Layout-Aware Single-Image Document Flattening

PU LI, MAIS, Institute of Automation, CAS and School of Artificial Intelligence, UCAS, China
WEIZE QUAN, MAIS, Institute of Automation, CAS and School of Artificial Intelligence, UCAS, China
JIANWEI GUO*, MAIS, Institute of Automation, CAS and School of Artificial Intelligence, UCAS, China
DONG-MING YAN, MAIS, Institute of Automation, CAS and School of Artificial Intelligence, UCAS, China

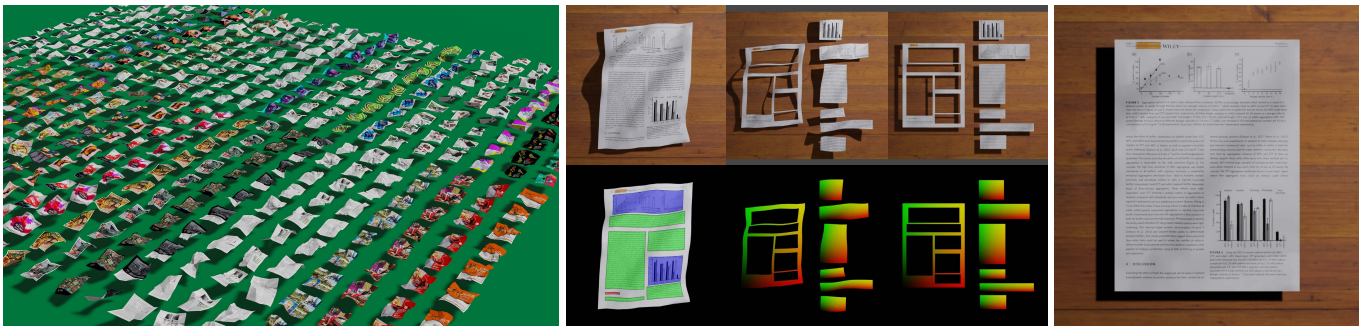

Fig. 1. Left: the large-scale dataset of deformed papers generated by our simulation method. Right: by analyzing the layout of the deformed document image, we propose a divide-and-conquer approach to rectify the document image. After predicting the global and local UVs (Middle) using neural networks, we merge the two and restore the flat paper image.

Single image rectification of document deformation is a challenging task. Although some recent deep learning-based methods have attempted to solve this problem, they cannot achieve satisfactory results when dealing with document images with complex deformations. In this paper, we propose a new efficient framework for document flattening. Our main insight is that most layout primitives in a document have rectangular outline shapes, making unwarping local layout primitives essentially homogeneous with unwarping the entire document. The former task is clearly more straightforward to solve than the latter due to the more consistent texture and relatively smooth deformation. On this basis, we propose a layout-aware deep model working in a divide-and-conquer manner. First, we employ a transformer-based segmentation module to obtain the layout information of the input document. Then a new regression module is applied to predict the global and local UV maps. Finally, we design an effective merging algorithm to correct the global prediction with local details. Both quantitative and qualitative experimental results demonstrate that our framework achieves favorable performance against state-of-the-art methods. In addition, the current publicly available document flattening datasets have limited 3D paper shapes without layout annotation and also lack a general geometric correction metric. Therefore, we build a new large-scale synthetic dataset by utilizing a fully automatic rendering method to generate deformed documents with diverse shapes and exact layout segmentation labels. We also propose a new geometric correction metric based on our paired document UV maps. Code and dataset will be released at https://github.com/BunnySoCrazy/LA-DocFlatten.

CCS Concepts: • **Computing methodologies** → **Image manipulation**.

Additional Key Words and Phrases: Document Image Rectification, Document Layout Analysis, Deep Neural Networks, Geometric Models

ACM Reference Format:
Pu Li, Weize Quan, Jianwei Guo, and Dong-Ming Yan. 2023. Layout-Aware Single-Image Document Flattening. *ACM Trans. Graph.* XX, XX, Article XX (December 2023), 18 pages.

*Corresponding author: Jianwei Guo (jianwei.guo@nlpr.ia.ac.cn)

Authors' addresses: Pu Li, MAIS, Institute of Automation, CAS and School of Artificial Intelligence, UCAS, China; Weize Quan, MAIS, Institute of Automation, CAS and School of Artificial Intelligence, UCAS, China; Jianwei Guo, jianwei.guo@nlpr.ia.ac.cn, MAIS, Institute of Automation, CAS and School of Artificial Intelligence, UCAS, China; Dong-Ming Yan, MAIS, Institute of Automation, CAS and School of Artificial Intelligence, UCAS, China.

0730-0301/2023/12-ARTXX $15.00
https://doi.org/

1 INTRODUCTION

With the rapid development and popularity of portable photographing devices, *e.g.*, cameras and smartphones, photography has become the primary method to digitize documents. Compared with documents scanned by a flatbed scanner under fixed working conditions, camera-captured document images often suffer from various distortions and uneven illumination. This phenomenon can be explained by three aspects. (1) Paper documents often have physical deformations such as wrinkling and bending. (2) The position and angle of the handheld camera may cause the document image surface to be non-orthogonal to the line of sight. (3) The lighting conditions of the shooting scene are complicated. These undesirable drawbacks of document images inevitably affect the user's reading experience and harm the downstream tasks, such as optical character recognition (OCR), document layout analysis (DLA), *etc.* Consequently, document image rectification has attracted increasing attention and become an important topic for many document-related tasks.

Previous conventional approaches mainly follow two research directions, *i.e.*, 2D image processing and 3D shape reconstruction. Document unwarping based on 2D image processing estimates the distortion by extracting the low-level features in the document images. These specific features usually contain letter/word [Fawzi et al. 2015; Zandifar 2007], boundaries [Brown and Tsoi 2006; Tsoi and Brown 2007], and text lines [Kim et al. 2015; Lavialle et al. 2001; Mischke and Luther 2005]. However, the performance of these methods heavily depends on the quality of extracted low-level features.

Document unwarping based on 3D shape reconstruction first recovers the 3D surface of document images and then maps it onto a plane. Some existing methods depend on specific hardware devices, such as multi-camera 3D scanning system [Brown et al. 2007], laser range scanner [Zhang et al. 2008], and structured light [Brown and Seales 2001; Sun et al. 2005]. However, these expensive equipment requirements and specific lighting conditions may limit their applicability. Due to these limitations, some other approaches reconstruct the 3D shape surface based on the simplified parametric model assumption. These methods are either designed using the shading information [Courteille et al. 2007; Tan et al. 2006; Wada et al. 1997], or using the visual cues [He et al. 2013; Liang et al. 2008; Meng et al. 2018]. This kind of approach often suffers from non-trivial optimization costs and achieves limited performance due to complex distortions and inaccurate visual cue detection.

Recently, deep learning-based methods have been proposed to rectify the warped document and achieve promising performance. These methods usually model the unfolding process by estimating a pixel-wise displacement flow. Different aspects have been focused on by these approaches: 3D coordinate regression [Das et al. 2019; Markovitz et al. 2020], foreground/background partition [Xie et al. 2021a], iterative optimization [Feng et al. 2021b], and local patch [Feng et al. 2021a; Li et al. 2019]. Although previous methods perform well for simple deformations, they tend to be less effective when unwrapping document images with complex deformations. This situation can be attributed to the following problems. First, most previous networks do not take advantage of the layout information provided by the document content. Second, the paper shapes of the documents in the current datasets are not diverse and complex enough.

For the network design, we propose a layout-aware rectification method for single-image document flattening. The core idea is to combine the UV maps of the entire document (from the global view) and the layouts (from the local view) to obtain the robust and exact deformation flow estimation, as shown in Fig. 1 (right). Specifically, our network contains a layout segmentation branch, a UV regression branch, and a UV map fusion module. The layout segmentation branch adopts an encoder-decoder architecture to predict the layout of the input document image. The UV regression branch also follows an encoder-decoder design to estimate the UV maps of the entire image and local patches extracted using the predicted layout. The UV map fusion module combines the local and global UV maps to obtain the final deformation flow.

In addition, most existing methods are evaluated on the real image benchmark [Ma et al. 2018], which contains 130 pairs of flat and deformed document images. However, this dataset does not provide strictly pixel-wise correspondence between image pairs, and thus cannot precisely measure the distortion degree of unwarped document images. Furthermore, [Das et al. 2019] introduced a Doc3D dataset via a rendering method. To obtain the 3D meshes of deformed documents, they first captured point clouds with a depth camera, and then extracted the meshes. They also applied simple flipping and random cropping for data augmentation. However, the diversity of these collected meshes is limited, and the collection process of point clouds is also tedious. The dataset contributed by [Li et al. 2019] is designed for the patch-based method, which only contains data pairs of 1,000 complete documents, and the deformation of the documents is relatively simple. To improve the dataset quality, we propose a fully automatic 3D mesh generation method based on a physical simulation model. Some representative samples rendered by our method are shown in Fig. 1 (left). The main contributions of our work include:

- We propose a layout-aware document geometric rectification network based on Transformer. By introducing a novel loss function and adopting a divide-and-conquer strategy in UV space, we design our network to be layout-aware, thereafter allowing for robust estimation of the deformation flow.
- We introduce a paper shape simulation method based on a physical mass-spring system. Instead of the tedious manual scanning, our approach enables a fully automatic synthesis pipeline to efficiently construct a large-scale dataset with diverse paper shapes and exact layout annotations for network training.
- We construct a synthetic document dewarping evaluation dataset along with an improved distortion metric called *Mean Pixel-level Distance* (MPD). It allows us to fairly evaluate geometric correction performance without being disturbed by shadows, document content, and image resolution.

2 RELATED WORK

2.1 Document Rectification

2.1.1 Traditional Methods. As mentioned above, conventional methods for document image dewarping can be classified into two categories: 2D image processing and 3D shape reconstruction.

2D image processing. In the literature, the most used low-level features extracted from document images include letter/word, boundaries, space, and text lines. Ulges *et al.* [2005] corrected the perspective and page curl distortion by estimating the local line spacing and the cell shape around the letters. Stamatopoulos *et al.* [2011] proposed a coarse-to-fine strategy to rectify the document images. This method first restores the large distortions based on detected word and text lines, and then word-level normalization is applied using baseline correction. However, the estimation of the continuous representation of text lines is non-trivial. Kim *et al.* [2015] adopted the discrete representation of text lines and text blocks proposed by [Koo and Cho 2010]. With this simple representation, the document dewarping can be easily modeled as an energy minimization problem. On the basis of the features of text lines and characters, Fawzi *et al.* [2015] rectified the skew and perspective distortions of camera-captured document images. Instead of using text lines themselves, Salvi *et al.* [2015] estimated a more accurate 2D distortion grid from white space lines. Takezawa *et al.* [2017] mainly solved

the perspective distortions on camera-captured document images with the RANSAC algorithm and the Radon transform. However, the detection and perception of low-level features in the deformed document images is non-trivial and thus limits the application of those image processing-based methods.

3D shape reconstruction. The 3D surface of document images can be reconstructed by exploiting the shading variations in a single image [Tan et al. 2006; Wada et al. 1997; Zhang et al. 2009]. In addition, the curved text lines and boundaries are also prevalent cues for guiding 3D surface reconstruction. Cao *et al.* [2003] modeled the document surface as a cylindrical surface, which is estimated using the extracted baselines of text lines. Liang *et al.* [2008] estimated the 3D page shape by using the texture flow fields, which rely on the extracted local text line and vertical character stroke directions. Different from them, Tian *et al.* [2011] utilized the domain knowledge, *i.e.*, line structure and local stroke statistics, to better estimate 2D distortion grids. Meanwhile, they proposed a shape-from-texture formulation to reconstruct 3D surfaces with no restrictions on the shape. In [He et al. 2013], book boundaries, including two horizontal boundary curves and six corners, were used to reconstruct the 3D page shape. Based on the assumption that document pages tend to curve into a developable surface, Meng *et al.* [2014] recovered two 3D curves using two beams and interpolated a developable surface to these two curves. These methods usually add simplification constraints on the reconstructed 3D surface, which limits the performance when unwarping the document image with complex geometric deformations.

