Content-Aware Image Resizing by Quadratic Programming

Daniel Freedman
Hewlett-Packard Labs
Haifa, Israel
daniel.freedman@hp.com

Renjie Chen
Math Dept, Zhejiang University
Hangzhou, China
renjie.c@gmail.com

Zachi Karni
Hewlett-Packard Labs
Haifa, Israel
zachi.karni@hp.com

Craig Gotsman
Computer Science Dept, Technion
Haifa, Israel
gotsman@cs.technion.ac.il

Ligang Liu
Math Dept, Zhejiang University
Hangzhou, China
ligangliu@zju.edu.cn

Abstract

We present a new method for content-aware image resizing based on a framework of global optimization. We show that the basic resizing problem can be formulated as a convex quadratic program. Furthermore, we demonstrate how the basic framework may be extended to prevent foldovers of the underlying mesh; encourage the magnification of salient regions; and preserve straight line structures. We show results demonstrating the effectiveness of the proposed method by comparing with four leading competitor methods.

1. Introduction

In many applications, it is desirable to change the dimensions of an image or video, sometimes altering the aspect ratio of the content in the process. This is already common in photographic printing on different paper sizes, and in the conversion of video content from one format to another, such as standard 4:3 to HD 16:9. It will become even more common as improvements in printing technologies encourage customized printing, which in turn will require the embedding of given image content into different template sizes, on demand.

The simplest solution to the problem of changing the dimensions of an image is uniform resizing, which stretches all parts of the image equally along each dimension. More sophisticated algorithms for converting 4:3 format to 16:9 format stretch the image non-uniformly: the stretch is minimal in the center of the image, where it is more noticeable by the viewer, and gradually increases towards the periphery.

However, it may be desirable to resize different parts of the image differently, depending on the image content. For example, where there is interesting detail, we may not wish to deform the image too much; namely, we will desire a close to *isotropic* scaling. However, where there is not much in the way of interesting detail, we will not mind stretching. Indeed, the stretching in these regions may be even greater than that implied by the original user constraint. This idea is typically referred to as *content-aware image resizing*, and was first introduced in the Seam Carving work of Avidan and Shamir [1].

Karni et al. [6] presented a technique for content-aware image resizing which performed well compared to existing methods, including Seam Carving [1, 7, 9, 8] and Optimized Scale and Stretch [12]. This technique proposed an energy function, which was then minimized by an iterative technique which was guaranteed to reach a local minimum. While the technique possessed many advantages, it also had a key disadvantage: it did not compute the global minimum of its energy function. This issue manifested itself in two ways. First, there is the standard problem with local optimization, namely that the local optimum may be far from the global optimum; in practice, there are cases when the local minimum reached by the algorithm in [6] does not lead to ideal results. Second, there are extra terms which one may wish to add to the energy function to encourage certain desirable behavior in the algorithm; these include terms to prevent the formation of foldovers, and to magnify important regions (both of which will be described in greater depth shortly). Within a local optimization framework, the addition of such terms often serves to confuse the algorithm, producing local optima which are even further from the global optimum.

In this paper, we present a new method for content-aware image resizing, with four main contributions:

1. **Global Optimization:** We describe an energy function for resizing which can be globally optimized, as it is based on a convex programming – in particular, quadratic programming – formulation.

2. **Foldover Prevention:** Foldovers occur when the underlying mesh representing image blocks is not properly embedded in the plane. Several methods, such as [12, 6], try to discourage foldovers, but to our knowledge, our method is the first to *guarantee* no foldovers.

3. **Magnification of Important Regions:** Content-aware image resizing typically requires that the aspect ratio of important regions is mostly preserved; however, this

does not guarantee that the important regions occupy a significant portion of the final image. Indeed, they may be quite small. Since, in many cases, the essence of content-aware resizing is emphasizing the most important content in the image, we incorporate extra energy terms which encourage the magnification of important regions.

4. **Straight Line Preservation:** (a) Our method encourages the grid lines (horizontal and vertical) to remain fairly straight. (b) We provide a semi-automatic tool for the preservation of arbitrary straight line structures in the image, such as an umbrella handle.

1.1. Related Work

The problem of content-aware image resizing was first addressed in the Seam Carving work of Avidan and Shamir [1]. The idea in this work in to remove "seams," i.e. curved columns or rows of pixels, from the image such that the least important seams are removed first. This paradigm was later extended in a number of ways [7, 9, 8, 2], including for video. Other techniques which focus on video content are those of Wolf et al. [13] and Wang et al. [11].

