A Deep Learning Approach for Digital Color Reconstruction of Lenticular Films

Stefano D’Aronco, Giorgio Trumpy, David Pfluger, and Jan Dirk Wegner

Abstract—We propose the first accurate digitization and color reconstruction process for historical lenticular film that is robust to artifacts. Lenticular films emerged in the 1920s and were one of the first technologies that permitted to capture full color information in motion. The technology leverages an RGB filter and cylindrical lenticules embossed on the film surface to encode the color in the horizontal spatial dimension of the image. To project the pictures the encoding process was reversed using an appropriate analog device. In this work, we introduce an automated, fully digital pipeline to process the scan of lenticular films and colorize the image. Our method merges deep learning with a model-based approach in order to maximize the performance while making sure that the reconstructed colored images truthfully match the encoded color information. Our model employs different strategies to achieve an effective color reconstruction, in particular (i) we use data augmentation to create a robust lenticule segmentation network, (ii) we fit the lenticules raster prediction to obtain a precise vectorial lenticule localization, and (iii) we train a colorization network that predicts interpolation coefficients in order to obtain a truthful colorization. We validate the proposed method on a lenticular film dataset and compare it to other approaches. Since no colored groundtruth is available as reference, we conduct a user study to validate our method in a subjective manner. The results of the study show that the proposed method is largely preferred with respect to other existing and baseline methods.

Index Terms—deep learning, lenticular film, color reconstruction.

I. INTRODUCTION

Historical films represent a precious source of information for how the world and life looked like in the past. Film digitization is essential to protect these valuable records from deterioration and also to facilitate their visualization. Digitization and visualization of historical films is not always a trivial process, as, in some cases, these media might rely on a specific analog device for their visualization. As a result, a software adaptation of the original analog process is needed to display them today.

Lenticular film, made popular by Eastman Kodak under the name Kodacolor in the late 20s, represents one of the earliest technologies that allowed amateur filmmakers to record color movies [1]. The technology consisted of a black-and-white film embossed with vertical cylinders, called lenticules, and an RGB filter placed in front of the camera lens. This combination allowed to encode the color information as silver densities in the horizontal spatial dimension within each lenticule. After being exposed in the camera, the film underwent reversal processing. Its visualization was done with a dedicated projection device that reversed the recording process, using an RGB filter in front of the lens to create color images from the black-and-white film. Different approaches can be adopted to recreate the color images of lenticular film with digital tools. The digitization of the projected, colored film is time consuming and requires a special, obsolete projection device. Scanning the black-and-white lenticular film, on the other hand, is definitely more practical, but the additive colors created optically during analog projection have to be created numerically via software. Two main steps are required to colorize a scanned lenticular film: i) identify the lenticules on the film, and ii) extract the grayscale values inside each lenticule to finally reconstruct the RGB image. The lenticules on the scanned film can be detected by looking at their boundaries which appear like dark vertical straight lines. After the boundaries of a lenticule are identified, the RGB information, which is stored in the horizontal spatial dimension between the boundaries, can be easily extracted. An example of a scanned input film, grayscale, with the color reconstruction is shown in Fig. 1.

The appropriate color reconstruction strategy for a scanned, lenticular film depends on the final, intended application. For instance, the principle dear to the film preservation community is to obtain the closest possible reproduction to the original, analog projection process, accurately simulating how the images would look like when visualized with the historical, analog projector. Another less orthodox aim would be creating images that simulate how the scenes represented in the lenticular film would look if they were shot with modern cameras by hallucinating some details and altering the colors. We aim to obtain a digital version of the lenticular film that somewhat improves the visualization w.r.t. the analog one, while maintaining the truthfulness dictated by the resolution and color information encoded in the lenticular film. In particular, we do not aim to suppress specific artifacts like scratches or other large defects potentially present on the lenticular film in order to fully preserve the historical value of the pictures. At the same time we aim to build a network that is more robust w.r.t. existing digital lenticular color reconstruction algorithms, and avoid severe artifacts that arise from an erroneous lenticule localization.