2.1.2 Deep Learning-Based Methods. With the advance of deep neural networks, *e.g.*, convolutional neural networks (CNNs), more attention has been paid to deep learning-based solutions for document image rectification. Das *et al.* [2017] applied a CNN model to detect creases and then performed a 2D boundary reconstruction based on polynomial regression. Later, Das *et al.* [2019] proposed DewarpNet, which explicitly regresses the 3D shape of document paper and then estimates the deformation flow field. Instead of learning the distortion flow in the entire image, Li *et al.* [2019] designed a patch-based processing pipeline, including patch partition, patch-level flow estimation, patch stitching, and illumination adjustment. Burden *et al.* [2019] rectified the camera-captured document image with a content-ware rectification framework, which sequentially processed text and non-text regions. To extract the valid structural cues and suppress the interferences of clutter backgrounds and blank margins in the document images, Liu *et al.* [2020] designed an adversarial gated unwarping network. Xie *et al.* [2021b] estimated the reference points and control points to construct the rectification mapping. Similar to the iterative manner in [Dasgupta et al. 2020], Feng *et al.* [2021b] progressively corrected the geometric distortion with a lightweight recurrent architecture. Xie *et al.* [2021a] jointly solved the document image rectification and background removal with a fully convolutional neural network. Das *et al.* [2021] proposed a patch-based method similar to [Li et al. 2019], whereas they divide 3D shape maps and employ a network to stitch pieces. Feng *et al.* [2021a] proposed a Transformer-based framework consisting of a geometric unwarping module and an illumination correction module. The former captures the global context with the self-attention operation to address the geometry distortion, and the latter corrects the illumination distortion to enhance the visual quality. To automatically handle images with wide margins, Zhang *et al.* [2022] first removed the margins and did preliminary dewarping, and then iteratively predicted the displacement flow. Ma *et al.* [2022] collected a training dataset consisting of real photos and for the first time trained a network on a mixture of real and synthetic images. Recently, Feng *et al.* [2023] introduced the first learning-based method specifically designed for rectifying unrestricted document images. Both Jiang *et al.* [2022] and Feng *et al.* [2022] utilized the geometric information of text lines to aid in flattening, where the former also introduced boundary constraints, and the latter predicted 3D shapes as cues for global correction. However, the document layout contains not only text but also tables, lists, and pictures. Therefore, compared with the methods using text lines, our approach exploits more types of document content, thus it is still applicable when there is a lack of text in the document.

Ma *et al.* [2018] developed an end-to-end network called DocUNet, which is designed to digitally flatten a distorted document image by stacking two U-Net networks to regress pixel-wise moving vectors. However, their training dataset was constructed by warping the mesh in a 2D plane, which may not accurately capture the realistic creases and real-world lighting and shadow information due to the lack of dimensionality. In contrast, we use a Transformer-based network and simulate the paper distortion in 3D space. [Das et al. 2019] also constructed a Doc3D dataset, including 3D meshes and 2D images. Their 3D meshes are manually captured via scanning and have not very rich shapes. By contrast, we propose a fully automatic rendering pipeline to produce deformed document images with a simulated mesh generation method. Furthermore, different from existing methods that also utilize the forward flow [Li et al. 2019; Ma et al. 2018; Xie et al. 2021a], we propose a layout-aware document flattening network, which simultaneously perceives the global and local geometric deformation. Our design mitigates the challenge of predicting the UV map and enhances the robustness of deformation flow estimation.

2.2 Document Layout Analysis

Next, we briefly review some recent works related to document layout analysis (DLA). We refer readers to the survey [Binmakhashen and Mahmoud 2019] for more details. Borges Oliveira *et al.* [2017] first embedded the pre-segmented document blocks into 1D signatures, and then introduced 1D CNN to identify the types of these blocks. DLA can also be regarded as a kind of semantic segmentation, and thus fully convolutional network (FCN) [Long et al. 2015] is adapted to DLA task [He et al. 2017; Wu et al. 2021]. He *et al.* [2017] proposed a multi-scale, multi-task FCN to produce page segmentation and contour detection jointly. They also introduced a conditional random field (CRF) to improve the final segmentation results. Wu *et al.* [2021] modified the FCN with a proposed dynamic residual feature fusion module, and designed a dynamic selection mechanism for efficient limited-input fine-tuning. To facilitate the research about DLA, especially deep learning-based methods, Zhong *et al.* [2019] collected a large-scale dataset, PubLayNet,

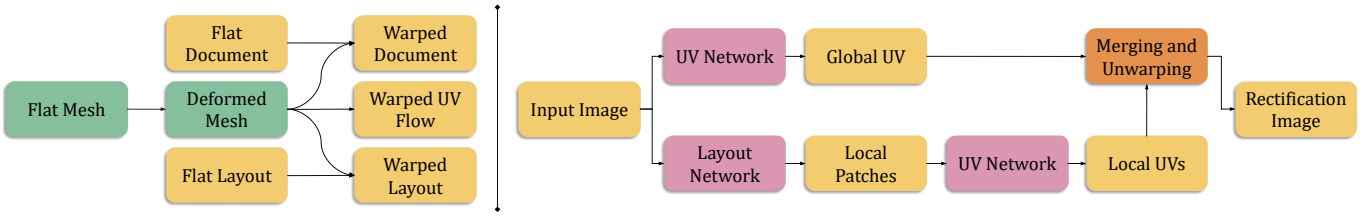

Fig. 2. Overview of dataset construction (left) and geometric rectification (right). Color blocks represent 2D data (yellow), 3D data (green), neural networks (pink), and non-neural processing modules (orange). During the training phase, warped document images are used as the inputs of the neural networks, warped UV flows serve as the labels of the UV Network, while warped layouts are the labels of the Layout Network.

by automatically annotating the layout of over 1 million PDF documents. Due to the tedious pixel-wise segmentation annotations for ancient document layout analysis, Davoudi *et al.* [2021] proposed an unsupervised representation learning method for ancient DLA. A sparse autoencoder is designed to learn the representation of image patches, and then a feed-forward network is applied to classify each pixel into semantic categories. In this paper, we design a document flattening framework with the aid of the results of layout analysis. In addition, we merge the PubLayNet dataset [Zhong et al. 2019] into our deformed document image rendering method.

3 OVERVIEW

3.1 Problem Formulation

Given a distorted document image $\mathbf{S}$ as input, we aim to obtain the flatten version $\mathbf{D}$ by learning a mapping as follows:

$$\mathbf{D}(u, v) = F(\mathbf{S}(x, y)), \tag{1}$$

where (x, y) is the 2D space coordinate, and (u, v) is the texture coordinate of the document in $\mathbf{D}$. The relationship between two coordinates can be represented via a backward mapping F_{bw}:

$$(x, y) = F_{bw}(u, v). \tag{2}$$

Following [Li et al. 2019; Ma et al. 2018], we first predict a forward mapping F_{fw} as an intermediate result. Compared with F_{bw}, F_{fw} describes the mapping of a flat document texture to a warped document image:

$$(u, v) = F_{fw}(x, y). \tag{3}$$

As we assume that all of the document content is visible, there exists a pixel-to-pixel correspondence between the input and output images, making the above two mappings invertible. After converting F_{fw} to F_{bw}, the RGB color of each pixel in the resulting image can be obtained by

$$\mathbf{D}(u, v) = \mathbf{S}(F_{bw}(u, v)). \tag{4}$$

We set the background values in F_{fw} to (-1, -1) to differentiate them from the non-negative values in the foreground. For simplicity, we use UV to denote F_{fw} in the following.

3.2 Our Approach

In this paper, we rectify the single distorted document image with a layout-aware transformer-based framework. Our work covers dataset construction and network design, each of which we briefly describe below.

Dataset Construction. The reviewed literature has not presented publicly available datasets containing the exact 3D meshes and layout segmentation labels. Therefore, we synthesize a large-scale dataset with the rendering tool Blender©. The overview of dataset construction is shown in Fig. 2 (left). Our approach extends the rendering pipeline [Das et al. 2019] with more diverse and controllable 3D paper shape generation and layout segmentation annotation. Specifically, we use the mass-spring system to represent paper shape, and apply random external forces on vertices to simulate the deformation of the paper. We use the document images with fine-grained layout annotations [Zhong et al. 2019] as texture. This scheme allows us to render an infinite number of deformed paper images with corresponding UV labels and layout ground truth.

Geometric Rectification. Our rectification pipeline is shown in Fig. 2 (right), in which the whole process contains three stages. In Stage I, we employ the UV regression network and layout segmentation network to predict the global coarse UV map and document layout, respectively. In Stage II, we utilize the layout segmentation results as masks to extract local patches in the input document image and feed them into the UV network to predict locally accurate UV maps. In Stage III, we merge the global and local UV maps using the Poisson blending approach and transform the combined result into an inverse UV map, which is then used to predict the final flat document image.

4 DATASETS

In this work, we propose a fully automatic rendering method for generating deformed document images. Our dataset is similar to the Doc3D dataset [Das et al. 2019], but with three key differences: (1) The meshes utilized by Doc3D are manually scanned, while our meshes are automatically generated based on a physical simulation model. (2) We introduce the document layout into the rendering pipeline, which is not included in Doc3D. (3) The Doc3D dataset contains 3,000 uniquely shaped document meshes, while ours contains 66,000 distinct meshes. Fig. 3 illustrates our rendering process.

4.1 Mesh Simulation

We use the quadrilateral mesh to represent the shape of the paper. After setting the mesh to a soft body with stiff quads, the mesh becomes a mass-spring model. In a quadrilateral mesh, adjacent vertices are connected by explicit edges, and each pair of opposite corners is connected by virtual edges. The interior force acting on vertex $\mathbf{v}_i$, generated by the structural spring between vertex $\mathbf{v}_i$ and

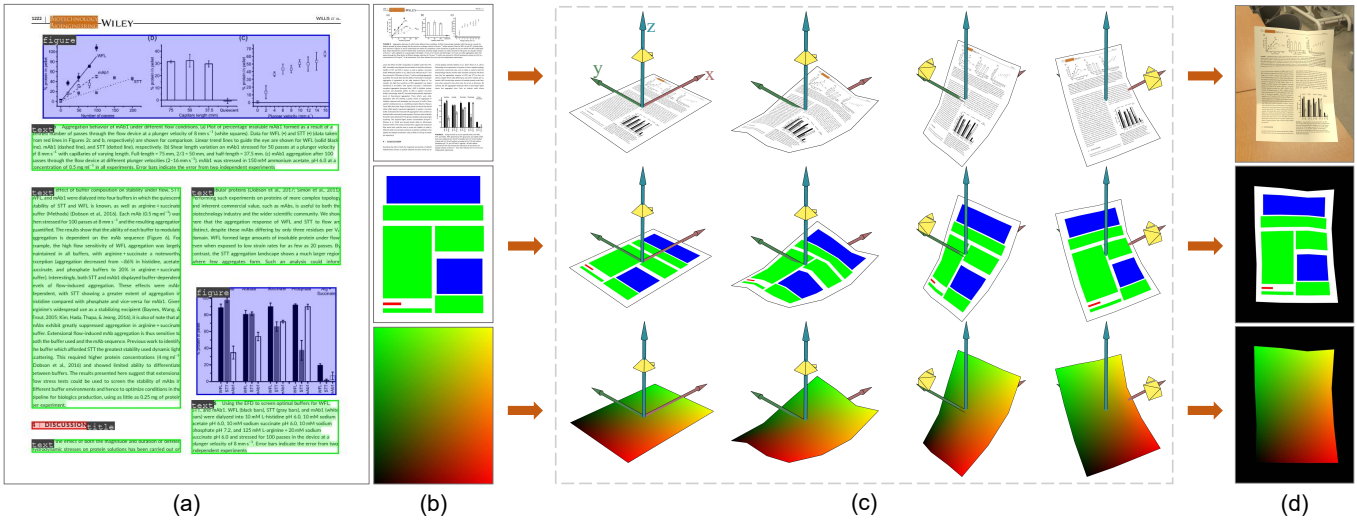

Fig. 3. Given an image and the corresponding layout bounding boxes (a) from [Zhong et al. 2019], we first render the rectangular segmentation map (middle in b) corresponding to the flat document. Then we use the flat document image (upper in b), layout label, and flat UV flow (bottom in b) as the mesh textures to render deformed images. Specifically, we first deform the flat mesh into a crumpled paper-like shape, then rotate along the y-axis to avoid the bottom illegal area, and rotate along the z-axis to obtain various backgrounds. The camera's position and orientation are adjusted in sync with the mesh's transform. The final rendered results contain the deformed paper image, layout label, and UV label (d).

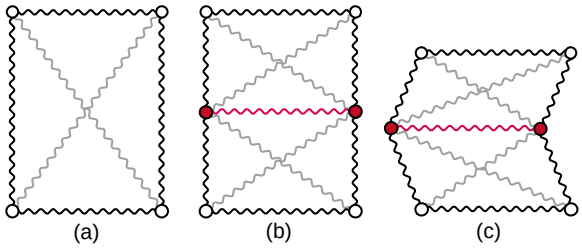

Fig. 4. For a basic quadrilateral mesh under a mass-spring system (a), in addition to the internal forces on the black edges, internal forces also appear on the diagonals, as represented by the gray lines. To increase the number of movable vertices, we add a new edge (red), which acts as a hinge in (b) and brings significant deformation to the mesh (c) when the vertices receive external forces.

