Scene Carving: Scene Consistent Image Retargeting

Alex Mansfield[1], Peter Gehler[1], Luc Van Gool[1,2] and Carsten Rother[3]

[1] Computer Vision Laboratory, ETH Zürich, Switzerland
[2] ESAT-PSI, KU Leuven, Belgium
[3] Microsoft Research Ltd, Cambridge, UK
{mansfield, pgehler, vangool}@vision.ee.ethz.ch, carrot@microsoft.com

Abstract. Image retargeting algorithms often create visually disturbing distortion. We introduce the property of scene consistency, which is held by images which contain no object distortion and have the correct object depth ordering. We present two new image retargeting algorithms that preserve scene consistency. These algorithms make use of a user-provided relative depth map, which can be created easily using a simple GrabCut-style interface. Our algorithms generalize seam carving. We decompose the image retargeting procedure into (a) removing image content with minimal distortion and (b) re-arrangement of known objects within the scene to maximize their visibility. Our algorithms optimize objectives (a) and (b) jointly. However, they differ considerably in how they achieve this. We discuss this in detail and present examples illustrating the rationale of preserving scene consistency in retargeting.

1 Introduction

The increasing diversity of modern displays calls for methods able to transform images so as to best exploit the display form factor. Such *media retargeting* has received much attention lately [1, 2, 4–8, 10, 11, 14, 18, 19, 22, 23]. Recent success can be attributed to two developments: firstly, the use of "content-aware" algorithms with more accurate image models; secondly, the formulation of the problem as a graph labelling problem, for which efficient solvers exist [3, 21].

Most existing approaches are fully automatic, using low level visual saliency to determine image region importance. These suffer problems with structured objects, which low level saliency is not able to detect. However, we assume that a *relative depth map* is available, provided by the user. By a relative depth map, we refer to object segmentations with a depth order label, as illustrated in Fig. 1(b).

Given this depth map, our novel retargeting algorithms are capable of retargeting such that objects are protected (i.e. not distorted) and maintain their correct depth ordering. We term this condition *scene consistency.* We extend the well-known seam carving algorithm [1] to achieve this. To the best of our knowledge, these are the first retargeting algorithms that are able to re-arrange objects such that object occlusions are created, as illustrated in Fig. 1.

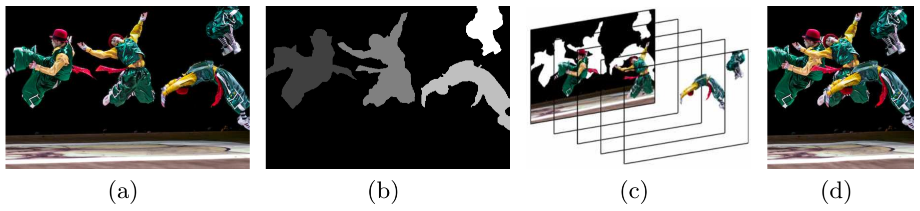


Fig. 1. For image (a) with relative depth map (b), illustrated in 3D in (c), we produce the scene consistent retargeted image (d) by the new *scene carving* algorithm

We acknowledge that assuming our additional input is a strong assumption, but the improvement in the output can make its acquisition worthwhile. Furthermore, recent developments allow the input to be acquired relatively easily. Firstly, efficient interactive user interfaces are now available for such annotation. In our work we make use of an interface employing the GrabCut algorithm [13], with which all our depth maps were created within a few minutes. Secondly, recent work [9, 16] has begun to succeed in detecting occlusion boundaries and acquiring 3D models from single images. These techniques could be used in automating, at least partially, the annotation process. Thirdly, commercial stereo cameras are hitting the market.[4] With state-of-the-art stereo depth estimation techniques [17], this technology may allow complete automation of this process.

In the next section we discuss related work on image retargeting. In Sect. 3 we discuss the properties of scene consistent retargeted images. Sections 4 and 5 contain the proposed algorithms. Real world examples are shown in Sect. 6 and we conclude with a discussion on future work in Sect. 7.

2 Related Work

There exists a large body of literature on media retargeting. In this section we discuss work which is most relevant to ours. Please note that we focus on image retargeting, although many algorithms have been extended to video.

On Retargeting. Two main strategies exist for image retargeting: minimizing applied distortion or maximizing similarity between the input and output images.

Arguably the simplest retargeting methods are cropping and scaling. These methods usually are not content aware and tend to give inferior results to algorithms that are. Some work exists on content-aware scaling and cropping [15, 19, 20] but these methods alone have limited ability to retain content or can cause distortions such that interesting parts of the image are no longer clearly visible.

Seam carving [1, 14] has received a lot of attention due to its elegance. It iteratively removes connected paths of pixels so as to minimize the resulting

[4] E.g. Fuji FinePix 3D W1. `www.fujifilm.com/products/3d/camera/finepix_real3dw1`.

distortion. It can be thought of as forgetting the input image altogether, as the distortion it measures is relative only to the previous image. Together with our algorithms and other extensions [7, 8, 15], it falls under the first strategy. We build on it for reasons of speed and because of the ability to explicitly control the modifications of pixels. We discuss this in greater detail in Sect. 4.1.

