MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Image and Video Retargetting by Darting

Matthew Brand

TR2009-032 July 2009

Abstract

This paper considers the problem of altering an image by imperceptibly adding or removing pixels, for example, to fit a differently shaped frame with minimal loss of interesting content. We show how to construct a family of convex programs that suitably rearrange pixels while minimizing image artifacts and distortions. We call this ”darting” on analogy to a tailor’s darts-small edits are discreetly distributed throughout the fabric of the image. We develop a reduction to integer dynamic programming on edit trellises, yielding fast algorithms. One- and two-pass variants of the method have 0(1) per-pixel complexity. Of the many edits that darting supports, five are demonstrated here: image retargeting to smaller aspect ratios: adding or moving or removing scene objects while preserving image dimensions: image expansion with gaps filled by a rudimentary form of texture synthesis; temporal video summarization by ”packing” motion in time; and an extension to spatial video retargetting that avoids motion artifacts by preserving optical flow.

Int. Conf. Image Analysis and Recognition

Image and video retargetting by darting

Matthew Brand

Mitsubishi Electric Research Labs

Abstract. This paper considers the problem of altering an image by imperceptibly adding or removing pixels, for example, to fit a differently shaped frame with minimal loss of interesting content. We show how to construct a family of convex programs that suitably rearrange pixels while minimizing image artifacts and distortions. We call this "darting" on analogy to a tailor's darts—small edits are discreetly distributed throughout the fabric of the image. We develop a reduction to integer dynamic programming on edit trellises, yielding fast algorithms. One- and two-pass variants of the method have $O(1)$ per-pixel complexity. Of the many edits that darting supports, five are demonstrated here: image retargetting to smaller aspect ratios; adding or moving or removing scene objects while preserving image dimensions; image expansion with gaps filled by a rudimentary form of texture synthesis; temporal video summarization by "packing" motion in time; and an extension to spatial video retargetting that avoids motion artifacts by preserving optical flow.

1 Introduction

The proliferation of diverse imaging devices and formats raises an interesting problem: How to re-compose an image to fit new boundary—usually smaller—without cropping or distorting interesting parts of the scene? In recent years, striking results have been demonstrated with three approaches:

Recomposition methods break an image into pieces to be reassembled in visually pleasing ways [1,2,3]. Although most general, its advocates point out that it is equivalent to the NP-hard "jigsaw puzzle" problem; scalable and broadly useful approximations have yet to be found.

Non-homogeneous rescaling computes new coordinates for pixels [4] with saliency-based penalties for squeezing pixels together, somewhat like a block of soft rubber that has been stiffened where the image is interesting. The fast and attractive quadratic formulation is equivalent to Tutte's embedding of a weighted graphs in the plane [5]. As such, it is vulnerable to unwanted embedding inversions where parts of the image fold over other parts, i.e., pixel order is not necessarily preserved.

Seam-carving removes or adds pixels along a continuous seam spanning some visually smooth part of the image [6]. Repeated seam carving can produce remarkable results, even though this greedy procedure optimizes no global objective. There is no penalty for distortions, but if no seam transects a scene object, it is left intact. By the same token, carved seams tend to distort and damage simple image contours. A "forward energy" reformulation in the framework of graph cuts avoids some of these arti-

Fig. 1. Fast darting with backpropagation. Even though many unconnected groups of pixels are removed, contours are well preserved—even the pixel-wide tether lines. Figure 2 shows the darts.

facts and enables carving in XYT (video) [7], but computation is quite slow and limited to greedy seam removal. Curiously, a more efficient formulation was available[1].

This paper introduces algorithms for image *darting*—inserting and/or removing pixels to make an image fit a new boundary, much as a tailor darts a piece of fabric to fit a mannequin. Darting has many of the strengths of the above schemes. Like 2D seam carving, there is a very fast integer algorithm. Unlike carving, darting optimizes the quality of the final—not intermediate—image, and the edits do not form seams. Like non-homogeneous scaling, darting computes new pixel placements subject to free-form boundary constraints. Unlike scaling, it explicitly minimizes local contrast artifacts and nonlocal distortions. Like recomposition, darting can recruit texture from elsewhere in the image to fill gaps. Unlike recomposition, the optimization problem is convex.