Closer in spirit to the technique presented here is the Optimized Scale and Stretch method of Wang et al. [12], as well as the Local-Global algorithm of Karni et al. [6]. Both of these works pose the problem of resizing as a continuous image deformation problem, rather than the discrete problem that characterizes Seam Carving. However, both of these methods minimize the relevant energy by iterative methods, arriving at a local minimum in both cases. The current method, by contrast, emphasizes a global optimization framework.

We will endeavor to relate the more relevant prior art to the proposed method in the body of the paper, particularly in the sections dealing with experimental results.

2. The Quadratic Programming Formulation

In this section, we present the main algorithmic contribution of the paper: casting the content-aware image resizing problem as the solution to a quadratic program. The main advantage of this approach is that the global optimum of the corresponding energy function can be found. We begin by fixing notation. We then move on to the definition of the resizing energy function, as well as the formulation the relevant content-aware constraints. Finally, we show that the resulting constrained optimization problem is a convex quadratic program.

2.1. Problem Setup

Given an image, the bottom right corner is moved from its original position to a desired location, while the top left corner is fixed. This essentially rescales the x- and y-axes, which changes the image aspect ratio. We assume that the original image is partitioned into rectangular blocks by horizontal and vertical grid lines, forming a rectangular mesh. Our goal is then to compute a new quadrangular mesh for the resized image. The new quadrangles should be chosen such that aspect ratios of the most important rectangles are essentially preserved, that is they are stretched by the almost the same amount in both the x- and y-directions; whereas unimportant rectangles can be stretched and deformed to a greater extent. Importance will be measured by *image saliency*, which we discuss at greater length in the next section.

We denote the original coordinates of the mesh by $x^k_{i,j}$. The indices (i, j) indicate the position in the grid – i is a discretized x-coordinate which runs from 1 to m and j is a discretized y-coordinate which runs from 1 to n; the index $k = 1, 2$ denotes the x- and y-coordinates, respectively. Similarly, we denote the coordinates of the resized quadrangular mesh by $f^k_{i,j}$. The quads themselves are labeled by the index of their upper left corner, that is by the indices (i, j). Finally, the stretch factors of the image in the x- and y-directions are denoted s^1 and s^2, respectively.

The goal is thus to compute the new quadrangular mesh, $f^k_{i,j}$, by formulating and solving a suitable optimization problem.

2.2. The Energy Function

Recall that we would like to place constraints, for each rectangle, on the amount of relative stretching in each of the x- and y-directions. We begin, therefore, by quantifying the amounts of stretch of the (i, j)-th rectangle in the x- and y-directions, through the introduction of the new variables $a^k_{i,j}$, which we refer to as the *stretch variables*. Specifically, we would like the transformation of each rectangle to be as close as possible to an axis-aligned affine transformation, which would leave the new quad as close to a rectangle as possible. This is captured by the following energy or cost term:

$$\begin{aligned} E^k_{i,j} &= \left[(f^k_{i+1,j} - f^k_{i,j}) - a^k_{i,j}(x^k_{i+1,j} - x^k_{i,j})\right]^2 \\ &+ \left[(f^k_{i+1,j+1} - f^k_{i+1,j}) - a^k_{i,j}(x^k_{i+1,j+1} - x^k_{i+1,j})\right]^2 \\ &+ \left[(f^k_{i,j+1} - f^k_{i+1,j+1}) - a^k_{i,j}(x^k_{i,j+1} - x^k_{i+1,j+1})\right]^2 \\ &+ \left[(f^k_{i,j} - f^k_{i,j+1}) - a^k_{i,j}(x^k_{i,j} - x^k_{i,j+1})\right]^2 \\ &\equiv E^{k,1}_{i,j} + E^{k,2}_{i,j} + E^{k,3}_{i,j} + E^{k,4}_{i,j} \end{aligned} \tag{1}$$

The per-quad energy $E^k_{i,j}$ ensures that for each of the 4 edges of the quad (i, j), the edge in the new mesh is close to a stretch of the corresponding edge in the old mesh by a factor of $a^k_{i,j}$. (In fact, $E^k_{i,j}$ looks at the k^{th} coordinate, so there are two such energy terms, one for x and one for y.)

The total energy is then the sum of all the per-quadrangle energies:

$$E(f, a) = \sum_{i=1}^{m-1} \sum_{j=1}^{n-1} \sum_{k=1}^{2} \sum_{l=1}^{4} E^{k,l}_{i,j} \tag{2}$$

where f is shorthand for the entire collection of variables $\{f^k_{i,j}\}$, and likewise for a.

