Image Orientation Estimation with Convolutional Networks

Philipp Fischer*, Alexey Dosovitskiy, Thomas Brox

Department of Computer Science
University of Freiburg
{fischer,dosovits,brox}@cs.uni-freiburg.de

Abstract. Rectifying the orientation of scanned documents has been an important problem that was solved long ago. In this paper, we focus on the harder case of estimating and correcting the exact orientation of general images, for instance, of holiday snapshots. Especially when the horizon or other horizontal and vertical lines in the image are missing, it is hard to find features that yield the canonical orientation of the image. We demonstrate that a convolutional network can learn subtle features to predict the canonical orientation of images. In contrast to prior works that just distinguish between portrait and landscape orientation, the network regresses the exact orientation angle. The approach runs in real-time and, thus, can be applied also to live video streams.

1 Introduction

Sometimes, taking a picture can take time. Everybody in front of the lens is smiling happily, but while the amateur photographer tries to get all lines perfectly horizontal struggling with the lens distortion, the subjects of interest start to become a little uneasy. Would it not be nice, if one could just press the button and the orientation of the picture would be corrected automatically?

The inertial sensors, which are already built into modern cameras to correct the orientation of pictures in 90 degree steps, could potentially do the job, but this function is usually not implemented. In this paper, we present a way to automatically correct the image orientation based just on the visual data. It can be applied as a post-processing step to pictures taken with any camera, including older models without inertial sensors.

Orientation correction is a long standing task in document analysis [21, 3, 16, 13, 9, 2]. However, all these methods exploit the special structure of document images, such as text layout in lines and precise shapes of letters. In the general setting, the task is harder, since important features, such as text or picture boundaries are not available and even image features, such as the horizon or other dominant horizontal or vertical lines in the scene can be missing. Figure 1 shows an example, where traditional line-based approaches will most likely fail.

* Supported by a scholarship of the Deutsche Telekom Stiftung

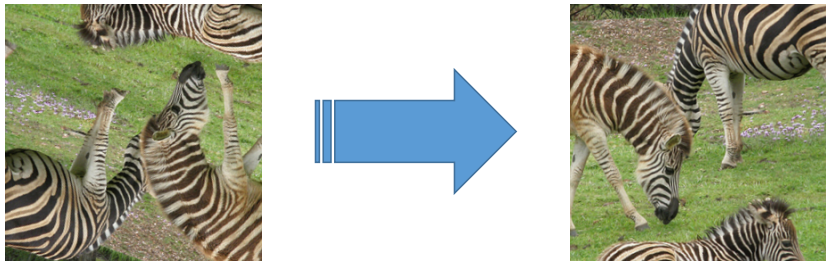

Fig. 1. Automatic orientation adjustment. The rotation angle of the input image is estimated by a convolutional network. With this information the image can be adjusted to its canonical orientation. In this example, there are hardly any useable horizontal or vertical lines. The existing lines are in fact confusing.

In such cases, the result depends on very subtle features that require some understanding of the image content. In the last three years, deep convolutional networks have been shown to be very good at learning such features. The initial success was on classification tasks [8], but also other problems that require the use of initially unspecified features, such as depth map prediction from single images, have been approached successfully with convolutional networks [4].

In this paper, we train a convolutional network to predict the orientation of an image. We consider the problem at three difficulty levels. In the easiest setting, we assume that the rotation is at most $\pm 30°$. The task becomes more difficult if the rotation can be between $-45°$ and $+45°$, as this can lead to confusion between horizontal and vertical lines and, consequently, to an ambiguity between the landscape and portrait orientation. The most difficult setting is orientation estimation without any prior knowledge about the coarse orientation, i.e., all angles (0-360 degrees) are equally likely.

Training powerful, deep convolutional networks used to require a large amount of annotated training data. Often this is a strong restriction as the collection of such data can be very tedious. For the present task, however, training data can be generated very easily in almost arbitrary amounts, since any unlabeled set of images can serve as training set. Training samples can be generated by just rotating these images by random angles.

We demonstrate that the proposed networks successfully learn to predict the orientation with an average accuracy of 3 degrees in the setting with $\pm 30°$. It outperforms baselines built upon Hough transform or Fourier transform by a large margin. The network runs in real time on a GPU. Hence, it can also be used to stabilize a live video stream. Experiments with a webcam show that the network generalizes and has not learned to make use of potential artifacts that result from the synthesized training and test set.