We design a multistage pipeline that combines deep learning and model-based stages. The first stage is a segmentation network that performs a binary classification on the input image, discriminating between pixels located on lenticules’ boundaries and pixels within the lenticules. We then leverage this prediction in order to extract the lenticules’ boundaries in a vector format, rather than a raster prediction. For each detected lenticule, the RGB values encoded are extracted creating an image where for each pixel column only one of the RGB channels is known. A second CNN, trained on a separate dataset, is then used in order to interpolate
Colorization is a more general problem that is not exclusively tailored to old photos restoration. A survey on different colorization approaches and applications can be found in [9]. In the simplest colorization settings, the aim is to reconstruct the RGB triplets starting from the single luminance component. Some details need to be hallucinated because many real-life objects exist in a wide range of colors. An effective pipeline for colorization of historical images is the DeOldify project [10]. DeOldify is able to colorize single images or videos, starting from a grayscale image, and uses a Generative Adversarial Network (GAN) in order to obtain realistic-looking, colorized images.

Most of the works about colorization, however, are not suited to deal with lenticular films because they are not designed to leverage the color information encoded in the spatial domain. If we assume that the lenticules are detected and the corresponding RGB values are extracted, the color reconstruction process of lenticular films shares a lot of similarities with demosaicing [11], [12]. The problem of demosaicing consists in reconstructing a full color image starting from an incomplete image, where only a single value among red, green, and blue is known for each pixel. This problem usually originates from the usage of an array color filters in front of an image sensor that creates a mosaic structure among the sensed color channels. In the case of lenticular film, the structure is composed of vertical stripes rather than a mosaic. Basic algorithms for demosaicing use rather simple interpolation techniques [12], [13]. The doLCE method, similarly to our pipeline, is composed of two stages: detect the lenticules, and colorize the image. The method, although effective on many images, suffers from two main limitations: i) the lenticule detection only works with, almost, perfectly vertical lenticules, ii) the colorization process assigns a single RGB triplet for the entire width of the lenticule thus drastically reducing the realistic-looking, colorized images.

Fig. 1: Example of a scanned lenticular film (top), and doLCE color reconstruction (center), proposed method (bottom)
Fig. 2: Side view section of the lenticular film camera. Tiny cylinders are embossed focusing the light colored on different parts of the emulsion.

Fig. 3: Left: Position of the red, green and blue components within the lenticules. Right: detail of a diagonal edge on a scanned lenticular film. As it can be seen the diagonal edge is still perceivable within the lenticules meaning that resolution in the horizontal direction. In this paper we aim at addressing those limitations to design a pipeline that can deal with tilted lenticules while at the same time increasing the detail level of the output image. We build upon our preliminary work \[17\] where we tested the feasibility of using a convolutional neural network to detect the lenticule boundaries. In our new work we include a refinement module for the lenticules extraction as well as a colorization network which preserves a higher level of detail from the grayscale input image.

III. BACKGROUND

The schematic diagram of a lenticular film camera is shown in Fig. 2. First, an RGB filter is placed in front of the camera lens. The lenticular film is characterized by a vertical array of hundreds of cylindrical lenses. The lenticules’ focal length is three times the curvature radius and is equal to the film thickness. Focal length and aperture of the camera were designed so that the photographic emulsion is exposed to the different color components \[18\].

Lenticular film was sold in 16 mm format, where the images occupies about 10 mm of film width. Considering that the width of each lenticule is about 43 µm, approximately 230 lenticules fit within each image. In order to extract the spatially-encoded color information, the film has to be scanned with the emulsion facing the image capture system. After scanning the original, historical film, the lenticules’ boundaries appear as dark straight vertical lines in the digital image. Estimating their width, in number of pixels, is straightforward because we know the scanning resolution. However, the lenticule width does vary slightly within each scanned film, this is possibly caused by: optical aberrations of the digital capture, limited manufacturing quality of the film itself, time degradation, or mechanical damage. From our dataset we empirically observed that the lenticules’ width inside a single frame varies by about 5% of the average width. Moreover, the film is often slightly misaligned with the scanner, which leads to the lenticules not being perfectly vertical. In this regard, we empirically observed a maximum rotation of about 1°.

In Fig. 3, we show where the colors are encoded within the lenticules. This image is obtained by shining white light through the original lens of a lenticular camera and capturing with a digital camera the light that was supposed to reach the film emulsion. In order to reconstruct the colors of the scanned grayscale image, we need to detect the lenticules, extract the color and fill each lenticule for its entire width with the RGB values. One question that rises when looking at the grayscale scanned film is how the color information is exactly encoded. Do the red-green-blue values inside each row of a lenticule refer to the same point in the captured scene? Or do they refer to different points in the scene where for each point we have access to only one of the RGB value? One way to answer this question is by carefully looking how diagonal edges look like in the grayscale images. Fig. 3b shows quite sharp diagonal edges between achromatic regions. Note that in white/gray/black regions the red-green-blue values in each lenticule should roughly always amount to the same value. As it can be seen, inside a single lenticule the diagonal edge is still visible, this means that valuable spatial horizontal information is contained within each lenticule. In other terms, different pixels in the horizontal direction within a lenticule map to different points in captured scene.