The second strategy requires a notion of distance between the input and output image. Many have been proposed and used in retargeting, based on e.g. patch colour similarity [2, 15, 19] with dominant colours [5], saliency with image gradients [22] or with face attention [18], and colour and gradient difference [11].

On Protecting Objects. Object protection (i.e. non-distortion) is important for the realism of synthesized images. In dynamic video synopsis [12], objects are detected using background subtraction, and protected in the synopsis. In [6] the user is requested to mark parts of the image where shape should be preserved. In [1], users can specify regions to be protected or removed during retargeting.

The method proposed by [18] is closest to our approach with regard to object protection. Importance maps are created automatically, from which important regions are detected. The retargeted output is constructed by removing the important regions, inpainting the resulting holes in the background, rescaling the background, and finally re-inserting and re-arranging the removed regions to create the output. The important regions thus avoid the rescaling, and so are protected. The authors show results which are visually pleasing, but the method relies on the strength of the inpainting algorithm. Also, unlike our methods, it is not able to create consistent object occlusions.

3 Scene Consistency

We first introduce the key concept of scene consistency. We model image formation as projection of flat fronto-parallel objects at different depths onto a background plane. An image can be decomposed into such a model as illustrated in Fig. 1(c). A retarget of the image is *scene consistent* if objects (1) are not distorted but kept as in the original image and (2) are placed in their correct depth ordering. We also define the concept of object consistency, which is held by retargets for which property (1) holds, that objects are not distorted.

This concept provides a formalization of scene realism, which we want to maintain during retargeting. To do so requires the model decomposition of the original image, which for a single image can be described simply in terms of a relative depth map, giving object segmentations each with a depth ordering label as illustrated in Fig. 1(b). Object segmentations alone allow scene consistent retargeting, by enforcing no distortion for the objects, but with the depth information, scene consistent occlusions may also be generated. The benefits of scene consistent retargeting are illustrated for a toy image in Fig. 2. Note that we distinguish between occlusions that require reappearance and those that do not, a distinction we find arises in practice. By “reappearance” we refer to pixels previously occluded becoming visible again while iterative retargeting.

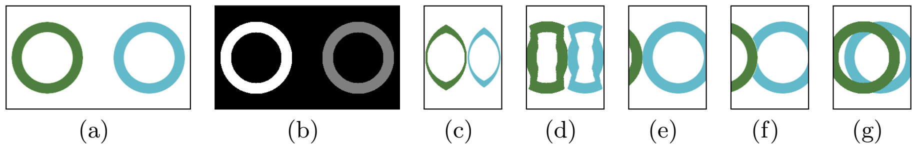


Fig. 2. Toy image (a) with depth layers (b) is retargeted by seam carving [1] (c), seam carving with object protection (Sect. 4.1, [1]) (d) and (e), seam carving with occlusions (Sect. 4.2) (f) and scene carving (Sect. 5) (g). Our two new algorithms (f, g) may form occlusions: in seam carving with occlusions (f), occlusions that do not require reappearance may be formed (see Sect. 4.3); in scene carving (g), all scene consistent occlusions may be formed

Occlusion in the original image means some parts of the model decomposition are unknown. We refer to these as *holes*. Holes constrain scene consistent retargeting: all holes must be kept occluded, to prevent the need to inpaint.

We use the following notation throughout. The image intensity is $I_{r,c}$ for pixels (r, c) in the image domain $\mathcal{P}$. An object map is defined over the same domain as $O(r, c) = o$ at pixels belonging to object $o > 0$; otherwise, $O(r, c) = 0$.

4 Towards Scene Consistent Seam Carving

In this section, we recap seam carving (S.C.) (Sect. 4.1), which we extend to be able to create scene consistent object occlusions (Sect. 4.2) by enabling seams to pass through occlusion boundaries. This extension we call *seam carving with object occlusions* (S.C.+Obj. Occ.). We discuss a complication of this algorithm, namely that it does not easily allow for object reappearance, in Sect. 4.3.

4.1 Seam Carving

Our algorithms build on seam carving with forward energy [14]. Seam carving greedily removes seams with minimum energy from an image. A seam is an 8-connected path through the image, containing a single pixel on each row (assuming vertical seams are removed as we do throughout without loss of generality). Removing pixel (r, c) causes the following distortions: it brings into horizontal contact its **L**eft $(r, c-1)$ and **R**ight neighbours $(r, c+1)$ in row r. Depending on where the seam passed in row $r-1$, it may additionally bring into vertical contact its **U**pper and **L**eft or its **U**pper and **R**ight neighbours. The energy of these distortions is captured in the following terms, used as illustrated in Fig. 3(a):

$$\begin{aligned} E^{\mathrm{LR}}_{r,c} &= |I_{r,c-1} - I_{r,c+1}| \\ E^{\mathrm{LU}}_{r,c} &= |I_{r,c-1} - I_{r-1,c}| \\ E^{\mathrm{UR}}_{r,c} &= |I_{r-1,c} - I_{r,c+1}| \ . \end{aligned} \quad (1)$$

These terms measure distortion by magnitude similarity of neighbouring pixels.