Fig. 5. Various mesh shapes generated by our simulation method.

$\mathbf{v}_j$ is given by:

$$\mathbf{f}_{\text{int}}(\mathbf{x}_i, \mathbf{v}_i, \mathbf{x}_j, \mathbf{v}_j) = \frac{\mathbf{x}_{ij}}{||\mathbf{x}_{ij}||}[k_s(||\mathbf{x}_{ij}|| - l_{ij}) + k_d(\mathbf{v}_j - \mathbf{v}_i)\frac{\mathbf{x}_{ij}}{||\mathbf{x}_{ij}||}], \quad (5)$$

where $\mathbf{x}_{ij}$ is the difference between the position vectors $(\mathbf{x}_j - \mathbf{x}_i)$ of two masses, l_{ij} is its rest length, k_s is the spring's stiffness, and k_d is the damping constant. In addition to internal forces, external forces are applied to the mesh to deform it, as shown in Fig. 4. For this purpose, we choose the Turbulence force field because it is random and does not shift the overall position of the mesh. For vertex $\mathbf{v}_i$, the external force it received can be expressed as follows:

$$\mathbf{f}_{\text{ext}}(\mathbf{x}_i) = turbulence(\mathbf{x}_i), \quad (6)$$

where $\mathbf{x}_i$ represents the position of $\mathbf{v}_i$. The turbulence force field in Blender© is implemented based on wavelet noise. We refer the reader to the original work [Kim et al. 2008; Pfaff et al. 2010] for additional details. To simulate complex shapes, we add new edges to the mesh. Without setting bending stiffness, each newly added edge can be regarded as a hinge, bringing a greater range of deformation to adjacent patches. The displacement of the vertex follows the formula:

$$\begin{aligned} \mathbf{f}_i^t &= \mathbf{f}_{\text{ext}}(\mathbf{x}_i^t) + \sum\nolimits_j \mathbf{f}_{\text{int}}(\mathbf{x}_i^t, \mathbf{v}_i^t, \mathbf{x}_j^t, \mathbf{v}_j^t), \\ \mathbf{v}_i^{t+1} &= \mathbf{v}_i^t + \Delta t \cdot \frac{\mathbf{f}_i^t}{m_i}, \\ \mathbf{x}_i^{t+1} &= \mathbf{x}_i^t + \Delta t \cdot \mathbf{v}_i^t. \end{aligned} \quad (7)$$

Following [Ma et al. 2018], we divide the types of paper deformation into two categories: curl and crease deformations. For the curl deformation, we divide the mesh into 8×8 quadrilateral faces

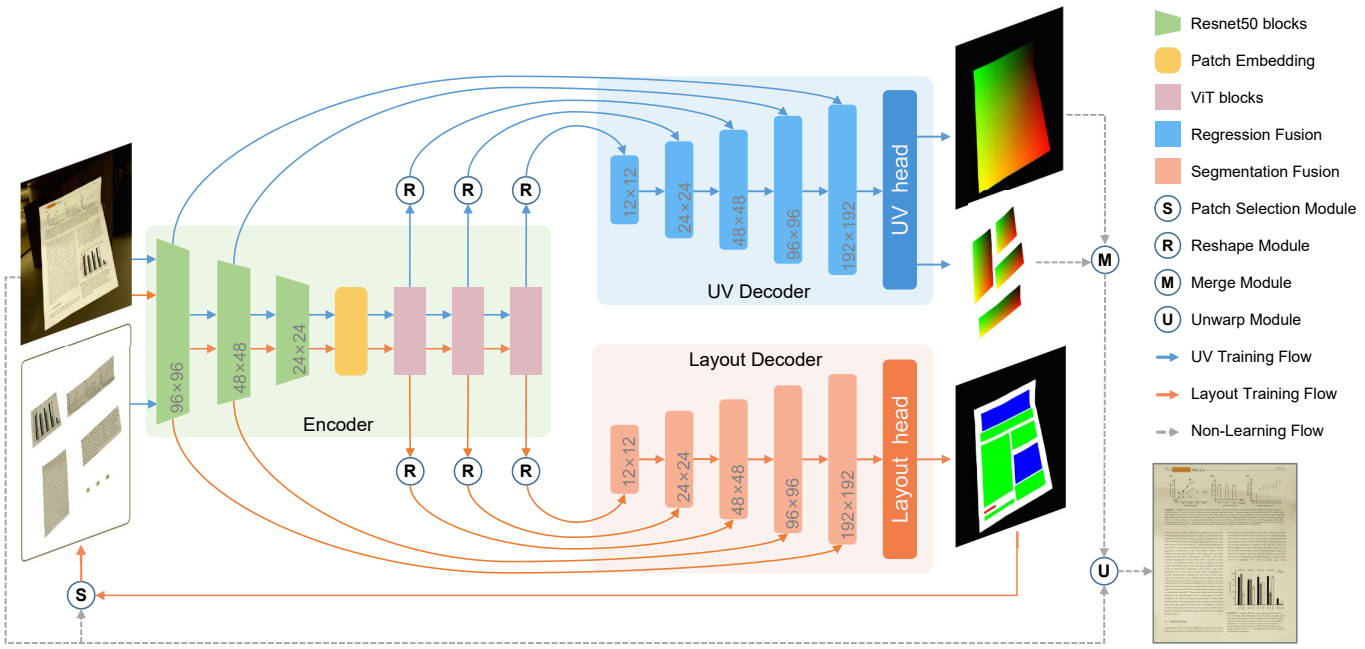

Fig. 6. The orange lines in the figure are the data flows of the layout network, while the blue lines describe the data flows of the UV network, and the gray dotted lines represent the general data flow. The numbers in the color blocks indicate the spatial dimensions of the output feature maps. We only draw one encoder in the figure because the structure of the encoders of the UV network and layout network is exactly the same, while their parameters are independent. Details of the reshape module and components marked with color blocks can be found in Sec. 5.1, and we detail the remaining modules in Sec. 5.2 and Sec. 5.3.

and enable smooth shading when rendering. For the crease deformation, we add edges with random positions and orientations to the mesh, with the number ranging from 5 to 50. The number of faces in the quadrilateral meshes varies from 6 to 951. Fig. 5 shows the mesh shape we generated. Our simulation method can generate 32 meshes per second, which is much more efficient than manual scanning. By setting the strength and the scale of the random force field or the number of edges in the mesh, we can simulate the paper shapes with different degrees of deformation. Furthermore, we can initialize a flat mesh of any size to cover various document sizes in the real world. Details of Blender's configuration and the meshes we generate are available in the supplemental materials.

4.2 Layout Segmentation Label Rendering

In PubLayNet [Zhong et al. 2019], the documents' layout is annotated with bounding boxes and polygonal segmentation. Our goal is to predict rectangular layout primitives in curved documents, therefore, we render all bounding boxes as solid rectangles. The layout of the paper documents is classified into five categories, namely, title, text, table, list, and image, and each category is illustrated with different color labels. Then, by using the document image and the layout segmentation label as the texture of the deformed paper mesh, we render the deformed paper image and the corresponding deformed layout segmentation label. Similar to the process in Doc3D [Das et al. 2019], we randomly sample the environment lights from 2,100 environment maps provided by the Laval Indoor HDR dataset [Gardner et al. 2017] when rendering the document images. In contrast, all external light sources are eliminated when rendering the layout labels. To make full use of the background image and avoid unrealistic black backgrounds when rendering the image, we rotate the paper and camera synchronously. Specifically, we take the center of the paper shape as the origin, the horizontal direction of the paper plane is parallel to the y-axis, the vertical direction is parallel to the x-axis, and the plane normal is $(0, 0, 1)$. We first rotate θ_y along the y-axis, where θ_y is randomly selected from range of $[60°, 90°]$ to avoid illegal areas. Then, we rotate θ_z along the z-axis, ranging from $0°$ to $360°$, to take advantage of different angles of the background. Accordingly, we calculate the corresponding camera position and orientation. For the shape of the paper, we use the scanned 24,000 meshes contributed by Doc3D [Das et al. 2019] and 66,000 meshes generated by our simulation. As a result, we rendered a total of 90,000 pairs of data, in which the shapes of all documents are unique.

5 GEOMETRIC RECTIFICATION

5.1 Network Architecture

As shown in Fig. 6, our network consists of two sub-networks, *i.e.*, the UV network and Layout network, which both adopt the encoder-decoder architectures. The two networks share the same encoder structure, but their decoders are slightly different because they are oriented toward pixel-level regression (UV prediction) and classification tasks (layout segmentation), respectively. To obtain high-resolution fine results, we also adopt a hierarchical refinement scheme, which has been widely used in many tasks [Amirul Islam

et al. 2017; Ding et al. 2020; Lin et al. 2017b,a]. Below we describe the details of our networks.

Hybrid ViT Encoder. Compared with CNNs that focus on capturing local information, the self-attention mechanism [Vaswani et al. 2017] enables vision transformers (ViTs) [Dosovitskiy et al. 2021] to capture long-range dependencies. Our geometric rectification network contains two sub-tasks of pixel-level regression and classification, therefore, we expect our encoder to work with both global contexts and local high-resolution details. We adopt a hybrid vision transformer architecture borrowed from [Dosovitskiy et al. 2021], which employs a ResNet-50 backbone and 12 transformer layers with patch size 16×16. In our network, we move stage 4 of ResNet-50 to stage 3, along with four downsampling layers, and the pixels of each spatial dimension in the final feature map correspond to a 16×16 image patch. Given an input image $S \in \mathbb{R}^{384\times384\times3}$ and denoting the i-th stage of ResNet as R_i, the output feature map after R_3 is $f_{R_3} \in \mathbb{R}^{1024\times24\times24}$. Then, 24×24 tokens with a dimension of 768 are obtained after patch embedding, which will go through 12 transformer layers maintaining the same dimensions. To maintain the high-resolution information, we add several skip-connections from the middle encoding layers to the decoder. Denoting the i-th layer of the Transformer as T_i, we directly extract the features after $\{R_1, R_2, T_6, T_9, T_{12}\}$ to form a multi-scale feature pyramid. While the outputs of ResNet-50 naturally form image-like shapes, the outputs of the Transformer layers are flattened. After reshaping them to restore the spatial dimension information, we employ 3×3 convolutions of strides 1, 2, and 4 to reduce the spatial resolution of feature maps gradually. The spatial resolutions corresponding to the final feature pyramid $\{P_1, P_2, P_3, P_4, P_5\}$ are $\{96 \times 96, 48 \times 48, 24 \times 24, 12 \times 12, 6 \times 6\}$, respectively.

Dual Decoders. For UV prediction and layout segmentation, we employ two structurally similar decoders (right side of Fig. 6). As the spatial resolutions of the feature maps are gradually reduced, we expect our decoder to fuse multi-scale features to obtain high-resolution predictions. Inspired by [Chen et al. 2020; Lin et al. 2017b; Ranftl et al. 2021; Xian et al. 2018], we employ a fusion module that mainly contains residual convolution blocks and pixel-wise summation. Formally, the intermediate result of the l-th fusion layer could be expressed as follows:

$$\begin{cases} ResConv(P^l), & l = 5 \\ ResConv(F^{l+1}_{fused} + ResConv(P^l)). & l < 5 \end{cases} \tag{8}$$

The fusion module further performs bilateral interpolation to increase the spatial dimension of the feature. The final output result of the l-th fusion layer is:

$$F^l_{fused} = proj(UP_2(\dot{F}^l_{fused})), \tag{9}$$

where UP_2 is a bilateral interpolation with a scale of 2, and $proj$ is a 1×1 convolutional layer used to adjust the number of channels of the output feature to align with the next fusion layer. The end head networks contain three convolutional layers to reduce the number of channels and restore the same spatial dimension as the input image. The output of the UV network is $\hat{\mathbf{F}}_{uv} \in \mathbb{R}^{384\times384\times2}$ and the output of the Layout network is $\hat{\mathbf{F}}_M \in \mathbb{R}^{384\times384\times7}$.