Finally, in order to perform an accurate color reconstruction, we need to know how to map the color values of the lenticular film in a standard RGB space. This mapping process can be derived from the transmission spectra of the RGB filter. We refer the reader to the Appendix A for further information about this process.

IV. PROPOSED METHOD

We denote the input image with \( x \), which is a grayscale image of size \( H \times W \) pixels. The proposed pipeline is split in two stages: the first stage is responsible for extracting the lenticules boundaries, whereas the second one is responsible for reconstructing the RGB triplet inside each lenticule.

A. Lenticule Boundaries Detection

The lenticule boundaries appear like straight vertical valleys (dark ridges) in the input grayscale film scan. What makes
boundary detection hard is that such valleys are only visible in parts of the image that are sufficiently bright, see for instance Fig. 7b. Moreover, lenticules are in reality often not exactly vertical but slightly misaligned because of the scanning process. Furthermore, the distance between lenticule boundaries, though being rather regular, is not exactly constant across the image, preventing us from fitting a regularly spaced grid. These artifacts are major reasons for failure of doLCE [16], a method built on well-established image processing tools for certain easy cases: clearly visible, almost perfectly vertical lenticule boundaries for at least some part of the image along the vertical direction. In order to generate a large dataset with clean lenticule labels for training our convolutional neural network (CNN), validation, and testing, we use successful lenticule detection cases obtained with doLCE. Since one major goal of our approach is greater robustness against artifacts (where doLCE fails), we artificially create many hard cases via data augmentation.In general, any modern network architecture for image segmentation could be used for lenticule boundary detection. Training is done by minimizing the cross-entropy loss between the ground truth and the network prediction:

\[
\min_{\theta} \mathbb{E}_{x,z_{gt}}[\text{CE}(z(x), z_{gt})],
\]

where CE represents the pixelwise cross entropy loss, \(z(x)\) is the network output (\(z \in [0,1]\)) and \(z_{gt}\) is the ground truth segmentation map. As already mentioned successful doLCE reconstructions occur exclusively when the lenticule boundaries are perfectly vertical, creating a bias in the training dataset as the neural network is never exposed to tilted lenticules. To correct for this bias, we perform data augmentation by randomly rotating (maximum rotation is set to \(\pm 1^\circ\)) the input-output patches so that the network learns how to deal with this situation. We train the network with patches of 256 \times 256 pixels taken from input images \(x\), whereas the groundtruth is obtained from lenticule boundary predictions using the doLCE algorithm.

B. Lenticules Refinement and Color Extraction

The output of the neural network is a raster grayscale image that highlights the lenticule boundaries. In order to proceed with the color reconstruction, we need to extract the lenticule locations in a vector format. Thus we first define a vectorized representation of the lenticule boundaries as two sequences \(t = (t_1, t_2, ..., t_M)\) and \(b = (b_1, b_2, ..., b_M)\), where \(t_n\) and \(b_n\) correspond to the location (measured in pixels) of the \(n\)-th lenticule at the top and at the bottom of the image \(x\). The total number of lenticule boundaries \(M\) in a single image can easily be estimated from \(z\) by simply measuring the average width of the lenticules. Intuitively, we need to look for the set of straight lines defined by \(t\) and \(b\) that best overlap with the raster lenticule boundaries predicted by the network, thus minimizing the following metric:

\[
\min_{u,t} \sum_{m=0}^{M} \sum_{h=0}^{H} z(h, b_m + \frac{ht_m}{H}).
\]

The objective of the minimization problem corresponds to integrating the pixel values of \(z\) that lie on each of the \(M\) straight lines defined by vectors \(t\) and \(b\). If lenticules are detected correctly, then \(z\) is equal to one on the boundary and zero everywhere else. Therefore, this metric is minimized when the fitted line lies exclusively on pixels located on a lenticule boundary. The objective function is, however, ill-posed as it does not guarantee that the entries in \(t\) and \(b\) fit different lenticules in the image. To address this problem, we need specific regularization terms. Since we know that the width of the lenticules should be rather constant across the image, we can use this to enforce two regularization terms. We can penalize large deviations from the average lenticule width, as well as large width variations between consecutive lenticules. To this end, we formulate the following quadratic penalties:

\[
\begin{align*}
    r_1(t, b) &= (Dt - \hat{w})^T(Dt - \hat{w}) + (Db - \hat{w})^T(Db - \hat{w}), \quad (3) \\
    r_2(t, b) &= t^T H^T H t + b^T H^T H b, \quad (4)
\end{align*}
\]

where \(D\) is a matrix of dimension \((M-1) \times M\) whose entries are defined as

\[
D(i, i) = -1, D(i, i+1) = 1 \quad (5)
\]

and zero otherwise, whereas \(H\) is a matrix of dimension \((M-2) \times M\) whose entries are defined as:

\[
H(i, i) = 1, H(i, i+1) = -2, H(i, i+2) = 1, \quad (6)
\]

and zero otherwise, and \(\hat{w}\) is a rough estimate of the lenticule width. Such estimate can simply be obtained with a Fourier analysis on \(z\). The matrices \(D\), and \(H\) basically perform a first order, and second order respectively, finite difference on the vectors \(t\) and \(b\). The full optimization can be written as:

\[
\begin{align*}
    \min_{t,b} & - \sum_{m=0}^{M} \sum_{h=0}^{H} z(h, b_m + \frac{ht_m}{H}) + \lambda_1 r_1(t, b) + \lambda_2 r_2(t, b), \quad (7)
\end{align*}
\]

where \(\lambda_1\) and \(\lambda_2\) are positive parameters that balance the strength of the regularization terms. Note that the goal of the first regularization step is mainly to avoid fitting multiple vectorial lenticules on the same image location, whereas the second term enforces a certain smoothness on the width of nearby lenticules width. The optimization problem of Eq. equation 7 can effectively be solved with gradient based optimization methods. However, because of the term \(z(\cdot)\), the objective function is non-convex and has multiple local minima. As a result, it is important to provide a good initial guess to the gradient method in order to avoid being trapped in a local minimum. Fortunately, we observed that a simple peak detector applied to the output of the network to initialize \(t\) and \(b\) provides a rough estimate that is very effective, i.e., we did not observe any failure cases.

The optimal values of \(t\) and \(b\) can be used to extract the RGB values from the input image. We simply need to iterate through all the entries of the vectors, which define the different lenticules, and extract the pixel values corresponding to the red, green and blue colors, see Fig. 5. By carrying out this operation for all the rows of the input image \(x\), of shape \(H \times\)
Fig. 4: Overall pipeline of the proposed method. First the image is passed through a CNN for a raster lenticule boundaries detection. The prediction goes through a vectorial extraction and refinement, at this point the RGB values for each lenticule are extracted. Finally a Demosaicing network takes care of filling the voids in the RGB channels and obtain a fully colorized image.

Fig. 5: RGB value extraction. Left: detail of an input grayscale image with the RGB location within each lenticule highlighted. Right: The RGB columns are extracted and used to build the striped color image.

W, we obtain a new image $H \times 3(M - 1) \times 3$. Note that $M$ is the number of detected lenticule boundaries and we extract only lenticules that are entirely contained in the frame. The newly constructed image has three channels and while it has the same number of rows, the width is reduced because there are only three pixels per lenticule width. In order to rescale the image to the original proportion, we down-sample the image vertically to $H' = \frac{3(M - 1)H}{W}$. We then obtain an image of shape $H' \times 3(M - 1) \times 3$ that has the same proportions of the input image and only one color channel is defined for each column.

C. Color Reconstruction

One trivial way to fill the voids in the color channels is to use some conventional interpolation technique, such as linear or cubic, between the valid pixels of each color channel. We instead decide to pursue an alternative approach, which proved to be effective in similar scenarios. It consists of training a CNN in order to fill the voids, similarly to what is done in recent super-resolution and demosaicing methods [13], [14]. Since we do not have a sufficient amount of analog reconstructed color images from lenticular films to train a neural network, we need to find an alternative training procedure. Fortunately, it is rather straightforward to create striped RGB images from fully colored ones. It is in fact sufficient to delete two RGB channels for each column of the image to replicate images that resemble the output of the RGB channel extraction module in our pipeline (see Fig. 4). We can then use this synthetic dataset to train a network that reconstructs the fully colored image. At test time we can use the trained network to colorize the lenticular images. It is important to perform a proper data augmentation during training though to be sure that a potential domain shift between training set and test set does not hurt model performance. To have a more robust network, we perturb images with additive noise, and by randomly changing the contrast and brightness of the training images.