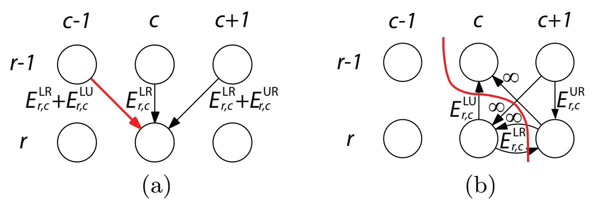


Fig. 3. Graphs for dynamic programming (a) and graph cut (b) optimization of forward energy seam carving. Only terms related to pixel (r, c) are shown. The red arc in (a) corresponds to the red cut in (b), removing pixels $(r-1, c-1)$ and (r, c)

The seam that corresponds to minimal energy can be efficiently found using dynamic programming (D.P.), or using a graph cut (G.C.) [14]. In the latter, the problem is cast as a binary graph labelling problem. The corresponding graph is shown in Fig. 3(b). After the graph is cut, the pixel on each row directly left of the cut is the seam pixel, as exemplified by the red arc and cut in Fig. 3. These two frameworks are equivalent but have different properties [14].

The aim of this paper is to maintain scene consistency in retargeting. A simple method for preventing object distortion is given in [1], which we refer to as *seam carving with object protection* (S.C.+Obj. Prot.). The energies of all arcs pointing to pixels that belong to an object are set to infinity:

$$E^{\mathrm{LR}}_{r,c} = E^{\mathrm{LU}}_{r,c} = E^{\mathrm{UR}}_{r,c} = \infty \quad \forall (r,c) \in \{(r,c) : O_{r,c} > 0\} \ . \tag{2}$$

This ensures that no seams pass through objects. As seams are progressively removed, objects are moved together until they abut. Continuing to remove seams, with infinite energy, would lead to great distortion (see Fig. 2(d)). For object consistency we enforce that seams may then pass only through edges of the image, resulting instead in a cropping (see Fig. 2(e)).

Neither of these methods allows seams to cut through the occlusion boundaries, moving objects behind one another. This would allow more flexibility for seams to be removed. In the next section we present an algorithm to do this.

4.2 Seam Carving with Object Occlusions

We now describe *seam carving with object occlusions* (S.C.+Obj. Occ.). This algorithm behaves like seam carving in background regions, but protects objects and allows seams to pass through occlusion boundaries between objects, as illustrated in Fig. 4(a). Two modifications are made, to the energies at occlusion boundaries and to the graph structure, with the use of "supernodes".

Occlusion Boundaries. For background pixels that border the edge of the image or an object, the standard forward energy does not apply. Removing these pixels can be viewed as an occlusion, with no visual distortion created. We replace the energies for these pixels with a small value $u_\mathrm{s} = 10$. For $E^{\mathrm{LR}}_{r,c}$,

$$E^{\mathrm{LR}}_{r,c} = u_\mathrm{s} \quad \forall (r,c) \in \{(r,c) : O_{r,c} = 0 \wedge ((r,c-1) \notin \mathcal{P} \vee (r,c+1) \notin \mathcal{P})\} \\ \cup \{(r,c) : O_{r,c} = 0 \wedge (O_{r,c-1} > 0 \vee O_{r,c+1} > 0)\} \tag{3}$$

gives the formal condition for use of this term, with similar definitions for $E^{\mathrm{LU}}_{r,c}$ and $E^{\mathrm{UR}}_{r,c}$. This energy modification could also be applied to S.C.+Obj. Prot.

Introducing Supernodes. We must allow seams to run along object occlusion boundaries while protecting objects. With occlusions possible, object protection cannot be ensured by infinite energy terms as in (2). Consider the object in Fig. 4(a) that is occluded and separated into two parts. Consistency requires that seams pass all visible parts of an object on the same side, so seam (b) in the figure is invalid. As can be seen, consistency does not exhibit optimal substructure and cannot be optimized with dynamic programming.

We resolve this problem by considering the graph cut formulation and modifying the graph *structure* to protect objects. We introduce *supernodes*, nodes that subsume a group of pixel nodes. A supernode takes only a single label, so pixels subsumed by the supernode are assigned the same label.

Supernodes are constructed as follows, as illustrated in Fig. 4(b). Recall that in the graph cut formulation, the seam pixels are those directly left of the cut (c.f. Fig. 3(b)). We take the object closest to the camera and create a supernode from all object pixels as well as their right neighbours. This procedure is now iterated from the closest to the furthest object. At each step all object pixels and their right neighbours are included in the supernode, if they are not already in an existing supernode (e.g. the node in the second row, fourth column in 4(b)).

Energy Terms for Supernodes. The energy of object-background occlusion was defined in (3). We now define the energy of object-object occlusion. We set the energy terms of pixels in the occlusion boundary to a term u_o where

$$u_\mathrm{o} = \frac{u_\mathrm{obj}}{|\{(r,c) : O_{r,c} = o\}|} \tag{4}$$

for a fixed constant u_obj. Setting this constant high increases the energy of occlusion of an object pixel, and even more so for smaller objects. We use $u_\mathrm{obj} = 10^7$. Note that the borders of the image are treated in the same way, as an occluding object. Note also that if the occlusion is not valid, because it would lead to reappearance of part of an object behind another or because the objects next to each other are at the same depth, we can simply merge the supernodes for the two objects to prevent any further occlusion occurring.