2 Darting as a linear or quadratic program

We illustrate the basic idea of darting by constructing a linear program (LP) to narrow an image by removing pixels, then proceed to more efficient and flexible formulations. In this LP, new horizontal coordinates $\{x_{ij}\}_{1\le i\le I;1\le j\le J}$ are computed for all pixels in an $I\times J$ image such that some are occluded by their neighbors, effectively removing them.

The LP's main constraint is that the result must fit and fill the target boundary with no folds or gaps. Formally, $\forall_{1\le i\le I;1\le j\le J}\, x_{1j}=1, x_{Ij}=I_{\text{new}}, x_{ij}\le x_{i+1,j}\le x_{ij}+1$.

The LP objective is to minimize visible artifacts (e.g., contrasts induced by newly adjacent pixels) and distortions (e.g., nonrigid motions of related pixels). These conditions are flagged by auxiliary indicator variables. We flag a horizontal adjacency caused by the occlusion of k pixels to the right of pixel ij with variable $a_{ijk}\ge \max(0,1-x_{i+k+1,j}+x_{ij})$. We flag a vertical adjacency caused by a k-pixel image shear with variable $s_{ijk}\ge \max(0,1-x_{i+k,j+1}+x_{ij})$. For selected points along an image contour one

[1] As a historical aside, the idea of forward energy, its *minimal* graph-cut formulation, and the extension of that to XYT were originally proposed to the seam carving authors at a seminar [8]. The formulation published 14 months later in [7] differs from the original proposal [8] in the addition of a completely redundant constraint (gratuitous graph edges in [7, figs. 4,15]).

Fig. 2. The darts of figure 1. Pin-stripes of removed (resp., retained) pixels are due to a girth$=90$ (resp., dartwidth≤ 12) constraint.

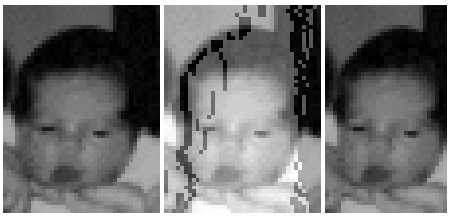

Fig. 3. Fast (one-pass) darting on a low-res image. From left to right: original; darts; result. Because no backpropagation is used, darts are made inside the face.

can flag an increase in slope with variable $c_{abcd} \geq \max(0, z_{abcd}(x_{ab} - x_{cd}) - (y_{ab} - y_{cd}))$ for some constant z_{abcd}; if vertical ordinates are fixed, curvature changes are flagged by indicator $d_{abcdef} \geq \max(0, z_{abcd}(x_{ab} - x_{cd}) - (x_{ef} - x_{cd}))$. Each of these distortions and artifacts is then penalized in the LP objective, e.g., the cost of a shear is the increase in contrast (if any) plus a penalty for distorting salient image content.

If the LP uses only adjacency and shear variables, the constraint matrix is unimodular, therefore the optimum is integral-valued. Otherwise the solution may be fractional and can be rounded (or rendered as is). More complicated distortions, e.g., perturbations of geometric ratios, and more complicated motions, e.g., local scaling or bending of image regions, can be captured in an otherwise identical quadratic program (QP).

A key property of the LP is the "convexification" of costs. Because image textures have some periodic structure, a "raw" edit cost based purely local contrasts can have an exponential number of local optima. Instead, the LP uses upper bounds on raw costs that are convex increasing in the size of each shear or deletion. This arises naturally because the indicator variables have chained values, e.g., by construction, $a_{ijk} \Rightarrow a_{ij,k-1}$, so each edit cost incorporates the costs of smaller edits. We adjust the cost coefficients in the LP objective to obtain the tightest convex upper bound on the raw costs.