A straightforward colorization network implementation would simply output a 3-channel image with the same spatial dimension of the input. In that case, however, the network would be able to colorize each pixel of the image with any possible color. Since we are still concerned about preserving the historical truthfulness of the colors, such a network architecture could potentially alter the hue of the picture. To alleviate this problem, we design a network that can colorize an output pixel by using exclusively a convex-combination of the closest valid pixels of the same color. We can think of this as the network is predicting the interpolation coefficients to reconstruct each pixel color (Fig. 6). In our design, we decide to use the six closest neighbors of the same color for each output pixel.\footnote{On a pixel column where one of the RGB values is available, we exclusively use the color values located on that column.}
Fig. 6: Detailed diagram of the Color reconstruction process. For each pixel, and for each color channel, the network outputs six positive weights that sum to one, these values represent the interpolation coefficients. For each pixel location in the input image the 6 closest values for each channel are extracted and concatenated in the feature dimension creating a feature map of shape $H' \times W' \times 6$ for each color channel. By performing a dot products in the feature dimension for each color channel we obtain the final RGB triplet.

In order to train the network we use a combination of $l_1$ loss and adversarial loss:

$$\min_{\theta} \max_{\varphi} \gamma \mathbb{E}_{x,y_B}[l_1(y(x), y_B)] + \mathbb{E}_{x}[\log(1 - f_D(y(x)))] + \mathbb{E}_{y_B}[\log(f_D(y_B))],$$

where $\theta$ represent the learning parameters of the colorization network, $\varphi$ the parameters of the discriminator network, and $\gamma$ is a weight parameter that balances the two losses. Using an adversarial loss allows us to improve the quality of the reconstructed image as we are able to obtain sharper output images.

V. EXPERIMENTS

A. Datasets

Our lenticular dataset consists of 2010 scanned frames of 12 lenticular films. Each frame is cropped to exclude the film perforations from the image. The average size of the images is about $2550 \times 3650$ pixels. Lenticule width can range from 12 to 20 pixels depending on the spatial resolution of the scan. This dataset is composed of images that were correctly colorized by doLCE [16]. Segmentation label masks that identify the lenticule boundaries are available alongside the grayscale input images. Note that such boundaries may not always perfectly correspond to the real ground truth locations, but nevertheless represent predictions that are sufficiently good to reconstruct the image colors. Because doLCE can only deal with perfectly vertical lenticules, we perform random rotations to train the network to generalize to tilted lenticules. We split the dataset into train, validation, and test subsets with 1810, 100, 100 images in each subset, respectively. During the training of the lenticule detection network, images are cut into patches of $256 \times 256$ pixels and a random rotation of maximum $1^\circ$ is applied to the input-output pair. No rotation is applied during test time and the input image is cut into patches, processed by the network, and then re-combined to form the whole image prediction. Moreover, we have access to additional images where the doLCE algorithm led to unsuccessful colorization as it failed to detect the correct lenticule boundaries. In these scans the lenticules were either tilted so that the vertical fit of doLCE performed poorly, or the images presented some challenging parts (e.g., extended dark areas) which would require the usage of some regularization technique. These additional images will be used to verify whether the proposed method is indeed more robust than the doLCE algorithm.

To train the colorization network we leverage a second dataset div2K [19]. This dataset is composed of 1000 high resolution RGB images with a large diversity of contents. We first randomly crop the images with patches of $256 \times 256$ pixels and artificially create a stripe-like structure by deleting two of the RGB channels for each column to obtain the same pattern of the original lenticular images. Random additive noise, contrast variation, and brightness variation of the images are performed during training to make the colorization network more robust. At test time, the model is used to colorize the lenticular images, for which no real groundtruth colorized version exists. The input images are cut into patches and then recombined to obtain a full image prediction.

