials, using different material type combinations.

Table 3 shows the ꟻLIP average differences for the three objects in the dataset (across the different test light directions and the different material combinations) between the two relightings of the downsampled images (with coefficients or RGB interpolation). Average and median values are rather small.

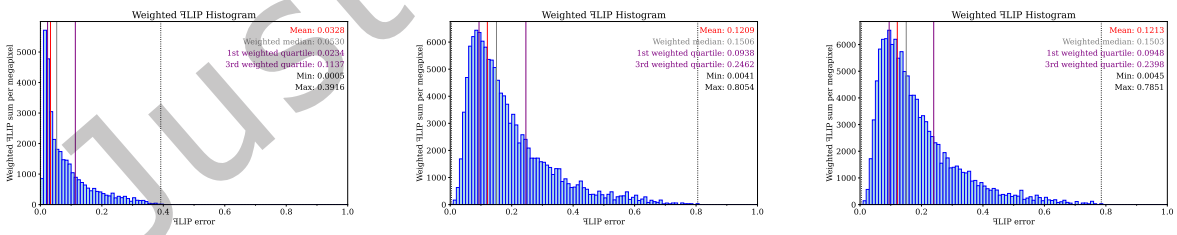
To understand that the difference is negligible, we must compare these values with those reported in Tables 4 and 5. These tables show the corresponding differences measured between each of the two sets of relighted images and the ground truth images resampled of the same factor with linear interpolation. It is evident that the errors introduced by the coefficient interpolation are negligible with respect to the relighting model errors. Furthermore, we note that the values in Tables 4 and 5 are nearly identical, meaning that there is no practical degradation of the results.



(a) Resized (0.5x) ground truth (b) Resized (0.5x) relighted image (c) Relighted image from resized (0.5x) coefficient maps



(d) ƎLIP (b) vs. (c) (e) ƎLIP (b) vs. (a) (f) ƎLIP (c) vs. (a)



(g) ƎLIP histogram (b) vs. (c) (h) ƎLIP histogram (b) vs. (a) (i) ƎLIP histogram (c) vs. (a)

Fig. 6. (a) test image from SynthRTI object 3 material 9, resampled by a factor 0.5. (b) Image relighted with the corresponding light direction with Neural RTI trained on the corresponding training set, resampled by a factor of 0.5. (c) Image relighted with the same light direction from NeuralRTI feature maps resampled by a factor of 0.5. (d) ƎLIP map showing perceived differences between the two relighted and resampled images (b,c). (e,f) ƎLIP maps showing perceived differences between resampled ground truth images and two relighted images (b,c). Image resolution: 320×320 pixels.

Object	Mean	Median	1st quartile	3rd quartile	Min	Max
1	0.012	0.015	0.011	0.023	0.000	0.097
2	0.030	0.044	0.024	0.083	0.001	0.360
3	0.028	0.042	0.021	0.086	0.001	0.345
Average	0.024	0.034	0.019	0.064	0.000	0.267

Table 3. Resampling coefficient vs output ꟻLIP difference.

Object	Mean	Median	1st quartile	3rd quartile	Min	Max
1	0.091	0.120	0.081	0.170	0.005	0.325
2	0.158	0.209	0.127	0.341	0.004	0.901
3	0.125	0.161	0.100	0.264	0.002	0.789
Average	0.125	0.164	0.103	0.258	0.004	0.671

Table 4. Resampling coefficient vs ground truth ꟻLIP difference.

Object	Mean	Median	1st quartile	3rd quartile	Min	Max
1	0.089	0.120	0.081	0.169	0.005	0.321
2	0.158	0.210	0.127	0.341	0.004	0.900
3	0.124	0.160	0.099	0.263	0.003	0.786
Average	0.124	0.163	0.102	0.258	0.004	0.669

Table 5. Resampling output vs ground truth ꟻLIP difference.

Visually, the error introduced by the coefficient interpolation should be visible in the regions with a sudden change in the material properties. The SynthRTI Multi-material data used for our test include surfaces with different patches assigned with different BRDF parameters, as shown in Figure 6. In this example, there are sudden changes of color and roughness along horizontal lines and a vertical transition between dielectric and metallic properties in the middle of the image. Both the relighted images (b,c) appear very similar to the downsampled ground truth one (a). Looking at the ꟻLIP map showing the perceptual difference between relighting with resampled coefficient, it is possible to see that very few artifacts appear near the materials' boundaries (only near the metallic-dielectric transition) and the highest differences between coefficient and RGB resampling (d) correspond to regions with higher geometrical complexity. However, the ꟻLIP values are small compared with the differences measured between the relighted images and resampled ground truth image corresponding to the relight direction (e,f). This fact can also be seen in the corresponding error histograms (g,h,i).

5.3 Rendering performance

The performance of the novel adaptive rendering tool has been tested on a specific inspection task on high-resolution images. The dataset chosen for this test is a high-resolution RTI capture of textile artifacts from the Oseberg Find, coming from a Viking Age burial mound at Oseberg in south Norway. Data is courtesy of Tomasz Łojewski (AGH University of Science and Technology, Kraków). The resolution of the processed images is 6,240×4,140 pixels.