The LP is very sparse, but the more image structure we want preserved, the more indicator variables we need. This, and dependence on inherently continuous convex solvers, is computationally costly. Thus the remainder of this paper develops a fast integer algorithm for darting, in which an upper bound on the LP objective is minimized via dynamic programming on a trellis of possible image edits. One variant of this algorithm achieves the global optimum, another handles nonconvex costs; and another offers $O(1)$ integer operations per pixel, regardless of the number of pixels removed. All are directly applicable to video.

3 Fast darting by dynamic programming

To introduce the dynamic programming (DP) solution for darting, consider the problem of finding an optimal set of pixel deletions in just a single scanline. As described above,

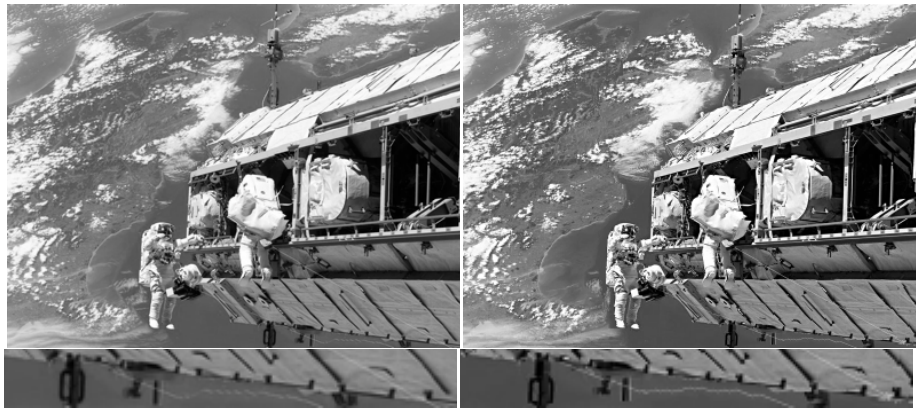

Fig. 4. Shrinking both dimensions. At left, the seam carving result. Note the broken coastline above the left astronaut, the broken tether line at bottom (zoomed below), the clipped antenna at top, and the fused continents near the antenna. At right, a darting result is free of those artifacts and generally does a better job of retaining image texture and packing it together. I.e., it keeps more land and clouds and less ocean. Intensities differ slightly between the two images because the seam carving code post-processes the image with pixel blends to conceal carving artifacts.

removing a pixel incurs two local costs that reflect the perceptual impact of bringing previously unconnected pixels together:

- An *meet cost* for bringing pixels within the scanline together, e.g., induced contrast. This can also incorporate any texture cost associated with the deletion, e.g., lost image energy, saliency, etc.
- A *shear cost* for sliding pixels from different scanlines into contact, e.g., increased contrast. This can also incorporate any geometric cost associated with the move, e.g., for bending a contour or distorting a face.

Meet costs are determined entirely by image content; shear costs depend on both image content and the set of darts in neighboring scanlines. Assume for the moment that those darts are known. Then the optimal set of edits for the current scanline can be identified as the min-cost path on an *edit trellis* having the grid structure shown in figure 5: A node in column i, row j represents the event that pixel from column $i+j-1$ in the original image will be placed in column i in the darted image, with associated shear cost vis-a-vis neighboring scanlines. This implies that $j-1$ pixels have already been deleted in previous columns. Node i,j is linked to all nodes $i+1,k$ with $k \geq j$. If $k > j$, the link has a meet cost associated with bringing pixel $i+j-1$ into adjacency with pixel $i+k$ by removing $k-j$ pixels.

Min-plus DP will find the optimal path through this trellis with time complexity linear in the total number of links. Figures 5 shows how to control the trellis girth and link bandwidth so that time complexity is linear in the number of pixels and independent of the number of deletions per row. In practice this works out to a small number of integer operations per pixel. Figures 1,3,4, & 7 were generated using this trellis and the simplest possible cost: increased contrast.

