Recovering Badly Exposed Objects From Digital Photos Using Internet Images

Florian M. Savoy,[a,b] Vassilios Vonikakis,[a] Stefan Winkler,[a] Sabine Süsstrunk[b]

[a] Advanced Digital Sciences Center, University of Illinois at Urbana-Champaign, Singapore

[b] School of Computer and Communication Sciences,
École Polytechnique Fédérale de Lausanne, Switzerland

ABSTRACT

In this paper we consider the problem of clipped-pixel recovery over an entire badly exposed image region, using two correctly exposed images of the scene that may be captured under different conditions. The first reference image is used to recover texture; feature points are extracted along the boundaries of both the source and reference regions, while a warping function deforms the reference region to fit inside the source. The second reference is used to recover color by replacing the mean and variance of the texture reference image with those of the color reference. A user study conducted with both modified and original images demonstrates the benefits of our method. The results show that a majority of the enhanced images look natural and are preferred to the originals.

Keywords: Clipping, enhancement, dynamic range, color transfer, texture transfer, inpainting

1. INTRODUCTION

When typical users photograph a scene, their aim is generally to capture an image that best reproduces what they perceive. Cameras can usually approximate the behavior of human vision for low dynamic range scenes. However, in high dynamic range conditions a considerable discrepancy exists between what humans perceive and what is captured by a camera. While we adapt to the varying illumination conditions, the limited dynamic range of the sensor results in badly exposed image regions with clipped pixels values.

In order to go beyond the limitations of the sensor, one can consider two strategies. Some techniques combine images of the same scene captured with varying exposure parameters.[1] However, they require an implementation in the camera itself. When such a feature is not available, post-processing algorithms must be applied to recover the clipped regions. The fact that there is often a true loss of data makes this problem very challenging.

As opposed to single image approaches that may fail when the underlying assumptions are not fulfilled,[2–6] our method uses multiple images that can be found among the abundance of publicly available digital photos on the Internet: we attempt to recover an entirely clipped image region using correctly exposed instances of it.

A typical application scenario is the following: a tourist visits a monument and takes a photograph. However, the illumination conditions or the camera settings are not appropriate, resulting in clipped pixels inside the object. Upon returning home, he searches the web for images of the monument and chooses two as follows:

1. A correctly exposed image of the same object taken at approximately the same viewpoint (*texture reference*). It will be used to recover the texture inside the object. This image can be taken with completely different camera models, settings, or illumination conditions.

2. A correctly exposed image of the object under approximately the same illumination conditions as in his original photo (*color reference*). It will be used to recover the color of the object. Once again, camera models and settings can be different. Furthermore, the viewpoint can strongly vary.

(a) Original image (b) Texture reference image (c) Color reference image (d) Enhanced image

Figure 1: Example of original, texture reference, color reference and enhanced images

Figure 1 shows an original image and its enhanced version, as well as the two chosen references.

The proposed approach works as follows: We first segment the object instances in the original and reference images. We use *GraphCut* algorithms, which we enhance based on the specifics of our problem. The extracted boundaries are the only common information between the original and the texture reference, as the background can strongly vary and the original object instance is clipped. Since a small viewpoint variation between those two images is inevitable, we compute a warping function using feature points along their respective boundaries. The aim of this function is to deform the texture reference instance of the object to fit inside the boundary of the source. Color distribution statistics of the color reference instance are then transferred to the enhanced image.

We conduct a user study with 16 pairs of original and enhanced images to evaluate the performance of our method. The participants had to choose the image they prefer and grade its naturalness. The majority of participants preferred the enhanced image in all cases except two.

The paper is organized as follows: Section 2 describes the related literature. Section 3 presents the details of the proposed method. The user study and evaluation of our approach are described in Section 4. Section 5 concludes the paper.

2. RELATED WORK

2.1 Clipped pixel recovery from a single image

In order to recover saturated areas, one can consider several paradigms. The most common one makes use of information present in non-saturated areas of the same image.