Occlusion boundaries cannot be carved with the algorithm so far described if it is not possible for an 8-connected seam to pass through them. We therefore relax the connectivity constraint around objects, allowing seams to jump through horizontal occlusion boundaries. We do this by not attaching to supernodes the infinite cost arcs that enforce this constraint (e.g. the arc from $(r-1, c+1) \rightarrow (r,c)$ in Fig. 3(b)). An example of a seam this allows is the red cut in Fig. 4(b).

4.3 Limitation to Non-Reappearance

The described algorithm can only *remove* pixels, hence the need to prevent occlusions that would cause part of an object to reappear. We would like to relax this constraint and include an energy term for this reappearance, as in many images this is necessary to create useful occlusions as in Fig. 2(g). However, we found that extending the algorithm so far described to this would lead to an energy with higher order potentials, which are in general non-submodular and cannot be optimized efficiently. We demonstrate this with an example before describing, in Sect. 5, an algorithm which does not suffer this limitation.

Consider Fig. 5. Without reappearance, seams simply determine object movements: if the object is to the right of the seam, it is moved left, and if it is to the left, it maintains its position, relative to the left edge of the image. Without loss of generality we assume the same rule even with reappearance. The seams passing through the boundaries between the blue and purple objects simply determine the behaviour at this boundary: seams on the left (c) and (f) lead to reappearance on the right, and similarly with seams (d) and (e). Hence the reappearance energy can be associated with passing through the boundaries. However, no such relationship exists for the occlusion boundary between the blue and green objects (seams (e) and (f)), where the reappearance also depends on the purple object. In general, it would be necessary to encode the reappearance energy to depend on the positioning of all of the objects. This energy would contain higher-order potentials and in general be non-submodular.

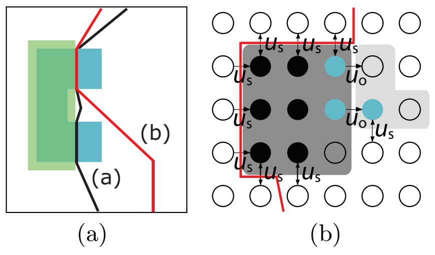


Fig. 4. Left: The blue "C" shaped object is occluded (indicated by transparency) and thus split into two separate parts. Hence the red seam (b) does not preserve object consistency, while seam (a) does.
Right: Two objects, their corresponding supernodes and changed energy terms. The object with black pixels is closest and creates the supernode containing nodes in the dark grey area. The supernode of the blue object is the light shaded region. Also shown are those energy terms that changed compared to seam carving. The red line indicates a possible cut along the objects

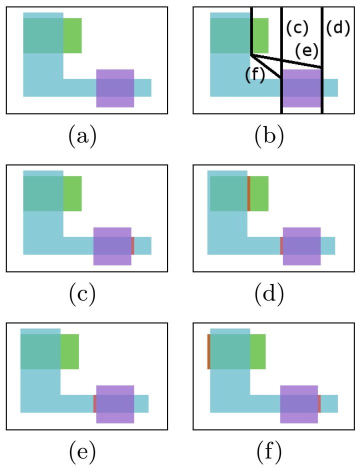


Fig. 5. S.C.+Obj. Occ. with reappearance requires higher order terms. The seams passing through occlusion boundaries in (a) are shown in (b) resulting in (c) to (f). The objects are shown with transparency, with purple in front of blue in front of green. Reappearing pixels are highlighted in red

5 Scene Carving

In seam carving, including the described extensions, the seam determines the movement of objects. This led to the problem that objects and background reappearance could not be optimized for efficiently. We resolve this problem by using a layered decomposition (Sect. 5.1) and adding the possibility of removing background holes (Sect. 5.2). This yields the *scene carving* (Sc. Carve) algorithm.

5.1 Layered Decomposition

The main idea of scene carving is the use of a layered image decomposition as illustrated in Fig. 1(c). Each object is stored in a separate layer. The last layer is referred to as the *background image*. This contains the background, with holes where the background is occluded by objects. From this representation an image can be created by "flattening" the layers onto the background image. Scene consistency is inherent if object layers are only translated in the plane, but have no pixels removed. We then only find seams in the background image.

This decomposition allows us to store an over-complete representation of the image. Pixels that are occluded in the flattened image are still stored in their respective layer and thus may reappear at a later iteration, as in Fig. 2(g).

The algorithm proceeds as shown in Fig. 6. At each iteration we consider all object positionings, and for each find the seam in the background image. Since the background image contains no objects, as in S.C., this can be done efficiently using dynamic programming. We calculate the total energy as the sum of the seam energy and object positioning energy, and take the joint minimum. Note that for V object movements and N objects, there are V^N object positionings to test at each iteration. We use the $V = 2$ movements of S.C.: the object stays in the same position or moves one pixel to the left, relative to the left of the image. In Sect. 5.3 we describe a speed up for this combinatorial problem.