2.3. The Constraints

We now introduce three types of constraints.

Boundary Constraints These constraints ensure that the coordinates of points on the rectangular boundary of the original image remain on the rectangular boundary of the new image. There are 4 groups of such constraints, corresponding to each side of the rectangular boundary; for example, for the left side we have that $f^1_{1,j} = 0$ for all j. The other 3 sets of boundary constraints are similar.[1]

Saliency Constraints The second set of constraints are more interesting, and encode the key idea in content-aware image resizing: that more important quads should better preserve their aspect ratio, and less important quads can be more severely stretched or deformed. Suppose that for each quad (i, j), we have a scalar measure of its importance or saliency, given by $\lambda_{i,j}$. Practically, the saliency measure can be as simple as a measure of the magntidue of the image gradient, or it can be a more complex measure, such as that found in the recent work on saliency, e.g. [4]. It may even be provided in semi-automatic fashion by the user. For the moment, we will take the saliency as given.

Now, assume, without loss of generality, that the user chooses to stretch the x-axis more than the y-axis; the reverse situation can be treated analogously, by switching the roles of the variables. The stretch per quad, which may be computed as $a^1_{i,j}/a^2_{i,j}$, will ideally be close to 1 if a quad is very important. We thus impose an upper bound $\psi_{i,j}$ on the amount of stretch we allow per quad, which varies *inversely* with the importance/saliency $\lambda_{i,j}$. If the quad is very important, namely $\lambda_{i,j}$ is large, then $\psi_{i,j}$ will be just a little above 1; whereas if the quad is unimportant so that $\lambda_{i,j}$ is small, then $\psi_{i,j}$ will be large. Thus, the constraints are

$$1 \le a^1_{i,j}/a^2_{i,j} \le \psi_{i,j}$$

In practice, we take $\psi_{i,j}$ to be the following piecewise linear function of $\lambda_{i,j}$:

$$\psi_{i,j} = \begin{cases} \psi_{max} & \text{if } \lambda_{i,j} \le \lambda_1 \\ \left(\frac{\lambda_2 - \lambda_{i,j}}{\lambda_2 - \lambda_1}\right)\psi_{max} + \frac{\lambda_{i,j} - \lambda_1}{\lambda_2 - \lambda_1} & \text{if } \lambda_1 \le \lambda_{i,j} \le \lambda_2 \\ 1 & \text{if } \lambda_{i,j} \ge \lambda_2 \end{cases}$$

which involves the choice of three parameters, namely ψ_{max}, λ_1, and λ_2.

Non-negativity Constraints The third set of constraints are simple non-negativity constraints on the stretch variables: $a^k_{i,j} \ge 0$.

[1]Recall that the index (i, j) is essentially a discretized (x, y), not a matrix index.

2.4. The QP Structure of the Problem Elucidated

First, begin by rewriting the energy function from Equations (1) and (2). For convenience, let us stack the variables we need to optimize over, that is the $\{f^k_{i,j}\}$ and the $\{a^k_{i,j}\}$ in a big[2] vector, denoted by z. Then each term $E^{k,l}_{i,j}$ can be written as

$$E^{k,l}_{i,j} = (c^{k,l}_{i,j} z)^2$$

where $c^{k,l}_{i,j}$ is a sparse (only three non-zero entries) row vector having the same length as the column vector z. Let us then stack all of the row vectors $c^{k,l}_{i,j}$ on top of each other, to get a matrix C. Then the energy can be simply written as

$$E = \|Cz\|^2$$

Note that z is a column vector of length $2mn+2(m-1)(n-1)$, and C is matrix of dimensions $8(m-1)(n-1) \times 2mn+2(m-1)(n-1)$.

Now, let us turn to the constraints. The boundary constraints are of the form $f^1_{1,j} = 0$ for all j for the left side of the image, with similar expressions for the other three sides. These are *linear* equality constraints. The content-based constraints are of the form $1 \le a^1_{i,j}/a^2_{i,j} \le \psi_{i,j}$, which can be rewritten as two linear inequality constraints:

$$a^1_{i,j} \ge a^2_{i,j} \quad \text{and} \quad a^1_{i,j} \le \psi_{i,j} a^2_{i,j}$$

Of course, the non-negativity constraints on the stretch variables, $a^k_{i,j} \ge 0$, are also linear. We can thus write all of the constraints together[3] as the matrix-vector expression $\tilde{H}z \ge h$. The problem is then compactly rewritten as

$$\min_z \|Cz\|^2 \quad \text{subject to} \quad \tilde{H}z \ge h$$

This is a standard quadratic program (QP).