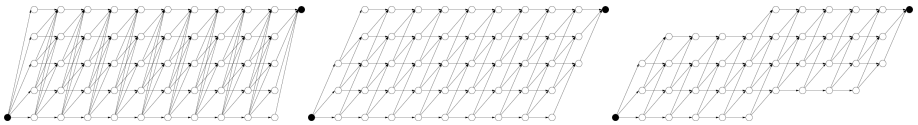

Fig. 5. LEFT: DP trellis for deleting $r = 4$ pixels from a scanline of $p + r = 14$ pixels. Each column in the trellis corresponds to a column in the result; diagonal links correspond to pixel deletions. Nodes have shear costs; diagonal links have meet costs. Evaluation takes $O(pr^2)$ time. CENTER: The trellis restricted to at most $f = 2$ pixel deletions per dart. Controlling bandwidth this way ensures strictly linear $O(prf)$ time. RIGHT: The trellis further trimmed to girth $g = 4$. This limits the concentration of darts and yields $O(pgf)$ compute time, independent of r.

3.1 Global optimality and convexity

If the meet and shear costs are convexified, the problem of finding the optimal joint darting of all scanlines is isomorphic to the LP of the previous section, and thus globally convex. To convexify a cost sequence $[c_1, c_2, \cdots]$, we use the tightest convex nondecreasing upper bound $[c'_1, c'_2, \cdots]$, with $c'_i = \max(c_i, 2c'_{i-1} - c'_{i-2})$. I.e., costs, as a function of edit size, have nondecreasing differences.

To find the optimal joint darting of all scanlines, consider a scheme in which each scanline DP is updated whenever its shear costs change due to updates in neighboring scanlines. Each update is equivalent to an optimal *integral* move within the equivalent LP polytope, along the dimensions spanned by the LP variables associated with that scanline. Such cost-reducing moves are always available because the LP constraints form a polytope with obtuse dihedral angles everywhere[2]; colloquially, it is has no corners to get stuck in. For some cost functions, the true optimal move in that subspace may be *fractional*, but because the LP constraints and objective all involve integer coefficients of bounded value, this fraction can be described with a finite (and indeed fairly small) amount of precision. This can be accommodated by expanding the edit trellis to represent fractional placements of pixels, e.g., halfway between columns. It follows immediately that iterative updating of such trellises will yield the global optimum.

Time-to-convergence depends on how far information must propagate along "stiff" structures in the image. In the next section, we show how to efficiently propagate this information *before* trellis evaluations, yielding a fast non-iterative non-fractional algorithm that produces high-quality results.

3.2 Non-iterative solution via backpropagation

Consider a fast one-pass algorithm in which one sequentially processes the scanlines from top to bottom, or from the middle scanline outwards. Each scanline's shear costs come from its just-processed neighbor. Although this clearly suboptimal scheme only propagates information outward, it works well in simple images, e.g., figure 3.

[2] Precisely, the polytope is a Cartesian product of high-dimensional parallelepipeds with every vertex having a majority of 135° angles, except for two extreme vertices that correspond to the high-cost options of cropping left and right sides of the image.

One can obtain considerably better results by augmenting the local costs with a bound on the impact that edits in the current scanline will have on costs in future scanlines. This will force DP to optimize an upper bound on the *global* cost. The main insight is that if DP deletes a pixel in the current scanline, in the next scanline it must either delete the corresponding pixel or to shear and delete a nearby pixel. Under the optimality principle, DP eschews the locally cheapest alternative only if doing so enables a greater savings elsewhere. Therefore the cheapest alternative in some fixed window gives an upper bound on the impact of a single deletion on costs in the next scanline. Applying this argument recursively from the final scanline(s) back to the current scanline gives an upper bound on how much a local pixel deletion can increase the full-image darting cost. This bound can be calculated for all pixels in an efficient min-plus accumulation that we call *backpropagation*: We augment the deletion cost of a pixel in one scanline with the minimum shear-plus-augmented-deletion cost in a small window on the next scanline.