2 Related work

We are not aware of work on precise estimation of the orientation of general natural images. Existing methods either reduce the domain of application or they do not predict the precise angle but only classify between a restricted set of standard orientations.

Some methods explicitly use fine structure of the image to estimate the rotation. Wei et al. [20] make use of interpolation artifacts introduced by applying rotation to digital images. However, this method would not work for images which were not taken upright. Solanki et al. [15] estimate the rotation of printed images by analyzing the pattern of printer dots. Clearly, this method does not apply to images taken with a digital camera.

Horizon detection [5, 10] is a special case of image orientation estimation. However, the horizon is not visible in most photos.

In other works the continuous-valued prediction task has been reduced to classification by restricting the rotations to multiples of 90°. This problem can be solved fairly efficiently [17, 19].

While we are not aware of applications of neural networks to estimating orientation of general images, there has been work on pose estimation with neural networks, in particular head and face orientation estimation [14, 18, 12]. Interestingly, the following networks trained to estimate orientation of arbitrary images also work well for face images.

3 Experimental setup

3.1 Data

Ideally, for training a network to predict orientation, one requires a dataset of natural images annotated with how much their rotation angle deviates from the upright orientation. In theory it would be possible to collect such a dataset using a camera with a sensitive tilt sensor like an accelerometer. However, such a procedure would be time-consuming and expensive, while being susceptible to accelerated motions of the camera.

Hence, we rather use the publicly available Microsoft COCO dataset [1] as training set and apply rotations artificially. This makes data collection trivial, but the resulting data is noisy: Microsoft COCO includes tilted images and images with undefined orientations. For the test set we discarded those images (see examples in Figure 2). While this procedure is infeasible for the large training set, our results show that a network trained on these noisy data still performs very well.

Augmentation. To prevent overfitting, we perform image augmentation, i.e., we apply random transformations to input samples during network training on the fly. We use translations (up to 5% of the image width), brightness adjustment in the range $[-0.2, 0.2]$, gamma adjustment with $\gamma \in [-0.5, 0.1]$ and Gaussian pixel noise with a standard deviation in the range $[0, 0.02]$.

Fig. 2. Examples of images that were dismissed for the test set. We discarded slanted images, framed images and images which do not have a well-defined orientation.

Manually verified test set. Images in the COCO dataset are not always perfectly straight. In order to precisely evaluate the performance of our method and compare it to the baselines, we manually selected a subset of 1030 images from the COCO validation set. For these images we ensured that they are correctly oriented. Moreover, we labeled test images as "easy" if they contained significant vertical or horizontal lines, for example, buildings, horizon, walls or doors. Other images we labeled as "difficult". In total there are 618 "easy" images and 412 "difficult" ones. Figures 3, 5 and 6 show several example test images, both "easy" and "difficult".

3.2 Tasks and networks

We consider orientation estimation at three difficulty levels: $\pm 30°$, $\pm 45°$, and the full $360°$. We call the corresponding networks trained for these tasks Net-30, Net-45, and Net-360. For all three tasks we built upon the AlexNet architecture from Krizhevsky et al. [8] implemented in Caffe [6] and pretrained on ImageNet. This architecture consists of 5 convolutional layers, followed by 3 fully connected layers. After each fully connected layer, a rectified linear unit is used as nonlinearity. Additionally normalization and dropout are applied. For more details see [8].

Using ImageNet pretraining worked better than training the network from scratch despite the good availability of training data for our tasks. It seems that the class labels from ImageNet help learn semantic features that are useful for the task but too difficult for the network to learn from the orientation objective. After pretraining, all fully connected layers were initialized with Gaussian noise. Moreover, the last prediction layer of the AlexNet was replaced by one with two output units. We chose two output units to distinguish clockwise and counter-clockwise rotations. The first output is to be active for positive rotations, while the second one is active for negative rotations. Given an angle α, the desired output vector is $[\max(0, \alpha), \max(0, -\alpha)]$. We trained the network with L1 loss.

We also set up a network with 4 class outputs corresponding to 0, 90, 180, and 270 degrees. It is trained with a 4-way softmax output and cross-correlation loss. This network can be used for the conventional task of distinguishing landscape from portrait and upside-down images. We used this network to make a coarse prediction in the 360 degree task before estimating the exact angle with the regression architecture described above.