5.2 Seams in the Background Image

Since the seam does not carry the burden of determining object movement it may pass anywhere in the background image, including through holes. The only restriction is to ensure that all holes are occluded in the resulting image. We now define the energy of such a seam in the background image.

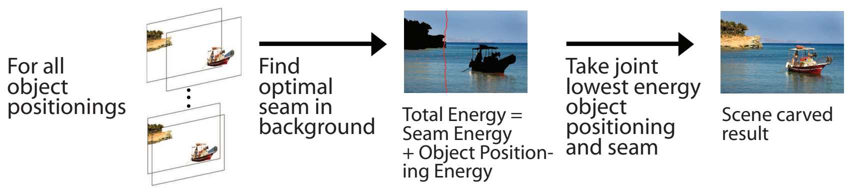

For all object positionings
Find optimal seam in background
Total Energy = Seam Energy + Object Position-ing Energy
Take joint lowest energy object positioning and seam
Scene carved result

Fig. 6. Scene carving jointly optimizes for a new object positioning and a seam to be removed from the background image

Distortion Domain. We can distinguish two choices for seam energies, calculating the distortion of either (1) the flattened image or (2) the background only. S.C. advocates (1) and our extension in Sect. 4.2 also follows this rationale.

However strategy (1) comes at the expense of allowing high distortion to be created in the background image at no cost behind objects. This could severely limit our ability to move objects in further iterations and allow increased distortion in the background. Empirical results show that this occurs in our images. See Fig. 8(f) and 8(g), where this method is referred to as Sc. Carve-D.[5]

We therefore take the second approach (2) and optimize at each iteration jointly for the highest fraction of objects to be visible and for the minimally distorting seam in the background image. This leads us to the *scene carving* algorithm. We pay the cost for introducing distortions that are not currently visible (but may be at future iterations), therefore sacrificing some potential improvement in the image at this iteration for a potentially better result image.

Seam Energy. We noted we find a seam only in the background image, so we are able to use D.P. for better runtime behaviour than S.C.+Obj. Occ. We construct the seam energy as follows. We reuse the graph of S.C. with the energies of (1). Energy terms for pixels next to the image boundary or holes are set as in (3) to a small constant, here $u_{\mathrm{s}} = 6$. As seams may pass through holes, we set energy terms for hole pixels to a non-infinite constant u_{h}. Given a binary hole mask H taking value 0 where the background is known and 1 otherwise:

$$E^{\mathrm{LR}}_{r,c} = E^{\mathrm{LU}}_{r,c} = E^{\mathrm{UR}}_{r,c} = u_{\mathrm{h}} \quad \forall (r,c) \in \{(r,c) : H_{r,c} = 1\} \ . \tag{5}$$

We set $u_{\mathrm{h}} = 0$ to encourage removal of hole pixels.

Remaining hole pixels constrain object movement, as all must be kept occluded by an object. This constraint is ensured by setting the following energy:

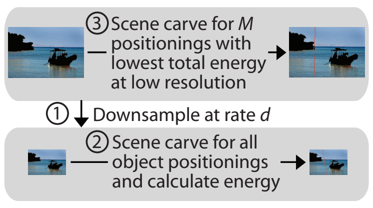
$$\begin{aligned} E^{\mathrm{LR}}_{r,c} = E^{\mathrm{LU}}_{r,c} = E^{\mathrm{UR}}_{r,c} &= \infty \quad \forall r,c \in \{c : c > c^{\max}_r \vee c < c^{\min}_r\} \qquad (6)\\ \text{where:} \qquad c^{\min}_r &= \max\{c : H_{r,c} = 1 \wedge O_{r,c} > 0 \wedge O_{r,c-1} = 0\}\\ c^{\max}_r &= \min\{c : H_{r,c} = 1 \wedge O_{r,c} = 0\} \ . \end{aligned}$$

This constrains the seam at a row r to pass between the columns $c^{\min}_r$ and $c^{\max}_r$.[6]

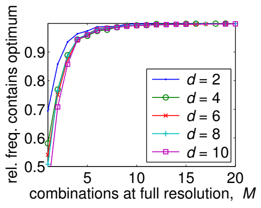
Object Positioning Energy. We compute the *final energy* by adding to the energy of the optimal seam an object positioning energy term: the negative of the unary used in S.C.+Obj. Occ. (4). At each iteration we take the joint object positioning and background image seam with the lowest energy.

[5] Details on how to define the energy for (1) and optimize it, taking all changes into account, can be found in the supplementary material, along with additional results.

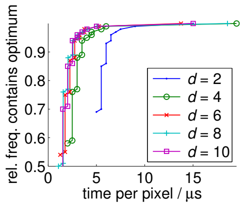
[6] Small scale non-convexities in object segmentations can limit seams through this constraint, so we remove these by simple dilation and erosion processes.