3. Extensions

3.1. Prevention of Foldovers

Up until now, we have not guaranteed that the quadrangular mesh we compute is embeddable in the plane. In fact, this is a central difficulty with methods that embed mesh graphs, either triangular or quadrangular: it is often a challenge to guarantee that the mesh will be properly embedded in the plane, without folding over on itself. Examples of the problems caused by foldovers are given in [6].

Within our framework, however, it turns out to be relatively straightforward to explictly prevent foldovers. In particular, what is needed is extra constraints, of the form

$$f^1_{i+1,j} - f^1_{i,j} \ge 0, \quad i = 1, \ldots, m-1;\ j = 1, \ldots n$$
$$f^2_{i,j+1} - f^2_{i,j} \ge 0, \quad i = 1, \ldots, m;\ j = 1, \ldots n-1$$

[2]Recall that for $\{f^k_{i,j}\}$, the indices range as follows: $i = 1, \ldots, m; j = 1, \ldots, n; k = 1, 2$. For $\{a^k_{i,j}\}$, the indices range as follows: $i = 1, \ldots, m-1; j = 1, \ldots, n-1; k = 1, 2$.

[3]Note that the equality constraints are written as inequality constraints using the usual trick: $Az = b$ becomes $Az \ge b$ and $-Az \ge -b$.

(a) (b) (c) (d)

Figure 1. Scaling by a factor of two, with different w_{size} values. (a) Original image. (b) $w_{size} = 0$. (c) $w_{size} = 10$. (d) $w_{size} = 100$.

The first constraint ensures that we do not have vertical foldovers, and the second constraint prevents horizontal foldovers.

In the above set of constraints, it is theoretically possible for an entire column or row to collapse: it will not fold over onto itself, but its width will become 0. If we wish to avoid this situation, we can be more conservative, and require that quads have a minimum positive dimension:

$$f^1_{i+1,j} - f^1_{i,j} \geq \delta^1, \quad i = 1, \ldots, m-1; \; j = 1, \ldots n$$
$$f^2_{i,j+1} - f^2_{i,j} \geq \delta^2, \quad i = 1, \ldots, m; \; j = 1, \ldots n-1$$

Adding these constraints to the set of constraints already given, i.e. $\tilde{H}z \geq h$, gives a larger set $Hz \geq h$. Now, our quadratic program becomes

$$\min_z \|Cz\|^2 \quad \text{subject to} \quad Hz \geq h$$

and foldovers are explicitly disallowed.

3.2. Encouraging Salient Regions to Be Large

The energy function described in Equations (1) and (2) is designed to preserve the aspect ratio of highly important, or salient, regions. However, the actual size of the resulting quad does *not* affect the energy. This is clearly undesirable: we would prefer to encourage the more salient regions of the image to be larger, since emphasizing the most important content in the image is perhaps the essence of content-aware resizing. We therefore add a term to the energy which precisely achieves this effect.

Our new term is of the form

$$E_{size} = -\sum_{i,j} \lambda_{i,j}(a^1_{i,j} + a^2_{i,j})$$

which encourages quads (i, j) with high saliency to have large stretch variables $a^k_{i,j}$, and hence to be large themselves. In order to bound the effect of this term, we add constraints of the form

$$a^k_{i,j} \leq \max\{s^1, s^2\}$$

where s^k is the stretch of the image in the k^{th} direction.

We then replace our original energy E by

$$E' = E + w_{size}E_{size}$$

where w_{size} is a scalar which emphasizes how important the size term is compared the original energy term. Note that the resulting energy is still convex, and the new constraints are linear, thus resulting in a convex program as before. In Figure 1, we show the effect of the new term with different values for w_{size}; the child's face, which is the most salient region of the image, is magnified as desired.

There is an additional advantage to this new term. As we have already noted, the original energy term E has a degeneracy, namely that the energy is only concerned with the *ratio* of the stretch variables, $a^1_{i,j}/a^2_{i,j}$, and not with the values of the individual stretch variables $a^1_{i,j}$ and $a^2_{i,j}$. This degeneracy can lead to numerical instabilities when searching for the minimum. (This is akin to the problem of searching for the minimum of a function of two variables which is shaped like a trough.) The addition of the new term, E_{size}, removes the degeneracy, and leads to better numerical behavior; indeed, the time required to find the global optimum is observed to actually decrease.

3.3. Preservation of Straight Lines

Structure preservation during image retargeting is gaining more and more attention lately [14, 10]. With our general QP framework, we can easily incorporate extra terms to preserve the structures, by casting the structures as some linear combination of the underlying grids. The following sections will show how we preserve the most common and simplest structures i.e. straight lines.