B. Implementation Details

The lenticule detection network has a U-Net structure [20] with a ResNet encoder [21] pretrained on ImageNet [22]. The model has been implemented using PyTorch [23] and the Segmentation Models Package [24]. The architecture of the colorization network is the same as the one of the lenticule detection network. Only the final layer changes as in this case the output is not a binary segmentation map but the interpolation coefficient to reconstruct the color of each pixel. The two networks are trained in the same way: we use the Adam optimizer [25] for 50 epochs with an initial learning rate of 0.001 for the lenticule segmentation network and 0.005 for the colorization network. The learning rate is decreased by half every 10 epochs for both networks. The $\gamma$ hyperparameter of Eq. equation 8 is set to 10. The rough lenticule width estimate $\hat{w}$ is obtained via a spectral analysis of the lenticule segmentation map in the horizontal direction. The spectrum shows a clear peak at the frequency corresponding to the periodical pattern of lenticular film. The optimization problem of Eq. equation 7 is solved using the L-BFGS-B method [26] implemented in the scipy library [27]. The hyperparameters of the regularization terms are set to $\lambda_1 = 1$. and $\lambda_2 = 10$. As we
do not have access to actual lenticule locations groundtruth, it was not possible to perform any objective maximization for the selection of the hyperparameters. As a result the values were selected by a subjective analysis of the color reconstruction results, seeking for the values that led to lesser artifacts. The output of this optimization is used to extract the RGB values inside each lenticule. We vertically interpolate the resulting image to maintain the same proportions like the input image as described in Section IV. We further apply a median filter in the vertical direction before the interpolation to reduce noise.

C. Lenticules Detection

In Fig. 7 we show an example of lenticule boundary detection and their refinement. The network can detect lenticule boundaries where they are visible and is robust to low contrast areas, too. In some cases, boundary detection becomes blurry (bottom left of Fig. 7(c)) and too inaccurate for correct color reconstruction. Our refinement module successfully corrects for these effects by fitting straight lines (Fig. 7(d)) vertically across the entire image.

Although we use the lenticule boundary predictions of doLCE to train the segmentation network, it is not entirely meaningful to use those in order to evaluate our method quantitatively. Recall that our goal is not to build a network that could replicate the old method but rather to create a network that is more robust, and able to deal with pictures where the old method failed to reconstruct the colors. The best way to validate successful lenticule detection of our approach is to look at images where doLCE failed. We show three of such examples in Fig. 8 where our proposed method is able to correctly predict the lenticule boundaries even in images where the doLCE algorithm failed. In this case the failures can be associated to different causes. For the first two images (top and center) the artifacts are caused by the tilted lenticules. When lenticules that are slightly tilted are fitted with vertical ones we usually observe a color shift (green in this case) from the bottom to the top of the image. This is caused by the fact that if the detection is correct at the bottom (top) of the image it gets progressively misaligned when moving up (down), causing a color shift. The artifact in the bottom image is instead of a different nature. We speculate that the high brightness of the sun behind the pole causes some confusion in the doLCE lenticule detection, whereas for the proposed model, the CNN is able to learn from the successful predictions how the lenticule grid usually looks like and detects the lenticule boundaries correctly.

Finally, we analyze how the lenticule width changes within the images. Figure 9 shows the horizontal variation of the average lenticule width for three images of the dataset.

![Fig. 7: (a) input image; (b) detail of the input image; (c) lenticule boundaries predicted by the network; (d) lenticule boundaries after the refinement module.](image_url)

![Fig. 8: (a) colorized images using the doLCE algorithm; (b) same images colorized using the lenticule detection pipeline proposed in this work.](image_url)

![Fig. 9: Horizontal variation of the average lenticule width for three images of the dataset.](image_url)
one image along the horizontal direction. In Fig. 9 we show the evolution of the lenticule width for some images of the dataset. As it can be seen, the lenticules width is not constant and reveals a clear low frequency component where in some parts of the image lenticules are consistently larger (or smaller) than the average. We observed similar patterns for several images of the dataset. Because of these low frequency variations we cannot have a fixed lenticule width for the entire image as by doing so, a progressive misalignment would build up, leading to severe colorization artifacts.

D. Color Reconstruction

To evaluate the digital colorization process, we compare our proposed method (Fig. 10(e)) with three other methods: the doLCE colorization process (Fig. 10(b)), nearest interpolation (Fig. 10(c)), and cubic interpolation (Fig. 10(d)). Note that doLCE spreads the same RGB value over the entire lenticule width creating a visible stripy patterns where pixels throughout the horizontal direction of a lenticule have the same color value (Fig. 10(b)). Nearest interpolation slightly improves the quality of color reconstruction at the cost of losing details (Fig. 10(c)) that appear in the original grayscale image versions (Fig. 10(a)). Our proposed learning-based method (Fig. 10(e)) is able to preserve most details and provides overall sharper images compared to bicubic interpolation (Fig. 10(d)).

