Can You Read Me Now? Content Aware Rectification using Angle Supervision

Amir Markovitz*, Inbal Lavi*, Or Perel, Shai Mazor, and Roee Litman

Amazon Web Services
{amirmak, ilavi, orperel, smazor, rlit}@amazon.com

Abstract. The ubiquity of smartphone cameras has led to more and more documents being captured by cameras rather than scanned. Unlike flatbed scanners, photographed documents are often folded and crumpled, resulting in large local variance in text structure. The problem of document rectification is fundamental to the Optical Character Recognition (OCR) process on documents, and its ability to overcome geometric distortions significantly affects recognition accuracy. Despite the great progress in recent OCR systems, most still rely on a pre-process that ensures the text lines are straight and axis aligned. Recent works have tackled the problem of rectifying document images taken in-the-wild using various supervision signals and alignment means. However, they focused on global features that can be extracted from the document's boundaries, ignoring various signals that could be obtained from the document's content.
We present CREASE: Content Aware Rectification using Angle Supervision, the first learned method for document rectification that relies on the document's content, the location of the words and specifically their orientation, as hints to assist in the rectification process. We utilize a novel pixel-wise angle regression approach and a curvature estimation side-task for optimizing our rectification model. Our method surpasses previous approaches in terms of OCR accuracy, geometric error and visual similarity.

1 Introduction

Documents are a common way to share information and record transactions between people. In order to digitize mass amounts of printed documents, the hard copies are scanned and text is extracted automatically by Optical Character Recognition (OCR) systems, such as [11,12]. In the past, most documents were scanned in flatbed scanners. However, the past few years have seen a rise in the use of smartphones, and with it the use of the smartphone camera as a document scanner. Camera captured documents such as receipts are often folded, curved, or crumpled, and vary greatly in camera angles, lighting and texture conditions. This makes the OCR task much more challenging compared to scanned images.

* - Equal Contribution

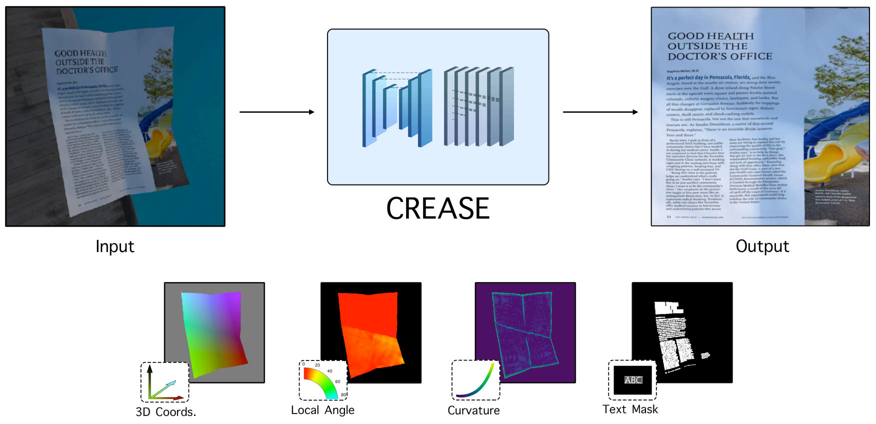

Fig. 1: **Overview of CREASE.** CREASE is a document rectification method that learns content based signals in addition to the document's 3D structure in order to estimate a transformation used for rectifying the image. On the left is a synthetically generated input image, and on the right is the image rectified using the transformation that CREASE predicted. The bottom images are the supervision signals, from left to right: The 3D coordinates of the warped document, the angle deformation map, the curvature map, and the text mask.

Recent OCR methods have had great success in recognizing text in very challenging scenarios. One example is scene text recognition [1,18] which aims to recognize text in natural images. The text is often sparse, and may also be rotated or curved. Another scenario is retrieving the content of a document with *dense* text, that poses the challenge of detecting and recognizing many words that are closely located.

While recognizing dense text, and similarly detecting sparse curved text, had been studied thoroughly, the combined problem of both dense and warped text detection and recognition has received significantly less attention. Many text detectors assume axis-aligned text and struggle with deformed lines [8,26], while text recognition systems struggle with fine deformations on the character level. Taking this into account, a line of works proposed to rectify the document as a pre-process to the recognition phase. Recent methods harnessed the power of deep learning to solve this task [6,13], but put more emphasis on the page boundaries and less emphasis on the contents of the document.

In this paper, we present CREASE: *Content Aware Rectification using Angle Supervision*. This method performs document rectification by relying on both global and local hints, with an emphasis on content. Our method predicts the 3D structure globally while simultaneously optimizing for the local structure of both the text orientation, the location of folds and creases, and the output backward map. CREASE provides results that are superior in readability, similarity and geometric reconstruction.