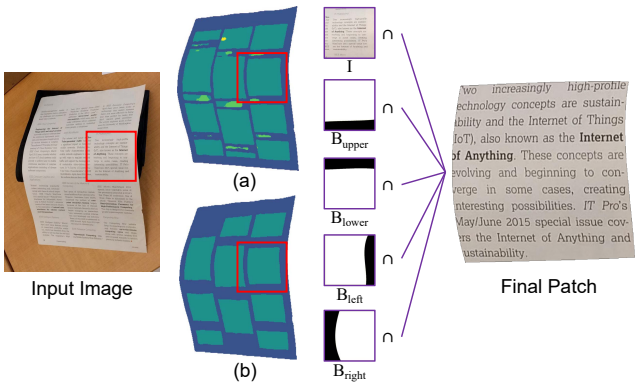

Fig. 7. Illustration of extracting local patch according to layout segmentation (a). The unqualified connected areas are removed in the raw segmentation, and the clean ones are left in (b).

Loss Function. We define two objective terms for UV prediction sub-task, the first one is an element-wise $\mathcal{L}_1$ loss, which measures the distance between the predicted result $\hat{\mathbf{F}}_{uv}$ and ground truth $\mathbf{F}_{uv}$:

$$\mathcal{L}_{distance} = \left\|\hat{\mathbf{F}}_{uv} - \mathbf{F}_{uv}\right\|_1. \tag{10}$$

Additionally, we propose a novel layout-aware loss term encouraging the network to capture document layout:

$$\mathcal{L}_{layout} = \left\|\hat{\mathbf{F}}_{layout} - \mathbf{F}_{layout}\right\|_1, \tag{11}$$

where $\hat{\mathbf{F}}_{layout} \in \mathbb{R}^{384\times384\times1}$ is the layout segmentation converted from $\hat{\mathbf{F}}_{uv}$ and $\mathbf{F}_{layout}$ is the ground truth. The flat layout contains P units of $\{i \in P \mid \mathbf{p}_i, \mathbf{t}_i\}$, where $\mathbf{p}_i \in \mathbb{R}^2$ is the height and width of the layout unit and $\mathbf{t}_i \in \mathbb{R}^2$ describes the location of the layout unit. We obtain $\mathbf{p}_i$ and $\mathbf{t}_i$ from the ground truth of a flat layout which is provided by [Zhong et al. 2019], and $\mathbf{F}_{layout}$ is rendered during the dataset construction. We first calculate the signed distance field for each layout unit in UV space:

$$\begin{aligned} \hat{\mathcal{D}}_i &= dist(\hat{\mathbf{F}}_{uv} - \mathbf{t}_i; \mathbf{p}_i) \\ &= \left\|\max(\left|\hat{\mathbf{F}}_{uv} - \mathbf{t}_i\right| - \mathbf{p}_i, 0)\right\| \\ &\quad + \min(\max(\left|\hat{\mathbf{F}}_{uv} - \mathbf{t}_i\right| - \mathbf{p}_i), 0). \end{aligned} \tag{12}$$

We then convert the signed distance field to layout segmentation:

$$\hat{\mathbf{F}}_{layout} = \sum_i^P \Phi(-\eta \cdot \hat{\mathcal{D}}_i) \cdot c_i, \tag{13}$$

where Φ is a sigmoid function, η is a hyper-parameter that controls the sharpness of the layout boundary, c_i is the category to which the i-th layout unit belongs. Note the transformation process from $\hat{\mathbf{F}}_{uv}$ to $\hat{\mathbf{F}}_{layout}$ is fully differential. The overall objective of our UV network is defined as the combination of the above two terms:

$$\mathcal{L}_{uv} = \mathcal{L}_{distance} + \lambda \cdot \mathcal{L}_{layout}, \tag{14}$$

where λ is a balance factor.

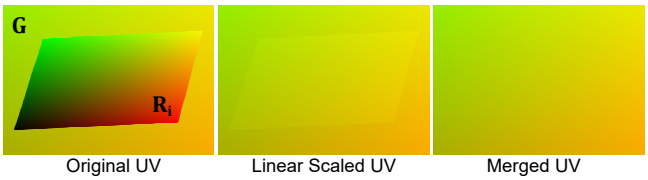

Fig. 8. An example of UV merging. The original local UV predicts the relative flow. We first linearly scale it to the global flow, and then apply Poisson blending to produce a seamless merged UV.

For the layout segmentation sub-task, we employ the cross-entropy loss as the semantic segmentation loss:

$$\mathcal{L}_{seg} = -\frac{1}{N}\sum_{j}^{N}\sum_{c=1}^{M} y_{j,c}\log(p_{j,c}), \tag{15}$$

where $N = 384 \times 384$ is the total number of pixels and $M = 7$ is the total number of segmentation categories, $p_{j,c}$ is the predicted probability that pixel j belongs to c class, and $y_{j,c}$ is a binary indicator (0 or 1) indicating whether pixel j belongs to c or not.

5.2 Merging Global UV and Local UV

To merge the local and global UV maps, we first extract the valid patch according to predicted layout segmentation with two steps (see Fig. 7), namely, patch selection and contour fitting.

Patch Selection. Among the layout segmentation results, each connected component is a candidate patch mask. As we have assumed that each primitive of the document layout is rectangular after flattening, segments that do not match this assumption should be eliminated. A valid patch mask should satisfy the following conditions: (1) it is a solid region without holes; (2) it has four distinct inflection points; and (3) it does not contain slender branches. We use a series of simple but effective operations to convert and filter each segment according to the above conditions. We denote the segmentation result as S, a 5×5 square structure element as K_5, and the i-th connected component of $S \circ K_5$ as S_i. We use the closing operation to fill in the inner holes, *i.e.*, $P_i = S_i \bullet K_{10}$. Denote the convex hull of P_i as H_i, we calculate the ratio of areas of P_i and H_i as $a_i = area(P_i)/area(H_i)$, and H_i is the number of Harris corners of P_i. We only keep the connected components with $a_i >= 0.97$ and $H_i = 4$ as patch masks.

Contour Fitting. After the patch selection stage, the selected patches are already aligned with our assumptions about the layout. However, the resolution of each patch is much smaller than that of the entire image (*i.e.*, 384×384). If the segmented patch is directly resized with the same size as the image fed into the UV network, then the contour of the patch will be rough. Therefore, we propose a method to fit the four edges of the patch region with four cubic curves, obtaining a patch mask with smooth edges. Specifically, for the segmented area of each patch, we first scale it to the original image resolution, use the Sobel operator to extract two horizontal and vertical edges, and fit each edge with a cubic curve. We take the fitted curve as the boundary to form four binary partition graphs, $\{B_{upper}, B_{lower}, B_{left}, B_{right}\}$, where the value of the patch region is 1, and the value of the other region is 0. The final patch image can be obtained by $\{I \bigcap B_{upper} \bigcap B_{lower} \bigcap B_{left} \bigcap B_{right}\}$ and will be fed back into the UV network for local prediction.

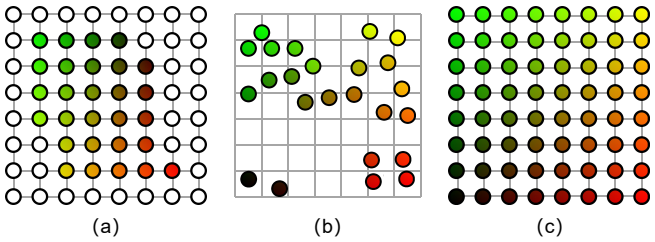

Fig. 9. Illustration of converting UV (a) to backward mapping (c). We first swap the UV and space coordinates to produce the intermediate result (b), where the points do not necessarily fall on the grid intersections. Then, we interpolate (b) to generate the full inverse map (c) and use it to sample the final rectified image.

Merging Strategy. Merging the UV maps of the local layout with the global one can be done on different targets, including forward UV maps, reverse UV maps, and flattened images. In this work, we choose to fuse the forward UV map (Fig. 8) because it is directly generated by the network and does not require post-processing operations that may introduce accumulated errors. Let $\mathbf{G}$ be the global UV area and $\mathbf{R}$ be the local UV areas, the simplest fusion method is the linear fusion, which is expressed as:

$$f_a = \begin{cases} g_a, & a \in (\mathbf{G} \setminus \bigcup_i \mathbf{R}_i) \\ l_a, & a \in \bigcup_i \mathbf{R}_i \end{cases} \tag{16}$$

where $\mathbf{R}_i$ is the UV map of i-th local patch. However, in our experiments, we found that linear fusion sometimes leads to misalignment of the edges of the fusion region. Therefore, we adopt an image blending method, namely, Poisson blending, to achieve seamless fusion. In general, Poisson blending uses the gradient domain of the image as guidance to achieve seamless merging between two images. Let N be the four-neighborhood of the pixel a, the boundary of $\mathbf{R}_i$ is $\partial\mathbf{R}_i = \{a \mid N_a \cap \mathbf{R}_i \neq \emptyset, a \in (\mathbf{G}_i \setminus \mathbf{R}_i)\}$. Our goal is to use the vector field of the local prediction results as guidance to solve $f|_{\mathbf{R}_i} = \{f_a, a \in \mathbf{R}_i\}$. The solution yields a discrete, quadratic optimization problem:

$$\min \sum_{<a,b> \cap \mathbf{R}_i \neq \emptyset} (f_a - f_b - v_{ab})^2 \quad s.t. \forall a \in \partial\mathbf{R}_i\ ,\ f_a = g_a. \tag{17}$$

Then, we use the Gauss-Seidel iteration method to drive an approximate solution efficiently. According to our experiments, one Poisson merging takes 13×10^{-4} seconds, and one linear merging takes 4×10^{-4} seconds. As the local UV prediction is a relative result, we need to transform it to the global position and linearly scale the UV value according to the maximum and minimum values of the edge positions of the global UV before merging.

5.3 Converting UV to Backward Mapping

As we stated in Sec. 3.1, we need to convert the forward UV map to the reverse UV map, obtaining the mapping from the deformed paper to the flat paper. The conversion process is shown in Fig. 9.

Specifically, we first take the values of all pixels in the predicted UV as coordinates, and use their coordinates as values to achieve part of the backward mapping. Then, we apply the nearest point interpolation method to complete the structured mapping, and use median filtering to remove outliers. Finally, we scale the inverse map to the exact resolution as the original image using bilateral interpolation upsampling, and use it to resample the original image to obtain the final flattened document image. Note that although the spatial size of the UV directly output by the network is 384×384, the resulting image has the exact resolution as the original image because we rectify the original image using the up-sampled inverse map. In other words, the input image to our framework can be of arbitrary resolution, and the final rectified image has the same resolution as the input image.

5.4 Implementation Details

Our model is implemented with PyTorch and is trained on a TITAN RTX GPU from NVIDIA®. We train the UV regression and layout segmentation networks separately with the same hyper-parameters. We employ 6,900 distinct document textures (such as papers, books, posters, and magazines) to train the UV branch network. In parallel, we train the layout branch network using 20,000 unique academic paper textures. Besides, we train the UV branch network with global images only because the shapes of a local patch and the entire document are both developable surfaces. After the neural network learns the UV of the severely deformed entire document, it can also correctly predict the forward flow of a single local content and the mildly deformed patch. Specifically, we train our networks with an Adam optimizer [Kingma and Ba 2015] and a batch size of 12. Each epoch consists of 16,500 iterations for the UV regression network and 8,000 iterations for the layout segmentation network. The initial learning rate is set as 6×10^{-5} and reduced by a factor of 0.25 if the loss does not improve in the last two epochs. For hyper-parameters in Eq. (13) and Eq. (14), we set $\eta = 250$ and $\lambda = 0.05$ by default, which generally works well in our experiments. The UV branch and layout branch networks contain 1.2×10^8 parameters each. We adopt the interpolation function in SciPy 1.7.3 to perform the nearest point interpolation in Sec. 5.3 and use the filtering algorithm from OpenCV 4.5.2.

6 EXPERIMENTAL RESULTS

We thoroughly compare our approach with recent state-of-the-art methods, including DocProj [Li et al. 2019], DewarpNet [Das et al. 2019], FCN-based [Xie et al. 2021a], DocTr [Feng et al. 2021a], Piece-Wise [Das et al. 2021], DDCP [Xie et al. 2021b], DocScanner [Feng et al. 2021b], Marior [Zhang et al. 2022], PaperEdge [Ma et al. 2022], RDGR [Jiang et al. 2022], and DocGeoNet [Feng et al. 2022] on two public benchmarks and our synthetic dataset. Then, we conduct ablation studies to verify the effectiveness of the components of our method.

6.1 Evaluation on Real Image Benchmark

DocUNet benchmark. [Ma et al. 2018] collected an evaluation dataset for document unfolding, named as DocUNet Benchmark, which is widely used in existing works. They scanned 65 unique documents, manually deformed each sheet into two different shapes, and then photographed them with a camera, creating 130 images of the deformed documents. The origins of document paper include magazines, receipts, posters, and academic articles, and the shape of the paper contains different degrees of deformation. Note that the 64-th pair of distorted images in the original DocUNet benchmark rotates by 180 degrees, which does not match the ground truth image. We have fixed this issue before evaluation.