Unfortunately it is not possible to have an objective evaluation of the different methods since we do not have access to a groundtruth reference images. We can however design a user study where we ask participants to evaluate which method in their opinion leads to the best looking image. To do this we selected 108 images, from the test set plus some failures cases, we then displayed 4 color reconstructed images: the former doLCE colorization, the proposed lenticule detection network with cubic interpolation, the proposed lenticule detection network with nearest interpolation, and the full proposed method with the colorization network. For each single frame we asked the users to select, in their opinion, which color reconstruction looked better, according to level of details, sharpness, and absence of color artifacts. In total 1232 frames were rated by 33 participants. The results of the user study are shown in Fig 11. As it can be seen users selected about 60% of the time the proposed method as the one generating the best looking picture. As expected the cubic interpolation ranked second, as it was selected as the best one about 30% of times, followed by the doLCE method and nearest interpolation.

In Fig. 12 we show some failure cases for the colorized images. In these images some color shifting is visible mainly the sky regions. This is caused by a misalignment between the actual lenticules and the predicted ones. In some cases large blue areas still pose a challenge for the overall method as the red color magnitude is much smaller than the blue one, and this makes the lenticule boundary look like it would be located
In this work, we proposed a new, learning-based method for color reconstruction of historical, lenticular films. A prerequisite for truthfully reconstructing the colored image is the accurate detection of the lenticule boundaries. Our method is thus composed of two consecutive steps: (i) a first deep neural network segments the individual lenticules on a scanned lenticular film. We circumnavigate manual ground truth annotations by training on a dataset where the labels are obtained automatically for artifact-free image samples using doLCE [16]. (ii) After the lenticule locations are refined and the RGB color triplets are extracted, a second deep neural network, trained on a synthetic dataset made of natural images, reconstructs the full color images by learning how to interpolate nearby valid color pixels. The conducted experiments show the validity of the proposed method compared to other existing and baseline methods for lenticular film color reconstruction. Due to the lack of colored groundtruth images, we conducted a user study to evaluate subjectively the proposed colorization technique: the study revealed that the proposed method was largely preferred among the considered methods.

ACKNOWLEDGMENTS

The present work leverages the fundamental concepts developed by Rudolf Gschwind and Joakim Reuteler, who initiated the doLCE project. The research would not be possible without the trust and support from Barbara Flueckiger. Special thanks go to the institutions and private collectors who provided the films used in the experiments: Agentur Karl Höffkes, Stiftung Deutsche Kinemathek, Cinémathèque suisse, Lichtspiel / Kinemathek Bern, Ralf Klee, Haus des Dokumentarfilms Stuttgart.

APPENDIX A

COLOR SPACE ESTIMATION

During the software color reconstruction the encoded color on the lenticular film must be assigned to the proper RGB space. The lenticular RGB space is determined by the transmittances of the color filter originally used in analog projection. Transmission spectra (Fig. 14 left) were measured with a double-beam spectrophotometer (Shimadzu UV-1800) and colorimetric calculations were performed [28] considering the 1931 CIE 2° standard observer [29] and the irradiance spectrum of a typical film projector (Kinoton FP38), which was measured with a spectroradiometer (Konica Minolta CS-2000). The resulting chromaticity values of the color primaries and the corresponding whitepoint are reported in the CIE 1976 UCS diagram (Fig. 14 right) [30].

In order to generate image files that convey the proper color information, the color values of the lenticular RGB space must be converted into a standard RGB space for correct visualization. The whitepoint of the lenticular RGB space (\(XYZ = [0.991, 1, 1.315]\)) does not correspond to any standard whitepoint, therefore a chromatic adaptation transform (CAT) is necessary [31]. In view of the linear von Kries model [32], the adaptation scaling must be performed at the cone response.
Fig. 13: Comparison between (a) input image, (b) analog reconstruction, and (c) color reconstruction with our proposed method.

Fig. 14: Left: Transmittance spectra of the color filter for lenticular film projection (inset picture). Right: Chromaticity values corresponding to red, green and blue filter’s sectors (circles) and corresponding white point (cross).

Fig. 15: Five steps of the RGB space conversion with chromatic adaptation transform.

Fig. 16: Two examples of color space conversion for correct visualization.

E). The resulting matrix is the following:

\[
\begin{pmatrix}
0.789 & 0.154 & 0.057 \\
-0.286 & 1.195 & 0.06 \\
-0.049 & 0.035 & 1.035
\end{pmatrix}
\] (9)

Two examples of images processed with the space conversion matrix are reported in Fig. 16.