③ Scene carve for *M* positionings with lowest total energy at low resolution
① Downsample at rate *d*
② Scene carve for all object positionings and calculate energy

(a)



(b)



(c)

Fig. 7. (a) describes a 3 step hierarchical approximation to speed up scene carving. We set the parameters based on (b) and (c). (b) shows the relative frequency that the M object positionings checked in step ③ includes the optimal full resolution positioning, (c) the relative frequency against optimization time per pixel, assuming $N = 5$ objects

5.3 Speeding Up

Scene carving has computational complexity D.P. $\times\, 2^N$ at each iteration. While dynamic programming is very efficient, this algorithm is still infeasible for large numbers of objects. We use two approaches to give a speed up.

Firstly, for a constant factor speed up, we note that objects only affect the energy on the rows they span, c.f. (6). We iterate through object positionings in a unit distance code, reusing the graph above and below the object moved.

Secondly, we use a hierarchical speedup, as described in Fig. 7(a). We set $M = 5$ and $d = 6$ based on the following analysis. On 11 images containing 2-8 objects we removed 300 seams using scene carving at the full resolution and at lower resolutions. Our results are shown in Fig. 7(b) and Fig. 7(c). Choosing $d = 6$ (red curve) and $M = 5$ places us at the "knee" of the trade-off curves of Fig. 7(b). Here, the optimal object positioning is obtained approximately 97% of the time. Fig. 7(c) then shows that if we want to find the optimum approximately 97% of the time, greater downsampling would not increase the speed.

This method is still combinatorial in the number of objects, but with a lower multiplying factor. In most cases we expect a low number of objects to be labelled (up to 10), such that optimizing over all combinations of positionings is feasible.

6 Results

We now present results for our algorithms, and compare these results to those gained from our implementation of seam carving.[7] For convenience the key properties of these algorithms are summarized in Table 1.

The power of our algorithms can be demonstrated with the example of the *People* image (from [11]) in Fig. 8. Seam carving (Fig. 8(c)) can be seen to create visually disturbing distortion of the people. Ensuring object consistency prevents this, but because there is no occlusion handling, this results in a cropped image with the two left-most people removed completely. (Fig. 8(d)).

[7] All code is available at `www.vision.ee.ethz.ch/~mansfiea/scenecarving/` under the GNU General Public License.

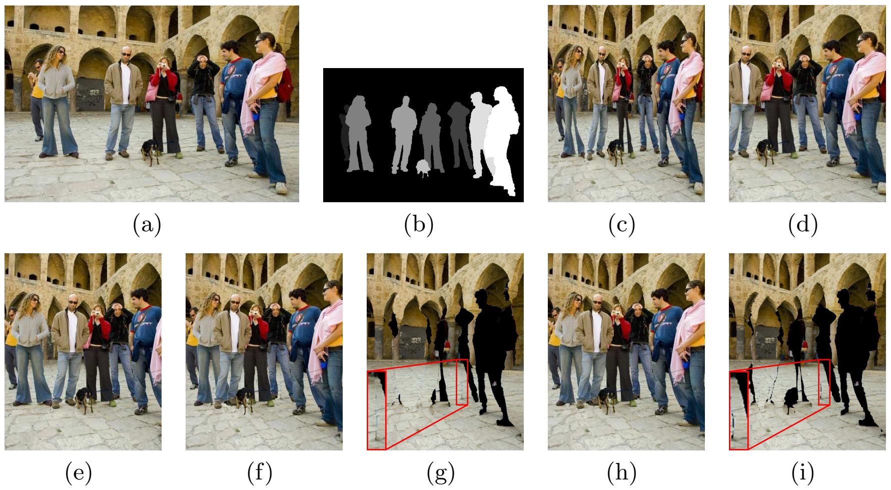


Fig. 8. *People* image (a) with depth map (b) retargeted by S.C. (c), S.C.+Obj. Prot. (d), S.C.+Obj. Occ. (e), Sc. Carve-D. (f) with bkg. image (g), Sc. Carve (h) with bkg. image (i) (300 seams removed). Note the distortion introduced by S.C., and cropping with S.C.+Obj. Prot. and S.C.+Obj. Occ. Sc. Carve keeps all objects, with the red boxes highlighting background distortion from Sc. Carve-D. not in Sc. Carve

Our algorithms guarantee scene consistency. S.C.+Obj. Occ. (Fig. 8(e)) moves the people together until reappearance would occur. Further seams are removed at the edges of the image, again cropping one person out. With Sc. Carve-D., reappearance is possible, but the ability to hide high gradients behind parts of objects allows distortion to be created in the background image, which are visible in the resulting image. These distortions are shown in Fig. 8(g), highlighted in the red box. Scene carving (Fig. 8(h)) is able to keep all people in the image, scene consistently, combined with a pleasing background (Fig. 8(i)).

Further results, demonstrating the same effects, are shown in Fig. 9.

Limitations of our methods can also be seen in these images. For example, in the *Boat* image, it can be seen that our freedom to edit the background image has shrunk the boat reflection so it no longer spans the whole boat. Another effect, caused by inaccurate segmentations, is shown in the *London eye* image, where sky can be seen through the wheel, where the building should be visible.