CREASE predicts the mapping of a warped document image to its “flatbed” version. First, we estimate the 3D structure of the input document. Then, we transform this estimation into a mapping, specifically, a backward mapping. Finally, the mapping is used to resample the warped image into the flattened form. A general overview of CREASE is given in Figure 1. Our contributions are as follows:

1. We present a per-pixel angle regression loss that complements the 3D structure estimation by optimizing different aspects of the rectification process.
2. We present a curvature estimation task, which predicts the lines along which the document is crumpled or folded, complementing the per-pixel angle regression loss by emphasizing its discontinuities.
3. The losses are learned as side tasks, focusing on the areas of the document that contain strong signals regarding the text orientation, and are optimized alongside the 3D structure estimation in an end-to-end optimization process.
4. We reduce the relative OCR error in a challenging warped document dataset by 20.2% and the relative geometric error by 14.1%, compared to the state-of-the-art method.

We train CREASE using synthetic data, which provides us with intricate details regarding each document in our training set without requiring manual annotation: the ground-truth transformation for every pixel, the 3D coordinates, angles, curvature values for every pixel, and the text segmentation mask.

We present visual and quantitative results and comparisons on both synthetic and real evaluation datasets. We also present a detailed study of the contribution of each individual model component.

2 Background

Many works have addressed the problem of extracting text from documents captured in challenging scenarios. These works have focused on different elements that improve OCR accuracy, such as illumination and noise correction [16,2], resolution enhancement [27], and document rectification [7,13,6]. This work focuses on the last problem of rectifying a warped document from a single image by prediction of the 3D model.

Early document rectification methods used hand crafted features to detect the structure of a document. These methods usually made strong assumptions on the deformation process such as smoothness [10,4] and folding structure [7]. Several works utilized special equipment to capture the 3D model of the document [3], or reconstructed the 3D model from multi-view images [25].

More recently, approaches such as [13,6,16] have used deep learning to rectify single-image, camera-captured documents, and were designed to solve the rectification problem by directly predicting the document warp. These works placed the focus on the aforementioned warp, and gave no special treatment to the content of the document, i.e., the data we wish to ultimately recognize.

The first in this line of works is DocUnet [13], which used a stacked hourglass architecture to predict the original 2D coordinates of each pixel in the warped document. This prediction, in essence, gives the forward mapping from the rectified image to the warped one, which can be inverted to get the final result.

A followup work presented DewarpNet [6], which added three learned post-processing components for calculating the backward map, the surface normals and the shading, each as a separate hourglass network. Additionally, the original 2D forward map of [13] was replaced by a prediction of its 3D counterpart, and the stacked hourglass was substituted by a single hourglass network for this prediction. This method mostly relied on the document boundary, and did not explicitly address the document's content in the rectification module.

Another work by Li et al. [16] focused mainly on uneven background illumination, however it did this by predicting the document warp. This work computed a forward map in a method similar to [13], but divided the prediction into three phases. First, a local, patch-based network predicted the gradients of the forward map. Then, a graph-cut model stitched these patch-predictions into a global warp. Finally, the un-warped image underwent an illumination correction. The local and global level prediction allowed the warp estimation to take into account both the document boundaries and areas in the center of the document, but this pipeline required an expensive patch-stitching process and worked best on input documents with minimal background. Additionally, this method was not end-to-end trainable and did not take into account the document content.

One of the key points in our approach is the importance of predicting text orientation in the warped image. The notion of text angle prediction was previously explored in scene text recognition at the word (or, object) level [28,19,17], as opposed to our pixel-level approach.

The EAST text detector by Zhou et al. [28] and FOTS detector by Liu et al. [17] both predicted the angle for each word detection candidate in conjunction with other parameters, like bounding box size and quadrangle coordinates. Ma et al. [19] extended upon the Faster-RCNN [20] architecture by adding rotated anchors to accommodate for arbitrarily oriented text. It is important to stress that scene text methods deal with a sparse set of words, and moreover, each word is rectified separately. Documents, on the other hand, benefit more from a rectification process before word localization, due to the denser text but also due to a stronger prior on the structure thereof. To make use of this prior, CREASE applies an angle regression loss at the pixel level, focused on the salient text areas in the document, and optimized in an end-to-end manner over the predicted backward map.

3 Method

We design CREASE to exploit a document's content and geometry on both local and global levels. CREASE addresses different aspects of the input, such as global structure, creases and fold lines, and per-pixel angular deformation. This allows