Table 1. Quantitative comparison with previous geometric rectification methods on the DocUNet benchmark. Note that we follow DewarpNet [Das et al. 2019] and DocTr [Feng et al. 2021a] to report ED and CER on their selected 50 and 60 pairs of images, respectively. Bold indicates the **best** result, and underline indicates the second-best result. MS is short for MS-SSIM. *Ours w/o SP* is the result of our model trained on the dataset rendered using Doc3D meshes only, without simulated paper.

Methods	MS↑	LD↓	AD↓	ED↓	CER↓
Distorted	0.246	20.51	1.012	2111.6/1552.2	0.535/0.509
DocProj	0.293	18.51	0.994	1812.1/1293.4	0.445/0.430
DewarpNet	0.474	8.39	0.426	885.9/525.5	0.237/0.210
FCN-based	0.448	7.84	0.434	1792.6/1031.4	0.421/0.316
DocTr	0.511	7.76	0.396	724.8/464.8	0.183/0.175
Piece-Wise	0.492	8.64	0.468	1069.3/743.3	0.268/0.262
DDCP	0.473	8.99	0.453	1442.8/745.4	0.363/0.263
Marior	0.478	7.27	0.403	776.2/593.8	0.193/0.214
PaperEdge	0.473	7.81	0.392	777.8/**375.6**	0.201/0.154
RDGR	0.497	8.51	0.461	729.5/420.3	0.172/0.156
DocGeoNet	0.504	7.71	0.380	713.9/379.0	0.182/0.151
DocScanner	0.518	7.45	0.334	**632.3**/390.4	**0.165**/**0.149**
Ours w/o SP	0.511	7.24	0.307	644.7/482.5	0.178/0.182
Ours	**0.526**	**6.72**	**0.300**	695.0/391.9	0.175/0.153

Table 2. Quantitative comparison with previous geometric rectification methods on the DIR300 benchmark.

Methods	MS↑	LD↓	AD↓	ED↓	CER↓
Distorted	0.3169	39.58	0.771	1500.56	0.5234
DewarpNet	0.4921	13.94	0.331	1059.57	0.3557
FCN-based	0.5035	9.75	0.331	1939.48	0.5099
DocTr	0.6160	7.21	0.254	699.63	0.2237
DDCP	0.5524	10.95	0.357	2084.97	0.5410
PaperEdge	0.5836	8.00	0.255	**508.73**	0.2069
DocGeoNet	0.6380	6.40	0.242	664.96	0.2189
Ours w/o SP	0.6410	6.12	0.204	548.17	0.1967
Ours	**0.6518**	**5.70**	**0.195**	511.13	**0.1891**

DIR300 benchmark. [Feng et al. 2022] constructed the DIR300 dataset, which contains 300 pairs of images from 300 different documents. Compared to the DocUNet benchmark, the photos in DIR300 have more diverse backgrounds and illuminations. Besides, DIR300 contains more severely deformed documents, where the authors deliberately increased the degree of deformation of the document when constructing the dataset.

Image similarity metrics. Multi-Scale Structural Similarity (MS-SSIM) [Wang et al. 2003] and Local Distortion (LD) [You et al. 2018]

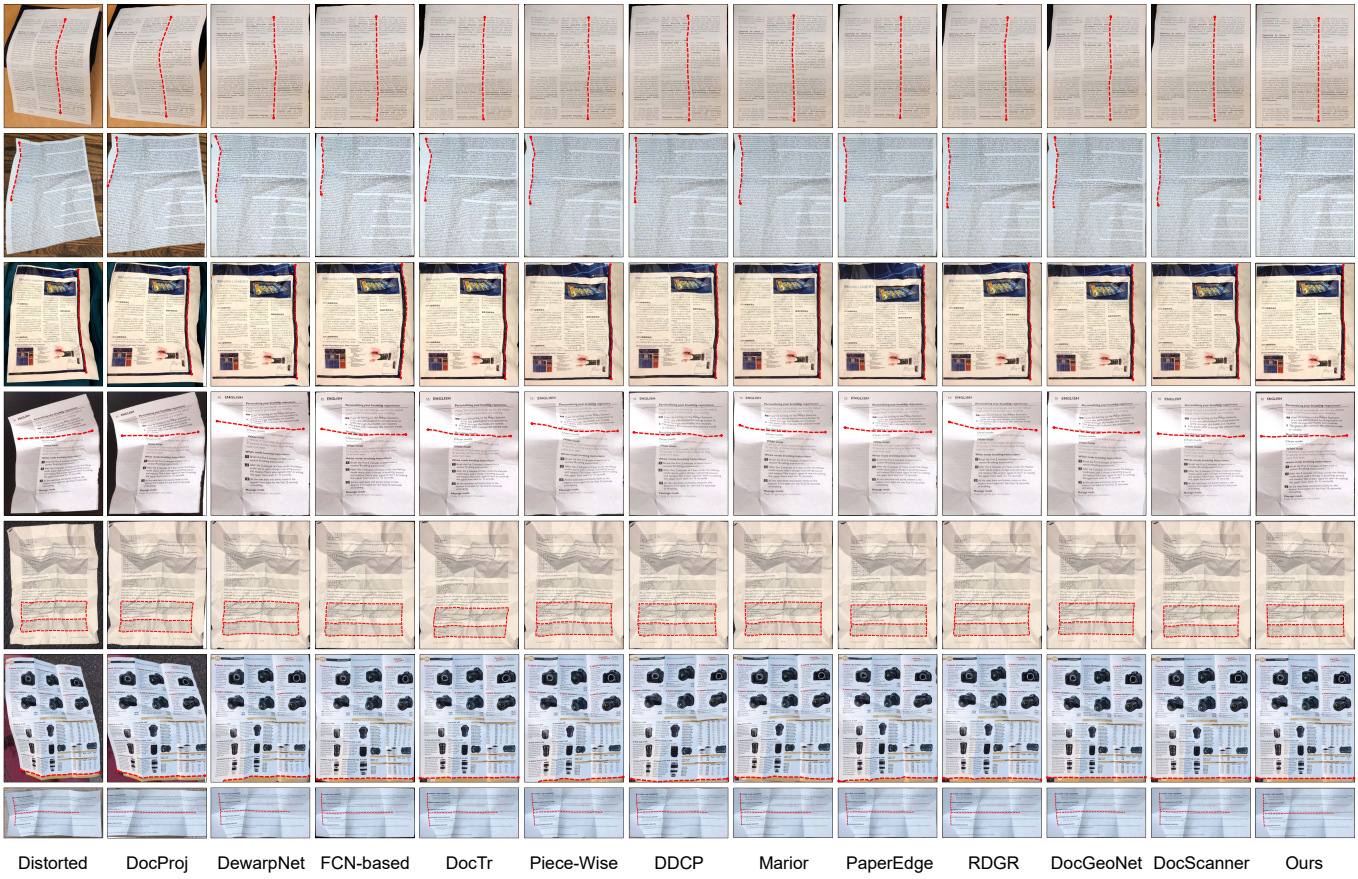

Fig. 10. Qualitative comparisons on DocUNet benchmark. The dotted lines in the rectified images should be horizontal or vertical if the corrections are perfect.

are two quantitative metrics that have been widely used in previous methods. SSIM uses a sliding window method to calculate luminance, contrast, and structure comparison of the two images. MS-SSIM builds a Gaussian pyramid of two images to calculate the weighted summation of SSIM over multiple resolutions. In keeping with previous work, we set the weights for each level as: 0.0448, 0.2856, 0.3001, 0.2363, 0.1333. LD measures the average deformation of each pixel by performing dense image registration using SIFT-flow [Liu et al. 2010] between two images. AD effectively enhances the MS-SSIM by employing translation and scaling techniques to reduce the global offset present within the flattened image. Additionally, it incorporates image gradients as weight factors, thereby diminishing the error of LD within the textureless regions. We calculated the three above metrics using Matlab R2019a in our experiments.

OCR metrics. Given the reference and recognized text, Edit Distance (ED) is the minimal number of substitutions (s), insertions (i), and deletions (d) to convert the recognized text to the reference text. Based on ED, Character Error Rate (CER) is used to calculate the ratio instead of quantity. It is defined as $CER = (s + i + d)/N$, where N is the total number of characters in the reference text. To perform OCR evaluation on the DocUNet benchmark, [Das et al. 2019] and [Feng et al. 2021a] selected 50 and 60 pairs of images, respectively. We follow these settings and calculate ED and CER on the same two subsets for a fair comparison. Furthermore, we select 150 images in our dataset and the same 90 images as [Feng et al. 2022] in DIR300 for OCR evaluation. The OCR recognition in our experiments is based on Tesseract v5.0.1.2022011 and PyTesseract v0.3.9 *. All ground truth strings are recognized using PyTesseract from the flat document images.

Comparison results. In Tab. 1, we first compare our approach against the previous methods on the DocUNet benchmark. The results show that our method produces significantly better results in terms of image similarity metrics. In addition, we conduct qualitative comparison in terms of global image (Fig. 10) and local regions (Fig. 11). We select some challenging test cases with complex deformations or large perspective angles for evaluation. It can be seen that our method corrects the distorted image to be smoother (take the dotted lines as the reference). Furthermore, our method presents more complete and straight contours compared with other methods.

We further report the quantitative and qualitative comparison results on the DIR300 dataset in Tab. 2 and Fig. 12. Note that Piece-Wise [2021], RDGR [2022], Marior [2022], and DocScanner [2021b] did not release their code or test results on DIR300, so they were

*https://pypi.org/project/pytesseract/

Fig. 11. Qualitative comparisons of local rectifications on DocUNet benchmark. The dotted lines are added for better visibility.

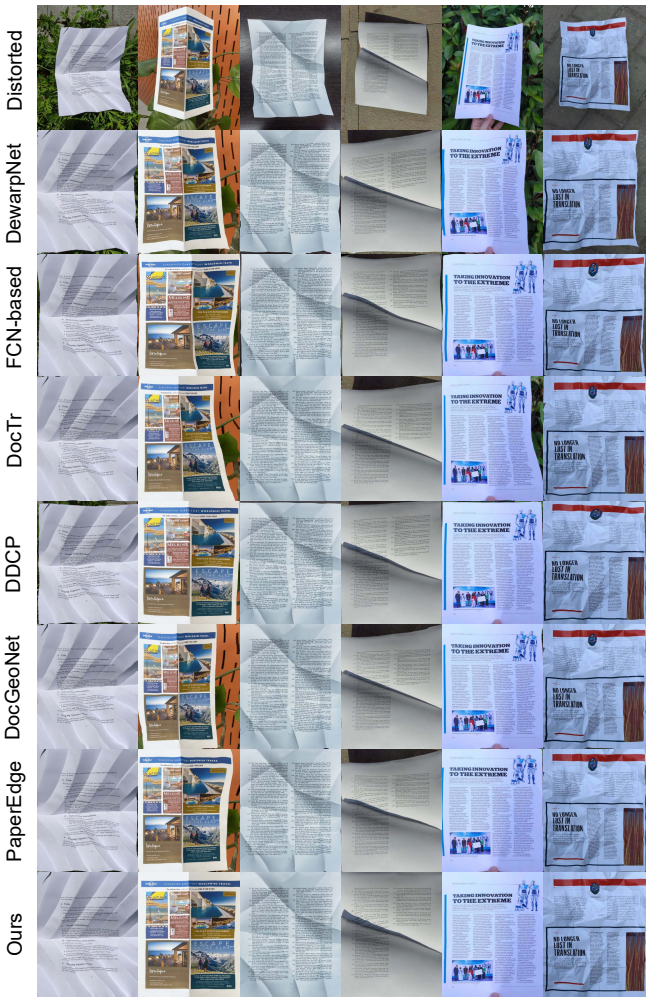

Fig. 12. Qualitative comparisons on DIR300 dataset.

not included in the comparison. The results in Tab. 2 show that our method outperforms previous methods on MS-SSIM, LD, and CER, and ranks second best on ED. It can be seen from Fig. 12 that our method is able to pull the outline of the paper to the edge of the image, making the corrected document texture fill the entire image. In contrast, other methods fall short of completely eliminating out-of-document regions.

6.2 Evaluation on Our Synthetic Dataset

Building synthetic evaluation dataset. Although existing benchmarks and evaluation metrics can intuitively measure the correction quality of flattened images, they have some drawbacks. We summarize them into the following three aspects: (1) The unevenness of illumination in the distorted documents is often preserved in the results of geometric rectification. This situation can lead to poor evaluation results even for perfectly geometrically corrected images. (2) Existing evaluation metrics are only suitable for certain textured documents. For example, OCR metrics can only evaluate text-rich

Table 3. Paper shape deformation classification for our synthetic dataset. We used this taxonomy in Fig. 15, Tab. 6, and Tab. 8.

