Content-Aware Rotation

Kaiming He
Microsoft Research Asia

Huiwen Chang*
Tsinghua University

Jian Sun
Microsoft Research Asia

Abstract

We present an image editing tool called Content-Aware Rotation. Casually shot photos can appear tilted, and are often corrected by rotation and cropping. This trivial solution may remove desired content and hurt image integrity. Instead of doing rigid rotation, we propose a warping method that creates the perception of rotation and avoids cropping. Human vision studies suggest that the perception of rotation is mainly due to horizontal/vertical lines. We design an optimization-based method that preserves the rotation of horizontal/vertical lines, maintains the completeness of the image content, and reduces the warping distortion. An efficient algorithm is developed to address the challenging optimization. We demonstrate our content-aware rotation method on a variety of practical cases.

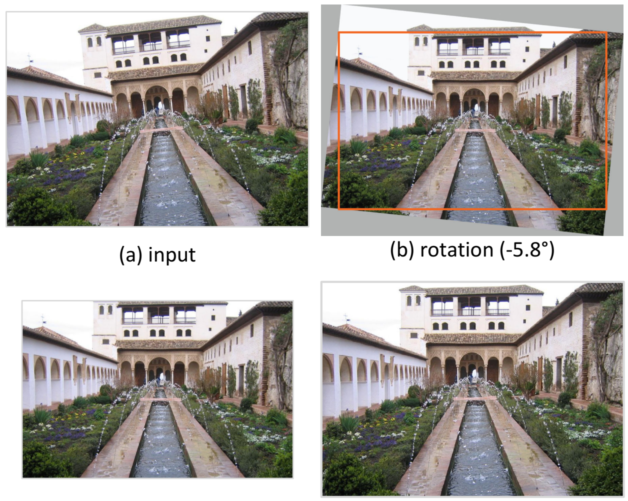

Figure 1. Image rotation. (a) An input tilted image. (b) The user rotates the image by -5.8° (clockwise 5.8°), such that its horizontal lines become level. (c) The rotated image is cropped by the largest inner upright rectangle. (d) Our content-aware rotation result.

1. Introduction

Image rotation is one of the fundamental image editing operations besides scaling and cropping. Photos casually shot by hand-held cameras/phones can appear tilted, which are sensitive to human eyes even when the rotation angle is small (Fig. 1(a)). On the other hand, artists may adjust the composition through manipulating the rotation [26]. For these reasons and others, image rotation has been incorporated in a majority of image editing softwares.

We assume the rotation angle has been given by users, algorithms [12], or sensors in devices. A rigidly rotated image inevitably has empty regions, and is often cropped to fit an upright rectangle (Fig. 1(b)). But this trivial solution may produce unwanted results: it may remove image content and hurt the integrity of the image (*e.g.*, the building in Fig. 1(c)). Actually, the cropping operation reduces the area of a typical photo by 20% even when the rotation angle is as small as 5° . A sophisticated solution is desired for common users and artists to reduce the loss of content when rotating images.

Despite little attention has been paid to the above issue on rotation, there have been an abundance of literatures on image scaling/resizing, popularly known as *retargeting* [4, 31, 29, 25, 32, 8]. Instead of cropping or uniformly scaling, the pioneer work Seam Carving [4] resorts to a *content-aware* fashion. The philosophy is that the perceptual sensitivity is not equal among different image content. So it is possible to produce visually pleasing results if the manipulations are less noticeable, *e.g.*, removing/inserting seams as in [4]. This method has been implemented as the *Content-Aware Scaling* feature in the commercial software Adobe Photoshop [2]. Recent image retargeting methods are mostly based on warping [31, 29, 32, 8], and they are designed to preserve high-level content like local shapes and straight lines. A common motivation of warping methods is to maintain the completeness of the image content at the price of distortion (as unnoticeable as possible).

With a similar motivation, in this paper we present *Content-Aware Rotation*. We design a warping method that keeps the image content inside an upright rectangle while *creating the perception of rotation*. We are driven by several observations on human vision studies [13, 23, 22, 11]. First, the perception of image tilting is mainly due to the

*This work was done when Huiwen Chang was an intern at MSRA.

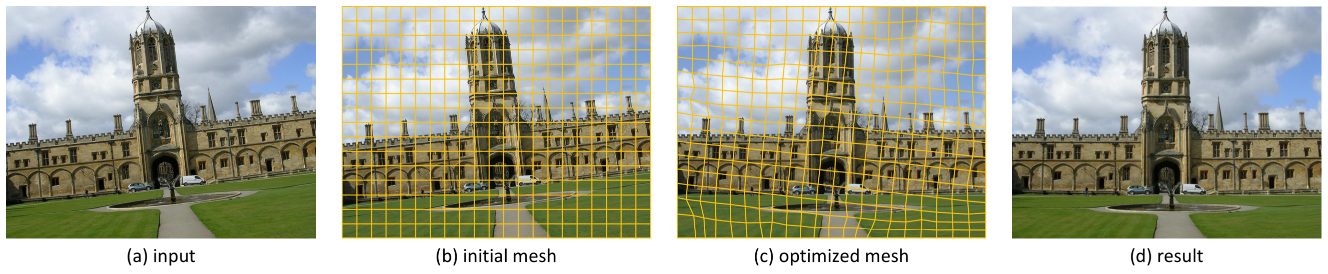

Figure 2. Content-aware rotation using warping. (a) An input image to be rotated by 6.1° . (b) The initialized grid mesh. (c) The optimized mesh. (d) The warping result.

