EUROGRAPHICS 2012 / P. Cignoni, T. Ertl
(Guest Editors)

Volume 31 (2012), Number 2

Data-Driven Object Manipulation in Images

Chen Goldberg[†1] Tao Chen[2] Fang-Lue Zhang[2] Ariel Shamir[1] Shi-Min Hu[2]

[1]The Interdisciplinary Center, Israel
[2]TNList, Department of Computer Science and Technology, Tsinghua University, China

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

Figure 1: *An example of object-based image manipulation: augmenting an object in a scene. (a) Original image with object (horse) selected in red, and augmentation (jockey) sketched in green (b) The output image containing the added jockey and other regions of the source horse (saddle, reins, shadows) (c) Alternative results (d) Internet photos used as sources.*

Abstract

We present a framework for interactively manipulating objects in a photograph using related objects obtained from internet images. Given an image, the user selects an object to modify, and provides keywords to describe it. Objects with a similar shape are retrieved and segmented from online images matching the keywords, and deformed to correspond with the selected object. By matching the candidate object and adjusting manipulation parameters, our method appropriately modifies candidate objects and composites them into the scene. Supported manipulations include transferring texture, color and shape from the matched object to the target in a seamless manner. We demonstrate the versatility of our framework using several inputs of varying complexity, for object completion, augmentation, replacement and revealing. Our results are evaluated using a user study.

1. Introduction

Modifying objects in a photograph using pixel-space techniques is a difficult and time consuming task demanding considerable user proficiency. Consider for example, the task of adding a jockey on top of a free-running horse in a photo (Figure 1). Today, one can find plenty of images containing jockeys in various poses and shapes on the internet and use them. The most natural results would be obtained by composing objects (jockeys) extracted from similar contexts (similar horses). For better composition, some regions of the source context (saddle, reins, shadows) should be pasted onto the target photograph as well. Hence, one needs to search for a photo of a racing horse which matches as closely as possible the horse in the target photograph, regarding both shape and context. Manually this is a tedious task, which may require sifting through tens of thousands of internet images. Next, desired objects need to be segmented from the candidate images, deformed to overcome differences in proportions, view angle, pose etc., and seamlessly blended to overcome differences in shape, illumination and texture.

† This work was performed while Chen Goldberg was a visiting researcher at Tsinghua University, Beijing, China

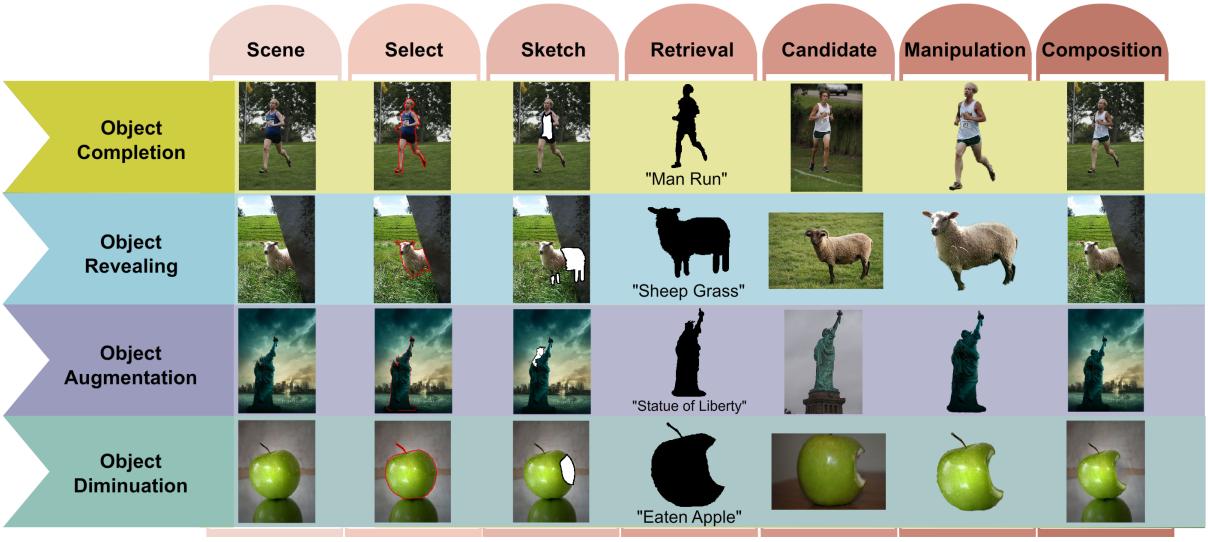


Figure 2: *Our general pipeline for different usage scenarios is similar. Note that the purposes of the user's sketches are different (top to bottom): the area to be completed, the partially occluded shape, the region to change, and the region to be removed. The composition step in the second row also includes a user provided affine transformation.*

This example illustrates that advanced image manipulation requires *object based* techniques and not just image-based.

In this paper we present a data-driven object-based image processing pipeline. Similar to previous works [HE07, CCT*09, JDA*11, HZZ11] we rely on the wealth of images on the internet, but concentrate on object-based manipulations, while reducing the amount of manual processing. User interactions are concentrated where they are essential. The user selects the "context" for the object, appropriate keywords to describe it, and in some cases a rough sketch to fill in the desired object's shape. For instance, in the above example, the horse serves as a good context for the jockey, since it determines their appearance in terms of both pose and illumination, and can be used for querying online images. Keywords are used to describe the desired manipulated object – e.g. "race horse" instead of "horse". A rough sketch of the jockey, along with the horse, provides a more salient shape for the object sought. The retrieved results can then be combined with the photo using the context to determine the desired positioning, proportions and coloring of the target object. We use a set of heuristics to find good correspondences between the objects, both externally and along the merging boundary. All these are obtained from correspondences between the two *objects* and not the whole images.