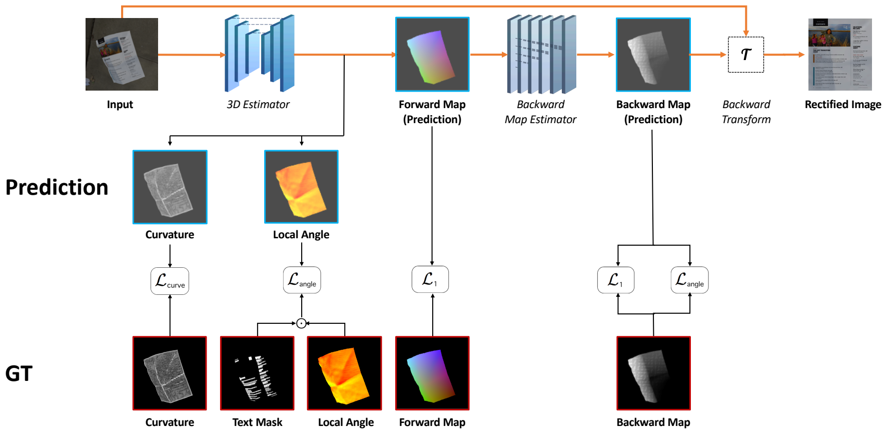

Fig. 2: **Architecture of CREASE.** The rectification process (in orange arrows) contains two steps: a 3D estimator predicts the 3D coordinates of the document in the image, and a backward map estimator that infers the backward map from the 3D estimation. The input image is rectified using the backward map. Red and blue frames denote ground truth supervisions and predictions, accordingly. Black arrows denote the training process, and the losses used for optimization. Training is performed first on the 3D estimation model, and is then fine-tuned in an end-to-end fashion.

it to capture an accurate mapping for the entire document globally, and for fine-grained features such as characters and words locally (without detecting them explicitly).

First, we present the general architecture of our model (sub-section 3.1). Next, we present the properties of documents that CREASE relies on: the flow field angles (sub-section 3.2), and the curvature estimation (sub-section 3.3). Finally, we present the optimization objective tying the various signals together (sub-section 3.4).

3.1 Architecture

CREASE is comprised of a two-stage network, illustrated in Figure 2. The first stage is used for estimating the location of each pixel in a normalized 3D coordinate system, the warp field angle values, and the curvature in each pixel. The 3D estimation module is followed by a backward mapping network. This network transforms the estimated 3D coordinate image into a backward map that can be used for rectifying the input image.

3D Estimation. The first stage provides per-pixel estimation for 3D coordinates (based on [6]) along with the angle and curvature outputs, used as side-tasks. A Unet [21] based architecture is used for mapping an input image into the angle, curvature and 3D coordinate maps. The three maps are used for supervision, and the 3D coordinate map also functions as the input for the backward mapping stage.

Backward Mapping. The second network stage transforms the 3D coordinate map outputted from the first stage to the backward mapping of the image. This mapping describes the transformation from the warped image to the rectified result. In other words, the backward map determines for every pixel in the rectified (output) image domain, its location in the input image.

The authors of [13] used a straightforward implementation of deducing the backward map by inverting their UV forward map prediction. This inversion is done by 'placing' the pixels of the forward map in the rectified image based on their values, and performing interpolation over the resulting non-regular grid[1]. This inversion is very sensitive to noise in the forward map, e.g., if two neighboring pixels swap their predictions. The same task was addressed in [16] using a parallel iterative method that isn't applicable for an end-to-end differential solution.

We rely on the DenseNet [9] based model provided by [6] for the backward mapper in our solution. The work of [6] introduced a learned model for this warp and trained it in a manner independent of the input texture. We rely on their work for transforming our 3D coordinate maps into backward maps. The use of a differentiable backward mapper allows for end-to-end training of the model using a combined objective, optimizing the 3D estimation and backward mapping networks jointly using a combined objective, as discussed in sub-section 3.4.

3.2 Angle Supervision

While the 3D estimation network learns the global document structure, we wish the network will also be aware of the local angular deformation that each point of the document has undergone during the warping. Angular deformation estimation complements the 3D regression used in [6] because it is more sensitive to small deformations that might warp parts of words. To calculate this value, we warp a local Cartesian system from the source to the target image. We use angular deformation estimation in two places in our framework.

Angle from backward map. The first place we estimate the angle is the backward map, where in each pixel we create two infinitesimal vectors ε_x and ε_y, respectively directed at the x and y directions. We then measure the *rotation* that ε_x and ε_y undergo due to the warping process, and denote the resulting angles as θ_x and θ_y. This process is illustrated in Figure 3. These angles capture the *rotation* and *shear* parts of a local affine transform, without the *translation* and *scale* counterparts that are captured by the coordinate regression.

[1] For more details see the Matlab code in [13]