3.3.1 Encouragement of Straight Grid Lines

In image resizing applications, it is often important to encourage the grid lines of the underlying rectangular mesh to remain straight. Grid lines which are noticeably curved can

lead to significant artifacts, which are easily seen; see, for instance, the examples given in [12]. In our case, straight grid lines come essentially for free. The per-quad energy of Equation (1) is designed to encourage the resulting quad to be a rectangle. In particular, it contains terms of the form

$$\left[(f^2_{i+1,j} - f^2_{i,j}) - a^2_{i,j}(x^2_{i+1,j} - x^2_{i,j})\right]^2$$

Note that $x^2_{i+1,j} - x^2_{i,j} = 0$, which results in a term of the form

$$(f^2_{i+1,j} - f^2_{i,j})^2$$

which encourages a horizontal edge in the original mesh to remain horizontal in the final mesh. Similarly, terms of the form

$$\left[(f^1_{i+1,j+1} - f^1_{i+1,j}) - a^1_{i,j}(x^1_{i+1,j+1} - x^1_{i+1,j})\right]^2$$

become, on noticing that $x^1_{i+1,j+1} - x^1_{i+1,j} = 0$,

$$(f^1_{i+1,j+1} - f^1_{i+1,j})^2$$

This term encourages the preservation of vertical edges.

For an example of this phenomenon in a real image, see Figure 3(b); the lines in this grid have been kept quite straight. The preservation of straight grid lines does not occur naturally with many other methods, as is noted in [12]. For example, Wang *et al.* [12] add an extra term to their energy to achieve this effect.

3.3.2 Preservation of Arbitrary Straight Lines

We have noted that the optimization problem we solve encourages the preservation of straight axis-aligned grid lines. However, there are often straight lines present in images that do not fall on grid lines; for example, the umbrella in the left image of Figure 2. Nothing in our algorithm guarantees that such straight lines will remain straight; in fact, it is often observed that the algorithm (and indeed all resizing algorithms) will bend such lines, see the right image of Figure 2. In this section, we outline a semi-automatic technique for preserving arbitrary straight lines.

Figure 2. Resizing does not respect arbitrary straight lines. Left: original image. Right: stretching the x-axis by a factor of 1.5 results in a broken umbrella handle.

In principle, we can detect the dominant feature lines in the image automatically [3]. Alternatively, the user can identify a line segment that s/he wishes to preserve by specifying its two end points. This line segment intersects the edges of the original quadrangular mesh in a number of places. A particular such edge will have endpoints $x_{i,j}$ and $x_{i',j'}$, so that the point of intersection P of the edge with the line segment may be denoted $\gamma x_{i,j} + (1-\gamma)x_{i',j'}$, where $\gamma \in [0,1]$. In fact, we will denote the ℓ^{th} such point $P_\ell = \gamma_\ell x_{i_\ell,j_\ell} + (1-\gamma_\ell)x_{i'_\ell,j'_\ell}$, and its transformation as $\tilde{P}_\ell = \gamma_\ell f_{i_\ell,j_\ell} + (1-\gamma_\ell)f_{i'_\ell,j'_\ell}$.

Our goal is to ensure that the set of intersection points remain colinear after the resizing transformation; that is, that the set $\{\tilde{P}_\ell\}$ lies on a line. Now, suppose that the normal to the line in the transformed image is known to us, and may be written $\alpha = [\alpha^1 \quad \alpha^2]^T$. In reality we will not have this information, but let us proceed as if we had, and return to the issue of estimating it shortly. Then we must have that

$$\alpha^T(\tilde{P}_{\ell+1} - \tilde{P}_\ell) = 0$$

It can be verified by simple substitution that this leads to a linear constraint on the $f_{i,j}$'s in which there are exactly 8 terms. There is one such constraint for each pair of consecutive intersection points of the original line segment with the original quadrangular mesh edges. We may then add these constraints, which are linear, to the quadratic program without difficulty.

Let's now revisit the issue of estimating the normal to the line segment in the transformed image. There are several possibilities here, and we sketch one simple method. We first run the resizing algorithm without the above straight line constraints. We then collect all transformed intersection points $\{\tilde{P}_\ell\}$ – which in general, will not lie on a line – and extract the principal direction of the line using PCA. We have found that this heuristic yields very good results. Another possibility is to perform an L_1 type minimization for fitting the best normal.

The proposed QP framework is flexible, and allows for arbitrary linear constraints, among which the straight line preservation constraint is but one example.