Table 1. Properties of the algorithms tested

	Scene consistent	Creates occlusions	With re-appearance	Optimization
S.C. [1, 14]	×	×	×	DP or GC
S.C.+Obj. Prot. [1, 14]	✓	×	×	DP or GC
S.C.+Obj. Occ. (Sect. 4.2)	✓	✓	×	GC
Sc. Carve (Sect. 5)	✓	✓	✓	DP$(5 + 2^N/6^2)$

Table 2. Time taken to produce results with our *Matlab/Mex* implementation

	People	*Sledge*	*Dancers*	*London eye*	*Boat*
No. objects	8	5	4	3	2
Size	640×427	1024×759	500×333	1024×683	1016×677
No. seams removed	300	500	200	500	500
S.C. [1, 14]	22s	62s	14s	49s	64s
S.C.+Obj. Prot. [1, 14]	28s	69s	9s	60s	54s
S.C.+Obj. Occ. (Sect. 4.2)	4515s	46152s	328s	2079s	19941s
Sc. Carve (Sect. 5)	352s	596s	73s	711s	619s

Table 2 shows the time taken to produce our results. In all cases, Sc. Carve is the fastest algorithm that allows for object occlusions. S.C.+Obj. Occ., while a non-combinatorial optimization problem, in practice produces a graph that is slow to optimize. S.C. and S.C.+Obj. Prot. are much faster, but may respectively lead to object distortion, or cropping and bad background distortion.

7 Conclusions and Future Work

In this work we considered the problem of scene consistent image retargeting. We developed two algorithms to perform such image retargeting, given a relative depth map: *seam carving with object occlusions* and *scene carving*.

The former was derived by making use of supernodes, enabling correct occlusion handling for the first time. This algorithm has the appealing property of requiring a single optimization in each step. However, accounting for reappearing material leads to graphs which cannot be optimized efficiently. Even without reappearance, the graph can be slow to optimize in practice.

Scene carving utilizes a layered decomposition of the image to allow flexible object re-arrangement. We find the joint global optimum seam and rearrangement at each iteration with dynamic programming, at the expense of an overall combinatorial problem. We presented a more efficient hierarchical approximation, which still finds the global optimal in almost all iterations.

In summary, we recommend scene carving as the better algorithm, given that it is usually faster and produces visually superior results. Seam carving with occlusions may be competitive only when very many objects are present.

There are several possible routes to be followed. First we want to automate relative depth map creation using either high-level computer vision such as object detection, or stereo vision. Also, the seam carving algorithm can be understood as "forgetting" the previous input at each iteration. Other methods optimize an energy defined between the input and output image [5, 11, 15, 18, 19, 22]. We plan to derive a similar retargeting method for our problem scenario.

Acknowledgements. We thank the following users of Flickr for placing their work under the Creative Commons License: badkleinkirchheim (*Sledge*), William Hamon (*Dancers*), Alain Bachellier (*Boat*), wallyg (*London eye*).

References

1. Avidan, S., Shamir, A.: Seam carving for content-aware image resizing. In: SIGGRAPH (2007)
2. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: PatchMatch: A randomized correspondence algorithm for structural image editing. In: SIGGRAPH (2009)
3. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. PAMI 23(11) (2001)
4. Cho, T.S., Butman, M., Avidan, S., Freeman, W.: The patch transform and its applications to image editing. In: CVPR (2008)
5. Dong, W., Zhou, N., Paul, J.C., Zhang, X.: Optimized image resizing using seam carving and scaling. ACM Trans. Graph. 28(5) (2009)
6. Gal, R., Sorkine, O., Cohen-Or, D.: Feature-aware texturing. In: Eurographics Symposium on Rendering (2006)
7. Grundmann, M., Kwatra, V., Han, M., Essa, I.: Discontinuous seam-carving for video retargeting. In: CVPR (2010)
8. Han, D., Wu, X., Sonka, M.: Optimal multiple surfaces searching for video/image resizing - a graph-theoretic approach. In: ICCV (2009)
9. Hoiem, D., Stein, A., Efros, A., Hebert, M.: Recovering occlusion boundaries from a single image. In: ICCV (2007)
10. Krähenbühl, P., Lang, M., Hornung, A., Gross, M.: A system for retargeting of streaming video. In: SIGGRAPH (2009)
11. Pritch, Y., Kav-Venaki, E., Peleg, S.: Shift-map image editing. In: ICCV (2009)
12. Rav-Acha, A., Pritch, Y., Shmuel, P.: Making a long video short: Dynamic video synopsis. In: CVPR (2006)
13. Rother, C., Kolmogorov, V., Blake, A.: "GrabCut": interactive foreground extraction using iterated graph cuts. In: SIGGRAPH (2004)
14. Rubinstein, M., Shamir, A., Avidan, S.: Improved seam carving for video retargeting. In: SIGGRAPH (2008)
15. Rubinstein, M., Shamir, A., Avidan, S.: Multi-operator media retargeting. ACM Trans. Graph. 28(3) (2009)
16. Saxena, A., Sun, M., Ng, A.: Make3d: Learning 3-d scene structure from a single still image. IEEE PAMI 31(5) (2009)
17. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vision 47(1-3) (2002)
18. Setlur, V., Takagi, S., Raskar, R., Gleicher, M., Gooch, B.: Automatic image retargeting. In: Int. Conf. on Mobile and Ubiquitous Multimedia (2005)
19. Simakov, D., Caspi, Y., Shechtman, E., Irani, M.: Summarizing visual data using bidirectional similarity. In: CVPR (2008)
20. Suh, B., Ling, H., Bederson, B.B., Jacobs, D.W.: Automatic thumbnail cropping and its effectiveness. In: UIT: ACM Symposium on User Interface Software and Technology (2003)
21. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., Rother, C.: A comparative study of energy minimization methods for markov random fields with smoothness-based priors. PAMI 30(6) (2008)
22. Wang, Y.S., Tai, C.L., Sorkine, O., Lee, T.Y.: Optimized scale-and-stretch for image resizing. In: SIGGRAPH Asia (2008)
23. Wolf, L., Guttmann, M., Cohen-Or, D.: Non-homogeneous content-driven video-retargeting. In: ICCV (2007)