For completeness, consider the case where $n \geq 2$ adjacent pixels are deleted. Due to the min operation, they may all have the same backpropagated cost, but clearly the future darts envisioned by that cost cannot be made more than once. Imagine making those future darts n pixels wide (DP will likely find a cheaper set of darts). On average, the raw meet costs of those darts will grow by a factor $< n$, because pixel values are bounded, while the raw shear costs will will grow by a factor $\ll n$, because the size of each shear remains constant. Therefore the (summed) backpropagated costs give, on average, an upper bound on future raw costs for adjacent deletions, and an upper bound on future convexified costs for well-separated deletions.

Backpropagated costs discourage DP from starting a dart inside of a foreground scene object (e.g., on the smooth cheeks of a face), because in some future scanline the boundary of that object will have to be sheared or partly deleted. In figure 1, this causes most darts to start in the textured clouds rather than the smooth interior of the space station.

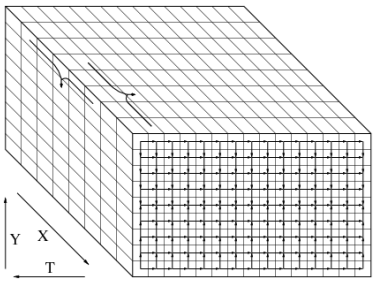

Fig. 6. An XYT video volume with min-plus cost backpropagation diagrammed for one pixel in each of the XY and XT faces. This is recursively computed over all pixels, giving the dataflow pattern shown on the YT face. Converging arrows on the YT face represent a max-plus operation.

3.3 Video retargetting

These methods are particularly suitable for spatially darting an XYT video volume. Each X scanline receives updated shear costs from its predecessor scanlines in time and in space (and thus a flying wedge of scanlines in YT can be darted in parallel). Note that temporal shear costs must be motion-compensated (and this will delay processing some scanlines in the wedge). Min-plus back-propagation is extended to video by taking the max of the min-plus results in successor pixels in both Y and (motion-compensated) T. See figure 6 for a schematic and figure 7 for an example darting of a sequence in which