We recorded a short inspection sequence that involved manipulating the lighting, followed by zooming, another lighting adjustment, and finally panning the dataset. Representative frames are Figure 7). We recorded user interaction and repeated the sequence with and without the adaptive rendering optimization, with data resident on a local HTTP server and cleared cache, with the browser taking the entire full-HD screen.

The visualization has been benchmarked on a low-end laptop with an Intel Coffee Lake GT2 graphics controlling a full-HD display.

Without optimization, the renderer achieves, on average, 14.79fps and recomputes on average 393,216 pixels/frame. In this benchmark, the recomputed pixel count is about 20% of the 1,920×1,080 viewport size, since caching reduces the required relighting during panning motion and zooming on the dataset achieves a 2× magnification. It should be noted, also, that the renderer performs asynchronous loading. Since rendering takes some time, there are visual discontinuities when new tiles arrive and are incorporated within the view.

When adaptive optimization is enabled with a target frame rate of 30fps, the renderer achieves a performance of 35.16fps (i.e., close to the target). The higher and more constant performance is obtained by adaptively reducing the number of shaded pixels per frame during the motion to an average of 148,226. Since about half of the pixels are shaded in the adaptive version, there is some slight blurring during motion with respect to the non-adaptive version, but with the advantage of a much smoother constant frame rate. When the image becomes still, the renderer progressively improves quality until the maximum is attainable.

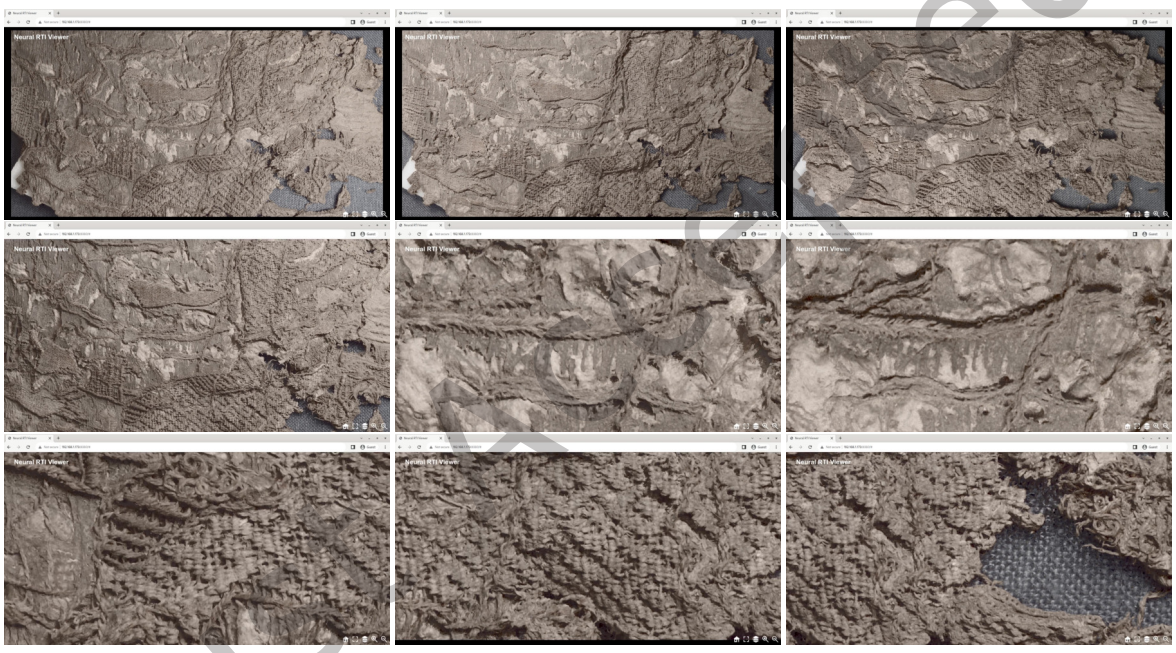


Fig. 7. Selected frames from the inspection of the Oseberg Find dataset used for benchmarking (adaptive version). The frames are ordered left to right, top to bottom. The sequence starts with the interactive motion of the light (frames 1-3), followed by a zooming sequence (frames 3=5), another light motion (frames 5-6), and a final panning (frames 7-9). The application is running on a laptop with integrated graphics. Resolution of source images: 6,240×4,160 pixels. Resolution of the display viewport: 1,920×1,080 pixels

The interactive performances are also maintained when visualizing multiple layers. In the example shown in Figure 1 right, a neural representation is used for the entire image, while a PTM representation is displayed inside a visualization lens. Here, the user directly manipulates camera, light, and lens with touch interaction on a 98-inch 4K multitouch display. The example shows the possibility of using the web-based interface for museum installations by configuring the browser to run in kiosk mode. Similarly, interactive performance is also obtained in the web-based visualization on a laptop of the multiple layers shown in Figure 4.