All of the networks were trained using the Adam optimization method [7]. The momentum parameters we fixed as recommended in [7] to $\beta_1 = 0.9$ and $\beta_2 = 0.999$. As the main learning rate we set $\lambda = 1e{-}4$ and then decrease it in steps until we reach $\lambda = 2e{-}7$. Additionally we use a slow start, i.e. starting with a low learning rate $\lambda = 1e{-}6$ and increasing it until the main learning rate is reached.

3.3 Baseline methods

We did not find prior work on precisely estimating the orientation of natural images. Hence, for comparison we evaluated three simple baselines that are built upon the "Straighten image" function in Matlab Central [11] and use two traditional computer vision techniques: Hough transform and Fourier transform. All methods are based on dominant gradient or line orientations to straighten the image.

In the first method, which we refer to as *Hough-var*, edges are detected with a Prewitt filter. We then perform a Hough transform and compute the variance of each column in the Hough space, that is, for each angle we compute the variance of all values corresponding to this angle. The Hough space is folded to a range of 90 degrees such that angles with the same value modulus 90 contribute to the same bin. For the Hough space we chose a bin size of 0.5 degrees. The angle estimate is computed as the angle with maximum variance.

Also the second method, *Hough-pow*, uses the Hough transform. We use the Canny edge detector and then apply the Hough transform. We raise all the values to a power α (we found $\alpha = 6$ to work well in the experiments) and sum each column. Again, after folding the space to a range of 90 degrees, the estimated angle is the one for which this sum takes the largest value.

The third method makes use of the fast Fourier transform (FFT), we hence refer to it as *Fourier*. After performing FFT, we again fold the space such that

Task	Net-30	Net-45	Net-360	Net-rough+45	Hough-var	Hough-pow	Fourier
±30°-all	**3.00**	4.00	19.74	19.64	11.41	10.62	10.66
±30°-easy	**2.17**	2.83	19.48	17.90	8.44	7.04	8.64
±30°-hard	**4.26**	5.75	20.12	22.24	15.88	15.99	13.69
±45°-all	-	**4.63**	20.64	19.24	16.92	13.06	16.51
±45°-easy	-	**3.24**	21.26	19.29	14.08	9.01	13.32
±45°-hard	-	**6.71**	19.70	19.15	21.16	19.13	21.28
±180°-all	-	-	**20.97**	**18.68**	-	-	-
±180°-easy	-	-	**20.29**	**18.70**	-	-	-
±180°-hard	-	-	**21.98**	**18.65**	-	-	-

Table 1. Average absolute errors of the estimated angles. The networks clearly outperform the baseline methods and achieve very good accuracies on the fine orientation adjustment tasks. The full orientation estimation task is significantly harder. Here it helps to first classify the coarse orientation succeeded by a fine adjustment.

rotations with a difference of 90 degrees match. For each angle, we then sum all magnitudes corresponding to this angle to obtain its score. The angle with the maximum score determines the estimated image orientation.

4 Results

We evaluated the networks on the three subtasks and compared them quantitatively to the above mentioned baselines. Additionally we show some qualitative results, demonstrating strengths as well as limitations of the approach. As an example application, we use an orientation-estimating network for real-time video stabilization.

4.1 Fine orientation adjustment

We first evaluated the scenario in which the rough image orientation is known, and only fine adjustment, in the range of $[-30; 30]$ degrees or $[-45; 45]$ degrees, is needed. Table 1 shows the average error in degrees for the test set. The average error of the network is much lower than that of the three baselines. The histograms in Figure 4 reveal that the better performance is mainly because

Fig. 3. Orientation adjustment with Net-30 and Hough-pow. Although dominant horizontal and vertical lines are missing in most of these examples, the network predicts the correct angle.