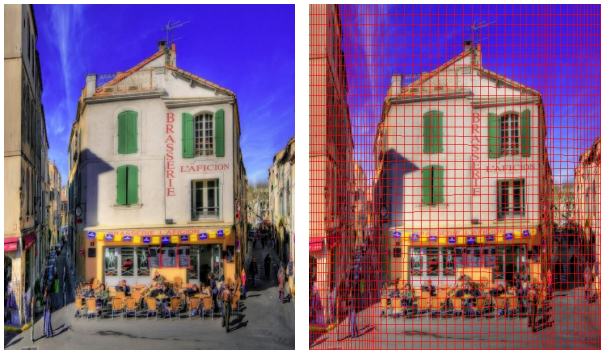
4. Results

We present experimental results comparing the proposed Quadratic Programming (QP) method with several competing techniques: Optimized Scale and Stretch (OSS) [12], Local-Global (LG) [6], Seam Carving (SC) [1, 9], and the multi-operator extension of Seam Carving (MOP) [8]. These results were generated using code acquired from the respective authors.

We briefly discuss some implementation details for our QP method. The initial quadrangular mesh is a grid of identical squares of between 10-32 pixels in both dimensions. We set the maximum aspect ratio ψ_{max} allowed for each quad to be 3 times the average scaling ratio of the whole image. To make the comparison fair, we use the same method as [12] to compute the saliency of the images, and use this saliency as input for all the methods. After we normalize the saliency

to lie in the range $[0, 1]$, we set $\lambda_1 = 0.25$ and $\lambda_1 = 0.75$. That is, the quarter of the quads with lowest saliency have $\psi_{i,j} = \psi_{max}$, while the quarter of the quads with highest saliency have $\psi_{i,j} = 1$. Our QP approach was implemented using the quadratic programming solvers in the CVX MATLAB toolbox [5]; in this environment, the resizing operation takes 2-12 seconds (depending on the grid size) on an image of size 1024×768.

Due to the large number of competing methods, we do not have sufficient space to show the deformations of the underlying grids in all cases. Instead, we show just one example generated by QP in Figure 3



(a) (b)

Figure 3. Brasserie image from Figure 4. (a) Image resized by QP. (b) The deformed quadrangular mesh overlaid on the image.

The results are shown in Figure 4. As discussed by [6], the aspect ratio of the deformation is the only thing that matters for energy-based methods, including OSS [12], LG [6] and our QP method. Since expansion in one dimension can be effectively achieved by contraction in the other dimension followed by uniform scaling in both dimensions, we only show results of contraction in the horizontal dimension. In the first row, the Brasserie example shows the power of the energy term in our QP approach which encourages the magnification of salient regions: the Brasserie, which is the clear focal point of this image, is enlarged, while at the same time its aspect ratio is maintained.[4] This contrasts with the other four methods: OSS maintains the aspect ratio, but produces a much smaller Brasserie; while SC, MOP, and LG squash the Brasserie. A similar result may be seen in the second row, in which the fish is the most salient object; QP produces the largest fish, while maintaining its aspect ratio.

In the third row, the motorcycle example shows the problems that can be caused by SC, and by extension MOP: important details can be removed. In the case of SC, both wheels (though particularly the front one) have been deformed; in the case of MOP, the back of the motorcycle has been cropped. Close inspection reveals that LG also deforms the wheels somewhat, but in a smoother manner than SC. QP and OSS produce similarly good results; QP yields a slightly larger motorcycle, thus perhaps is slightly better.

In the fourth row, the table with wine bottles shows deficiencies of OSS, as well as SC and MOP. In all three cases, the table and wine bottles – the most salient part of the image – do not maintain their aspect ratios; they are stretched vertically. Furthermore, the tables gain a measure of asymmetry. LG produces a good result in this case, comparable to the result of QP. As in the previous example, QP produces a slightly larger table, yielding a slightly better result than LG. Note that the vertical lines (chairs, windows) in this example are well preserved by QP due to the fact that QP encourages straight grid lines.

The fifth row shows an example where SC performs well. Here the most salient object is the woman, and the result of the SC algorithm is to remove some of the less salient wall. QP produces an almost identical result, though with less deformation of the background grafitti and bricks. (SC shrinks a particular column of bricks, while leaving others untouched; this type of behavior is not surprising, given the way SC operates By contrast, QP shrinks the bricks much more uniformly.) The other three methods, OSS, LG, and MOP produce inferior results, in which the woman is not as large.