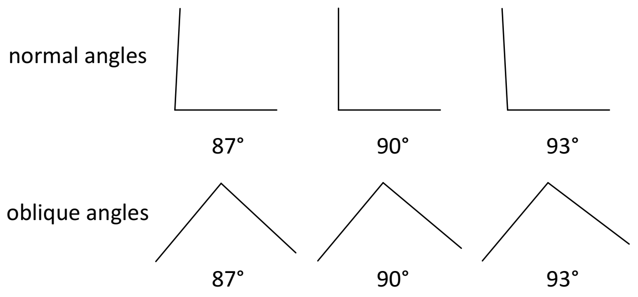

Figure 3. Perception of angles. Top: for normal angles, human eyes can easily notice the tilted vertical lines. Bottom: for oblique angles, human eyes are not sensitive to the absolute angle values. This figure is due to [13].

perception of the tilted horizon (or lines parallel to it) [11]. This is perhaps because the human visual system is horizontally binocular. Second, the human eyes are sensitive to the right angles (90°) that are "normal" [13, 23], *i.e.*, right angles with one edge being horizontal. As a result, the vertical lines are easily noticeable if tilted. Last, the human eyes are **not** sensitive to the absolute values of angles when the angles are acute/obtuse [22] or when they are "oblique" right angles (*i.e.*, not "normal") [13, 23]. The readers can experience this in Fig. 3.

The above studies suggest that the human vision system is very sensitive to the horizontal/vertical lines but less so to others. Driven by these studies, we propose to preserve the orientation of the horizontal/vertical lines (after rotation) so as to create the perception of rotation. The orientations of other lines are relaxed because they are less noticeable to human eyes. This adaptive rotation makes it possible to maintain the image integrity with less distortion (Sec. 3.4).

Algorithmically, we optimize a grid mesh with several considerations. We constrain the horizontal/vertical lines (after rotation) to be horizontal/vertial to create the perception of rotation, but relax other lines. We constrain the boundary vertexes on the image boundary to maintain the completeness of the content. We also minimize local distortion to preserve shapes. We formulate these considerations as an energy function, and minimize it by a half-quadratic splitting technique [28, 19]. Our method is very fast ($<$1s for a multi-megapixel image) and produces visually compelling results (*e.g.*, Fig. 1(d)).

Our method performs particularly well for "small" rotation angles like $<$10° . We find this range covers most real cases of correcting careless camera tilt. We also notice that human eyes are very sensitive to a rotation angle as small as 3° or even smaller (the famous *Leaning* Tower of Pisa leans at about 3.99° [1]), so the correction is still desired despite the angles are "small". We believe our solution can meet the need in many practices.

This paper mainly has these contributions: (i) to the best of our knowledge, this is the *first* study on content-aware rotation based on human perception studies; (ii) we develop an optimization-based warping method to create the perception of rotation; (iii) our method is practically fast and high-quality, and can be a useful tool from common users.

2. Related Work

We review the related methods and tools on image rotation, retargeting, and warping.

Image Rotation Consider a popular way of manual rotation [1]: the user drags a straight line aligned to any horizontal/vertical line, and then the software rotates the image so that the dragged line becomes horizontal/vertical. This user interface is inspiring: the human eyes are sensitive to tilted horizontal/vertical lines. Our method harnesses this property to create the perception of rotation.

A method [12] that automatically estimates a rotation angle from a single image is based on a similar motivation. This method finds the vanishing points and determines the horizon accordingly. The rotation angle that rectifies the horizon is chosen. Beyond 2D in-plane rotation, the method in [20] automatically estimates 3D rotation angles (a homography matrix). This method harnesses both horizontal and vertical vanishing lines. Given the estimated rotation angle (2D/3D), conventional methods [12, 20] transform the image rigidly and globally. On the contrary, our purpose is to design locally adaptive transforms. We only consider 2D in-plane rotation in this paper.

[1]en.wikipedia.org/wiki/Leaning_Tower_of_Pisa

Image Retargeting Retargeting refers to the problem of changing image sizes (or most usually, aspect ratios). Uniform scaling may lead to notorious distortion, while cropping may remove desired content. The Seam Carving method [4] removes/inserts the most unnoticeable seams to resize the image. This method may produce visible discontinuities when the number of seams increases because it is greedy. This problem is addressed by a variety of optimization-based warping methods [31, 29, 32, 8]. The warping strategy allows to introduce smooth distortion that are less noticeable to human eyes. Further, the mesh-based warping methods are able to preserve high-level perceptual properties like local shapes [29, 32, 8] and straight lines [8].

Image Warping Warping is a successful solution to many other computer vision/graphics applications besides retargeting. The method in [16] introduces an "as-rigid-as-possible" manipulation of shapes or images. The studies in [18, 7, 6] use the warping strategy to adaptively project wide-angle/panaramic images. Warping has also been adopted in image morphing [9], video stabilization [21], and panorama rectangling [14].

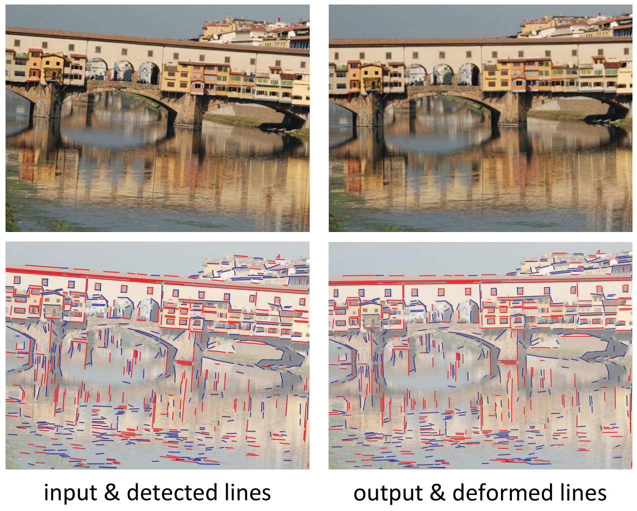

Figure 4. Top: the input and the content-aware rotation result. The rotation angle is 5.2° . Bottom: the detected lines in the input and their deformed counterparts in the output. Here the "canonical" lines are marked as red - they are the lines to be horizontal/vertical after rotation.