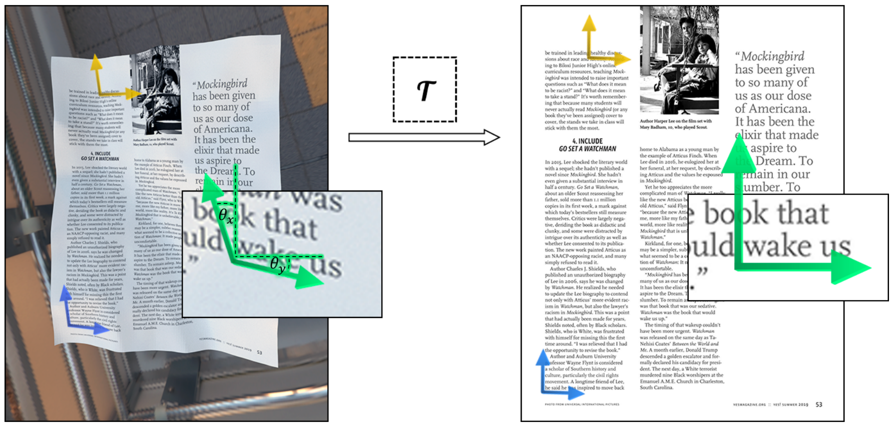

Fig. 3: **Per-Pixel Angle Calculation.** Illustration of the local warp angle calculation in three locations on the document. The green point is enlarged to provide better details, showing the resulting angles per axis, denoted by θ_x and θ_y. This process is used to penalize for angle prediction errors in the the backward map. A similar process is done for the forward map, by selecting points from the warped image and transferring them using $\boldsymbol{\tau}$.

Auxiliary Angle Prediction. In addition to deriving the angles from the backward map, we predict them directly from the 3D estimator network as two auxiliary prediction maps. These are learned in parallel to the 3D coordinates prediction, as shown in Figure 2, to better guide the training, and are not used during test time. Specifically, each of the two angles θ_x and θ_y is derived from its own pair of channels, followed by a Cartesian-to-polar conversion (see supplementary material for details). This conversion yields, in addition to angles θ_x and θ_y, corresponding magnitude values denoted ρ_x and ρ_y. We use the magnitude values as 'angle-confidence' to penalize the angle loss proportionally. This is beneficial since predictions that have small magnitude are more sensitive to small perturbations.

Angle Estimation Loss. We employ a per-pixel angle penalty on the two aforementioned predictions: one as derived from the backward map, and the other as an auxiliary prediction. The per-pixel prediction provided by the 3D estimation network is masked by a binary text segmentation map. Often, the strongest deformations appear around the borders, and far from content. Masking content-less areas allows the loss to target areas of interest, and avoid bias towards the highly deformed boundaries. Our loss minimizes the smallest angle (modulo 2π) between each of the predicted angles $\{\theta_x, \theta_y\}$ and their ground truth counterparts $\{\hat{\theta}_x, \hat{\theta}_y\}$. The per-pixel loss for angles is therefore:

$$L_{angle}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\rho}}) = \sum_{i \in \{x,y\}} \hat{\rho}_i \odot (\|\theta_i - \hat{\theta}_i\| - \pi) \mod 2\pi, \tag{1}$$

where $\odot$ denotes the Hadamard product. In the backward map angles the loss is without the confidence values $\hat{\rho}$, that are set to $\mathbf{1}$, because the angles are derived from the backward map and are not predicted as an auxiliary.

3.3 Curvature Estimation

A key observation we utilize in this work is that the surface of a crumpled document behaves in manner similar to a 2D piecewise-planar surface. Each interface between two approximately-planar sections introduces a section of higher curvature, and higher local distortion. We wish to give the network a supervision signal that indicates the presence of such high curvature.

Intuitively, the more crumpled the paper, the more creases or discontinuities the warp function exhibits. The *curvature map* highlights non-planar areas of the paper, where 3D and angle regression might be less accurate. A point in the middle of a plane would have zero curvature, while a point at the tip of a needle would have the maximal curvature value. To generate this signal, we utilize the 3D mesh used to generate each document image.

Formally, for a paper mesh $\mathcal{M}$ we calculate a curvature map $H(\mathcal{M})$ using the Laplace-Beltrami operator, as defined for meshes in [23]. The mean curvature per mesh vertex $\mathbf{v}_i \in \mathbb{R}^3$ is obtained by:

$$H(\mathcal{M})_i = || \sum_{j \in N_i} (\mathbf{v}_i - \mathbf{v}_j) ||_2. \tag{2}$$

The maps used for supervision are created by thresholding the curvature, to avoid noise and slight perturbations while emphasizing the actual lines defining the global deformation for the paper. The maps are used as supervision and are predicted as an additional segmentation mask by the 3D estimation network.

3.4 Optimization

The optimization of our model consists of two stages: An initial training stage for the 3D estimation network using the side-tasks, followed by an end-to-end fine tuning stage in which the network is optimized w.r.t. a combined loss term.