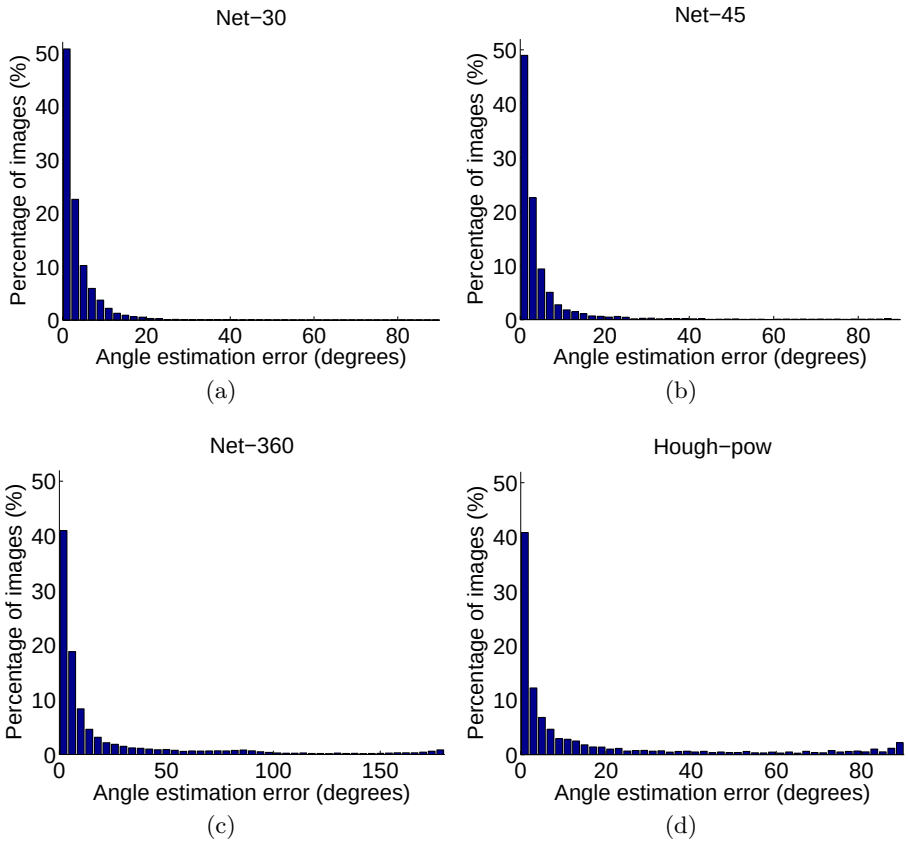

Fig. 4. Error histograms for the three networks on the $\pm 30^\circ$, the $\pm 45^\circ$, and the full orientation estimation task, as well as for the Hough-pow baseline on the $\pm 45^\circ$ task. The networks fail less often than the baseline. Note that Net-30, Net-45 and Net-360 are run on progressively harder tasks.

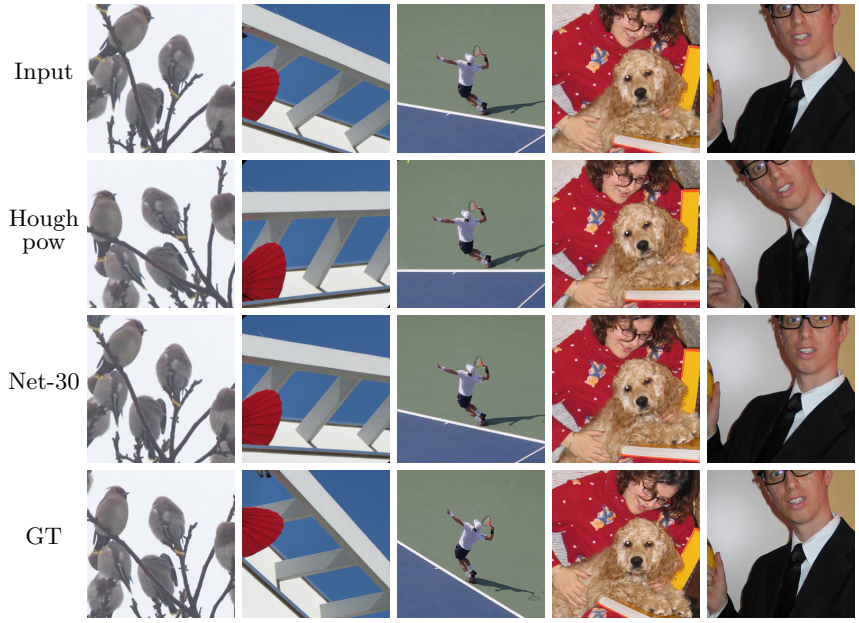

Fig. 5. Failure cases with Net-30 (samples with particularly large errors between 14 and 38 degrees). For comparison we also show the Hough-pow results. Even for humans it is hard to decide on the optimal orientation.