3. Algorithm

We suppose the rotation angle Δ is given by users or automatic methods like [12], and is fixed. We further suppose that given this rotation angle, the horizon (or the lines parallel to the horizon) would become horizontal after rigid rotation. This is generally the case in practice. But our method can easily adapt to "tilted composition" if a tilted horizon is the artists' purpose.

We optimize a quad mesh for warping. The warping result is expected to maintain the orientations of horizontal/vertical lines (after rotation) but relax others. To this end, we design an energy function that can manipulate the rotations of lines in different orientations.

3.1. Line Extraction and Quantization

We first extract and quantize the lines that will be used in the mesh optimization. This step is on the input image. We extract lines using the code of [27][2] (Fig. 4 left). Then these lines are cut by the input mesh grid, so each resulting line is within a single quad. Denote the orientation of the x-axis as 0. We compute the orientations of the detected lines. The orientations are offset by periods of π such that they are in the range $[-\Delta, \pi - \Delta)$ in the input image. If the image is rigidly rotated by Δ, these lines would be in the range $[0, \pi)$ in the output.

We uniformly quantize the range $[-\Delta, \pi - \Delta)$ into M=90 bins, each bin covering 2° . As in [8], our energy function will encourage all the lines in the same bin to share a common rotation angle. As such, a long straight line will be kept straight, and parallel lines will be kept parallel. We denote the common rotation angle in the m-th bin as θ_m. The set $\{\theta_m\}_{m=1}^{M}$ forms a $M \times 1$ vector $\boldsymbol{\theta}$.

We pay special attention to the lines that would become horizontal/vertical after rotation: they are in the bins m=1,M (horizontal) or $\frac{M}{2}$,$\frac{M}{2}+1$ (vertical). We refer to these four bins as "canonical" bins. We want to preserve the rotation of the lines in these bins (marked as red in Fig. 4).

3.2. Energy Function

We design an energy function that encourages the following properties. (1) The lines in the canonical bins are strictly constrained to be rotated by Δ. This is for creating the perception of rotation. (2) The vertexes in the input mesh boundary are constrained to be on the upright rectangular boundary of the output. This is for maintaining the content completeness. (3) The local distortion is minimized.

We put a regular quad mesh on the input image (Fig. 2). We denote the position of a vertex as $\mathbf{v}_i = (x_i, y_i)^{\mathrm{T}}$, and concatenate all vertexes $\{\mathbf{v}_i\}$ into a long vector $\mathbf{V}$. Our energy function is with respect to $\mathbf{V}$ and $\boldsymbol{\theta}$. It has the following terms:

Rotation Manipulation The energy E_R manipulates the rotation angles of the lines:

$$E_R(\boldsymbol{\theta}) = \sum_m \delta_m (\theta_m - \Delta)^2 + \sum_m (\theta_m - \theta_{m+1})^2 \quad (1)$$

The first term is a data term. It encourages the rotation angles to follow the desired Δ. The weight $\delta_m = 10^3$ if bin m is canonical, and $\delta_m = 0$ otherwise. In this way we impose

[2]www.ipol.im/pub/art/2012/gjmr-lsd/

strong constraints only on the horizontal/vertical lines. The second term is a smoothness term. It encourages the two neighboring bins to be similar. It smoothly propagates the impact of the canonical bins to the non-canonical ones.

The energy $E_R(\boldsymbol{\theta})$ allows to rotate the non-canonical lines by some angles different from Δ, so enables non-rigid and adaptive rotation. The energy $E_R(\boldsymbol{\theta})$ is quadratic on the vector $\boldsymbol{\theta}$.

Line Preservation The line preservation energy builds a relation between the lines and the mesh vertexes.

For the k-th detected line, we can represent its two endpoints as bilinear interpolations of four vertexes. We compute a directional vector $\mathbf{e}_k$ by the difference of the two endpoints. So $\mathbf{e}_k$ can be written as a linear function of the vertexes $\mathbf{V}$ (that is, $\mathbf{e}_k = P_k\mathbf{V}$ for some P_k). Also, we use $\mathbf{u_k}$ to denote the directional vector of this line in the input image. Denoting the bin of this line as $m(k)$ with the expected rotation angle $\theta_{m(k)}$, we consider the following energy measuring the distortion of line rotation:

$$E_L(\mathbf{V}, \boldsymbol{\theta}, \mathbf{s}) = \frac{1}{K}\sum_k \|s_k R_k \mathbf{u}_k - \mathbf{e}_k\|^2. \tag{2}$$

where K is the number of lines, the matrix $R_k = \begin{bmatrix} \cos\theta_{m(k)} & -\sin\theta_{m(k)} \\ \sin\theta_{m(k)} & \cos\theta_{m(k)} \end{bmatrix}$ is a rotational matrix, and s_k is a scale associated to the line k. Intuitively, we rotate the input vector $\mathbf{u}_k$ by $\theta_{m(k)}$ and scale it by s_k, and then we measure its distortion to $\mathbf{e}_k$.

We assume the scale s_k is independent to each other. Minimizing E_L with respect to each s_k gives us $s_k = (\mathbf{u}_k^{\mathrm{T}}\mathbf{u}_k)^{-1}\mathbf{u}_k^{\mathrm{T}}R_k^{\mathrm{T}}\mathbf{e}_k$. Substituting s_k into (2) we obtain:

$$E_L(\mathbf{V}, \boldsymbol{\theta}) = \frac{1}{K}\sum_k \|\left(R_k U_k R_k^{\mathrm{T}} - I\right)\mathbf{e}_k\|^2. \tag{3}$$

where $U_k = \mathbf{u}_k(\mathbf{u}_k^{\mathrm{T}}\mathbf{u}_k)^{-1}\mathbf{u}_k^{\mathrm{T}}$ and I denotes a unit matrix. Intuitively, this term encourages the angle between $\mathbf{e}_k$ and $\mathbf{u}_k$ to be $\theta_{m(k)}$, such that the line is to be rotated by $\theta_{m(k)}$.