3D Estimation Model. Initially, we optimize the 3D estimation model using a loss objective that includes the 3D coordinate estimation loss and the aforementioned auxiliary losses, described in Equation (3). We denote the predicted and ground-truth normalized world coordinates $\mathbf{C}$ and $\hat{\mathbf{C}}$. The first loss term is the L_1 loss over coordinate error, similarly to [6]. The second term is the angle loss term presented in Equation (1), masked by the binary text segmentation mask $\hat{\mathbf{D}}$, averaged over all text containing pixels. The last term is the curvature estimation L_2 loss. The 3D estimation loss is:

$$L_{3D} = ||\mathbf{C} - \hat{\mathbf{C}}||_1 + \hat{\mathbf{D}} \odot L_{angle} + ||\mathbf{H} - \hat{\mathbf{H}}||_2. \quad (3)$$

End-to-end Fine Tuning. The first stage of our model may either be trained individually, or as part of an end-to-end architecture. When training the model end-to-end, the backward map B is inferred and used for penalizing the predicted 3D coordinates by the final result. We penalize the resulting backward map $\hat{B}$ by the L_1 loss as was done in [6], and additionally using our angle loss from Equation (1). We append these penalty terms to the one in Equation (3), resulting in the following combined end-to-end loss:

$$L_{combined} = L_{3D} + ||\mathbf{B} - \hat{\mathbf{B}}||_1 + L_{angle}. \quad (4)$$

4 Experiments

We evaluate CREASE on a new evaluation set comprised of 50 high resolution synthetic images, as well as on real images from the evaluation set proposed by [13]. The synthetic dataset is generated with both warp and text annotations, useful for OCR based evaluations and for evaluating the individual stages of our model. We provide geometric, visual, and OCR based metrics, as well as qualitative evaluations. We compare our results to Dewarpnet [6] trained on our training set, using the code and parameters that were published by the authors. As the method of Li et al. [16] isn't directly applicable to the task at hand, it is not evaluated in this section. Discussion and comparison to [16] are provided in the supplementary material.

All models were trained on 15,000 high resolution images rendered using an extension of the rendering pipeline provided by [6]. Our extensions include the generation of our supervision signals: text, curvature and angles in addition to the 3D coordinates provided by the original rendering pipeline. These added signals come at negligible cost and have no affect in test time. Further details regarding dataset generation are provided in the supplementary material.

4.1 Evaluation metrics

OCR Based Metric. To correctly evaluate any word related metric, we must first obtain a set of aligned word location pairs, i.e., a matching polygon for each ground-truth bounding box in the predicted rectified image domain. Given the density and small scale of words in documents, a naive coordinate matching scheme is likely to fail, as a small global shift is to be expected even in the best case scenario.

During evaluation, we rectify an input image twice: using the network's predicted backward map, and using the ground-truth map. We then use an OCR engine for extracting words and bounding boxes from the rectified images.

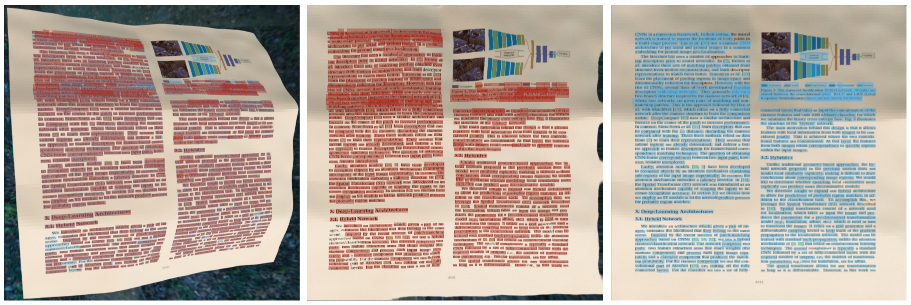

Fig. 4: **OCR Polygon matching.** Frames overlayed with OCR bounding boxes. Left: Input image overlayed with the warped polygons from images rectified using a predicted transformation (center) and the ground-truth transformation (right). Purple areas denote a correct match, blue and red areas were detected only in the ground-truth and prediction rectified images, respectively.

To properly match bounding boxes, we perform the matching stage in the input image domain, visualized in Figure 4. Each bounding box extracted from a rectified image is warped back and becomes a polygon in the input (warped) image domain.

We define polygon intersection as our distance metric and match pairs using the Hungarian algorithm [14]. With the paired prediction and ground-truth word boxes we can evaluate the *Levenshtein distance* [15], or edit distance, denoted by E_d. We first calculate the edit distance for each word in each document, then calculate the average edit distance over all the words in the dataset.

Following [6], we use an off-the-shelf OCR engine (Tesseract 4.0 [22]). This engine is quite basic, and does not reflect the advances and robustness of more modern OCR models. However, the vast majority of recent OCR methods are targeted at scene-text, with the number of proposed text instance detections often limited to 100-200. Thus, they are not suited to handle *dense* document text. As an alternative, there are a few commercial products designed to handle dense text recognition [12,11] that are far more advanced than Tesseract. We choose one of them, [11], for an additional evaluation. Results are presented in Tables 1 and 2.

Geometric and Visual Metrics. In addition to an OCR-based evaluation, we use two metrics for evaluating the geometric correctness and visual similarity of our results, *End Point Error (EPE)* and Multi-Scale Structural Similarity (MS-SSIM). The EPE metric is used to evaluate the calculated rectification warps and compare them to ground truth. Following [16], we include evaluation for this metric in our benchmark.