Our framework supports a variety of applications which all share the same basic processing scheme as shown in Figure 2. To manipulate a scene object in an input image, the user must first select and segment it from the background and occluders, if they exist. In addition, a rough sketch is sometimes obtained from the user. For object completion, the user sketches on the area that needs replacement. For object revealing and replacement, if the object is occluded in the scene, the user sketches its hidden shape (e.g. the sheep's rear in the second row of Figure 2), providing an estimated shape of the complete object. For the applications of augmentation or diminution, the user can sketch the details to add or to remove, respectively. Finally, the user provides keywords describing images with potentially relevant sources to be used for manipulation.

Candidate object instances are obtained from internet images matching the selected object's complete shape and the given keywords. The algorithm uses either the best retrieved candidates in offline batch-mode or presents them to the user for selection. The composition of candidate retrieved objects takes into account occlusions and object depth order. Each final scene is ranked and the top results are presented to the user in order. Consequently, the results can be refined manually by adjusting manipulation parameters, fixing automatic segmentation, and providing correspondences using a graphical user interface.

Our major contribution is the definition and implementation of an object-based data-driven approach for image manipulations. We present several applications changing the object's color, texture, shape and arrangement (position) in the image. These changes can be applied to whole objects but our framework is strongest when part of the object is occluded (object revealing), replaced (object completion), or modified (object augmentation / diminution). Finally, we present an evaluation of the results of our system with a user study.

2. Related Work

Our work conceptually follows works such as data-driven *scene completion* of Hays and Efros [HE07] and Sketch2Photo of Chen et al. [CCT*09]. In scene completion a large bank of online images is analyzed and used for retrieving scenes similar to the input image and used to seamlessly replace unknown regions. Sketch2Photo is a framework for automatically extracting objects from internet images for a given search query and composing them into a photo-montage using user sketches and keywords. As opposed to scene completion, dealing with scene objects raises new challenges: instead of retrieving a source image based on its scene descriptor we search for specific objects inside it using shape and color. Instead of matching images over slight transformations we match objects over non-rigid deformations. In addition, we need to take care of the object interactions with the scene and with other objects. In contrast to Sketch2Photo, we support many types of manipulations based on merging the retrieved internet objects with scene objects and not merely composing them onto a background canvas.

In general image completion, missing or removed areas in an image are completed mostly using known regions in the image. The problem is then treated as texture synthesis ensuring good blending [DCOY03] and continuity [CPT03, SYJS05]. The novel data-driven approach by Hays and Efros [HE07] uses other related scene images to complete scenery photographs. However, unlike scenery photos or textures, in our case most complex objects cannot be completed using texture synthesis, or by finding related scenes, thus motivating the use of object based methods.

In some special images, class-specific data is available within the scene itself. In RepFinder [CZM*10] an occluded element is completed with a user specified complete element by finding correspondences between them. However, RepFinder relies on almost identical repeated objects in the image. Our framework must be robust enough to match objects over variations in pose, texture, illumination, perspective, resolution and intra-class variance.

Finding correspondences between two objects in the presence of such variance is a hard task for computers. This relates to the problem of morphing [Wol98]. Most morphing works deal with parametric models (e.g. faces) and only few works attempt to find automatic correspondences for the general case (e.g. [GS98]). When a supervised training set is available the correspondence problem can be simplified by learning a deformable shape model and automatic landmark detection [CTCG95]. Our approach, however, is unsupervised and deals with general objects.

In recent years several works have focused on exclusively modifying specific classes of image elements using priors. Recently Johnson et al. [JDA*11] showed a method for improving realism in a CG scene by transferring color, tone and texture from photos of similar scenes. Other works [BKD*08, JMAK10, YWS*11] detect and replace full or parts of faces automatically in images. However, such methods cannot be trivially generalized to any object class, as they are specific to faces and use face-specific fiducial points for alignment and fine pose estimation.

3. Object Retrieval

The goal of the retrieval step is to find candidate objects to be used as sources for the target object manipulation. Our retrieval system relies on two object descriptors: shape and keywords. Keywords are used for reducing the number of images to a workable pool for which the shape features can be evaluated. Shape features are effective as many classes of objects have a characteristic shape (e.g., horse, cup).

Either the hard-segmentation, or the rough contour initializing the GrabCut segmentation of the scene (target) object S_t give a reliable shape contour for retrieval. In some applications (revealing and replacement) the object might either be occluded in the scene or cropped by the end of the image. Using only a partial shape significantly lowers the probability that an object with a similar partial shape can be found. In such cases, the user completes the missing part of the shape by sketching, and the complete shape outline is used for retrieval. Similarly, for any applications that alters the object's shape (augmentation and diminution), the modified shape is used for retrieval.

We use a retrieval system similar to Chen et al. [CCT*09]. First, images retrieved by the keyword search query are downloaded and preprocessed. Next, saliency is estimated on the image and then GrabCut is repeatedly used to extract the most salient part as the segmented source object using [CZM*11]. Suspected shape outliers are discarded using unsupervised clustering. For the remaining objects, shape features are calculated using the ShapeContext descriptor [BMP01], which encodes for reference points sampled on the shape, the distribution of distances to all other points. Shape context matching scores are evaluated for both the candidate shape and the flipped shape, by finding optimal matching over sampled feature points.