The term $E_L(\mathbf{V}, \boldsymbol{\theta})$ takes effect like the "angle coupling energy" in [8]. But unlike [8], we have decoupled from s and obtained a closed-form on $\mathbf{V}$ and $\boldsymbol{\theta}$. This simpler form allows us to handle the nonlinearity of $\boldsymbol{\theta}$ in optimization (Sec. 3.3). The energy $E_L(\mathbf{V}, \boldsymbol{\theta})$ is quadratic on $\mathbf{V}$.

Shape Preservation Various shape preservation functions have been proposed, *e.g.*, in [16, 32]. In this paper we adopt the one in [32]. This energy favors each quad to undergo a similarity transformation ("as-similar-as-possible"). The shape preservation energy E_S is:

$$E_S(\mathbf{V}) = \frac{1}{N}\sum_q \|(A_q(A_q^{\mathrm{T}}A_q)^{-1}A_q^{\mathrm{T}} - I)\mathbf{V}_q\|^2, \tag{4}$$

where N is the quad number, q is a quad index. Defined on the quad q, the 8×4 matrix A_q and the 8×1 vector $\mathbf{V}_q$ are:

$$A_q = \begin{bmatrix} \hat{x}_{q,0} & -\hat{y}_{q,0} & 1 & 0 \\ \hat{y}_{q,0} & \hat{x}_{q,0} & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ \hat{x}_{q,3} & -\hat{y}_{q,3} & 1 & 0 \\ \hat{y}_{q,3} & \hat{x}_{q,3} & 0 & 1 \end{bmatrix}, \quad \mathbf{V}_q = \begin{bmatrix} x_{q,0} \\ y_{q,0} \\ \vdots \\ x_{q,3} \\ y_{q,3} \end{bmatrix}. \tag{5}$$

Here $(x_{q,0}, y_{q,0})$, ..., $(x_{q,3}, y_{q,3})$ denote the four vertexes of a deformed quad, and $(\hat{x}_{q,0}, \hat{y}_{q,0})$, ..., $(\hat{x}_{q,3}, \hat{y}_{q,3})$ those of the input quad (see [32] for the derivation). The energy E_S is a quadratic function of $\mathbf{V}$.

It is possible to weight the terms in (4) to preserve faces [7] or other salient objects. In this paper we ignore these concerns and keep the presentation of our core idea simple.

Boundary Preservation To avoid the image content going outside an upright rectangular boundary, we constrain the boundary vertexes on this rectangle:

$$E_B(\mathbf{V}) = \sum_{i\in\text{left}} x_i^2 + \sum_{i\in\text{right}} (x_i - w)^2 + \sum_{i\in\text{top}} y_i^2 + \sum_{i\in\text{bottom}} (y_i - h)^2. \tag{6}$$

Here each summation is over the vertexes on each boundary. The values w and h are the width and height of an upright rectangle. We use a rectangle of the same size as the input image. The energy $E_B(\mathbf{V})$ is quadratic on $\mathbf{V}$.

Total Energy We optimize the following energy:

$$E(\mathbf{V}, \boldsymbol{\theta}) = E_S(\mathbf{V}) + \lambda_B E_B(\mathbf{V}) + \lambda_L E_L(\mathbf{V}, \boldsymbol{\theta}) + \lambda_R E_R(\boldsymbol{\theta}). \tag{7}$$

We use $\lambda_B = \infty$ (10^8) to impose hard boundary constraint. For λ_L and λ_R, our method involves the issue of parameter settings as in other multi-term warping methods [7, 6, 8]. We find $\lambda_L = \lambda_R = 100$ work well in various experiments. Throughout this paper we use this fixed setting.

3.3. Optimization

We use an alternating algorithm to optimize the energy $E(\mathbf{V}, \boldsymbol{\theta})$. We divide the problem into two subproblems, and iteratively optimize each of them:

Fix $\boldsymbol{\theta}$, solve for $\mathbf{V}$. In this case E is a quadratic function on $\mathbf{V}$. The solution is simply given by a sparse linear system. Because $\mathbf{V}$ consists of several hundreds of unknowns, it takes only a few milliseconds to solve this subproblem.

Fix $\mathbf{V}$, solve for $\boldsymbol{\theta}$. In this case the subproblem is to minimize:

$$\min_{\boldsymbol{\theta}} \quad \frac{\lambda_L}{K}\sum_k \|\left(R_k U_k R_k^{\mathrm{T}} - I\right)\mathbf{e}_k\|^2 + \lambda_R E_R(\boldsymbol{\theta}). \tag{8}$$

This is a nontrivial problem due to the nonlinearity in the first term. We adopt a *half-quadratic splitting* technique

[28, 19] to solve this problem. We introduce a series of auxiliary variables $\boldsymbol{\phi} = \{\phi_k\}_{k=1}^{K}$. Intuitively, each ϕ_k denotes the *individual* rotation angle of line k (while θ_m is the *common* rotation angle of the lines in bin m). Denote $\widehat{R}_k = \begin{bmatrix} \cos\phi_k & -\sin\phi_k \\ \sin\phi_k & \cos\phi_k \end{bmatrix}$ as the *individual* rotation matrix of line k. We rewrite the problem in (8) as:

$$\min_{\boldsymbol{\theta},\boldsymbol{\phi}} \quad \frac{\lambda_L}{K}\sum_k \| \left(\widehat{R}_k U_k \widehat{R}_k^{\mathrm{T}} - I\right) \mathbf{e}_k \|^2 \\ +\beta \sum_k (\phi_k - \theta_{m(k)})^2 + \lambda_R E_R(\boldsymbol{\theta}). \tag{9}$$

Here β is a penalty weight, and when $\beta \to \infty$ the solution to (9) converges to (8). The half-quadratic splitting technique warms up from a small β and gradually increases it to ∞. In each step when β has been fixed, the problem in (9) is split into two subproblems:

(i) Fix $\boldsymbol{\phi}$, update $\boldsymbol{\theta}$. This subproblem is

$$\min_{\boldsymbol{\theta}} \ \beta \sum_k (\phi_k - \theta_{m(k)})^2 + \lambda_R E_R(\boldsymbol{\theta}). \tag{10}$$

This is a quadratic function on $\boldsymbol{\theta}$ and is solved by a linear system. This step is fast because $\boldsymbol{\theta}$ has $M = 90$ unknowns.