ID	Shape characteristics of the distorted paper
1	Flat, without crease nor curve.
2	One noticeable crease with no curved surface.
3	Two noticeable creases with no curved surface.
4	Convex curved surface with no crease.
5	Complex curved surface with no crease.
6	At least three creases mixed with curved surface.

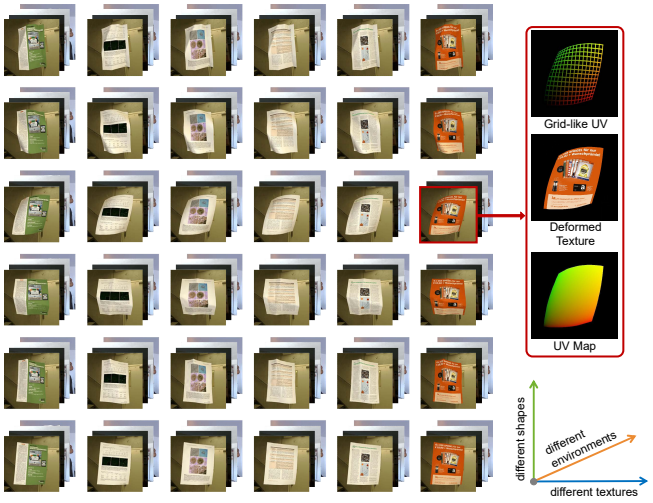

Fig. 13. Some examples of our synthetic evaluation dataset.

document images, while image similarity metrics only work on regions with varying textures. (3) Although the DocUNet benchmark contains papers with varying degrees of deformation, the textures and backgrounds of these images are also different, making it impossible to exclude the influence of other factors when measuring the robustness to purely varying degrees of deformation.

To overcome the above shortcomings, we construct a synthetic image dataset containing 1,500 deformed document images and propose new evaluation metrics based on UV flow. Compared to real image datasets and traditional evaluation methods, our evaluation method is independent of light, shadow, and document texture. Furthermore, by simply changing the shape of the document sheet while keeping the background image, global illumination, and document texture consistent, we can fairly compare the performance of various methods on sheets with varying degrees of deformation.

Fig. 13 shows some examples of our synthetic evaluation dataset. For a flat document texture $\mathbf{T}$, we first render a deformed image $\mathbf{P}_{warp}$ according to the method mentioned above, and the deformed texture image $\mathbf{T}_{warp}$ corresponding to the deformed image, the deformed UV flow $\mathbf{UV}_{warp}$ and deformed grid-looking UV $\mathbf{G}_{warp}$. We denote the mapping function from the flattened expression $\mathbf{P}_{warp}$ to the flattened image $\mathbf{P}_{flat}$ as $\mathcal{F}$, and we apply the function $\mathcal{F}$ to $\mathbf{T}_{warp}$, $\mathbf{UV}_{warp}$, and $\mathbf{G}_{warp}$ to obtain

$$\begin{aligned} \mathbf{P}_{flat} &= \mathcal{F}(\mathbf{P}_{warp}), \mathbf{T}_{flat} = \mathcal{F}(\mathbf{T}_{warp}), \\ \mathbf{UV}_{flat} &= \mathcal{F}(\mathbf{UV}_{warp}), \mathbf{G}_{flat} = \mathcal{F}(\mathbf{G}_{warp}), \end{aligned} \tag{18}$$

Table 4. The Coefficient of Variation (CV) and the running time of each distortion metric when processing images with the same or the symmetrical geometric distortion. For the original image (a), we obtain the image with shadow (b), the lower resolution image (c), the image with different document content (d), and the image with symmetrical deformation (e). Since their perspective angles are the same, an ideal evaluation metric should produce consistent results.

	Groundtruth	(a)	(b)	(c)	(d)	(e)	CV (%)	Time (s)
Figure							-	-
MS-SSIM	1.000	0.393	0.248	0.429	0.260	0.409	22.305	5.500E-01
LD	0.000	14.002	13.712	5.554	18.021	18.745	33.503	4.340E+00
AD	0.000	0.863	0.529	0.395	0.727	0.647	25.445	4.825E+00
DEPE	0.000	16.706	16.706	16.723	16.706	36.702	38.614	3.630E-04
MPD	0.000	18.608	18.608	18.627	18.608	18.627	**0.050**	**3.539E-04**

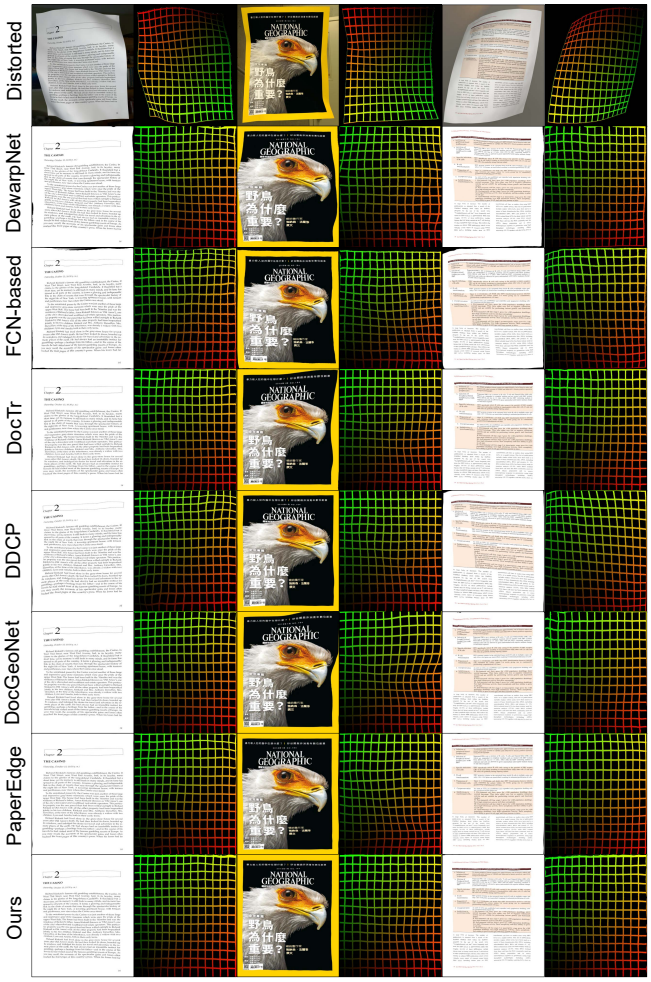

Fig. 14. Visual comparisons on our synthetic dataset. To eliminate the interference of light and shadow, and show the effect of geometric correction more clearly, we provide the $\mathbf{T}_{flat}$ (left) and $\mathbf{G}_{flat}$ (right) of each method.

where $\mathbf{T}_{flat}$ is the texture correction result without light and shadow. $\mathbf{G}_{flat}$ should have a regular grid appearance under perfect geometric correction, so we use it as a reference for visual evaluation.

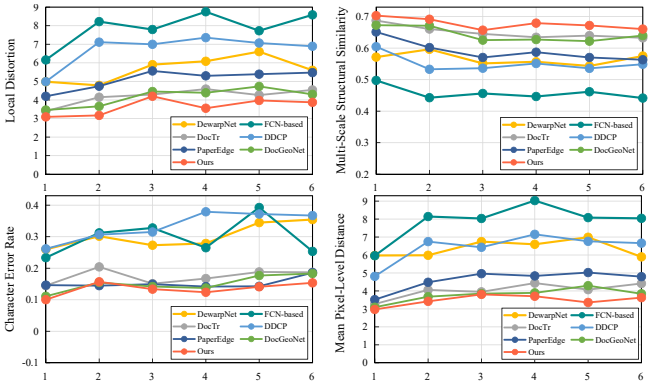

Fig. 15. Quantitative comparison of geometric correction methods on our synthetic dataset. The horizontal axis coordinates correspond to different paper deformation types, which are listed in Tab. 3. For Multi-Scale Structural Similarity, the higher the better, while the others are the opposite.

Table 5. Quantitative comparison with previous geometric rectification methods on our synthetic dataset.

Methods	MPD↓	MS↑	LD↓	AD↓	ED↓	CER↓
Distorted	18.869	0.205	20.50	0.728	1773.6	0.829
DewarpNet	6.390	0.566	5.66	0.188	772.0	0.302
FCN-based	7.912	0.458	7.87	0.300	838.8	0.297
DocTr	4.050	0.650	4.20	0.170	470.6	0.174
DDCP	6.427	0.552	6.74	0.300	948.0	0.334
PaperEdge	4.602	0.591	5.11	0.207	403.7	0.152
DocGeoNet	3.775	0.643	4.16	0.182	386.1	0.151
Ours w/o SP	3.981	0.664	4.10	**0.136**	449.0	0.171
Ours	**3.496**	**0.677**	**3.64**	0.138	**365.4**	**0.135**

$\mathbf{UV}_{flat}$ is the corrected UV flow, its ground truth value will be linearly distributed from 0 to 1 in the horizontal direction on the U channel and the vertical direction on the V channel.

MPD metric. To evaluate the document correction effect, [Li et al. 2019] first proposed to use End Point Error (EPE), which measures the pixel-wise Euclidean distance between the predicted flow and

the ground truth:

$$EPE = \frac{1}{HW}\sum_{p\in\Psi}(\hat{\mathbf{UV}}_{warp}(p) - \mathbf{UV}_{warp}(p)), \tag{19}$$

where $H\times W$ is the image resolution, p is the coordinate of a pixel, Ψ is the inner area of the image, $\hat{\mathbf{UV}}_{warp}$ and $\mathbf{UV}_{warp}$ are the predicted and ground truth for forward flow of the warped document. The main drawback of EPE is that it cannot evaluate methods that do not predict the forward flow of distorted documents, such as methods that directly generate backward flow [Feng et al. 2021a, 2022] or use 3D coordinates as intermediate results [Das et al. 2019]. To overcome this shortcoming, a simple approach is to change the calculation target of EPE from the forward flow of the warped document to the forward flow of the flattened document. We denote this metric as Dewarped End Point Error (DEPE):

$$DEPE = \frac{1}{HW}\sum_{p}(\hat{\mathbf{UV}}_{flat}(p) - \mathbf{UV}_{flat}(p)), \tag{20}$$

where $\hat{\mathbf{UV}}_{flat}$ and $\mathbf{UV}_{flat}$ are the predicted and ground truth for the forward flow of the dewarped document. As all corrective methods must generate flattening function $\mathcal{F}$, making $\hat{\mathbf{UV}}_{warp}$ available, DEPE is more general than EPE without the applicable restrictions. However, the valid area of $\hat{\mathbf{UV}}_{warp}$ may not completely overlap with the ground truth. In non-coincident areas, DEPE calculates the distance of the point to the origin instead of to the correct position. To solve this problem, we further propose the Mean Pixel-level Distance (MPD):

$$MPD = \frac{1}{HW}\sum_{p\in\Omega}(\hat{\mathbf{UV}}_{flat}(p) - p), \tag{21}$$

where Ω is the area covered by $\hat{\mathbf{UV}}_{warp}$. Note that there is no necessary belonging relationship between Ω and Ψ. MPD inherits the advantages of DEPE, *i.e.,* it can sensitively reflect the geometric error of details, and is not disturbed by texture, shadow, and background. Fig. 16 illustrates the difference between EPE, DEPE, and MPD more intuitively, indicating that MPD is able to circumvent calculations within invalid regions (represented by gray color).

We conduct an experiment to demonstrate the robustness of MPD, where the images of settings (a, b, c, d) in Tab. 4 have the exact same perspective distortion, and the distortion of setting (e) is vertically symmetric to the former. On the basis of (a), we add shadows (b), reduce the resolution (c), change the content of the document (d), and keep the texture consistency (e). We calculate the coefficient of variation and running time of the five compared metrics. The ideal evaluation metric should output consistent results when evaluating these five images. The result shows that MPD has the strongest robustness to shadow, resolution, and different content, making MPD a more equitable metric for evaluating geometric deformations than others. Besides, MPD also has the highest computational efficiency.

Comparison results. Applying MPD along with previous traditional metrics, we compare the performance of our method and several competitors on synthetic datasets, and the results are reported in Tab. 5. The results show that our method can bring a better geometric correction effect. We qualitatively compare the results of all methods in terms of flattened images and grid-looking UVs,

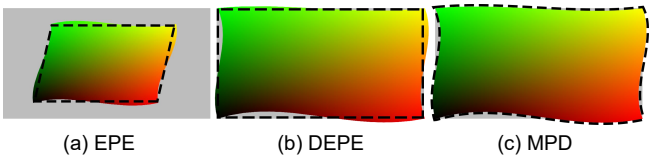

Fig. 16. The difference between EPE, DEPE, and MPD. The predicted flow is represented by the colored area, while the ground truth is encompassed by the black dashed line.