Most of the computation time spent on retrieval is shape independent. Only the last part compares the given shape to registered shapes. Hence, preprocessing can be performed offline while indexing each image using keywords. Using such preprocessing and given a query target shape candidate, source objects can be retrieved in a matter of several minutes. Relying on shape instead of appearance allows retrieving a large variety of objects from the same class. This is clearly desirable for applications such as object replacement and morphing, but is not optimal, e.g. for object revealing or completion where the most seamless completion of the same object is required. In this case, we could use more features to filter out undesirable results. In practice, we use user

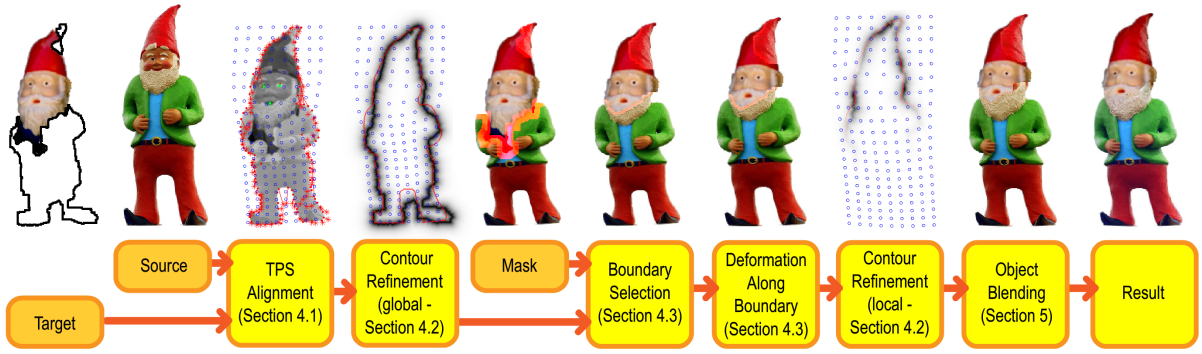


Figure 3: *The alignment deformation pipeline is common to most of the demonstrated applications combining a pair of objects along a mask. The mask is either given explicitly or implicitly, depending on the application. In this pipeline, we first optimize a global deformation by TPS alignment via key-point correspondences. Then, we apply an exterior contour alignment to refine the deformation on the exterior boundary. Next, we calculate an optimized internal boundary to minimize the visual artifacts along the boundary between the two objects. Lastly, deformation and contour alignment is applied along this internal boundary to reduce the distortion and shape discontinuity.*

selections and attempt to overcome appearance differences during the deformations and combining steps.

4. Alignment Deformations

In most cases the shape of the source and target objects do not match exactly. If only global characteristics are modified such as color or global shape, this does not pose a problem. However, when the two objects need to be combined, there is a need to deform the source object to match the target object. We focus our discussion on object completion, but other applications are similar. For object completion, the user provides a mask M on the target object destined for completion, and the system copies onto it the corresponding area of the source object. The required deformation needs to match the corresponding areas, at least along their boundaries. To overcome local differences along the mask boundary we copy a larger area from the source onto the target after alignment.

Previous works in image completion and blending consider the source and target areas to be aligned [ADA*04], or aligned up to similarity transformations [HE07, CZM*10]. In our case, due to scarcity of available data we must be able to compensate for large differences in pose, perspective and intra-class variation. To that effect we obtain a non-rigid deformation between the objects using Thin-Plate Splines (TPS) [Boo89]. Although distortions caused by similarity preserving deformations [SMW06] are less noticeable for some inputs, TPS is more robust in overcoming the extreme variations common in our data.

An overview of our deformation algorithm is summarized below and illustrated in Figure 3. In general, it involves 1. Finding a global deformation by TPS alignment, 2. Global deformation refinement by aligning exterior contour. 3. Finding a good boundary seam between source and target, and refining the deformation around this boundary:

Require: O_s,O_t - source and target patches
Require: M - user supplied mask
Require: X,Y - source and target corresponding points
Deform(O_s,O_t,M,X,Y)
$w_g \leftarrow$ estimate TPS from O_s to O_t using X,Y
$\hat{w_g} \leftarrow$ refine w_g to align contour of O_s with O_t
$B \leftarrow$ find seam minimizing $|\hat{w_g}(O_s) - O_t|$ near M
$p \leftarrow$ sample points from B
$w_l \leftarrow$ estimate TPS from $\hat{w_g}^{-1}(p)$ to p
$\hat{w_l} \leftarrow$ refine w_l to align contour of O_s with O_t along B
return $\hat{w_l}$

4.1. TPS Alignment

Let O_s be the source object and O_t the target object obtained from source (retrieved) and target (input) images accordingly using segmentations S_s and S_t. Correspondence between the objects is defined as a deformation w_g such that $w_g(O_s) \approx O_t$. Finding these correspondences is an ill-posed problem. No image feature can reliably capture similarity for all object classes. Instead, we find correspondences solely based on the objects shapes. As the source objects are retrieved based on their shapes, they are expected to have high correspondence with the target shape.