(ii) Fix $\boldsymbol{\theta}$, update $\boldsymbol{\phi}$. This problem is nonlinear, but the ϕ_k is independent of each other and can be solved separately. The subproblem involving ϕ_k is:

$$\min_{\phi_k} \ \frac{\lambda_L}{K} \| \left(\widehat{R}_k U_k \widehat{R}_k^{\mathrm{T}} - I\right) \mathbf{e}_k \|^2 + \beta(\phi_k - \theta_{m(k)})^2. \tag{11}$$

This is a single variable problem. It is possible to solve it using a gradient descent method. But we adopt a simpler look-up method inspired by [19]. Notice that if $\beta = 0$, the problem is solved by the angle between $\mathbf{e}_k$ and $\mathbf{u}_k$: $\phi_k = \angle(\mathbf{e}_k, \mathbf{u}_k)$ (this is the intuition of line preservation (3)); and if $\beta = \infty$, it is solved by $\phi_k = \theta_{m(k)}$. Intuitively, the problem in (11) trades between these two values. So we uniformly divide the range $[\angle(\mathbf{e}_k, \mathbf{u}_k), \theta_{m(k)}]$ into 100 discrete values, evaluate the cost in (11), and choose ϕ_k as the one that minimizes the cost. This is a very straightforward solution with clear intuitions.

Algorithm Summary. Our overall optimization method is in Algorithm 1. The angles $\boldsymbol{\theta}$ are all initialized as Δ (the given rotation angle). There is no need to initialize $\mathbf{V}$ because we start from updating $\mathbf{V}$. The penalty weight β warms up from $\beta_0 = 1$, and stops at $\beta_{\text{max}} = 10^4$ with the increment $\beta_{\text{inc}} = 10$. The number of the outer iteration is fixed at $\text{iter}_{\text{max}} = 10$, which is empirically sufficient for producing good visual quality.

We use a mesh with around 400 square quads in this paper. Because the number of unknown is small (several hundreds), the optimization takes about 0.1-0.2 seconds. After optimizing the mesh, we deform the image using bilinear interpolation to produce the final result. The interpolation is the main timing bottleneck, depending on image sizes. Our current implementation takes about 0.5s to interpolate one 8-megapixel image. Our algorithm is implemented in C++ and run on a PC with an Intel Core i7 3.0GHz CPU and 8GB memory.

Algorithm 1 Content-Aware Rotation: Optimization

```
1:  Initialize $\boldsymbol{\theta}$.
2:  for iter $= 1$ to iter$_{\text{max}}$ do
3:      Fix $\boldsymbol{\theta}$, solve for $\mathbf{V}$ due to (7).
4:      Fix $\mathbf{V}$. Set $\beta = \beta_0$.
5:      while $\beta < \beta_{\text{max}}$ do
6:          Fix $\boldsymbol{\theta}$, solve for $\phi_k$ in (11) for all $k$.
7:          Fix $\boldsymbol{\phi}$, solve for $\boldsymbol{\theta}$ due to (10).
8:          Set $\beta = \beta \cdot \beta_{\text{inc}}$
9:      end while
10: end for
```

3.4. Discussions

Adaptive Rotation Algorithmically, our method is designed to strictly preserve the horizontal/vertical lines (after rotation) while relaxing others. In Fig. 5 we show an example illustrating this mechanism. In the input image Fig. 5(a), the green pencil is tilted +10.5°. This pencil will be horizontal if the image is rigidly rotated by $\Delta = +10.5°$. Our content-aware method maintains such rotation of this pencil, as in Fig. 5(b). But our method relaxes the rotation of other pixels (*i.e.*, lines in other orientations). In this interesting example, our method behaves like rearranging the pencils but not strictly rotating all of them.

This relaxation gives us more room to trade off other factors like shape preservation. It can be expected that if lines of all orientations are treated strictly, the result will be distorted more. To test this effect, we modify the data term weight δ_m in Eqn.(1) and set them as equal (10^3) for all the bins. Unlike our original way that adapts to the orientations, this modified way is non-adaptive and forces lines of all orientations to undergo strict rotation. Fig. 6(b)(c) shows the comparisons of these two ways. Fig. 6 (bottom) shows the rotation angle θ_m of each bin m. We see the non-adaptive way respects all rotation, but at the price of larger distortion (Fig. 6(c)). On the contrary, our adaptive way produces a visually more pleasing result (Fig. 6(b)). Fig. 6 (bottom) also reveals that our adaptive method constrains only the canonical bins.

Note the "horizontal/vertical" lines are w.r.t. the 2D image coordinate system (not the 3D world coordinate system). This is because in this paper we only model the perceptual sensitivities in the 2D coordinate system. We leave the case of 3D coordinate systems for future studies.

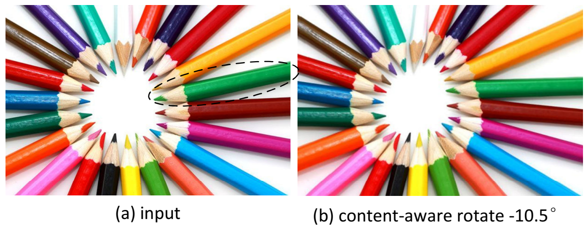

Figure 5. (a) The input image. The green pencil is +10.5° tilted. (b) Content-aware rotation (-10.5°). The green pencil is kept horizontal, but the others are relaxed. The input is from [24].