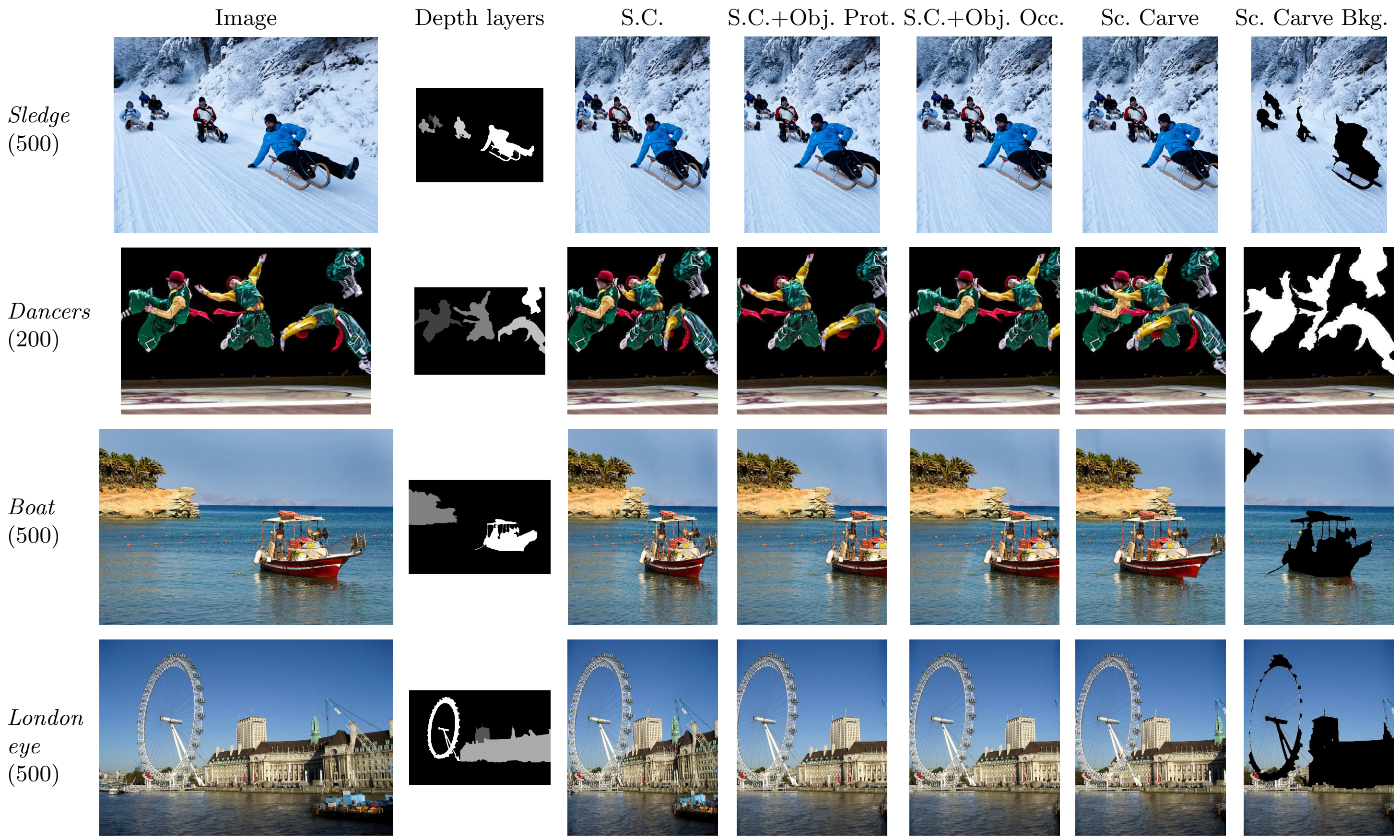


Fig. 9. Further results. The number by each image name indicates the number of seams removed. The final column of images shows the background image at the end of Sc. Carve. Holes are shown in black for all images, except for *Dancers* where they are shown in white