We estimate the deformation w_g using an approach similar to [BMP01] for automatic shape matching. We estimate the transformation that best aligns the two shapes, while also considering manual correspondences provided by the user. The non-rigid deformation is found by finding the correspondence and deformation simultaneously [GR96]. The sum of distances between corresponding source and target points is minimized by alternating between estimating the correspondences while keeping the transformation fixed, and finding the transformation while fixing the correspondences.

The correspondences are found using matching between

the shape context of sampled points on S_s and S_t. Bipartite Matching between these points are obtained by the Hungarian method [Kuh10]. The deformation is estimated using TPS. TPS smoothness is controlled using a regularization factor. A stiff-to-flexible approach is taken by decreasing shape control-points regularization thus allowing the TPS bending energy to increase. This regularization can be adjusted by the user. In addition, the user-set anchor points $\{X_u, Y_u\}$ indicating true correspondences $w_g(X_u) = Y_u$, are given regularization value of 0.

4.2. Exterior Contour Refinement

The resulting function w_g represents a dense correspondence between the source and target objects. However, source pixels in S_s may be mapped outside the target shape S_t. To minimize this effect, we refine the TPS by aligning the contours of both shapes. Our goal is to optimize w_g such that the average of the source contour pixels distances to nearest target contour pixels is minimized.

Inspired by Chamfer matching, which compares the shapes of two collections of curve fragments in linear cost [BTBW77], we extract the contours of the source object C_s and compute the distance transform D_t on the contours of the target object. We then iteratively refine the deformation to greedily minimize the SSD between $G = e^{-D_t/\sigma}$ (we set $\sigma = 5$) and the warped C_s (see Figure 4). To give more weight to high-proximity pixels over outliers we measure distance using a gaussian RBF. We optimize the TPS deformation directly in a similar manner to [LY05] by iteratively refining the target control-points using a gradient-descent process for minimizing:

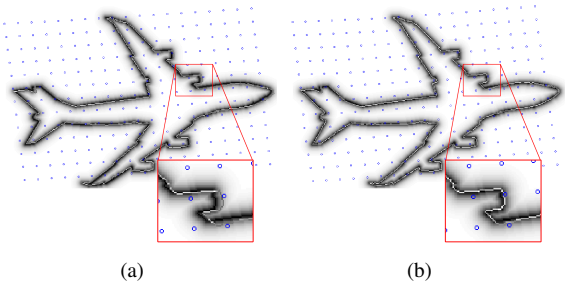
$$E(Y) = \sum_{p=(x,y)} L(p) \left\| C_s(p) - G(w(p;Y)) \right\|^2$$

where Y are the control-points, L is a spatially decreasing weight function, and w is the TPS deformation function determined by Y. The gradient-descent solution for optimizing the above equation relies solely on the gradients of G.

The control-points used for the original TPS are used for w, along with their original regularization terms. Thus, the user provided anchor points $\{X_u, Y_u\}$ are incorporated as hard constraints for stabilizing the refinement stage. After this contour refinement we obtain the final global TPS deformation $\hat{w_g}$.

4.3. Internal Boundary Refinement

The alignment $\hat{w_g}$ between the two shapes and their contours provides good global correspondence. However, along the boundary between the two objects there could still be evident discrepancies. To minimize these, we refine the internal boundary between the two objects and consequently deform the source object again to connect it smoothly along this boundary.



(a) (b)

Figure 4: *Fine Shape alignment with Direct TPS. (a) Result of TPS alignment (b) Result of refined TPS alignment*

Finding the Boundary. The user selected mask boundary M might not be optimal for connecting the source and target objects. The mask may pass through areas of semantic inconsistency between corresponding pixels including differences in texture, illumination and self-occlusions. These can cause poor visual blending but also lead to noticeable semantic mistakes. Such inconsistencies are characterized by locally high differences in brightness values between the target object O_t and the globally deformed source object $\hat{w_g}(O_s)$. Therefore, we seek to refine the boundary of M to minimize the sum of local squared brightness differences. Similar to [HE07] we find the optimal boundary B using Graphcut [BJ01] within a band of $R = 20$ pixels around M. The graph cut uses a median filtered brightness difference map between O_t and $\hat{w_g}(O_s)$.

Deforming along the Boundary. Along the consistent boundary B, we require the local deformation to map the source object pixel to the target object pixels:

$$\forall b \in B : w_l\left(\hat{w_g}^{-1}(b)\right) = b$$

where $\hat{w_g}^{-1}$ is the pseudo-inverse TPS of the global deformation. Unlike the global deformation, the locally deformed object is revealed to the user and should therefore show as little distortion as possible on the source object's side of B. To that affect we relax all point correspondence constraints from the global deformation (those that used the Shape Context features) and keep only the manual anchors: $w_l(X_u) = Y_u$. w_l regularization is set to zero for all control-points. Finally, to minimize shape discontinuity along the boundary we also employ contour refinement as described in Section 4.2 on w_l to obtain the final deformation $\hat{w_l}$ where locality is obtained by the spatial weight function L.

5. Composition

Following deformation, the source and target objects are first combined to form the compound object and then composed using alpha blending onto a base layer created from the target image. The user can also provide an affine transformation for the compound object moving it to a new location. The alpha matting for the combined object is obtained by merging the alpha mattings of both objects. Alpha matting for each

(a) (b) (c)

Figure 5: *Combining objects. (a) Optimal Blending boundary (b) Poisson blending (c) Poisson blending after local color mapping*

object is computed similar to Levin et al. [LLW06] using a trimap, where the unknown region is a four-pixels wide band along the object's boundary.