5.4 Image quality

Quantitative tests demonstrate that the accuracy of the relighting obtained with NeuralRTI is clearly better than that obtained by PTM or HSH.

We will see in the case study discussion of our case study section 6 that one improvement is due to the sharpness of the projected shadows. An even more evident visual improvement obtained with NeuralRTI is related to the relighting of specular materials, that is, for example, particularly evident for the artifact in Figure 5.

Since the neural representation has the same storage and bandwidth cost of PTM, and can be constructed from the exact same input data with a similar end-to-end pipeline, we expect that our approach for providing interactive performance inside web-based platforms will boost its adoption and increase the quality of relightable models.

6 CASE STUDY: ANALYSIS OF TEXTILE ARTIFACTS

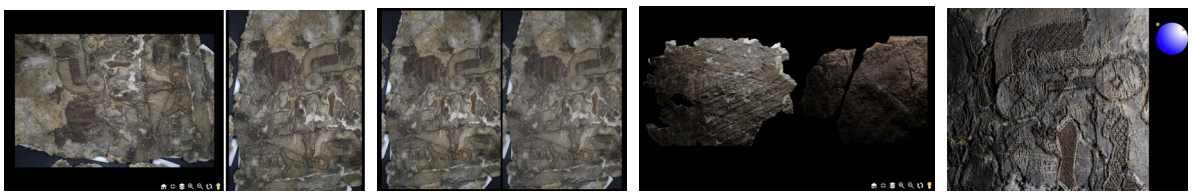
We evaluated the novel visualization tool for practical use on the already mentioned RTI capture of textile artifacts from the Oseberg Find. Oseberg Tapestries are highly fragmented, and an ongoing research is needed to identify matching fragments and virtually reconstruct them [17, 19, 54]. The real fragments are extremely fragile, so interaction with them should be minimal for conservation purposes. Therefore, the reconstruction efforts are primarily limited to digital solutions. Recent work has described the limitations of color images for this objective and proposed RTI as one of the alternative solutions [16]. The professional with expertise in computational reconstruction of fragmented archaeological textiles conducted a qualitative case study, where the tasks were the following:

- Detect figures and analyze the motifs of the fragments, which can be used for the identification of semantic and stylistic similarities among the fragments;
- Count threads, since the number of threads per centimeter can be one of the technical criteria of similarity between the fragments [16];
- Segment the object. Here, we mean not only separating the object's external outline from the background but also identifying holes and missing parts inside the object that may introduce significant noise to the computational algorithms if not detected properly.

The study was conducted using *OpenLIME* tool, where PTM and Neural relighting methods were compared. Besides, PTM versions were also visualized with *RTIViewer* software to draw comparisons with the usability of the *OpenLIME*. The evaluation was conducted on a color-calibrated 15.8-inch LCD display with a resolution of 3,840×2,400 px. The display was calibrated to the following parameters: sRGB, D65 White Point (x=0.313; y=0.329); Gamma=2.2; Peak luminance=80 cd/m^2. The images used for relighting shown throughout this section (Fig.8-12) were captured by Tomasz Łojewski.

6.1 Comparison between visualization tools

OpenLIME demonstrated an advantage in interactivity and multiple practical features. First of all, it enables *rotation* of the fragments, which is essential for proper interpretation of the motifs and figures (Fig. 8a). When the fragment was not oriented properly, the user had to turn their head, which was highly impractical. Surprisingly, such a simple feature is unavailable in *RTIViewer*. Second, *OpenLIME* enables loading multiple layers, allowing users to toggle between different relighting versions of the fragments. It helps to identify the best relighting method for a given task and fragment. Additionally, different layers can be visualized on the same plane – side by side – which can be different relighting versions of the same fragment or different fragments (Fig. 8b and 8c). This facilitates the comparison of the fragments to identify matches among them. The user can display desired fragments from the graphical user interface (GUI). And third, *OpenLIME* offers an additional option to control and visualize illumination direction. In addition to the standard light sphere, which is also available in *RTIViewer*,



(a) Rotation (b) Comparing versions (c) Multiple fragments (d) Illumination Rays

Fig. 8. (a) Possibility to rotate images significantly simplifies detection and interpretation of the figures; (b) *OpenLIME* permits to display different relighting versions of the same object on the same plane, which simplifies the choice of a relighting method (here, PTM and Neural); (c) *OpenLIME* also permits displaying different images at the same time, which helps with similarity comparisons for eventual matching for virtual reconstruction; (d) Light rays overlaid on the image offer a simple and interactive way to vary illumination angle. A light sphere is shown on the right. Resolution of source images: 6,240x4,160 px.

OpenLIME offers dynamic visualization of light rays overlaid on top of the fragment, which can facilitate the selection of optimal illumination angles without the distraction caused by observing the light sphere, which is usually displayed on the side of the screen (Fig. 8d). Other features of the *OpenLIME* that have been mentioned to be of help were *Home* button that returns to the default view and *zoom in* and *zoom out* buttons in the GUI.