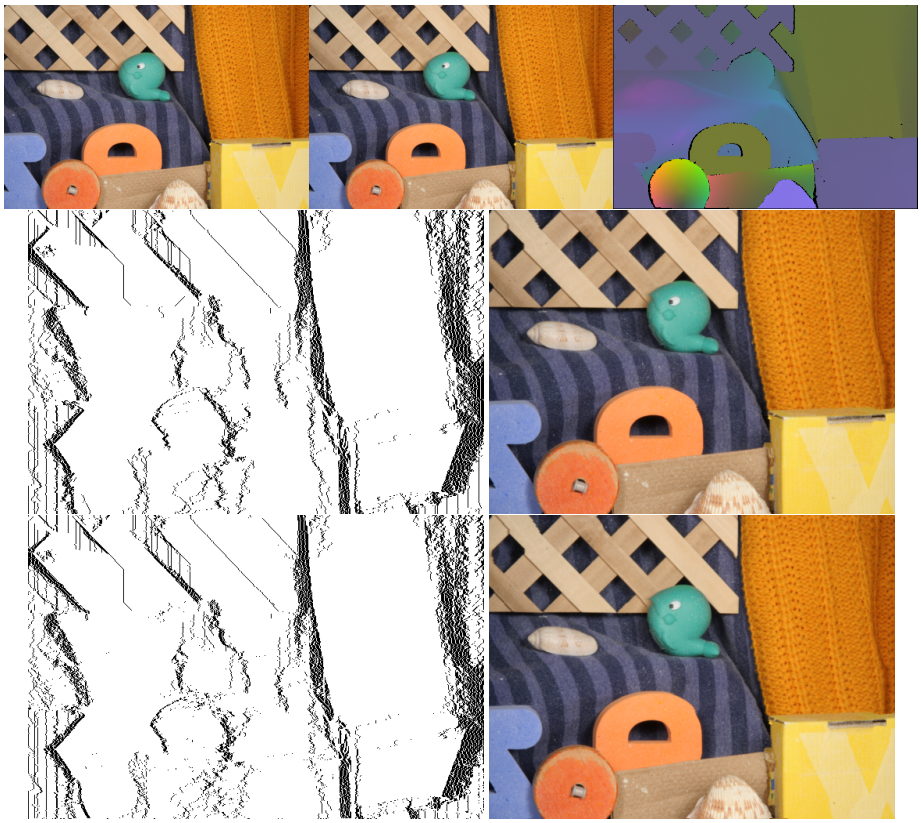

Fig. 7. A 2-frame sequence narrowed from 584 to 520 columns by darting. At top, the original frames and the forward flow, from the Middlebury flow dataset [9]. Every surface in this scene is moving in a different direction, including rotational and nonrigid motion. The largest motion is 5 pixels. The flow is dense but not complete. At left, the computed darts for the two frames are similar but not identical; they are also offset by the flow. At right the resulting images, which are consistent with the flow and animate smoothly.

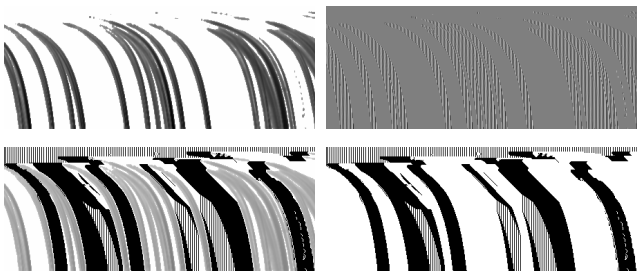

Fig. 8. Darting in time. Clockwise from upper-left: A YT video variance map; the same, striped for darting; darts marking scanlines to be removed; the same, superimposed on the variance map. The horizontal axis represents time.

every surface has a different non-simple motion. For such sequences, it is topologically impossible to carve a continuous seam without introducing motion artifacts. Since darts are discontinuous, they can track the diverging flows of the various surfaces without artifacts. This tracking is forced by the temporal shear costs, which cause DP to make the retargetted images maximally consistent with the optical flow of the original sequence.

3.4 Video packing by darting in time

One can "pack" an XYT video volume by darting along the T axis to shrink spatio-temporal gaps between moving objects. We demonstrate in figure 9 with a traffic-cam sequence. The camera is stationary and traffic moves vertically through the frame, so this particular video can be packed by removing horizontal scanlines that exhibit negligible variation in time. To do so, we constructed a YT variance map by calculating variance in a temporally sliding window at each pixel., then summing along horizontal scanlines. We negate the values in every other column of this map to produce high contrast where there is high variance. Darting the result identifies a set of low-variability video scanlines that can be removed from the video without distorting the temporal or spatial structure of high-variability scanlines. See figure 8. This example works well, but more sophisticated measures of motion will be needed for more general scenes. See [10] for a similar application which uses DP to carve seams in XT or YT.

Fig. 9. A frame from a traffic-cam video before and after temporal darting. Using the darting of the YT variance map in figure 8, horizontal scanlines from later frames are brought forward in time, packing the traffic together. This brings forward the truck in the left land and the car in the middle lane.

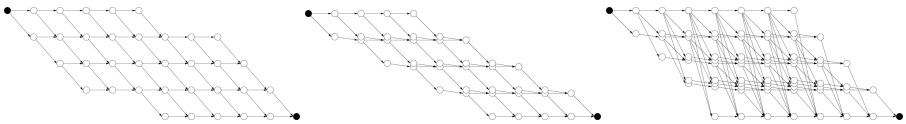

Fig. 10. Trellises for stretching an 8-pixel scanline to 12 pixels by pixel duplication. Each implements a different method for filling the gap by copying pixels. From left to right: unlimited copies (as in seam carving); at most one copy per pixel; copying pixel sequences (a rudimentary form of texture synthesis). Stacked *replica* nodes refer to the same edit, but in different DP contexts.

3.5 Stretching with a duplication trellis

One can also compute optimal *stretches* of the scanline with a of downward links that signify duplications, interpolations, or insertions of pixels. See figure 10 for three such

Fig. 11. Stretching a densely textured image using a trellis that opens gaps in regions with low color variation and fills them with texture from elsewhere in the image.

trellises, including one that supports a modest form of texture synthesis by copying pixel sequences via *replica* nodes. Downward links bear costs that discourage insertions where the image has high saliency, texture, or color variation. We had some success using backpropagated deletion costs. In figure 11 an insertion trellis with large copy offsets is used to stretch an image containing no smooth areas.