The MS-SSIM [24] metric quantifies how visually similar are the output images to the ground truth. Given that a small amount of shift is expected and

Table 1: Benchmark Comparison Using Tesseract OCR [22]. For E_d and EPE, lower is better, while or SSIM, higher is better.

	$\downarrow E_d$	$\downarrow$ EPE	$\uparrow$ SSIM
DewarpNet [6]	0.223 ± 0.014	0.051 ± 0.001	0.403 ± 0.004
Ours	0.178 ± 0.003	0.043 ± 0.002	0.411 ± 0.002
Improvement	+20.2%	+14.1%	+1.4%

Table 2: Benchmark Comparison Using A Commercial OCR Model [11]

	$\downarrow E_d$
DewarpNet [6]	0.109 ± 0.005
Ours	0.103 ± 0.001
Improvement	+5.1%

is not considered an error, a naive evaluation using L_1 or L_2 metrics is not suited for our evaluation. Therefore, following [6] we use the MS-SSIM metric which focuses on statistical measures rather than per-pixel color accuracy. Evaluating statistics rather than per-pixel accuracy also has its limitations, as character level rectification is a fine-grained task and improvements on this scale are not always manifested in this metric. In fact, SSIM is much more sensitive to small visual deformations in documents containing large amounts of text or sharp edges. Thus, we only use it to complement our finer-grained, OCR based metrics. For further discussion regarding the SSIM metric, see supplementary.

4.2 Implementation Details

Models are trained by first optimizing the 3D estimation network using 3D coordinates, text masks, curvature masks and local angle supervision signals to convergence. Starting from the converged 3D estimation models, we fine-tune our model in an end-to-end manner by using a fixed, pre-trained, differentiable backward mapper. We calculate the L_1 and angle losses over the output backward maps and back-propagate the losses to the 3D estimation network. Training is conducted using 15,000 high-resolution images rendered in Blender [5] using over 8,000 texture images. Further details regarding data generation are provided in the supplementary material.

4.3 Comparison to DewarpNet [6]

The first result we present is a comparison to the prior state-of-the-art trained on our training set, using the Tesseract [22] engine.

We show mean and standard deviation values over 5 experiments in Table 1. Our method improves the edit distance metric over the previous method by 4.5% absolute and 20.2% relative. We also see improvements in EPE and SSIM metrics, and a reduction in standard deviation for all three. The use of both

Table 3: Angle Loss Evaluation for the 3D Estimation Model.

Model	$\downarrow E_d$	$\downarrow$ EPE	$\uparrow$ SSIM
Vanilla	0.324 ± 0.169	0.066 ± 0.029	0.390 ± 0.024
Angles	0.246 ± 0.017	0.049 ± 0.002	0.403 ± 0.005
+ Mask	0.244 ± 0.014	0.052 ± 0.001	0.398 ± 0.005
+ Conf.	0.216 ± 0.017	0.049 ± 0.001	0.400 ± 0.005

Table 4: Ablation Study.

	$\downarrow E_d$	$\downarrow$ EPE	$\uparrow$ SSIM
Vanilla	0.324 ± 0.169	0.066 ± 0.029	0.390 ± 0.024
Angles	0.216 ± 0.017	0.049 ± 0.001	0.400 ± 0.005
Angles + Curvature	0.187 ± 0.005	0.043 ± 0.001	0.409 ± 0.007
E2E	0.223 ± 0.014	0.051 ± 0.001	0.403 ± 0.004
E2E + Angles	0.204 ± 0.015	0.051 ± 0.002	0.402 ± 0.005
E2E + Angles + Curvature	0.178 ± 0.003	0.043 ± 0.002	0.411 ± 0.002

angle regression and curvature estimation improves performance and stabilizes the optimization process, reducing the sensitivity to model initialization.

Next, we evaluate our method using the public online API of [11]. Results are presented in Table 2. In this case, our model still provides a 5.1% relative improvement. The commercial model [11] is superior to [22], reducing the mean edit distance from 0.178 to 0.103, yet CREASE still maintains a significant gap over DewarpNet of 0.6% absolute and 5.1% relative.

4.4 Evaluation using Real World Images

Figure 5 depicts a qualitative comparison between our rectification method and [6] on the real images provided by [13]. Notice how the text lines rectified using CREASE are better aligned and easier to read than the other method's outputs, especially for text near document edges. Additional examples are included in the supplementary material.