6.2 Comparison between relighting methods

The question for this study is whether Neural relighting offers any advantages over PTM for virtual reconstruction, in general, and for manual figure detection, thread counting, and segmentation, more specifically.

When observed under overhead illumination, under which photography of such artifacts is usually done, we notice that Neurally relighted images seem less chromatic and more faded (Fig. 9b) than those relighted with PTM (Fig. 9a). On the other hand, it is worth mentioning that a Neurally relighted one makes holes and gaps inside the object easier to segment out, which needs to be accurate for classification and clustering purposes (Fig. 9d). Color is an important attribute for figure identification and texture classification. However, in heritage artifacts that have undergone significant degradation, color can be highly unreliable, and if it is used, this is usually done based on color images, not RTI [16, 19]. The primary reason why RTI was suggested in the literature for such case study [16, 19] is to capture 2.5D surface topography due to color's unreliability, i.e. first, to identify the part of the threads that are elevated from the background, and second, to segment the object from the canvas. For this purpose, it is important to observe the fragments under grazing angle illumination instead, where elevated parts start casting shadows. This is illustrated in Fig. 10. While both PTM (Fig. 10a) and Neural (Fig. 10b) relighting make the figure more noticeable, the latter provides higher contrast, and the elevated parts cast sharper shadows in the Neural case. Fig. 10c shows that in the cracks and concave parts, PTM usually has lighter pixels, i.e., lower contrast with the surround. This observation has significant implications for thread counting. A number of threads has been reported to be one of the important criteria that experts use for comparing weaving techniques and identifying matching fragments. This is currently done manually. Fig. 11 shows that the threads stand out less under overhead illumination, while under grazing angle, the contrast is noticeably higher. Higher contrast facilitates manual thread counting, which is slightly higher for the Neurally relighted version (but figures in Neural still look less reddish). Recent research efforts have attempted to mimic human behavior and automatically count threads by pulse-counting, where contrast also plays a crucial role [11]. Therefore, Neural relighting can have implications not only for manual analysis but for computational solutions as well.

(a) PTM overhead (b) Neural overhead (c) PTM minus Neural (d) Neural minus PTM

Fig. 9. (a) and (b) illustrate parts of a fragment under overhead illumination with PTM and Neural relighting, respectively. It is noticeable that Neural has less chromatic appearance, while PTM has retained reddish tints. We subtracted Neural from PTM (illustrated in (c)), as well as PTM from Neural (illustrated in (d)), for R, G, and B channels separately. The resulting values were scaled up for visualization's sake. (c) illustrates a stronger reddish channel in PTM, while (d) shows that Neural is usually lighter in the holes where the background canvas is to be segmented.

(a) PTM - Grazing Angle (b) Neural - Grazing Angle (c) Pixel-wise Difference

Fig. 10. At grazing angles, elevated figures and threads start casting shadows that seem slightly sharper for the Neural relighting case (cf. (a) and (b)). (c) This illustrates pixels where PTM contains lighter pixels. We see that PTM is noticeably lighter in the cracks and concavities. The image is found by subtracting a grayscale version of Neural from a grayscale version of PTM, where negative numbers were set to 0. The resulting image was scaled up to better visualize the differing areas.

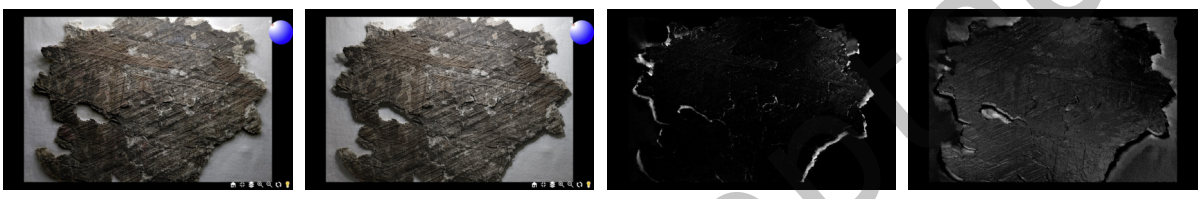
Finally, it is interesting to examine what implications this difference, in contrast, may have for segmentation purposes. Fig. 12 shows a fragment under grazing angle. We can observe that Neural relighting casts larger and sharper shadows around boundaries while brightening the rest of the background. This makes it easier to detect 2.5D (or 3D) surface variations and facilitates segmentation of both an object's outer contours as well as the holes and gaps inside the object.

6.3 Summary

The qualitative case study was carried out by an expert in virtual reconstruction. *OpenLIME* visualization tool has several practical advantages over *RTIViewer*, such as interactivity (e.g., rotation) and possibility to process multiple layers. Afterward, PTM and Neural relighting were compared for virtual reconstruction purposes. The target objects were archaeological tapestry artifacts. On the one hand, PTM offers better preservation of color information, and neurally relighted images look more achromatic or faded. On the other hand, Neural

(a) PTM overhead (b) Neural overhead (c) PTM grazing (d) Neural grazing

Fig. 11. Under overhead lighting, the 3D surface structure is less noticeable, and threads are more difficult to count, as shown in (a) and (b). PTM retains more reddish chromatic components (a), while Neural looks more faded (b). At the grazing angle, the thread structures clearly emerge, where Neural offers slightly higher contrast (d) than PTM (c).