REFERENCES
[1] M. Gordon, “Lenticular spectacles: Kodacolor’s fit in the amateur arsenal,” Film History, 2013.
[2] F. Stanco, G. Ramponi, and A. de Polo, “Towards the automated restoration of old photographic prints: a survey,” in The IEEE Region 8 EUROCON. Computer as a Tool., 2003.
[3] R.-C. Chang, Y.-L. Sie, S.-M. Chou, and T. Shih, “Photo defect detection for image inpainting,” in IEEE International Symposium on Multimedia, 2005.
[4] V. Bruni and D. Vitulano, “A generalized model for scratch detection,” IEEE Transactions on Image Processing, 2004.
[5] J. Yu, Z. Lin, J. Yang, X. Shen et al., “Generative image inpainting with contextual attention,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018.

[6] ——, “Free-form image inpainting with gated convolution,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019.

[7] Z. Wan, B. Zhang, D. Chen, P. Zhang et al., “Bringing old photos back to life,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020.

[8] ——, “Old photo restoration via deep latent space translation,” arXiv preprint arXiv:2009.07047, 2020.

[9] S. Anwar, M. Tahir, C. Li, A. Mian et al., “Image colorization: A survey and dataset,” 2020.

[10] A. Jason, “Deoldify,” [https://github.com/jantic/DeOldify] 2018.

[11] R. Kimmel, “Demosaicing: image reconstruction from color ccd samples,” IEEE Transactions on Image Processing, 1999.

[12] X. Li, B. Gunturk, and L. Zhang, “Image demosaicing: A systematic survey,” in Visual Communications and Image Processing 2008. International Society for Optics and Photonics, 2008.

[13] M. Gharbi, G. Chaurasia, S. Paris, and F. Durand, “Deep joint demosaicking and denoising,” ACM Transactions on Graphics, 2016.

[14] G. Qian, J. Gu, J. S. Ren, C. Dong et al., “Trinity of pixel enhancement: a joint solution for demosaicking, denoising and super-resolution,” arXiv preprint arXiv:1905.02538, 2019.

[15] Y. Zhang, K. Li, K. Li, B. Zhong et al., “Residual non-local attention networks for image restoration,” arXiv preprint arXiv:1903.10082, 2019.

[16] J. Reuteler and R. Gschwind, “Die farben des riffelfilms : digitale farbrekonstruktion von linsenrasterfilm,” in Rundbrief Fotografie, 2014.

[17] G. Trumpy, S. D’Aronco, J. D. Wegner, and J. Reuteler, “deep-doLCE. A Deep Learning Approach for the Color Reconstruction of Digitized Lenticular Film,” in Colour Photography and Film - Conference Proceedings. Associazione Italiana Colore, 2021.

[18] J. G. Capstaff, O. E. Miller, and L. S. Wilder, “The projection of lenticular color-films,” Journal of the Society of Motion Picture Engineers, 1937.

[19] E. Agustsson and R. Timofte, “Ntire 2017 challenge on single image super-resolution: Dataset and study,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2017.

[20] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical image computing and computer-assisted intervention. Springer, 2015.

[21] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.

[22] J. Deng, W. Dong, R. Socher, L.-J. Li et al., “Imagenet: A large-scale hierarchical image database,” in IEEE Conference on Computer Vision and Pattern Recognition, 2009.

[23] A. Paszke, S. Gross, F. Massa, A. Lerer et al., “Pytorch: An imperative style, high-performance deep learning library,” in Advances in Neural Information Processing Systems 32. Curran Associates, Inc., 2019.

[24] P. Yakubovskiy, “Segmentation models pytorch,” GitHub repository, 2020.

[25] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in 3rd International Conference on Learning Representations, 2015.

[26] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, “A limited memory algorithm for bound constrained optimization,” SIAM Journal on scientific computing, pp. 1190–1208, 1995.

[27] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland et al., “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” Nature Methods, 2020.

[28] “Recommended practice for tabulating spectral data for use in colour computations,” CIE International Commission on Illumination, Tech. Rep., 2005.

[29] “Colorimetry — part 1: Cie standard colorimetric observers,” CIE International Commission on Illumination, Tech. Rep., 2019.

[30] R. W. G. Hunt and M. R. Pointer, Measuring colour. John Wiley & Sons, 2011.

[31] N. Moroney, M. D. Fairchild, R. W. Hunt, C. Li et al., “The ciecam02 color appearance model,” in Color and Imaging Conference, 2002.

[32] M. H. Brill, “The relation between the color of the illuminant and the color of the illuminated object,” Color Research & Application, 1995.