In the eagle example of the sixth row, QP produces an eagle which is much closer in aspect ratio to the original. This is noticeable in particular in the wing on the right side of the image, which is quite strongly deformed by each of the four competing methods.

We now illustrate the effect of the semi-automatic straight line preservation method outlined in Section 3.3.2. In Figure 5, we show two examples in which the proposed method has destroyed a straight line structure in the image. In the top tow, the umbrella handle has effectively been broken; while in the bottom row, the wand in the child's mouth has been more subtly curved (see the ends, in particular). In both cases, as described in Section 3.3.2, we have selected the endpoints of the line segment that we wish to preserve. The results, shown in the third column of Figure 5, demonstrate the effectiveness of the approach.

Instead of using a quadratic penalty, we might use a p-norm penalty:

$$E_{i,j}^{k,l} = |c_{i,j}^{k,l} z|^p$$

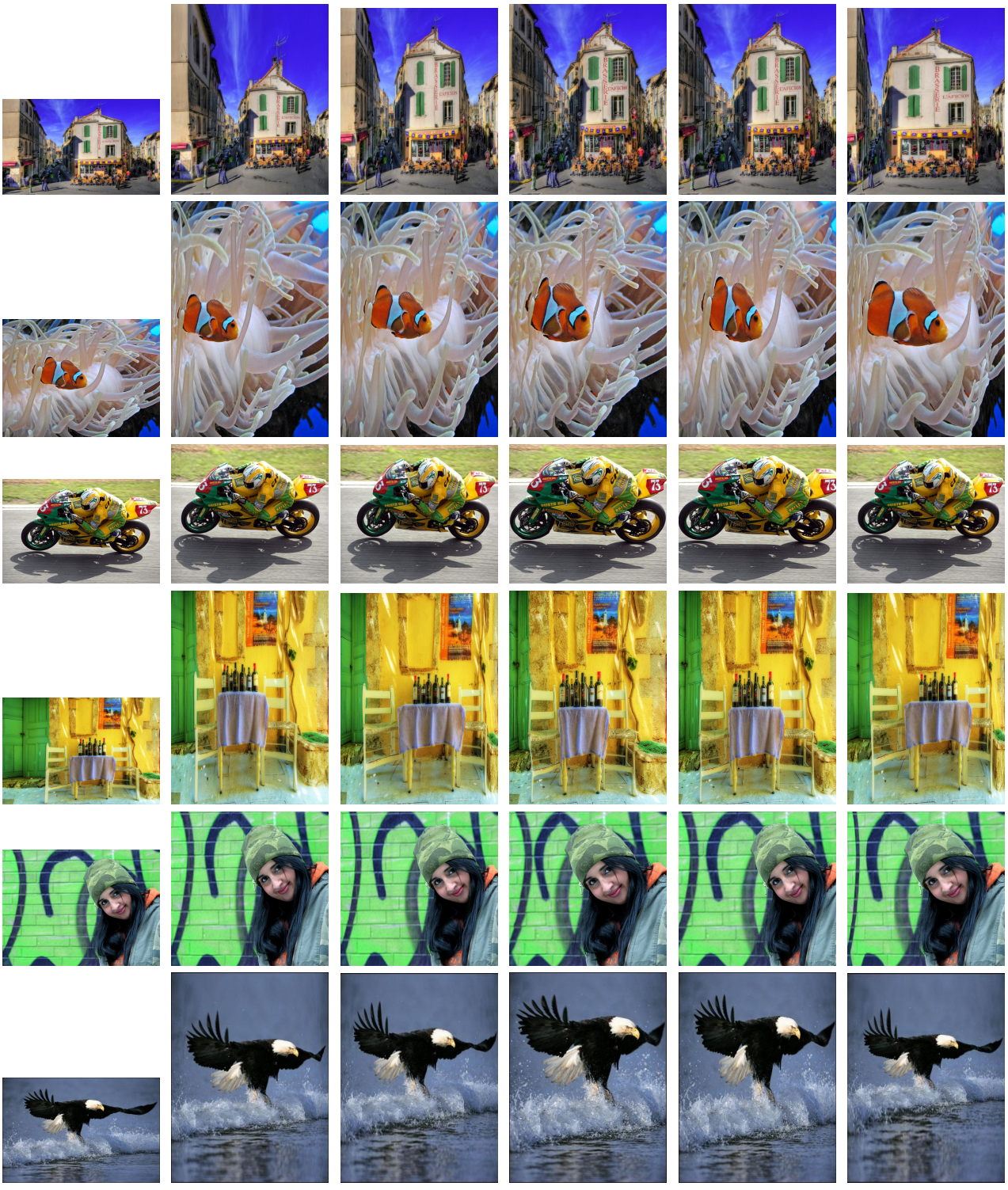
In this case, it is easy to verify that the resizing problem becomes

$$\min_z \|Cz\|_p \quad \text{subject to} \quad Hz \geq h$$

where $\|\cdot\|_p$ denotes the usual vector p-norm. In principle, any $p \geq 1$ results in a convex programming problem, for which a global minimum may be found. We have tried the cases $p = 1$ and $p = \infty$, but have not found substantial differences in the results compared to $p = 2$.