4.5 Angle Loss Evaluation

We show the contribution of the different elements of our angle-based loss presented in Section 3.2 for our metrics and for the OCR metric in particular in Table 3. 'Angles' refers to models trained with the angle loss applied to all image pixels, instead of only to those that contain text. '+ Mask' refers to applying the text mask over the loss, i.e., taking the loss only in text-containing pixels, using the mask denoted by $\hat{\mathbf{D}}$ in Equation (3). '+ Conf.' represents the use of the angle confidence values (denoted ρ in Equation (1)). When not used, we set ρ to 1 for all pixels. We report results averaged over 5 experiments each, as well as the standard deviation. For this experiment, the curvature estimation term

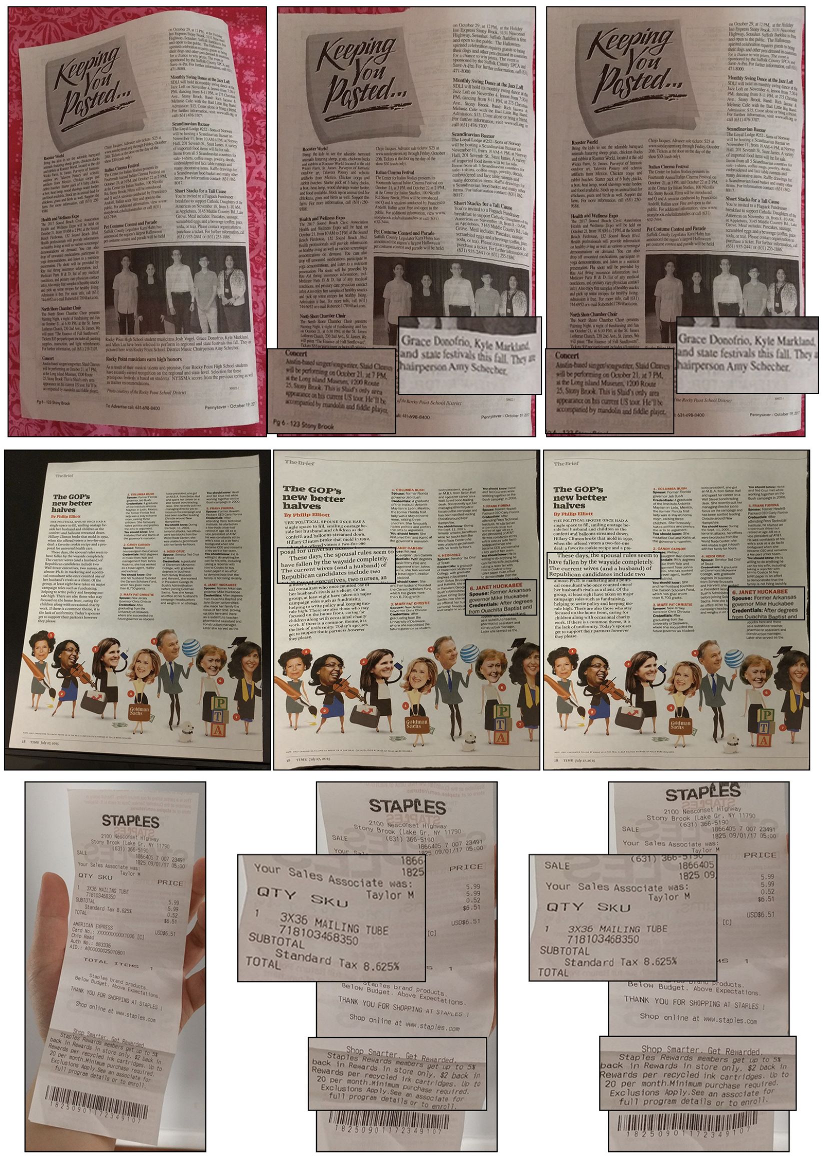

Fig. 5: **Visual Examples.** Results from the real image dataset of [13]. Left to right: input images, rectification of the input image according to output of [6] trained using our data, rectification of the input image according to our model's output transformation.

was omitted. Our contributions show a consistent improvement over the vanilla 3D estimation network and, in addition, a much more stable training framework with consistent results over multiple initializations.

4.6 Ablation Study

Table 4 shows the effect of each component of our method. Models trained using angle and curvature estimation are compared to vanilla models. We compare both models trained end-to-end (denoted $E2E$) and models trained separately. As seen before, the improvement in results is also accompanied by a decrease in standard deviation, especially for models trained using curvature estimation.

We evaluate the contribution of end-to-end training of our model using a fixed, differentiable backward mapper and losses derived from its results, i.e., the backward map and angle prediction errors (shown in Table 4). The top three rows refer to models that were not trained in an end-to-end fashion, while the three rows below (starting with 'E2E') refer to models trained end-to-end. 'Angles' and 'Curvature' denote the use of each of our two added auxiliary predictions.

The dual usage of the angle loss, in both the 3D estimation model and the end-to-end training, as well as the curvature estimation, result in much more readable rectification and a more stable training scheme than the previous state-of-the-art.

5 Conclusion

We presented CREASE, a content aware document rectification method which optimizes a per-pixel angle regression loss, a curvature estimation loss and a 3D coordinate estimation loss for providing image rectification maps.