3.6 Edits within the image

A *mixed* trellis is a node-wise merge of the deletion and duplication trellises such that non-replica nodes have both upward (deletion) and downward (insertion) links. If we selectively remove trellis arcs and nodes to "pin" certain pixels to desired columns, DP uses the remaining degrees of freedom in the trellis to shrink and stretch neighboring parts of the image to minimize collateral shear damage. This provides a facility to add, move, delete, or reshape image content. In figure 12 this is done to open a gap between panelists in a conference photo and copy in texture from the first panelist. The new panelist is positioned by DP to minimize artifacts on the tabletop.

Fig. 12. A panelist is inserted and tabletop contents are adjusted accordingly via DP on a mixed trellis.

4 Implementation notes

The images in this paper were made with simple demonstration codes written in octave and C. On a 2.4GHz Intel Core 2 Duo, the compiled code exhibits a throughput of 10^5 to 10^6 pixels/second, depending on trellis girth and bandwidth. Full-frame DVD-quality video will require flow and darting at 10^7 pixels/second. Parallel evaluation of scanlines

on a GPU using the aforementioned "flying wedge" would increase throughput by a factor of roughly 10^2, thus darting at 30 f.p.s. is conceivable on current hardware. Good and fast flow will be the bottleneck, but that literature is advancing rapidly.

5 Discussion

This paper frames image retargetting and editing as convex optimization problems where the variables of interest are pixel placements, and the objective is to minimize contrast and distortion artifacts so that naive observers find it difficult to tell which is the "before" image and which is the "after" image.

We showed how to construct a sequence of local edit trellises on which dynamic programming can minimize upper bounds on the global cost with $O(1)$ per-pixel time complexity. Variations on the basic trellis design allow us to efficiently shrink images, stretch images with texture-filled gaps, spatially and temporally "pack" video, and edit content into images. Many other kinds of edits are possible.

The principal limitation of this approach is that we do not yet have an efficient way to dart in all directions in a single optimization. One possibility is to dart in alternating directions, under the guidance of a bound on the total cost.

The outputs shown in this paper were obtained using simple colorspace distances for costs. Future results will surely benefit from more sophisticated measures of contrast, motion, and saliency in the psychophysical literature. Similarly, the extension to video depends heavily on having good flow (or videos with static or non-salient backgrounds); we are exploring routes to remove this dependency.

References

1. Setlur, V., Takagi, S., Raskar, R., Gleicher, M., Gooch, B.: Automatic image retargeting. In: Proc. Mobile and Ubiquitous Multimedia. (2005)
2. Simakov, D., Caspi, Y., Shechtman, E., Irani, M.: Summarizing visual data using bidirectional similarity. In: Proc. CVPR. (2008)
3. Cho, T.S., Butman, M., Avidan, S., Freeman, W.T.: The patch transform and its applications to image editing. In: Proc. CVPR. (2008)
4. Wolf, L., Guttmann, M., Cohen-Or, D.: Non-homogeneous content-driven video-retargeting. In: Proc. ICCV. (2007)
5. Tutte, W.T.: How to draw a graph. Proc. London Mathematical Society **13**(1) (1963) 743–767
6. Avidan, S., Shamir, A.: Seam carving for content-aware image retargeting. Proc. SIGGRAPH (2007)
7. Rubinstein, M., Shamir, A., Avidan, S.: Improved seam carving for video retargeting. Proc. SIGGRAPH (2008)
8. Brand, M.: Graph cut formulation for minimizing artifacts in seam carving—archived MERL seminar whiteboard photo and follow-on email correspondence with S. Avidan. (5/2007)
9. Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. In: Proc., ICCV07. (2007)
10. Li, Z., Ishwar, P., Konrad, J.: Video condensation by ribbon carving. IEEE Transactions on Image Processing (To appear, 2009)