Our QP approach works well for most images, however, like all the competing methods, it is also quite dependent on

[4] To observe that the aspect ratio is maintained, inspect Figure 3(b), in which the quads over the Brasserie remain squares.



(a) (b) (c) (d) (e) (f)

Figure 4. Comparing the proposed resizing method with other competing methods. (a) Original image. (b) Optimized Scale and Stretch (OSS) [12]. (c) Local-Global (LG) [6]. (d) Seam Carving (SC) [1, 9]. (e) Multioperator extension of Seam Carving (MOP) [8]. (f) Our Quadratic Programming (QP) method.

(a) (b) (c)

Figure 5. Preserving straight line structures. (a) Original image. (b) Resizing with QP without straight line preservation. (c) Resizing with QP with straight line preservation.

the quality of the saliency map, and can perform poorly when presented with bad saliency maps. Additionally, if too many hard linear constraints are imposed, the global optimum of the energy may be infeasible. In this case, we can always transform these hard constraints into soft ones by incorporating them into the quadratic energy. Then QP can still achieve the global minimum while preserving the linear structures as much as possible.

5. Conclusions

We have presented a new method for content-aware image resizing based on a convex programming approach to computing the deformation of the underlying quadrangular mesh; as a result, a global optimum of the corresponding energy function may be found. We have shown how foldover prevention, magnification of salient regions, and preservation of straight line structures may be easily incorporated within this framework. A comparison of the proposed method with four leading competitor approaches has shown its effectiveness.

It would be interesting to see whether the preservation of more geometric structures could be incorporated into the same framework, and whether this method could be used for other image deformation applications, e.g. for animating 2D content.

References

[1] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 26(3):10, 2007.

[2] W. Dong, N. Zhou, J.-C. Paul, and X. Zhang. Optimized image resizing using seam carving and scaling. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH Asia)*, 28(5), 2009.

[3] L. Fernandes and M. Oliveira. Real-time line detection through an improved hough transform voting scheme. *Pattern Recognition*, 41(1):299–314, 2008.

[4] V. Gopalakrishnan, Y. Hu, and D. Rajan. Random walks on graphs to model saliency in images. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1698–1705, 2009.

[5] M. Grant, S. Boyd, and Y. Ye. CVX: Matlab software for disciplined convex programming. *Available at http://www. stanford. edu/boyd/cvx*.

[6] Z. Karni, D. Freedman, and C. Gotsman. Energy-Based Image Deformation. In *Computer Graphics Forum*, volume 28, pages 1257–1268, 2009.

[7] M. Rubinstein, A. Shamir, and S. Avidan. Improved seam carving for video retargeting. *ACM Trans. Graph.*, 27(3):1–9, 2008.

[8] M. Rubinstein, A. Shamir, and S. Avidan. Multi-operator media retargeting. *ACM Transactions on Graphics, (Proceedings SIGGRAPH 2009)*, 28(3), 2009.

[9] A. Shamir and S. Avidan. Seam carving for media retargeting. *Commun. ACM*, 52(1):77–85, 2009.

[10] S.-F. Wang and S.-H. Lai. Fast structure-preserving image retargeting. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1049–1052, 2009.

[11] Y.-S. Wang, H. Fu, O. Sorkine, T.-Y. Lee, and H.-P. Seidel. Motion-aware temporal coherence for video resizing. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH Asia)*, 28(5), 2009.

[12] Y.-S. Wang, C.-L. Tai, O. Sorkine, and T.-Y. Lee. Optimized scale-and-stretch for image resizing. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH Asia)*, 27(5), 2008.

[13] L. Wolf, M. Guttmann, and D. Cohen-Or. Non-homogeneous content-driven video-retargeting. In *Proc. ICCV*, pages 1–6, 2007.

[14] Q. xing Huang, R. Mech, and N. Carr. Optimizing structure preserving embedded deformation for resizing images and vector art. *Computer Graphics Forum*, 28:1887–1896, 2009.