(a) PTM grazing (b) Neural grazing (c) PTM minus Neural (d) Neural minus PTM

Fig. 12. (a) PTM-relighted fragment under grazing angle illumination; (b) Neurally relighted fragment under grazing angle illumination; (c) Pixel-wise difference PTM minus Neural; (d) Pixel-wise difference Neural minus PTM; In difference images (c-d), calculations were done in grayscale and negative values were set to 0. The resulting differences were scaled up for visualization's sake.

relighting yields higher contrasts and sharper shadows under grazing angle illumination, which facilitates surface topography analysis, thread counting, segmentation, and texture classification. This makes neural relighting a promising avenue for cultural heritage applications. Interesting future work would be further enhancement of contrast with image enhancement algorithms, such as Retinex and STRESS [25].

7 CONCLUSIONS

We have reported on our work targeting the integration of complex relightable models based on neural encoding into modern web frameworks for the inspection of stratigraphic relightable models.

By relying on the fact that typical neural reflectance field approximations need only to implement a simple decoder (typically a low-depth fully connected network), we have shown that it is possible to directly encode weights and input data into web- and GPU-friendly formats that exploit common image formats and texturing features. By doing so, we have been able to implement the network in a way that does not significantly differ from the design of other typical RTI shaders. This solution has made it possible to avoid the incorporation of, and communication with, external deep-learning libraries, and to incorporate extra per-fragment filtering steps (e.g., gamma correction) without the need to resort to multipass solutions.

We have also empirically demonstrated that, in practice, the latent representation of nearby pixels in the employed network can be combined, filtered, and resampled without producing significant disturbing artifacts on the generated output image. This fact has made it possible to use filtering and resampling as major building blocks to build levels of detail and adaptive rendering solutions. It should be noted that the assumption that some

deep neural networks can model input data as flat and smooth distributions is specifically employed, e.g., in latent-space image interpolation approaches, where, if x and y are sampled from two respective domains X and Y, moving from x toward y in the latent space continuously produces realistic images from domain X to Y [5, 28]. While this property was only empirically verified for our network [10], an important avenue of research is the design of other (guaranteed) interpolable networks for reflectance function encoding. This sort of approach has already been explored for Neural BRDFs [51], but, to the best of our knowledge, not in reflectance modeling for RTI.

Furthermore, the solutions employed for ensuring interactivity through a combination of precomputed discrete levels of details, run-time adaptive resampling, smart caching, and exploitation of decoupling between lighting and camera control, are not limited to neural representations but promise to be general solutions to also efficiently incorporate other costly relighting methods into an interactive viewer. Interesting candidates are those based on RBF interpolation [15, 45].

The presented results on cultural heritage item inspections have shown the method's appeal and flexibility. As the method supports the same features of more standard PTM and HSH solutions, we expect it to become a possible plug-in replacement that provides a higher-quality experience with respect to current low-frequency solutions. To benefit the community, we plan to release its implementation as open source within the *OpenLIME* framework [37].

In future work, we also plan to test the potential impact for CH practitioners of allowing them to relight the surface using more complex illumination patterns (e.g., multiple lights) and other detail enhancement solutions, possibly combining RTI with physically-based rendering.

ACKNOWLEDGMENTS

The authors thank Tomasz Łojewski (AGH University of Science and Technology, Kraków) for the provision of the Oseberg Find data, Mercurio Imaging for capturing the modern carnelian beads, and the Diocesan Museum of Vicenza and Accademia delle Belle Arti of Verona for giving access to their artworks for digitization.

This study was partially carried out within the activities of the consortium iNEST (Interconnected North-Est Innovation Ecosystem) funded by the European Union Next-GenerationEU (Piano Nazionale di Ripresa e Resilienza (PNRR) – Missione 4 Componente 2, Investimento 1.5 – D.D. 1058 23/06/2022, ECS00000043) and partially supported by the project "REFLEX – REFLectance EXploration: improving the acquisition, distribution, and exploration of multi-light image collections for surface characterization and analysis" (PRIN2022) funded by the EU Next-GenerationEU PNRR, componente M4C2, investimento 1.1.

We also thank Fabio Marton for his collaboration in the development of OpenLime tools.

REFERENCES

[1] Moonisa Ahsan, Giuliana Altea, Fabio Bettio, Marco Callieri, Antonella Camarda, Paolo Cignoni, Enrico Gobbetti, Paolo Ledda, Alessandro Lutzu, Fabio Marton, Giuseppe Mignemi, and Federico Ponchio. 2022. Ebb & Flow: Uncovering Costantino Nivola's Olivetti Sandcast through 3D Fabrication and Virtual Exploration. In *Proc. GCH*. 85–94. https://doi.org/10.2312/gch.20221230

[2] Jonathan T Barron. 2019. A general and adaptive robust loss function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4331–4339.