Our method rectifies folded and creased documents using hints found in both local and global scale properties of the document, and provides a significant improvement in OCR performance, geometry and visual similarity based metrics. In our proposed two stage model, the first stage is used for predicting 3D structure, angles and curvature, while the second stage predicts the backward map. We utilize a pixel-level angle regression loss that is shown to be a beneficial side-task in both the 3D estimation and the end-to-end training. Furthermore, our 3D estimation model learns the angle side-task specifically on the words in the document, thus optimizing for readability in the rectified image, while the curvature estimation side-task complements the angle regression by mapping its discontinuities.

Extensive testing and comparisons show our method's superior performance over diverse inputs, using both real and synthetic evaluation data. We show an increase in OCR performance, geometry and similarity metrics that is consistent over all experiments and on a variety of documents.

References

1. Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J., Lee, H.: What is wrong with scene text recognition model comparisons? dataset and model analysis. In: International Conference on Computer Vision (ICCV) (2019), to appear
2. Bajjer Ramanna, V.K., Bukhari, S.S., Dengel, A.: Document image dewarping using deep learning. In: The 8th International Conference on Pattern Recognition Applications and Methods. International Conference on Pattern Recognition Applications and Methods (ICPRAM-2019), February 19-21, Prague, Czech Republic. Insticc (2019)
3. Brown, M.S., Seales, W.B.: Image restoration of arbitrarily warped documents. IEEE Transactions on Pattern Analysis and Machine Intelligence **26**, 1295–1306 (2004)
4. Burden, A., Cote, M., Albu, A.B.: Rectification of camera-captured document images with mixed contents and varied layouts. In: 2019 16th Conference on Computer and Robot Vision (CRV). pp. 33–40. IEEE (2019)
5. Community, B.O.: Blender - a 3D modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam (2018), `http://www.blender.org`
6. Das, S., Ma, K., Shu, Z., Samaras, D., Shilkrot, R.: Dewarpnet: Single-image document unwarping with stacked 3d and 2d regression networks. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
7. Das, S., Mishra, G., Sudharshana, A., Shilkrot, R.: The common fold: Utilizing the four-fold to dewarp printed documents from a single image. In: Proceedings of the 2017 ACM Symposium on Document Engineering. p. 125–128. DocEng '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3103010.3121030, `https://doi.org/10.1145/3103010.3121030`
8. Grüning, T., Leifert, G., Strauß, T., Michael, J., Labahn, R.: A two-stage method for text line detection in historical documents. International Journal on Document Analysis and Recognition (IJDAR) p. 285–302 (Jul 2018)
9. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)
10. Huang, Z., Gu, J., Meng, G., Pan, C.: Text line extraction of curved document images using hybrid metric. In: 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR). pp. 251–255 (Nov 2015). https://doi.org/10.1109/ACPR.2015.7486504
11. Inc., A.: Amazon textract, `https://aws.amazon.com/textract`
12. Inc., G.: Detect text in images, `https://cloud.google.com/vision/docs/ocr`
13. Ke Ma, Zhixin Shu, X.B.J.W.D.S.: Docunet: Document image unwarping via a stacked u-net. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (2018)
14. Kuhn, H.W.: The hungarian method for the assignment problem. Naval research logistics quarterly **2**(1-2), 83–97 (1955)
15. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals
16. Li, X., Zhang, B., Liao, J., Sander, P.V.: Document rectification and illumination correction using a patch-based cnn. ACM Transactions on Graphics (TOG) (11 2019)

17. Liu, X., Liang, D., Yan, S., Chen, D., Qiao, Y., Yan, J.: Fots: Fast oriented text spotting with a unified network. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (Jun 2018)
18. Lyu, P., Liao, M., Yao, C., Wu, W., Bai, X.: Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. Lecture Notes in Computer Science p. 71–88 (2018)
19. Ma, J., Shao, W., Ye, H., Wang, L., Wang, H., Zheng, Y., Xue, X.: Arbitrary-oriented scene text detection via rotation proposals. IEEE Transactions on Multimedia **20**(11), 3111–3122 (Nov 2018)
20. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems 28, pp. 91–99. Curran Associates, Inc. (2015)
21. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
22. Smith, R.: An overview of the tesseract ocr engine. In: Ninth International Conference on Document Analysis and Recognition (ICDAR 2007). vol. 2, pp. 629–633. IEEE (2007)
23. Sorkine-Hornung, O.: Laplacian mesh processing. In: Eurographics (2005)
24. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)
25. You, S., Matsushita, Y., Sinha, S., Bou, Y.B., Ikeuchi, K.: Multiview rectification of folded documents. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**, 505–511 (2016)
26. Yousef, M., Bishop, T.E.: Origaminet: Weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020)
27. Zheng, Y., Kang, X., Li, S., He, Y., Sun, J.: Real-time document image super-resolution by fast matting. In: 2014 11th IAPR International Workshop on Document Analysis Systems. pp. 232–236. IEEE (2014)
28. Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang, J.: East: An efficient and accurate scene text detector. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Jul 2017)