Combining Objects. We use Poisson cloning [PGB03] to stitch the source to the target object ensuring smooth gradient continuity across a seam. Since the source object is locally warped afterwards, the consistent boundary may not be the optimal seam for blending. Hence, we find a blending boundary having minimal gradient magnitudes over brightness differences [HE07] using Graph-cut. Note that for object alignment we used color information and not gradients, as gradients are useful for Poisson blending but lose a lot of information important for comparing similar objects.

Since poisson blending can be sensitive to the choice of boundary, especially when the boundary is small (Figure 5), we use color mapping of the target to source object prior to blending. Existing color transfer techniques try to match the color distribution either globally [RAGS01] or locally [TJT05]. We take advantage of the fact that our object images are roughly aligned and corresponding. We estimate the color mapping from pairs of corresponding pixels in the non-masked regions that have similar gradient magnitudes. We usually sample one percent pixels from the object to form these pairs. We learn a linear color transformation in L*a*b space by using least-squares fitting of these matching pairs. The transformation is applied to the source object image and the result is used before Poisson blending.

Object Layering. For the applications of object revealing and replacement each object may be occluded by other objects. The user-provided sketch then constitutes as the "hidden shape" of the object. Using the hidden shape we can estimate the layer order of overlapping objects in the scene. Assuming that such an order exists; if non-hidden pixels of an object overlap hidden pixels of another object then the former belongs to a higher layer than the latter. When there is no unique order, then the object overlapping with more hidden pixels is associated with the higher layer. If two objects only overlap in hidden pixels then their mutual ordering is ambiguous. A complete order is obtained using topological sorting of the given pairwise precedences.

Occlusion Estimation. An object can also be occluded by image pixels not belonging to any other object. For instance, the hand in Figure 6 occludes the mug handle, but is not explicitly recognized as an object. Here the hidden shape sketch reveals a small band of pixels belonging to an occluding area (Yellow region Figure 6c). Replacing the mug or transforming it may cause the results to move outside this area with unknown occlusions. This can cause the mug to unnaturally be exposed (Figure 6b). To prevent this we wish to know which nearby pixels are also occluding. Thus, for every object in the scene we attempt to label each pixel as either belonging to *object*, *background* or *occlusion*.

Pixels guaranteed to be labeled as *occlusion* are given by the hidden shape, and *object* pixels are given by the segmentation mask. Pixels belonging to the background, however, are completely unknown. Nevertheless we can reasonably assume that a small enough band of pixels around the *object* pixels excluding any *occlusion* pixels, belongs to the background (blue band Figure 6c). Based on this initial labeling, we predict a label (background/occluder) to each unknown pixel using a least-mean-squares approach to maximize local brightness consistency similar to Levin et al. [LLW04].

Base Layer. The base layer is obtained from the input image after removing pixels covered by scene objects and filling them using an implementation of Patch-Match [BSFG09]. To gain better results for completion, we utilized again object-based information. First, the base layer is not completed from foreground information found on objects and objects' estimated occluding regions. For instance, the road beneath the red car in Figure 8 is not completed from the yellow car because it's detected as an occlusion. Second, as image completion works well mostly for textured areas, complex (occluded) objects belonging to the base layer are completed using our object-based method instead of texture-based image completion.

6. Applications and Results

We demonstrate several applications utilizing our framework on a variety of input images. For our experiments we

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

Figure 6: *Occlusion Estimation. (a) Original image (b) The mug is replaced and previously occluded pixels remain occluding. Since the handle of new mug is bigger than previous mug some artifacts are noticeable (c) Three labels superimposed: Green: object, Blue: suspected background, Yellow: Occluded pixels (d) Composition after occlusion estimation.*

Figure 7: *Various applications by the framework. Columns left-to-right: inputs, candidate result and alternative results. Examples top-to-bottom: completion, augmentation, completion, augmentation, completion and whole-body replacement. No manual correspondences and refinements were provided.*

downloaded about 15,000 images per object from flickr.com. Caching and preprocessing the images takes several hours. Retrieval takes about 10 minutes per object (with no acceleration or parallelization). The rest of the pipeline, including deformation, blending and composition, takes at most 10 seconds per object. All experiments were performed on a PC with 2.66GHz dual core and 4GB RAM.

User Interface. We created a GUI for users experimentation (the supplementary video demonstrates this interface for two examples). The user first draws a rough approximation of the selected object and then provides the application specific sketch. Next, retrieval is performed and a candidate source object is chosen. A dialog box then appears which lets the user select an application and then displays the source, target and combined objects. The dialog allows the user to improve the result by refining the source and target object segmentation using GrabCut and specifying manual anchor points between the objects. In addition, the user is given control of the width of the consistent boundary band, which can have a significant effect on the results.

User Study. To evaluate the usability of our system and its results we conducted a small user study: eight subjects were given a short introduction to the user interface and then requested to perform four tasks. Their results were evaluated by five different subjects and compared to results of two Photoshop users, one a professional artist and the other a novice (the details of the user study can be found in the supplementary material). The results show that the novice users have quickly adapted to the system and have performed the tasks at significantly shorter times than the Photoshop users, and have also outperformed the novice Photoshop user in the evaluation scores. However it is no surprise that the professional Photoshop user has almost in all cases ranked highest in quality.

6.1. Combination-based Applications

The applications are divided into two groups: one for combining the objects along a boundary and the other for merging whole objects. The former applications are based on completion (i.e. combing objects along a mask) and have been the major focus of the work.