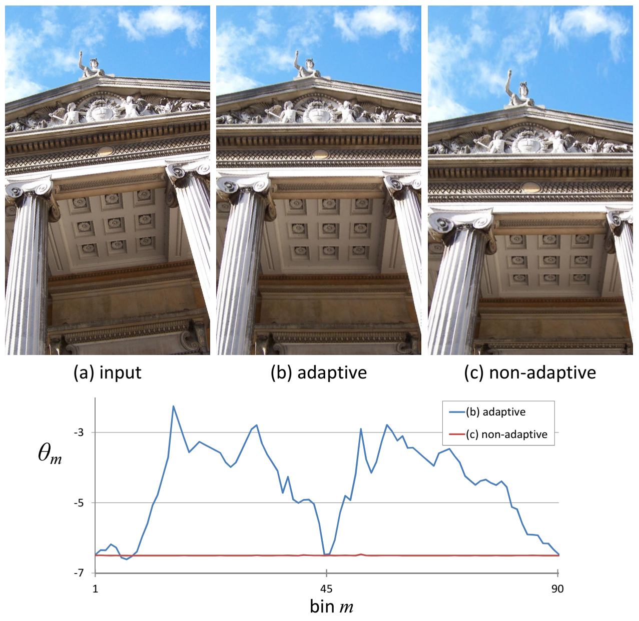

Figure 6. Content-aware rotation using adaptive and non-adaptive angles. The desired rotation angle is $\Delta = -6.5^\circ$. (a) Input. (b) Result using adaptive angles. (c) Result using non-adaptive angles. Bottom: the optimized rotation angle θ_m for each each bin m.

About Retargeting Our method is related to a line-preserving retargeting method [8] and a recent panorama warping method [14]. But our method has major differences. Unlike [8, 14] that only *preserve* the straightness or orientations of lines, our method *manipulates* the rotational angles $\boldsymbol{\theta}$ of the lines - this is the focus of image rotation. Our method is capable of discriminatively handling rotational directions (horizontal/vertical *vs.* others), while [8, 14] are not. Our energy function also leads to a new and more challenging optimization problem.

Without angle manipulations, existing retargeting methods including [8, 14] have no effect when applied for content-aware rotation. In image rotation, the aspect ratio of the output image is unchanged or almost unchanged[3]. If we apply the existing methods, we can only obtain trivial results that are identical or almost identical to the input.

[3]In "rigid rotation + cropping", the output aspect ratio depends on the cropping strategy. One could find a crop that preserves the input aspect ratio, or a largest inner crop that may change the aspect ratio.

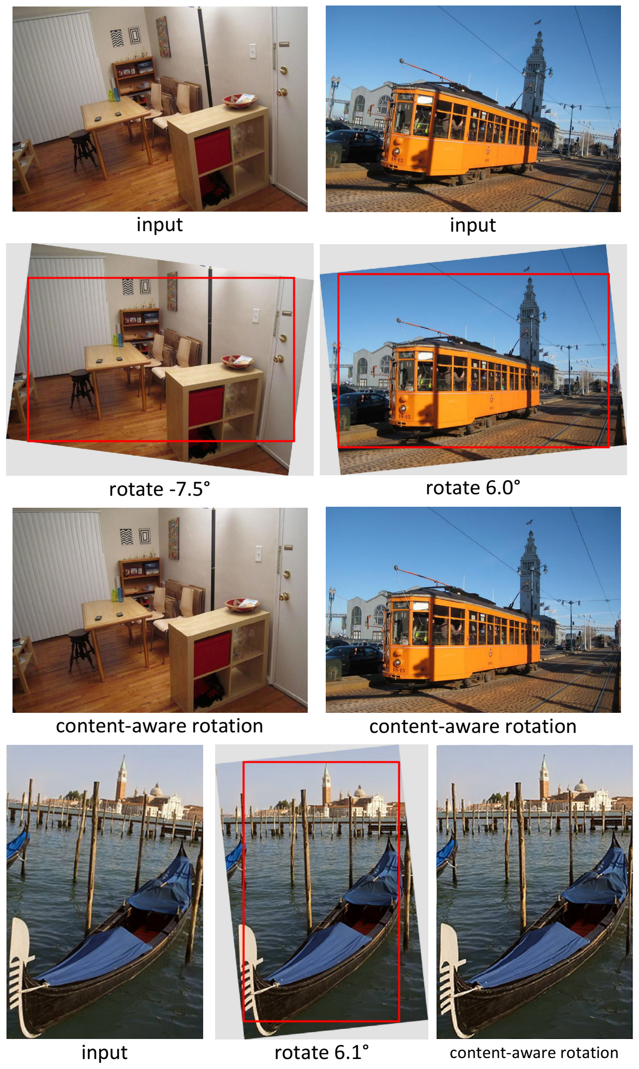

Figure 7. Rigid rotation + cropping vs. Content-aware rotation.

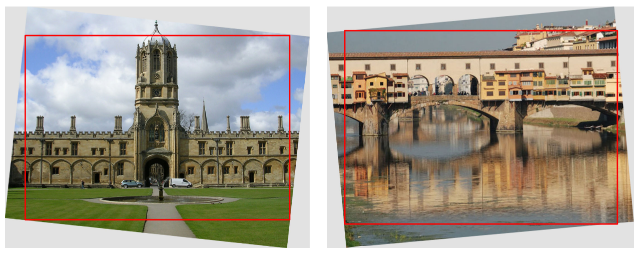

Figure 8. The cropping results of Fig. 2 and 4.

4. Experiments

We assume the rotation angles are given by the users. There is no other user interaction. All parameters are fixed.