Zhang and Brainard aim at estimating a corrected value of a saturated pixel using the principles of Bayesian estimation.[2] Their method considers two properties: the sensor responses from the non-saturated channels and a multivariate normal prior that captures the correlation in the responses across all color channels. Masood et al. present a method to recover the value of a saturated pixel using a smoothness prior.[3] They assume that neighboring pixels have similar channel ratios and color values. They use quadratic models for the reconstruction.

Guo et al. compress the dynamic range of well-exposed regions while expanding the dynamic range of over-exposed regions.[4] Lightness and color are treated separately. The lightness is recovered based on a likelihood of over-exposure and the color via neighborhood propagation as well as the confidence of the original color. Wang et al. add some high dynamic range details to over- and under-exposed regions of a low dynamic range image.[5] They assume that the textures of an over- or under- exposed part also exist in another well exposed regions of the image. The missing details in those regions can thus be inpainted by transferring the texture details of the corresponding well-exposed region. The mapping between similar textures is done by user interaction.

Didyk et al. distinguish various types of saturation, namely diffuse objects, reflective objects and light sources.[6] Their method enhanced the saturated areas considering their nature. The method works also on video clips. It uses a classifier that is semi-automatic, in the sense that the user can correct the classification at a certain frame before the automatic classification of the next.

Send correspondence to F. M. Savoy, E-mail: f.savoy@adsc.com.sg.

However all these methods are rather limited, since they incur a true loss of data. They all rely on various assumptions, and their success depends on the veracity of those. In our application scenario, the pixels of an image region representing a whole object can be clipped. It is unlikely that useful information can be found in the background, whose content is completely different from the saturated region.

2.2 Image enhancement using reference images

Image hosting websites are very common these days, giving everyone access to an abundance of pictures. It is thus interesting to make use of this data for computational photography tasks. The techniques listed here use various images to enhance a photograph. While they are not specifically aimed at recovering clipped pixels, the underlying ideas inspired our approach.

Cohen and Szeliski have set a new trend in photography by taking multiple pictures of a scene.[1] The idea is to combine them into an image that better conveys the atmosphere the user was experiencing at the time of the capture. They present multiple applications related to this paradigm. What we want to capture often does not happen at the exact same time as we press the camera button. The final image can thus consist of the frame which describes the atmosphere in the best way. Another idea presented in their paper is to capture two images of the scene, one with flash and one without. The first one is used to avoid noise, while the second captures the illumination properties of the scene. Furthermore taking multiple images with varying depths of field can increase the overall sharpness. Variations of the exposure can also help to increase the dynamic range. Finally, in an application called *Group shot*, they propose to create one image combining the best occurrences of each face among all the frames.

Some papers try to transfer the color style of one image to another. HaCohen et al. perform an overall color transfer for two images.[7] They must share some common content, but can be captured under a different lighting and rendered using a different tone-mapping process. Bychkovsky et al. perform global retouching of an image using a machine learning algorithm on a large database of input/output image enhancements done by photographers.[8] Two other papers use reference images to colorize gray-scale images: Liu et al. decompose the source image into reflectance and illumination components.[9] Their method then recovers the color in the reflectance domain. Chia et al. use interactive segmentation and text labeling to propose various colorizations to the user, based on multiple reference images found on the web.[10]

Finally, some publications use reference images to perform image completion tasks. Hays and Efros fill patch holes using similar image regions found on the web.[11] Whyte et al. improve this technique with a homography-based geometric registration step as well as a global affine transformation on the pixel values to match the photometric properties of the source image.[12] Garg et al. use *PCA*-based methods to retrieve a low-rank representation of the scene, which can be used for image reconstruction, occluder-removal, or field-of-view expansion.[13]

2.3 Clipped pixel recovery using reference images

In order to go beyond the limitations of existing methods for clipped pixel recovery (discussed in Section 2.1), we explore a novel approach using reference pictures taken under disparate conditions. Its aim is to apply the general idea discussed in Section 2.2 to this specific problem.