Object Completion. A given area on the object is completed seamlessly. In this application the user explicitly sketches the completion mask M to indicate the completion region. Here, the refined boundary is excluded from passing through M. Results can be seen in Figure 7.

Object Revealing. When partly occluded objects are translated, the occluded part that is revealed is completed seamlessly. The user sketches the hidden shape M_h of the occluded object. The framework combines the selected object with candidates along the hidden shape $M = M_h$. Results can be seen in Figure 8.

Object Augmentation/Diminution. For augmentation the user sketch is augmented to the object's shape, and for diminution the sketch is subtracted from the object's shape. In both cases the shape of the object changes, and the object is completed along the sketch (M = sketch). While the sketch itself doesn't explicitly affect inner shape details, the change of the shape along with the new semantics (i.e. search query) can change it. Consider for example, row 4 of Figure 2: the new diminished shape and the new semantics ("Eaten Apple") are meaningful enough to remove the side of the apple while adding the bite marks that accompanied it. Thus, the choice of keywords to describe the desired result here is crucial. See Figures 1 and 7 for results.

6.2. Merging-based Applications

Object Replacement. Scene objects can be replaced with other objects, either of similar object class or not. The framework is used to perform a highly regularized global warp, effectively estimating an affine deformation w_a. Using no local deformation the resulting largely undistorted source object is composed to replace the target object. If a hidden shape is provided then the composition considers occluding areas. Results are shown in Figure 7.

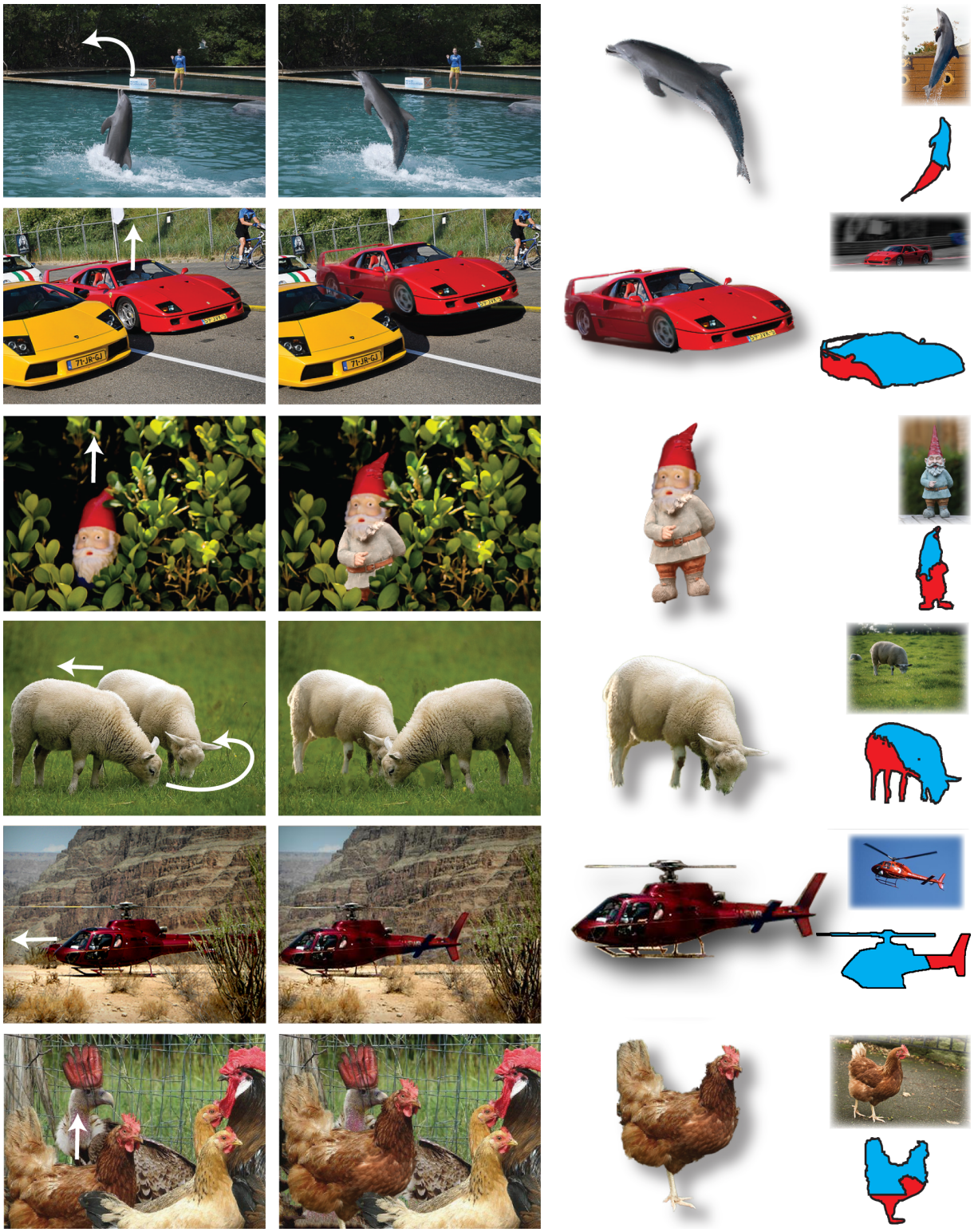


Figure 8: *Results for revealing occluded objects in a scene. Columns left-to-right: 1. Input image with arrows signifying affine transformations 2. result 3. reconstructed object and 4. shows the source image and the user segmentation (blue) with user sketch (red) superimposed. All results were processed automatically with the exception of the car example ("Ferrari f40") which used 12 manual anchor points and the "Garden Gnome" which used 3.*