[3] Johannes Behr, Peter Eschler, Yvonne Jung, and Michael Zöllner. 2009. X3DOM: a DOM-based HTML5/X3D integration model. In *Proceedings of the 14th international conference on 3D web technology*. 127–135. https://doi.org/10.1145/1559764.1559784

[4] Fabio Bettio, Moonisa Ahsan, Fabio Marton, and Enrico Gobbetti. 2021. A novel approach for exploring annotated data with interactive lenses. *Computer Graphics Forum* 40, 3 (2021), 387–398. https://doi.org/10.1111/cgf.14315

[5] Y. Chen, X. Xu, Z. Tian, and J. Jia. 2019. Homomorphic Latent Space Interpolation for Unpaired Image-To-Image Translation. In *Proc. CVPR*. 2403–2411. https://doi.org/10.1109/CVPR.2019.00251

[6] Zhe Chen, Shohei Nobuhara, and Ko Nishino. 2021. Invertible neural BRDF for object inverse rendering. *IEEE TPAMI* 44, 12 (2021), 9380–9395. https://doi.org/10.1109/TPAMI.2021.3129537

[7] CHI. 2019. Cultural Heritage Imaging website. http://culturalheritageimaging.org [Online; accessed 22-May-2023].

[8] DHLAB. 2017. RTI tools at DHLAB Basel. https://github.com/dhlab-basel/rti.js [Online; accessed 22-May-2023].
[9] Mark S Drew, Yacov Hel-Or, Tom Malzbender, and Nasim Hajari. 2012. Robust estimation of surface properties and interpolation of shadow/specularity components. *Image and Vision Computing* 30, 4-5 (2012), 317–331. https://doi.org/10.1016/j.imavis.2012.02.012
[10] Tinsae G Dulecha, Filippo A Fanni, Federico Ponchio, Fabio Pellacini, and Andrea Giachetti. 2020. Neural reflectance transformation imaging. *The Visual Computer* 36 (2020), 2161–2174. https://doi.org/10.1007/s00371-020-01910-9
[11] Riestiya Zain Fadillah, Davit Gigilashvili, Margrethe Havgar, Marianne Vedeler, and Jon Yngve Hardeberg. 2024. Automatic Thread Counting for Archaeological Textiles. In *Proc. InART*.
[12] Peter Fornaro, Andrea Bianco, Aeneas Kaiser, and Lukas Rosenthaler. 2017. Enhanced RTI for gloss reproduction. *Electronic Imaging* 2017, 8 (2017), 66–72. https://doi.org/10.2352/ISSN.2470-1173.2017.8.MAAP-284
[13] Duan Gao, Xiao Li, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. 2019. Deep inverse rendering for high-resolution SVBRDF estimation from an arbitrary number of images. *ACM TOG* 38, 4 (2019), 134–1. https://doi.org/10.1145/3306346.3323042
[14] Pascal Gautron, Jaroslav Krivánek, Sumanta N Pattanaik, and Kadi Bouatouch. 2004. A Novel Hemispherical Basis for Accurate and Efficient Rendering. *Rendering Techniques* 2004 (2004), 321–330. https://doi.org/10.2312/EGWR/EGSR04/321-330
[15] Andrea Giachetti, Irina Ciortan, Claudia Daffara, Giacomo Marchioro, Ruggero Pintus, and Enrico Gobbetti. 2018. A Novel Framework for Highlight Reflectance Transformation Imaging. *Computer Vision and Image Understanding* 168 (2018), 118–131. https://doi.org/10.1016/j.cviu.2017.05.014
[16] Davit Gigilashvili, Hana Lukesova, Casper Fabian Gulbrandsen, Akash Harijan, and Jon Yngve Hardeberg. 2023. Computational techniques for virtual reconstruction of fragmented archaeological textiles. *Heritage Science* 11, 259 (2023), 1–16.
[17] Davit Gigilashvili, Ha Thu Nguyen, Casper Fabian Gulbrandsen, Margrethe Havgar, Marianne Vedeler, and Jon Yngve Hardeberg. 2023. Texture-based clustering of archaeological textile images. In *The Proceedings of Archiving 2023 Conference*, Vol. 20. 139–142.
[18] Darya Guarnera, Giuseppe Claudio Guarnera, Abhijeet Ghosh, Cornelia Denk, and Mashhuda Glencross. 2016. BRDF representation and acquisition. *Computer Graphics Forum* 35, 2 (2016), 625–650. https://doi.org/10.1111/cgf.12867
[19] Casper Fabian Gulbrandsen. 2023. *Reconstructing the Original: Machine Learning Puzzle Assembly for Matching Archaeological Textile Fragments.* Master Thesis. Norwegian University of Science and Technology.