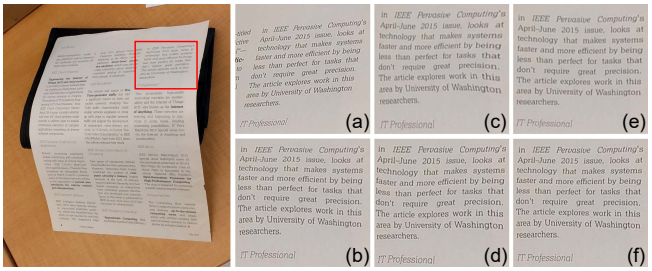

Fig. 17. Visual comparison of different merging strategies: the uncorrected local image (a), the rectification result using only the global rectification module (b), the results of merging on the backward map with linear strategy (c) and Poisson strategy (e), and the results of merging on UV with linear strategy (d) and Poisson strategy (f).

as shown in Fig. 14. We can see that our method recovers the flat document images with clear texture and reasonable shape (yellow boundary in the second row). On the right side of Fig. 14, the restored grid-looking UVs of our method are more regular. In addition, Fig. 15 reports the numerical comparisons of our method with six competitors on our synthetic dataset with six different deformation types. For image similarity evaluation, especially LD and MPD, our method shows clear advantages on all paper deformation categories. For OCR evaluation, our CER result curve is flatter, which indicates that our method is more robust to different deformations. These results consistently indicate the superiority of our method.

6.3 Ablation Studies

In the following, we conduct ablation studies to analyze the effectiveness of our constructed dataset, layout-aware assists, and different merging strategies. We also perform an ablation experiment on the backbones of the network, and the results are available in supplementary materials. For the DocUNet benchmark, all OCR results are obtained on the 60 pairs of images selected by [Feng et al. 2021a].

Simulated paper. To demonstrate the contribution of more diverse 3D paper shapes produced by our simulation method, we perform an ablation study on our synthetic dataset. Specifically, we train the UV network separately on the Doc3D dataset and our synthesized dataset. Tab. 6 shows that training with our synthetic paper dataset leads to better corrections than training with the Doc 3D dataset, which we attribute to the more diverse paper shapes in the synthetic dataset. In addition, the correction effect is further improved when training our network with both datasets.

Table 6. Ablation study on the extended training dataset with simulated paper shapes. Bold numbers represent the best, and the underlined setting means our final adoption. We use the same notations in Tab. 7 and Tab. 8

	Training Dataset		DocUNet Benchmark				Our Evaluation Dataset				
Setting	Doc3D	Simulated Paper	**MS-SSIM↑**	**LD↓**	**ED↓**	**CER↓**	**MPD↓**	**MS-SSIM↑**	**LD↓**	**ED↓**	**CER↓**
(a)	✓	✗	0.5013	7.238	482.5	0.182	3.512	0.6640	4.101	449.0	0.171
(b)	✗	✓	0.5082	6.980	415.0	0.168	3.646	0.6693	3.738	392.5	0.146
(c)	✓	✓	**0.5256**	**6.719**	**391.9**	**0.153**	**3.496**	**0.6772**	**3.643**	**365.4**	**0.135**

Table 7. Ablation study on the layout network branch and the layout-ware loss term. In the layout column, the numbers in place of the check marks represent the prediction accuracies of the enabled layout networks.

	Network Branch		Loss Term	DocUNet Benchmark				Our Evaluation Dataset				
Setting	UV	Layout	$+\mathcal{L}_{layout}$	**MS-SSIM↑**	**LD↓**	**ED↓**	**CER↓**	**MPD↓**	**MS-SSIM↑**	**LD↓**	**ED↓**	**CER↓**
(a)	✓	✗	✗	0.4886	7.898	466.1	0.173	4.581	0.6223	4.054	437.5	0.165
(b)	✓	✗	✓	0.4913	7.532	452.6	0.167	3.842	0.6397	3.748	399.6	0.150
(c)	✓	0.879	✗	0.4968	7.367	448.7	0.159	3.783	0.6360	3.881	374.9	0.140
(d)	✓	0.924	✗	0.5080	7.272	436.8	0.158	3.663	0.6442	3.765	373.3	0.140
(e)	✓	0.963	✗	0.5099	7.185	414.2	0.155	3.502	0.6559	3.713	366.6	0.137
(f)	✓	0.963	✓	**0.5256**	**6.719**	**391.9**	**0.153**	**3.496**	**0.6772**	**3.643**	**365.4**	**0.135**

Table 8. Ablation study on different merging strategies.

	Merging Target		Merging Method		DocUNet Benchmark				Our Evaluation Dataset				
Setting	UV	BM	Linear	Poisson	**MS-SSIM↑**	**LD↓**	**ED↓**	**CER↓**	**MPD↓**	**MS-SSIM↑**	**LD↓**	**ED↓**	**CER↓**
(a)	✓	✗	✓	✗	0.5169	7.3934	408.6	0.156	3.498	0.6654	3.752	390.1	0.141
(b)	✓	✗	✗	✓	**0.5256**	**6.7193**	**391.9**	**0.153**	**3.496**	**0.6772**	**3.643**	**365.4**	**0.135**
(c)	✗	✓	✓	✗	0.4811	7.3701	465.6	0.157	3.575	0.5379	4.064	461.4	0.172
(d)	✗	✓	✗	✓	0.5193	7.1169	466.4	0.165	4.220	0.5786	4.091	413.3	0.150

Layout-Awareness. We utilize the document layout in two aspects, namely, the $\mathcal{L}_{layout}$ loss term in the UV network and the layout branch network. We remove these two layout-aware components separately to verify their contribution to the overall correction effect. We also use three layout branch networks with different prediction accuracies to verify the influence of layout prediction accuracy on the final correction. Note that the layout loss only works on the UV branch, aiming to make the UV branch predict a more accurate forward flow. The layout branch further refines the prediction of the UV branch based on the predicted forward flow. Therefore, the layout loss and layout branch are independent of each other, and they sequentially improve the correction results.

As shown by the results in Tab. 7, both components can bring an improvement in document correction accuracy. Settings (c), (d), and (e) show that the more accurate the layout prediction, the more document correction may benefit from the assistance of the layout branch network. When both loss term and the layout network are enabled (see setting (f)), LD is reduced by 17% on the DocUNet benchmark, and MPD is reduced by 24% on our evaluation dataset.

Merging Strategies. Finally, we evaluate the different merging strategies in layout correction, and report the corresponding numerical results in Tab. 8. We use the linear and Poisson merging strategies on the forward UV ("UV" in Tab. 8) and reverse UV maps ("BM" in Tab. 8), respectively. Two conclusions can be drawn from the results. First, the performance of merging on UV is better compared with merging on backward map. Second, using Poisson blending on the UV map is the best merging strategy. In addition, we show a visual comparison of different merging strategies in Fig. 17. The visual results are also consistent with numerical results in Tab. 8.

6.4 Rectification Efficiency

In order to report the time spent by each module during inference, we run our method on a document image with a resolution of 2000×2000, containing five valid layout patches. Tab. 9 shows the running time of each module and the corresponding proportion in the total time. It can be seen that the non-neural *Patch Selection* and *Dewarping* modules were the most time-consuming, accounting for 41.32% and 35.81% of the total time, respectively. In addition, the local UV prediction also takes up 14.00% because it needs to be run multiple times.

6.5 Limitations

Our method still has several limitations. Firstly, when the content of the document does not have clear and distinct layout information, our layout correction module will fail to bring improvement, and the entire model will degenerate into global flattening. Secondly, since we use document edges as important visual cues, our method may fail when the outline of the document is incomplete or obscure (see

Table 9. The time consumption of each module in the inference phase.

Module	Time (s)	Proportion (%)
Global UV	0.0598	2.88
Layout Segmentation	0.0588	2.83
Patch Selection	0.8597	41.32
Local UV	0.2912	14.00
Merging	0.0659	3.17
Dewarping	0.7450	35.81
Total	2.0805	100

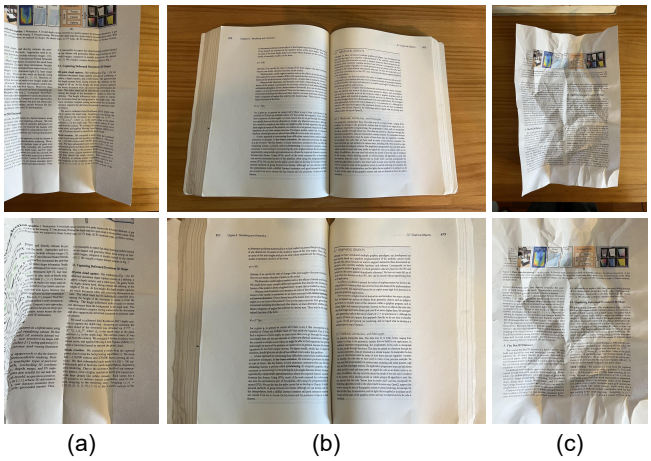

Fig. 18. Several failure cases of our approach. Each column represents a case, the top is the input, and the bottom is the output. Our method may produce inferior results when the input documents are incomplete (a), possess ambiguous boundaries (b), or are severely wrinkled (c).

Fig. 19. Our method only segments the largest layout patches when encountering documents with hierarchical layouts. However, the final rectification result remains satisfactory.

the left and middle columns of Fig. 18). Thirdly, our method only performs geometric correction, so for documents that are severely wrinkled, our rectified results still look bumpy due to uneven illumination (see the right column of Fig. 18). Finally, the proposed MPD can only be applied to synthetic datasets that have forward-flow ground truth.

7 CONCLUSION AND FUTURE WORK

We have proposed a layout-aware framework for rectifying a single distorted document image. After applying the layout segmentation network to predict the layout of the document, we employ the UV regression network to estimate the UV maps of the entire image and the layout patches. We then propose an effective fusion module to merge the local and global predictions and obtain a robust and complete UV map, which is leveraged for recovering the flat document. Extensive results illustrate the superiority of our proposed network. In addition, we adopt a fully automatic rendering method for generating warped document images with diverse 3D shapes, exact layout segmentation annotations, and paired deformed/flat document images. Correspondingly, we introduce a novel metric for measuring geometric correction performance. We will release the code and dataset upon acceptance to facilitate future research.

In the future, we plan to explore how to utilize more low-level features (*e.g.*, text lines and straight lines) to assist deep learning-based document rectification. Considering that all current neural-based document correction methods rely on strong supervision, rectification based on unsupervised learning or self-supervised learning is a challenging but interesting research direction. In addition, for documents containing hierarchical layouts (see Fig. 19), our approach only exploits the top-level layout patch and does not further segment the lower-level sub-layouts. This is because the segment labels in our current training set only contain the largest ones. We can see that our approach still reasonably rectifies such documents even without the most fine-grained segmentation. In the future, we would like to extend our dataset and explore the finer-grained utilization of the hierarchical layout to handle more complex documents.

ACKNOWLEDGMENTS

We thank the reviewers for their constructive comments. This work was supported in parts by NSFC (U22B2034, U21A20515, 62172416, 62172415, 62102418), and Youth Innovation Promotion Association of the Chinese Academy of Sciences (2022131).

REFERENCES

Md Amirul Islam, Mrigank Rochan, Neil DB Bruce, and Yang Wang. 2017. Gated feedback refinement network for dense image labeling. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. 3751–3759.

Galal M. Binmakhashen and Sabri A. Mahmoud. 2019. Document Layout Analysis: A Comprehensive Survey. *ACM Comput. Surv.* 52, 6 (2019).

Dário Augusto Borges Oliveira and Matheus Palhares Viana. 2017. Fast CNN-Based Document Layout Analysis. In *Int. Conf. Comput. Vis. Worksh.* 1173–1180.

Michael S. Brown and W. Brent Seales. 2001. Document restoration using 3D shape: a general deskewing algorithm for arbitrarily warped documents. In *IEEE International Conference on Computer Vision (ICCV)*, Vol. 2. 367–374.

Michael S. Brown, Mingxuan Sun, Ruigang Yang, Lin Yun, and W. Brent Seales. 2007. Restoring 2D Content from Distorted Documents. *IEEE Trans. Pattern Anal. Mach. Intell.* 29, 11 (2007), 1904–1916.