Figure 9: *Left three columns show colorization. Right three columns show morphing. No manual correspondences and refinements were provided.*

(a) (b)

Figure 10: *Failures of our system. (a) Shape-Pose ambiguity (b) Exact deformation, impossible blending*

Morphing. Instead of completely replacing the object the user can adjust the shape or appearance parameters by merging the source with the target object. Morphing is traditionally obtained by warping both source and target to the same interpolated space. To obtain 100% shape warp (replace target shape with source shape) we align the shape of O_s with that of O_t using the affine warping w_a used for object Replacement. 0% shape warp is obtained using $\hat{w}_g$. To obtain any intermediate shape we interpolate between the warped sample points: $w_s^{\lambda_s}(x) = \lambda_s \cdot w_a(x) + (1-\lambda_s) \cdot \hat{w}_g(x)$.

Next, we need to warp the target object to that same space. Using the pseudo-inverse TPS of $\hat{w}_g$, we warp the target object to the source object and apply $w_t^{\lambda_s}$. The target image deformation is therefore: $w_t^{\lambda_s}(x) = w_s^{\lambda_s}\left(\hat{w}_g{}^{-1}(x)\right)$. The appearance morph is trivially obtained by linearly interpolating the warped objects' color values. Results can be seen in Figure 9.

Colorization. An object in a monochrome photo can have its color resorted based on similar image objects. The local color mapping algorithm used in Section 5 is employed directly to modify the color distribution of the target object. Examples are shown in Figure 9 (note that no color input was provided).

Image montage. A user can sketch an object to be composed into the scene. The proposed framework generalizes Sketch2Photo by allowing a user to augment an empty object over a background scene.

7. Discussion and Limitations

With the distinction between "things" and "stuff" [Ade01], our framework is designed for exclusively manipulating "things"; these are characterized by a "well-defined boundary" [ADF10], which is essential for retrieval, deformation and segmentation. State of the art methods for manipulating images do not rely on priors and instead exploit image repetitions often characterizing "Stuff" (e.g. [BSFG09]).

In addition to having "well-defined boundary" a scene object in our framework requires a name. However, often naming an instance is either too hard (e.g., an obscure phone model) or too specific for a sufficient number of internet images to be found.

In this work retrieval and correspondences focused on the object's shape. Relying on shape alone can often lead to ambiguity (Figure 10a). Better results could be obtained by utilizing more information available in the target objects such as color, inner edges, and self-similarity, or learning "data-driven uniqueness" for each query as described in [SMGE11]. Such information can retrieve higher quality query results. In terms of correspondences, the more the shape is distinct the less anchor points are needed. Most of the examples in this paper did not require anchor points at all (only 2 out of 12 in Figure 7 and Figure 8 used 3 anchor points). Some recent improvements on Poisson blending [ZT11, DT10] could help to reduce the artifacts caused by illumination change. Our framework is able to overcome differences in shape, color and stochastic textures, but has its limitations. In some cases even with a subjectively good correspondence the variance in texture makes it difficult to perform seamless blending (Figure 10b).

8. Conclusion

We presented a simple data-driven framework for objects manipulation in images. Inspired by the approach of Hays and Efros [HE07] and Chen et al. [CCT*09] our framework is the first to use object-priors to manipulate general objects. We proposed methods for deforming and seamlessly combining objects while overcoming their complex structure and appearance. To that effect we introduced the novel use of direct thin-plate spline for the task of fine contour alignment.

We designed our framework to use human interaction only for problems which are easy for humans to solve and intractable for computers: the user provides object correspon-

dences, hidden shape and segmentation, the framework deals with making finer correspondences, retrieval, blending and composition. Overall with very little proficiency a user can perform high quality image manipulations.

Acknowledgements This work was supported by the National Basic Research Project of China (2011CB302205), the Natural Science Foundation of China (61120106007, 61103079), and the Israeli Science Foundation grant number 315/07.

References

[ADA*04] AGARWALA A., DONTCHEVA M., AGRAWALA M., DRUCKER S., COLBURN A., CURLESS B., SALESIN D., COHEN M.: Interactive digital photomontage. *ACM Transactions on Graphics 23* (August 2004), 294–302. 4

[Ade01] ADELSON E. H.: On seeing stuff: The perception of materials by humans and machines. In *Storage and Retrieval for Image and Video Databases* (2001). 9

[ADF10] ALEXE B., DESELAERS T., FERRARI V.: What is an object? In *IEEE CVPR* (2010), pp. 73–80. 9

[BJ01] BOYKOV Y., JOLLY M.-P.: Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images. In *IEEE ICCV* (2001), vol. 1, pp. 105 –112 vol.1. 5

[BKD*08] BITOUK D., KUMAR N., DHILLON S., BELHUMEUR P., NAYAR S. K.: Face swapping: automatically replacing faces in photographs. *ACM Transactions on Graphics 27* (August 2008), 39:1–39:8. 3

[BMP01] BELONGIE S., MALIK J., PUZICHA J.: Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24* (2001), 509–522. 3, 4