the network fails far less often in hard cases when traditional line features are absent. For Net-30, the average error for the $\pm 30°$ range reaches the accuracy level of the training data set. Allowing for up to $\pm 45°$ makes the task harder, since horizontal and vertical lines can now be confused. Figure 4b reveals that the accuracy is the same for good cases, but there are more failure cases at the far end of the histogram due to portrait vs. landscape confusion (please use zoom in the electronic version of the paper). Figure 3 shows some examples for the orientation adjustment. Visually, the results look very good. In Figure 5 we show cases that caused the largest errors. The optimal rotation for most of these examples is also hard to decide for humans.

4.2 Orientation estimation with full rotations

We also evaluated the case where the given image is rotated arbitrarily between 0 and 360 degrees. We tested two different approaches to solve this problem. First, we applied a network similar to those from section 4.1 with two output units. Since a direct prediction of the angle as a scalar would break continuity at one point of the cycle, the network predicts the cosine and sine of the angle and the actual angle is then obtained by the arc tangent. In a second approach, we

Fig. 6. Results with Net-360. **Top row:** input image. **Middle row:** adjusted output. **Bottom row:** ground truth. Despite large rotations, the network can often estimate the correct angle.

predict a 90 degree rough orientation followed by a fine orientation adjustment in the range $[-45, 45]$ degrees.

Table 1 reveals that the second approach performed better. In general, the full rotation task comes with significantly larger average errors than the fine adjustment. The histograms in Figure 4 reveal that this is mostly due to failure cases with very large errors. The left part of the histogram of Net-360 is still comparable to those of the other two networks. Figure 6 shows some examples for Net-360 including one larger failure case and one slight inaccuracy. It is worth noting that, in order to make use of semantic information such as faces, the network has to recognize faces although they can be upside-down in the input image.

4.3 Video stabilization

A single forward pass of our networks takes 45 milliseconds at an input resolution of 280×280 pixels on an NVIDIA GTX Titan GPU, which enables us to use it in real-time applications like video stabilization. Our video stabilization implementation can be used with any webcam and shows the original as well as the rectified image live on the screen. We used Net-45 in this experiment.

The network successfully generalizes to images captured with the webcam. Some examples are shown in Figure 7, and the reader is referred to the supplementary video for more examples. The fact that the network works with raw webcam images proves that it does not make use of interpolation artifacts which appear when artificially applying rotations. The network was trained to estimate rotations not larger than 45°, hence when presented with larger rotations,

it jumps to the 90°-rotated image (Figure 7 (a) and (b)). Interestingly, the network works well even in the absence of strong vertical and horizontal edges, successfully retrieving upright images; see Figure 7 (c) and (d).

5 Conclusions

We have presented an approach based on convolutional networks that can predict the orientation of an arbitrary natural photograph. This orientation prediction can be used to adjust the image to its canonical orientation. The use of convolutional networks is advantageous for this task, since large numbers of training samples can be generated synthetically and the network is able to learn subtle contextual features that allow it to estimate the correct orientation even when straightforward features, such as vertical or horizontal lines are not present in the image. The network runs in realtime, which allows us to apply it to live video streams. In future work the network architecture will be optimized to run in realtime on smaller graphic chips (such as the NVIDIA Tegra), which would make the approach also applicable to orientation estimation in quadcopters, providing acceleration-independent measurements besides IMUs.