[20] Masatoshi Hidaka, Yuichiro Kikura, Yoshitaka Ushiku, and Tatsuya Harada. 2017. WebDNN: Fastest DNN execution framework on web browser. In *Proc. ACM Multimedia*. 1213–1216. https://doi.org/10.1145/3123266.3129394
[21] Jacek Jankowski and Martin Hachet. 2015. Advances in Interaction with 3D Environments. *Comput. Graph. Forum* 34, 1 (2015), 152–190. https://doi.org/10.1111/cgf.12466
[22] Alberto Jaspe Villanueva, Moonisa Ahsan, Ruggero Pintus, Andrea Giachetti, and Enrico Gobbetti. 2021. Web-based Exploration of Annotated Multi-Layered Relightable Image Models. *ACM JOCCH* 14, 2 (2021), 24:1–24:31. https://doi.org/10.1145/3430846
[23] H. Jin, I. Liu, P. Xu, X. Zhang, S. Han, S. Bi, X. Zhou, Z. Xu, and H. Su. 2023. TensoIR: Tensorial Inverse Rendering. In *Proc. CVPR*. 165–174. https://doi.org/10.1109/CVPR52729.2023.00024
[24] Andrej Karpathy. 2022. ConvNetJS: Deep Learning in your browser. https://cs.stanford.edu/people/karpathy/convnetjs/ [Online; accessed 19-May-2023].
[25] Øyvind Kolås, Ivar Farup, and Alessandro Rizzi. 2011. Spatio-temporal Retinex-inspired envelope with stochastic sampling: a framework for spatial color algorithms. *Journal of Imaging Science and Technology* 55, 4 (2011). https://doi.org/10.2352/J.ImagingSci.Technol.2011.55.4.040503
[26] KUL. 2019. PLD software KU-Leuven. https://portablelightdome.wordpress.com/software [Online; accessed 22-May-2023].
[27] Chen Liu, Michael Fischer, and Tobias Ritschel. 2023. Learning to learn and sample BRDFs. *Computer Graphics Forum* 42, 2 (2023), 201–211. https://doi.org/10.1111/cgf.14754
[28] Yajing Liu, Zhiwei Xiong, Ya Li, Xinmei Tian, and Zheng-Jun Zha. 2023. Domain Generalization Via Encoding and Resampling in a Unified Latent Space. *IEEE Transactions on Multimedia* 25 (2023), 126–139. https://doi.org/10.1109/TMM.2021.3121564
[29] Yun Ma, Dongwei Xiang, Shuyu Zheng, Deyu Tian, and Xuanzhe Liu. 2019. Moving deep learning into web browser: How far can we go?. In *Proc. WWW*. 1234–1244. https://doi.org/10.1145/3308558.3313639
[30] Lindsay William Macdonald. 2015. *Realistic visualisation of cultural heritage objects.* Ph. D. Dissertation. UCL (University College London).
[31] Tom Malzbender, Dan Gelb, and Hans Wolters. 2001. Polynomial texture maps. In *Proc. SIGGRAPH*. 519–528. https://doi.org/10.1145/383259.383320
[32] Mozilla MDN. 2023. Image file type and format guide. https://developer.mozilla.org/en-US/docs/Web/Media/Formats/Image_types [Online; accessed 28-May-2023].
[33] Microsoft. 2008. DeepZooom. http://www.seadragon.com/developer/creating-content/file-formats/ [Online; accessed 22-May-2023].
[34] Steven Miller. 2022. MIND: Deep Learning in your browser. https://github.com/stevenmiller888/mind [Online; accessed 19-May-2023].
[35] Anastasia Moutafidou, Georgios Adamopoulos, Anastasios Drosou, Dimitrios Tzovaras, and Ioannis Fudos. 2018. Multiple Material Layer Visualization for Cultural Heritage Artifacts. In *Proc. GCH*. 155–159. https://doi.org/10.2312/gch.20181353

[36] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. 2022. Extracting triangular 3D models, materials, and lighting from images. In *Proc. CVPR*. 8280–8290. https://doi.org/10.1109/CVPR52688.2022.00810

[37] OpenLime Team. 2022. OpenLime: Open Layered IMage Explorer. URL: https://github.com/cnr-isti-vclab/openlime and https://github.com/crs4/openlime [Online; accessed 22-May-2023].

[38] Renato Pajarola and Enrico Gobbetti. 2007. Survey on Semi-Regular Multiresolution Models for Interactive Terrain Rendering. *The Visual Computer* 23, 8 (2007), 583–605. https://doi.org/10.1007/s00371-007-0163-2

[39] Giampaolo Palma et al. 2019. WebRTI Viewer. http://vcg.isti.cnr.it/rti/webviewer.php [Online; accessed 22-May-2023].