Comparisons with Rigid Rotation + Cropping In Fig. 1

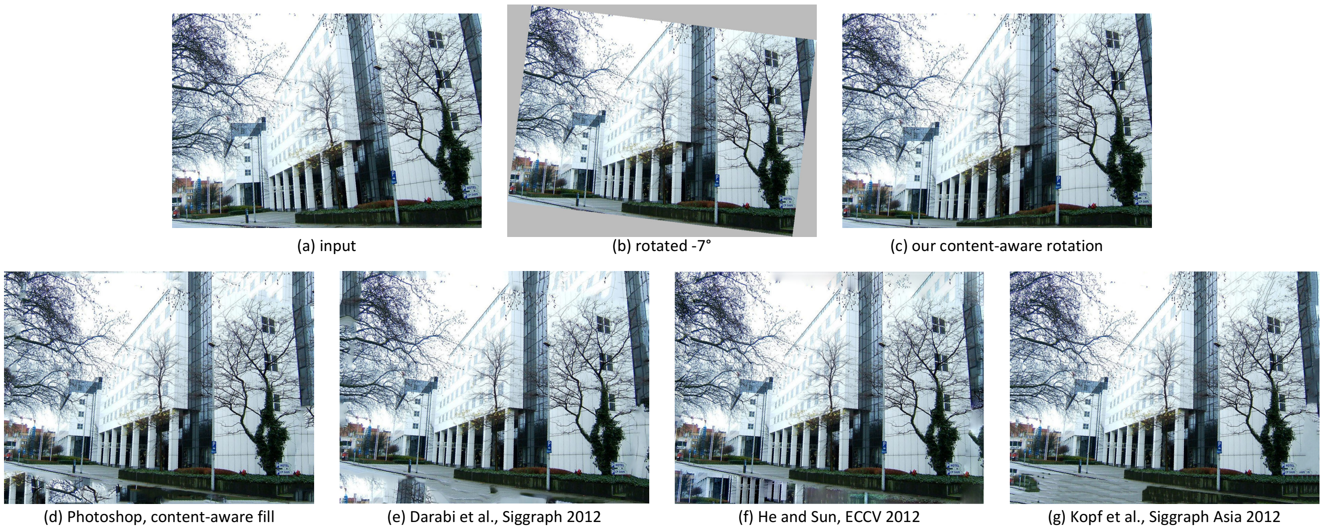
(a) input
(b) rotated -7°
(c) our content-aware rotation
(d) Photoshop, content-aware fill
(e) Darabi et al., Siggraph 2012
(f) He and Sun, ECCV 2012
(g) Kopf et al., Siggraph Asia 2012

Figure 9. Comparisons with state-of-the-art image completion techniques. (a) Input. (b) Rotated by -7° . (c) Our result. (d) Content-aware filling in Photoshop [3]. (e) Darabi *et al.*'s [10]. (f) He and Sun's [15]. (g) Kopf *et al.*'s [17].

and Fig. 7 we show comparisons with rigid rotation followed by cropping. We also show the cropping results of Fig. 2 and 4 in Fig. 8. Our content-aware rotation method creates perception similar to the results of rigid rotation, but maintains the integrity of the image content.

Comparisons with Image Completion Another possible solution is to fill the missing region (after rigid rotation) using image completion techniques, though they are not originally designed for this purpose. We compare with several recent state-of-the-art techniques in Fig. 9: Content-aware fill in Photoshop [3] (partially based on [30, 5]), Darabi *et al.*'s [10], He and Sun's [15], and Kopf *et al.*'s [17]. All these image completion methods are exemplar-based. They may produce poor results when the required examples are missing in the images. They also have limitations in synthesizing semantic content. Our warping solution does not suffer from these problems. Please see the supplementary materials for more comparisons.

From the viewpoint of users, image completion and our solution are not competing. Users need a new option when completion fails. We expect our novel solution would help the users to better solve their practical problems.

Generalizations In some special cases, the artist may deliberately tilt the images, known as "tilted composition" in photography [26]. Our content-aware rotation method can be easily generalized to this application, thanks to its flexibility on the angle manipulation. For simplicity we suppose the horizons are level in the input images (Fig. 10(a)), but other cases are compatible. Unlike in Sec. 3.1 where we considered "canonical" bins as those of *output* horizontal/vertical lines, here we consider them as those of *input* horizontal/vertical lines. In this way, the input horizons will be strictly rotated and others are relaxed. Fig. 10(c) shows two examples.

Limitations Like retargeting methods, our content-aware rotation method attempts to find visually unnoticeable operations. But this task becomes challenging when there is little room for such operations. This can be the case when the image content is visually important in many local regions, or the rotation angle is large. In this case our method may produce noticeable distortion, as in Fig. 11.

5. Conclusion

We have presented the perceptually motivated Content-Aware Rotation. We believe it is interesting and useful to harness the human vision properties for developing computer vision/graphics techniques.

Technically, we have introduced a warping method that can directly and flexibly manipulate the rotation angles. This formulation can possibly be used in other warping-based applications and improve the quality.

References

[1] Straighten an image. `helpx.adobe.com/photoshop/using/adjusting-crop-rotation-canvas.html`.

[2] `www.adobe.com/products/photoshop/content-aware-scaling.html`.

[3] `www.adobe.com/technology/projects/content-aware-fill.html`.

[4] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. In *SIGGRAPH*, 2007.

[5] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. In *SIGGRAPH*, 2009.