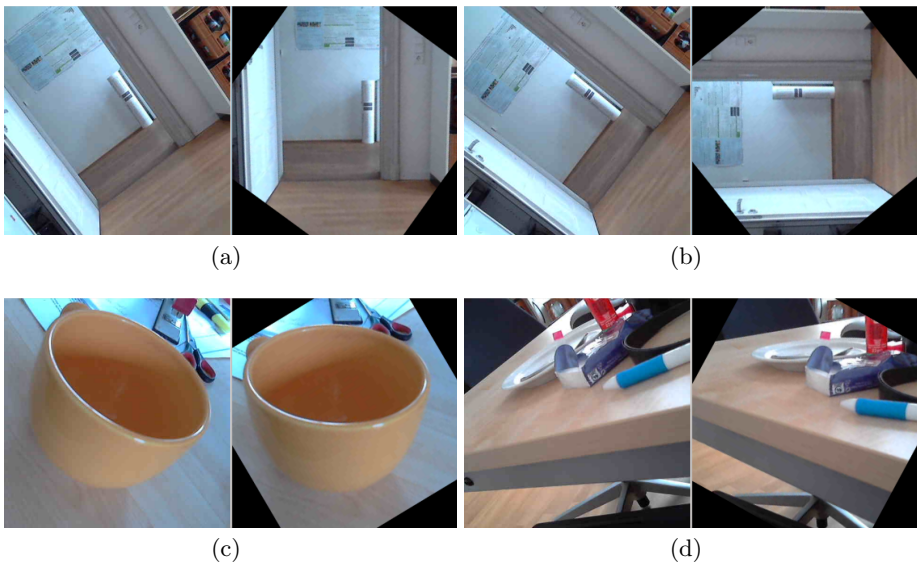

Fig. 7. Results of real-time video stabilization with Net-45. Images come in pairs: input on the left and the straightened image on the right. (a) and (b) demonstrate how the angle estimation flips at 45°, because the network expects angles between ±45°. In (c) and (d) the network predicts the correct orientation, even though the images do not contain useful vertical or horizontal gradients.

References

1. Microsoft COCO dataset. http://mscoco.org
2. Ávila, B.T., Lins, R.D.: A fast orientation and skew detection algorithm for monochromatic document images. In: Proceedings of the 2005 ACM Symposium on Document Engineering. pp. 118–126 (2005)
3. Chen, S.S., Haralick, R.M.: An automatic algorithm for text skew estimation in document images using recursive morphological transforms. In: ICIP. pp. 139–143 (1994)
4. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. NIPS (2014)
5. Fefilatyev, S., Smarodzinava, V., Hall, L.O., Goldgof, D.B.: Horizon detection using machine learning techniques. In: ICMLA. pp. 17–21 (2006)
6. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
7. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015), http://arxiv.org/abs/1412.6980
8. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1106–1114 (2012)
9. Kwag, H.K., Kim, S.H., Jeong, S.H., Lee, G.S.: Efficient skew estimation and correction algorithm for document images. Image Vision Comput. 20(1), 25–35 (2002)
10. Lipschutz, I., Gershikov, E., Milgrom, B.: New methods for horizon line detection in infrared and visible sea images. International Journal Of Computational Engineering Research (2013)
11. Motl, J.: Straighten image function in Matlab Central. http://www.mathworks.com/matlabcentral/fileexchange/40239-straighten-image
12. Osadchy, M., LeCun, Y., Miller, M.L.: Synergistic face detection and pose estimation with energy-based models. J. Mach. Learn. Res. 8, 1197–1215 (2007)
13. Peake, G.S., Tan, T.N.: A general algorithm for document skew angle estimation. In: ICIP (2). pp. 230–233 (1997)
14. Pingali, G.S., Zhao, L., Carlbom, I.: Real-time head orientation estimation using neural networks. In: ICIP. pp. 297–300 (2002)
15. Solanki, K., Madhow, U., Manjunath, B.S., Chandrasekaran, S.: Estimating and undoing rotation for print-scan resilient data hiding. In: ICIP. pp. 39–42 (2004)
16. Sun, C., Si, D.: Skew and slant correction for document images using gradient direction. In: 4th International Conference Document Analysis and Recognition (ICDAR'97). pp. 142–146 (1997)
17. Vailaya, A., Zhang, H., Member, S., Yang, C., Liu, F.I., Jain, A.K.: Automatic image orientation detection. In: IEEE Transactions on Image Processing. pp. 600–604 (2002)
18. Voit, M., Nickel, K., Stiefelhagen, R.: R.: Neural network-based head pose estimation and multi-view fusion. In: Proc. CLEAR Workshop, LNCS. pp. 299–304 (2006)
19. Wang, Y.M., Zhang, H.: Detecting image orientation based on low-level visual content. Computer Vision and Image Understanding 93(3), 328–346 (2004)
20. Wei, W., Wang, S., Zhang, X., Tang, Z.: Estimation of image rotation angle using interpolation-related spectral signatures with application to blind detection of image forgery. Trans. Info. For. Sec. 5(3), 507–517 (2010)
21. Yan, H.: Skew correction of document images using interline cross-correlation. CVGIP: Graphical Model and Image Processing 55(6), 538–543 (1993)