[Boo89] BOOKSTEIN F.: Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence 11* (1989), 567–585. 4

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D. B.: Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics 28* (July 2009), 24:1–24:11. 6, 9

[BTBW77] BARROW H. G., TENENBAUM J. M., BOLLES R. C., WOLF H. C.: Parametric correspondence and chamfer matching: two new techniques for image matching. In *Proceedings of IJCAI, Volume 2* (1977), pp. 659–663. 5

[CCT*09] CHEN T., CHENG M., TAN P., SHAMIR A., HU S.: Sketch2Photo: internet image montage. *ACM Transactions on Graphics 28*, 5 (2009), 124: 1–10. 2, 3, 9

[CPT03] CRIMINISI A., PÉREZ P., TOYAMA K.: Object removal by exemplar-based inpainting. *IEEE CVPR* (2003), 721. 3

[CTCG95] COOTES T. F., TAYLOR C. J., COOPER D. H., GRAHAM J.: Active shape models - their training and application. *Comput. Vis. Image Underst. 61* (January 1995), 38–59. 3

[CZM*10] CHENG M.-M., ZHANG F.-L., MITRA N. J., HUANG X., HU S.-M.: Repfinder: Finding approximately repeated scene elements for image editing. *ACM Transactions on Graphics 29*, 4 (2010), 83:1–8. 3, 4

[CZM*11] CHENG M.-M., ZHANG G.-X., MITRA N. J., HUANG X., HU S.-M.: Global contrast based salient region detection. In *IEEE CVPR* (2011), pp. 409–416. 3

[DCOY03] DRORI I., COHEN-OR D., YESHURUN H.: Fragment-based image completion. *ACM Transactions on Graphics 22* (July 2003), 303–312. 3

[DT10] DING M., TONG R.-F.: Content-aware copying and pasting in images. *The Visual Computer 26* (2010), 721–729. 9

[GR96] GOLD S., RANGARAJAN A.: A graduated assignment algorithm for graph matching. *IEEE Trans. Pattern Anal. Mach. Intell. 18* (April 1996), 377–388. 4

[GS98] GAO P., SEDERBERG T. W.: A work minimization approach to image morphing. *The Visual Computer 14* (1998), 390–400. 3

[HE07] HAYS J., EFROS A. A.: Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007) 26*, 3 (2007). 2, 3, 4, 5, 6, 9

[HZZ11] HUANG H., ZHANG L., ZHANG H.-C.: Arcimboldo-like collage using internet images. *ACM Transactions on Graphics 30* (2011). 2

[JDA*11] JOHNSON M. K., DALE K., AVIDAN S., PFISTER H., FREEMAN W. T., MATUSIK W.: Cg2real: Improving the realism of computer generated images using a large collection of photographs. *IEEE Transactions on Visualization and Computer Graphics 17* (2011), 1273–1285. 2, 3

[JMAK10] JOSHI N., MATUSIK W., ADELSON E. H., KRIEGMAN D. J.: Personal photo enhancement using example images. *ACM Transactions on Graphics 29* (April 2010), 12:1–12:15. 3

[Kuh10] KUHN H. W.: The hungarian method for the assignment problem. In *50 Years of Integer Programming 1958-2008*. 2010, pp. 29–47. 5

[LLW04] LEVIN A., LISCHINSKI D., WEISS Y.: Colorization using optimization. *ACM Transactions on Graphics 23* (2004), 689–694. 6

[LLW06] LEVIN A., LISCHINSKI D., WEISS Y.: A closed form solution to natural image matting. *IEEE CVPR* (2006), 61–68. 6

[LY05] LIM J., YANG M.-H.: A direct method for modeling non-rigid motion with thin plate spline. *IEEE CVPR* (2005), 1196–1202. 5

[PGB03] PÉREZ P., GANGNET M., BLAKE A.: Poisson image editing. *ACM Transactions on Graphics 22*, 3 (2003), 313–318. 6

[RAGS01] REINHARD E., ASHIKHMIN M., GOOCH B., SHIRLEY P.: Color transfer between images. *IEEE Comput. Graph. Appl. 21* (September 2001), 34–41. 6

[SMGE11] SHRIVASTAVA A., MALISIEWICZ T., GUPTA A., EFROS A. A.: Data-driven visual similarity for cross-domain image matching. *ACM Transaction of Graphics 30*, 6 (2011). 9

[SMW06] SCHAEFER S., MCPHAIL T., WARREN J.: Image deformation using moving least squares. *ACM Transactions on Graphics 25* (July 2006), 533–540. 4

[SYJS05] SUN J., YUAN L., JIA J., SHUM H.-Y.: Image completion with structure propagation. *ACM Transactions on Graphics 24* (July 2005), 861–868. 3

[TJT05] TAI Y.-W., JIA J., TANG C.-K.: Local color transfer via probabilistic segmentation by expectation-maximization. In *IEEE CVPR* (June 2005), vol. 1, pp. 747 – 754 vol. 1. 6

[Wol98] WOLBERG G.: Image morphing: a survey. *The Visual Computer 14* (1998), 360–372. 3

[YWS*11] YANG F., WANG J., SHECHTMAN E., BOURDEV L., METAXAS D.: Expression flow for 3d-aware face component transfer. *ACM Transactions on Graphics (Proc. SIGGRAPH)* (2011). 3

[ZT11] ZHANG Y., TONG R.: Environment-sensitive cloning in images. *The Visual Computer 27* (2011), 739–748. 9