[40] Ruggero Pintus, Moonisa Ahsan, Antonio Zorcolo, Fabio Bettio, Fabio Marton, and Enrico Gobbetti. 2023. Exploiting Local Shape and Material Similarity for Effective SV-BRDF Reconstruction from Sparse Multi-Light Image Collections. *ACM JOCCH* 16, 2 (June 2023), 39:1–39:31. https://doi.org/10.1145/3593428

[41] Ruggero Pintus, Tinsae Dulache, Irina Ciortan, Enrico Gobbetti, and Andrea Giachetti. 2019. State-of-the-art in Multi-Light Image Collections for Surface Visualization and Analysis. *Computer Graphics Forum* 38, 3 (2019), 909–934. https://doi.org/10.1111/cgf.13732

[42] Mara Pistellato and Filippo Bergamasco. 2023. On-the-Go Reflectance Transformation Imaging with Ordinary Smartphones. In *Proc. ECCV Workshops, Part I*. 251–267. https://doi.org/10.1007/978-3-031-25056-9_17

[43] Gilles Pitard, Gaëtan Le Goïc, Hugues Favrelière, Serge Samper, Simon-Frédéric Desage, and Maurice Pillet. 2015. Discrete Modal Decomposition for surface appearance modelling and rendering. In *Optical Measurement Systems for Industrial Inspection IX*, Vol. 9525. SPIE, 489–498. https://doi.org/10.1117/12.2184840

[44] Federico Ponchio et al. 2019. Relight. http://vcg.isti.cnr.it/relight/ [Online; accessed 22-May-2023].

[45] Federico Ponchio, Massimiliano Corsini, and Roberto Scopigno. 2018. A compact representation of relightable images for the web. In *Proc. ACM Web3D*. 1:1–1:10. https://doi.org/10.1145/3208806.3208820

[46] Marco Potenziani, Marco Callieri, Matteo Dellepiane, Massimiliano Corsini, Federico Ponchio, and Roberto Scopigno. 2015. 3DHOP: 3D heritage online presenter. *Computers & Graphics* 52 (2015), 129–141. https://doi.org/10.1016/j.cag.2015.07.001

[47] Peiran Ren, Yue Dong, Stephen Lin, Xin Tong, and Baining Guo. 2015. Image based relighting using neural networks. *ACM TOG* 34, 4 (2015), 111:1–111:12. https://doi.org/10.1145/2766899

[48] Leonardo Righetto, Fabio Bettio, Federico Ponchio, Andrea Giachetti, and Enrico Gobbetti. 2023. Effective interactive visualization of neural relightable images in a web-based multi-layered framework. In *The 21th Eurographics Workshop on Graphics and Cultural Heritage*. 57–66. https://doi.org/10.2312/gch.20231158

[49] Roberto Scopigno, Marco Callieri, Paolo Cignoni, Massimiliano Corsini, Matteo Dellepiane, Federico Ponchio, and Guido Ranzuglia. 2011. 3D models for cultural heritage: Beyond plain visualization. *Computer* 44, 7 (2011), 48–55. https://doi.org/10.1109/MC.2011.196

[50] Daniel Smilkov, Nikhil Thorat, Yannick Assogba, Charles Nicholson, Nick Kreeger, Ping Yu, Shanqing Cai, Eric Nielsen, David Soegel, Stan Bileschi, et al. 2019. Tensorflow.js: Machine learning for the web and beyond. *Proc. Machine Learning and Systems* 1 (2019), 309–321.

[51] Alejandro Sztrajman, Gilles Rainer, Tobias Ritschel, and Tim Weyrich. 2021. Neural BRDF representation and importance sampling. *Computer Graphics Forum* 40, 6 (2021), 332–346. https://doi.org/10.1111/cgf.14335

[52] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. 2020. State of the art on neural rendering. *Computer Graphics Forum* 39, 2 (2020), 701–727. https://doi.org/10.1111/cgf.14022

[53] Bruno Vandermeulen, Hendrik Hameeuw, Lieve Watteeuw, Luc Van Gool, and Marc Proesmans. 2018. Bridging Multi-light & Multi-Spectral images to study, preserve and disseminate archival documents. In *Proc. Archiving Conference*, Vol. 2018. 64–69. https://doi.org/10.2352/issn.2168-3204.2018.1.0.15

[54] Marianne Vedeler. 2019. *The Oseberg Tapestries*. Scandinavian Academic Press Oslo.

[55] VIPS. 2022. libvips:A fast image processing library with low memory needs. https://www.libvips.org/ [Online; accessed 09-May-2023].

[56] Zexiang Xu, Kalyan Sunkavalli, Sunil Hadap, and Ravi Ramamoorthi. 2018. Deep image-based relighting from optimal sparse samples. *ACM TOG* 37, 4 (2018), 126:1–126:13. https://doi.org/10.1145/3197517.3201313

[57] Mingjing Zhang and Mark S Drew. 2014. Efficient robust image interpolation and surface properties using polynomial texture mapping. *EURASIP Journal on Image and Video Processing* 2014, 1 (2014), 25. https://doi.org/10.1186/1687-5281-2014-25

[58] Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. 2021. NeRFactor: neural factorization of shape and reflectance under an unknown illumination. *ACM TOG* 40, 6 (2021), 237:1–237:18. https://doi.org/10.1145/3478513.3480496