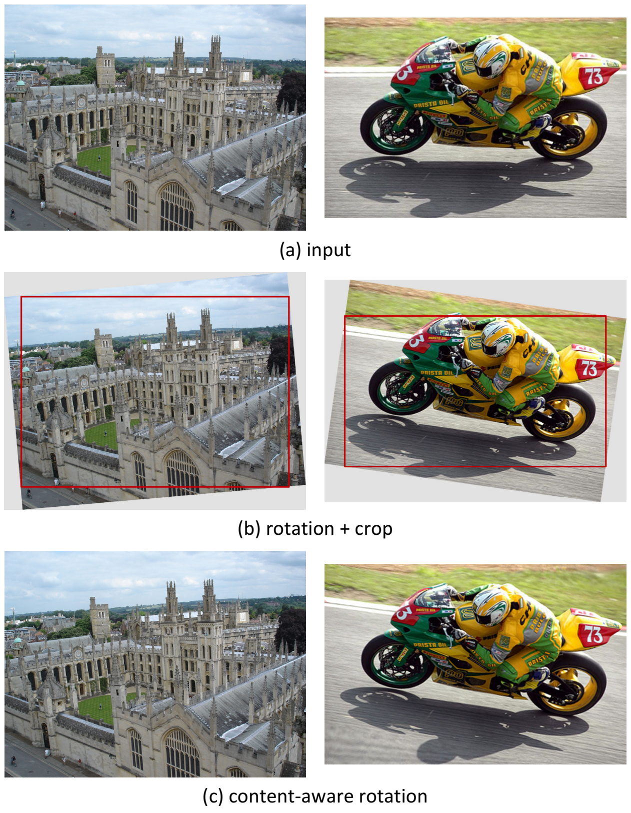

Figure 10. Deliberate tilting. The artists may tilt the upright images on purpose for a visual novelty. Our content-aware method can easily adapt to these purposes. The rotation angles are +5° (left) and -8° (right).

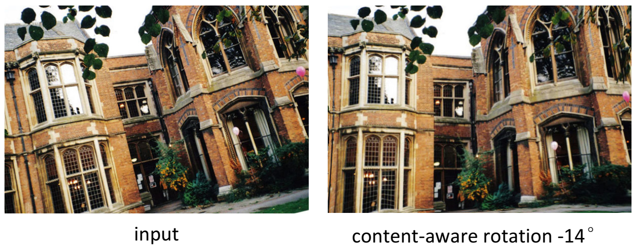

Figure 11. A failure case of our method.

[6] R. Carroll, A. Agarwala, and M. Agrawala. Image warps for artistic perspective manipulation. In *SIGGRAPH*, pages 127:1–127:9, 2010.

[7] R. Carroll, M. Agrawal, and A. Agarwala. Optimizing content-preserving projections for wide-angle images. In *SIGGRAPH*, pages 43:1–43:9, 2009.

[8] C.-H. Chang and Y.-Y. Chuang. A line-structure-preserving approach to image resizing. In *CVPR*, 2012.

[9] S. E. Chen and L. Williams. View interpolation for image synthesis. In *SIGGRAPH*, 1993.

[10] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen. Image melding: combining inconsistent images using patch-based synthesis. In *SIGGRAPH*, 2012.

[11] M. Freeman. *The Photographer's Eye: Composition and Design for Better Digital Photos*, chapter 1. The Image Frame. Focal Press, 2007.

[12] A. C. Gallagher. Using vanishing points to correct camera rotation in images. In *Computer and Robot Vision*, pages 460–467, 2005.

[13] E. Goldmeier. Similarity in visually perceived forms. *Psychological Issues*, 1972.

[14] K. He, H. Chang, and J. Sun. Rectangling panoramic images via warping. In *SIGGRAPH*, 2013.

[15] K. He and J. Sun. Statistics of patch offsets for image completion. In *ECCV*, pages 16–29, 2012.

[16] T. Igarashi, T. Moscovich, and J. F. Hughes. As-rigid-as-possible shape manipulation. In *SIGGRAPH*, 2005.

[17] J. Kopf, W. Kienzle, S. Drucker, and S. B. Kang. Quality prediction for image completion. In *SIGGRAPH Asia*, 2012.

[18] J. Kopf, D. Lischinski, O. Deussen, D. Cohen-Or, and M. Cohen. Locally adapted projections to reduce panorama distortions. In *Computer Graphics Forum*, pages 1083–1089. Wiley Online Library, 2009.

[19] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-Laplacian priors. In *NIPS*, 2009.

[20] H. Lee, E. Shechtman, J. Wang, and S. Lee. Automatic upright adjustment of photographs. In *CVPR*, 2012.

[21] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3d video stabilization. In *SIGGRAPH*, 2009.

[22] S. Nundy, B. Lotto, D. Coppola, A. Shimpi, and D. Purves. Why are angles misperceived? *Proceedings of the National Academy of Sciences*, 2000.

[23] I. Rock. *Indirect perception*, chapter 11. The Right Angle. The MIT Press, 1997.

[24] M. Rubinstein, D. Gutierrez, O. Sorkine, and A. Shamir. A comparative study of image retargeting. In *SIGGRAPH Asia*, page 160, 2010.

[25] M. Rubinstein, A. Shamir, and S. Avidan. Improved seam carving for video retargeting. In *SIGGRAPH*, 2008.

[26] J. Suler. *Photographic Psychology: Image and Psyche*. 2013.

[27] R. Von Gioi, J. Jakubowicz, J. Morel, and G. Randall. Lsd: A fast line segment detector with a false detection control. *TPAMI*, pages 722–732, 2010.

[28] Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 2008.

[29] Y.-S. Wang, C.-L. Tai, O. Sorkine, and T.-Y. Lee. Optimized scale-and-stretch for image resizing. In *SIGGRAPH Asia*, pages 118:1–118:8, 2008.

[30] Y. Wexler, E. Shechtman, and M. Irani. Space-time completion of video. *TPAMI*, pages 463–476, 2007.

[31] L. Wolf, M. Guttmann, and D. Cohen-Or. Non-homogeneous content-driven video-retargeting. In *ICCV*, 2007.

[32] G. Zhang, M. Cheng, S. Hu, and R. Martin. A shape-preserving approach to image resizing. In *Computer Graphics Forum*, pages 1897–1906. Wiley Online Library, 2009.