Michael S. Brown and Yau-Chat Tsoi. 2006. Geometric and shading correction for images of printed materials using boundary. *IEEE Trans. Image Process.* 15, 6 (2006), 1544–1554.

Alexander Burden, Melissa Cote, and Alexandra Branzan Albu. 2019. Rectification of Camera-Captured Document Images with Mixed Contents and Varied Layouts. In *IEEE Conf. Comput. Robot Vis.* 33–40.

Huaigu Cao, Xiaoqing Ding, and Changsong Liu. 2003. A cylindrical surface model to rectify the bound document image. In *IEEE International Conference on Computer Vision (ICCV)*, Vol. 1. 228–233.

Lei Chen, Rui Liu, Dongsheng Zhou, Xin Yang, and Qiang Zhang. 2020. Fused behavior recognition model based on attention mechanism. *Visual Computing for Industry, Biomedicine, and Art* 3, 1 (2020), 1–10.

Frédéric Courteille, Alain Crouzil, Jean-Denis Durou, and Pierre Gurdjos. 2007. Shape from shading for the digitization of curved documents. *Pattern Recog.* 18 (2007), 301–316.

Sagnik Das, Ke Ma, Zhixin Shu, Dimitris Samaras, and Roy Shilkrot. 2019. DewarpNet: Single-Image Document Unwarping With Stacked 3D and 2D Regression Networks. In *IEEE International Conference on Computer Vision (ICCV)*. 131–140.

Sagnik Das, Gaurav Mishra, Akshay Sudharshana, and Roy Shilkrot. 2017. The Common Fold: Utilizing the Four-Fold to Dewarp Printed Documents from a Single Image. In *ACM Symp. Document Engin.* 125–128.

Sagnik Das, Kunwar Yashraj Singh, Jon Wu, Erhan Bas, Vijay Mahadevan, Rahul Bhotika, and Dimitris Samaras. 2021. End-to-end Piece-wise Unwarping of Document Images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 4268–4277.

Tanmoy Dasgupta, Nibaran Das, and Mita Nasipuri. 2020. Multistage Curvilinear Coordinate Transform Based Document Image Dewarping using a Novel Quality Estimator. *CoRR* abs/2003.06872 (2020).

Homa Davoudi, Marco Fiorucci, and Arianna Traviglia. 2021. Ancient Document Layout Analysis: Autoencoders meet Sparse Coding. In *Int. Conf. Pattern Recog.* 5936–5942.

Henghui Ding, Xudong Jiang, Bing Shuai, Ai Qun Liu, and Gang Wang. 2020. Semantic segmentation with context encoding and multi-path decoding. *IEEE Trans. Image Process.* 29 (2020), 3520–3533.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *Int. Conf. Learn. Represent.* (2021).

Mohamed Fawzi, Mohsen. A. Rashwan, Hany Ahmed, Shaimaa Samir, Sherif M. Abdou, Hassanin M. Al-Barhamtoshy, and Kamal M. Jambi. 2015. Rectification of camera captured document images for camera-based OCR technology. In *IAPR Int. Conf. Document Anal. Recog.* 1226–1230.

Hao Feng, Shaokai Liu, Jiajun Deng, Wengang Zhou, and Houqiang Li. 2023. Deep Unrestricted Document Image Rectification. *arXiv* (2023).

Hao Feng, Yuechen Wang, Wengang Zhou, Jiajun Deng, and Houqiang Li. 2021a. DocTr: Document Image Transformer for Geometric Unwarping and Illumination Correction. In *ACM Int. Conf. Multimedia*. 273–281.

Hao Feng, Wengang Zhou, Jiajun Deng, Qi Tian, and Houqiang Li. 2021b. DocScanner: Robust Document Image Rectification with Progressive Learning. *CoRR* abs/2110.14968 (2021).

Hao Feng, Wengang Zhou, Jiajun Deng, Yuechen Wang, and Houqiang Li. 2022. Geometric Representation Learning for Document Image Rectification. In *Proceedings of the European Conference on Computer Vision*.

Marc-André Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. 2017. Learning to Predict Indoor Illumination from a Single Image. *ACM Trans. Graph.* 36, 6 (2017).

Dafang He, Scott Cohen, Brian Price, Daniel Kifer, and C. Lee Giles. 2017. Multi-Scale Multi-Task FCN for Semantic Page Segmentation and Table Detection. In *IAPR Int. Conf. Document Anal. Recog.*, Vol. 01. 254–261.

Yuan He, Pan Pan, Shufu Xie, Jun Sun, and Satoshi Naoi. 2013. A Book Dewarping System by Boundary-Based 3D Surface Reconstruction. In *IAPR Int. Conf. Document Anal. Recog.* 403–407.

Xiangwei Jiang, Rujiao Long, Nan Xue, Zhibo Yang, Cong Yao, and Gui-Song Xia. 2022. Revisiting Document Image Dewarping by Grid Regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4543–4552.

Beom Su Kim, Hyung Il Koo, and Nam Ik Cho. 2015. Document dewarping via text-line based optimization. *Pattern Recog.* 48, 11 (2015), 3600–3614.

Theodore Kim, Nils Thürey, Doug James, and Markus Gross. 2008. Wavelet Turbulence for Fluid Simulation. *ACM Trans. Graph.* 27, 3 (2008), 1–6.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Int. Conf. Learn. Represent.*

Hyung Il Koo and Nam Ik Cho. 2010. State Estimation in a Document Image and Its Application in Text Block Identification and Text Line Extraction. In *European Conference on Computer Vision (ECCV)*. 421–434.

Olivier Lavialle, X. Molines, Franck Angella, and Pierre Baylou. 2001. Active contours network to straighten distorted text lines. In *IEEE Int. Conf. Image Process.*, Vol. 3. 748–751.

Xiaoyu Li, Bo Zhang, Jing Liao, and Pedro V. Sander. 2019. Document Rectification and Illumination Correction Using a Patch-Based CNN. *ACM Trans. Graph.* 38, 6 (2019).

Jian Liang, Daniel DeMenthon, and David Doermann. 2008. Geometric Rectification of Camera-Captured Document Images. *IEEE Trans. Pattern Anal. Mach. Intell.* 30, 4 (2008), 591–605.

Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. 2017b. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. 1925–1934.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017a. Feature pyramid networks for object detection. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2117–2125.

Ce Liu, Jenny Yuen, and Antonio Torralba. 2010. Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence* 33, 5 (2010), 978–994.

Xiyan Liu, Gaofeng Meng, Bin Fan, Shiming Xiang, and Chunhong Pan. 2020. Geometric rectification of document images using adversarial gated unwarping network. *Pattern Recog.* 108 (2020), 107576.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. 3431–3440.

Ke Ma, Sagnik Das, Zhixin Shu, and Dimitris Samaras. 2022. Learning From Documents in the Wild to Improve Document Unwarping. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.

Ke Ma, Zhixin Shu, Xue Bai, Jue Wang, and Dimitris Samaras. 2018. DocUNet: Document Image Unwarping via a Stacked U-Net. In *IEEE Computer Vision and Pattern Recognition (CVPR)*.

Amir Markovitz, Inbal Lavi, Or Perel, Shai Mazor, and Roee Litman. 2020. Can You Read Me Now? Content Aware Rectification Using Angle Supervision. In *European Conference on Computer Vision (ECCV)*. 208–223.

Gaofeng Meng, Yuanqi Su, Ying Wu, Shiming Xiang, and Chunhong Pan. 2018. Exploiting Vector Fields for Geometric Rectification of Distorted Document Images. In *European Conference on Computer Vision (ECCV)*. 180–195.

Gaofeng Meng, Ying Wang, Shenquan Qu, Shiming Xiang, and Chunhong Pan. 2014. Active Flattening of Curved Document Images via Two Structured Beams. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. 3890–3897.

Lothar Mischke and Wolfram Luther. 2005. Document Image De-Warping Based on Detection of Distorted Text Lines. In *Int. Conf. Image Anal. Process.* 1068–1075.

Tobias Pfaff, Nils Thuerey, Jonathan Cohen, Sarah Tariq, and Markus Gross. 2010. Scalable Fluid Simulation Using Anisotropic Turbulence Particles. *ACM Trans. Graph.* 29, 6 (2010).

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. 2021. Vision transformers for dense prediction. In *IEEE International Conference on Computer Vision (ICCV)*. 12179–12188.

Dhaval Salvi, Kang Zheng, Youjie Zhou, and Song Wang. 2015. Distance Transform Based Active Contour Approach for Document Image Rectification. In *IEEE Winter Conf. on App. Comput. Vis.* 757–764.

Nikolaos Stamatopoulos, Basilis Gatos, Ioannis Pratikakis, and Stavros J. Perantonis. 2011. Goal-Oriented Rectification of Camera-Based Document Images. *IEEE Trans. Image Process.* 20, 4 (2011), 910–920.

Mingxuan Sun, Ruigang Yang, Lin Yun, G. Landon, W. Brent Seales, and Michael S. Brown. 2005. Geometric and photometric restoration of distorted documents. In *IEEE International Conference on Computer Vision (ICCV)*, Vol. 2. 1117–1123.

Yusuke Takezawa, Makoto Hasegawa, and Salvatore Tabbone. 2017. Robust Perspective Rectification of Camera-Captured Document Images. In *IAPR Int. Conf. Document Anal. Recog.*, Vol. 06. 27–32.

Chew Lim Tan, Li Zhang, Zheng Zhang, and Tao Xia. 2006. Restoring warped document images through 3D shape modeling. *IEEE Trans. Pattern Anal. Mach. Intell.* 28, 2 (2006), 195–208.

Yuandong Tian and Srinivasa G. Narasimhan. 2011. Rectification and 3D reconstruction of curved document images. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. 377–384.

Yau-Chat Tsoi and Michael S. Brown. 2007. Multi-View Document Rectification using Boundary. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. 1–8.

Adrian Ulges, Christoph H. Lampert, and Thomas M. Breuel. 2005. Document image dewarping using robust estimation of curled text lines. In *IAPR Int. Conf. Document Anal. Recog.*, Vol. 2. 1001–1005.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Adv. Neural Inform. Process. Syst.* 6000–6010.

Toshikazu Wada, Hiroyuki Ukida, and Takashi Matsuyama. 1997. Shape from Shading with Interreflections Under a Proximal Light Source: Distortion-Free Copying of an Unfolded Book. *Int. J. Comput. Vis.* 24, 2 (1997), 125–135.

Zhou Wang, Eero P Simoncelli, and Alan C Bovik. 2003. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, Vol. 2. Ieee, 1398–1402.

Xingjiao Wu, Ziling Hu, Xiangcheng Du, Jing Yang, and Liang He. 2021. Document Layout Analysis via Dynamic Residual Feature Fusion. In *Int. Conf. Multimedia and Expo.* 1–6.

Ke Xian, Chunhua Shen, Zhiguo Cao, Hao Lu, Yang Xiao, Ruibo Li, and Zhenbo Luo. 2018. Monocular relative depth perception with web stereo data supervision. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. 311–320.

Guo-Wang Xie, Fei Yin, Xu-Yao Zhang, and Cheng-Lin Liu. 2021a. Dewarping Document Image By Displacement Flow Estimation with Fully Convolutional Network. *CoRR* abs/2104.06815 (2021).

Guo-Wang Xie, Fei Yin, Xu-Yao Zhang, and Cheng-Lin Liu. 2021b. Document Dewarping with Control Points. In *International Conference on Document Analysis and Recognition*. Springer, 466–480.

Shaodi You, Yasuyuki Matsushita, Sudipta Sinha, Yusuke Bou, and Katsushi Ikeuchi. 2018. Multiview Rectification of Folded Documents. *IEEE Trans. Pattern Anal. Mach. Intell.* 40, 2 (2018), 505–511.

Ali Zandifar. 2007. Unwarping scanned image of Japanese/English documents. In *Int. Conf. Image Anal. Process.* 129–136.

Jiaxin Zhang, Canjie Luo, Lianwen Jin, Fengjun Guo, and Kai Ding. 2022. Marior: Margin removal and iterative content rectification for document dewarping in the wild. *arXiv preprint arXiv:2207.11515* (2022).

Li Zhang, A.M. Yip, M.S. Brown, and Chew Lim Tan. 2009. A unified framework for document restoration using inpainting and shape-from-shading. *Pattern Recog.* 42, 11 (2009), 2961–2978.

Li Zhang, Yu Zhang, and Chew Tan. 2008. An Improved Physically-Based Method for Geometric Restoration of Distorted Document Images. 30, 4 (2008), 728–734.

Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. 2019. PubLayNet: largest dataset ever for document layout analysis. In *IAPR Int. Conf. Document Anal. Recog.* 1015–1022.