To our knowledge, only two papers relying on a similar idea have been published before. Zhang et al. reconstruct a 3D model of the background of a scene using a multitude of photographs taken at that location.[14] The model can then be used for various other tasks, such as photometric enhancement, field of view expansion, annotation, foreground segmentation as well as 2D-to-3D conversion. Their method focuses on enhancing the background of an image, while ours applies to an object in the foreground. Joshi et al. on the other hand present a method to enhance a face image using good quality photographs of the person in question.[15] They only consider face images, while our work is more general.

3. DESCRIPTION OF THE METHOD

Our proposed method can be divided into three steps: segmentation, texture transfer, and color transfer. This section contains a detailed explanation of each step. Figure 2 gives an overview of the process.

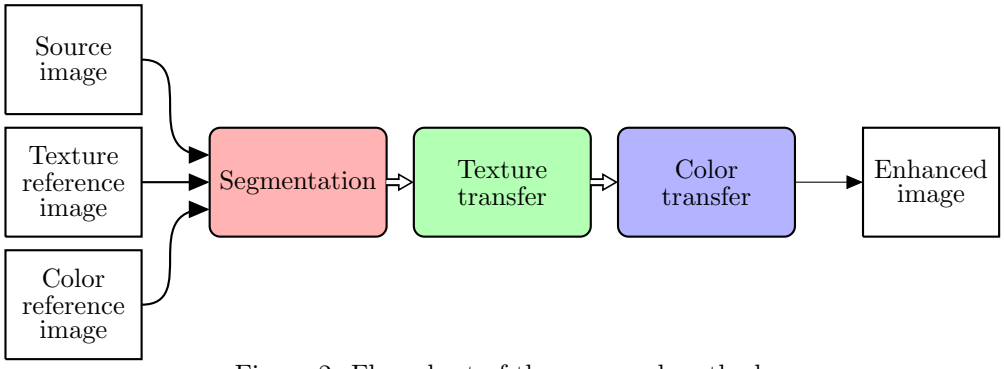


Figure 2: Flow chart of the proposed method.

3.1 Segmentation

A foreground/background segmentation is required in both source and reference images to extract the object to be recovered. The object of the source image is assumed to be clipped (under-/overexposed), which means that the technique can be simplified to selecting the correct white or black patch. Assuming that the segmentation in the source image is done, localizing the same object in the reference images can be assisted by using the shape of the source object as a prior.

3.1.1 Segmentation of the object in the source image

We use a *GraphCut* algorithm to perform the segmentations. This algorithm requires initial labels for all image pixels into foreground, background, and ambiguous cases. In order to automatically set the input labels, we first apply a low pass filter to the image, followed by thresholding for under- and over-exposed regions. We assume that the largest of those regions will be part of the clipped object and label it as foreground. We set the initial labels of the rest of the pixels as background, except those close to the foreground area. A graphical user interface has been implemented, allowing the user to modify the initial labels if the segmentation is not accurate enough, or if the first automatic estimation of foreground is wrong.

GraphCut uses Gaussian Mixture Models (GMM) to model the color of both foreground and background regions. In order to estimate a good number of components, we consider a subset containing 1% of the pixels of the region. We model them by expectation-minimization with a number of components varying from 1 to 10. The number leading to the GMM whose Bayesian information criterion is the smallest is considered as optimal. We then estimate a more accurate model using 10% of the pixels. The restriction to a subset of the pixels significantly increases the speed of the application without affecting the precision.

We then follow the work of Boykov and Jolly[16] to implement the segmentation. An overview of the algorithm is described here. For a more detailed explanation, refer to the related literature.[16,17] We consider the following energy minimization equation:

$$\min_{labels} \sum_{p \in \text{pixels of } I} \left(\mathrm{C}(p, l_p) + \lambda \cdot \sum_{q \in \text{neighbors of } p} \mathrm{S}(p, q, l_p, l_q) \right), \tag{1}$$

where I is the image, l_p, $l_q \in \{\text{foreground}, \text{background}\}$ are the labels of pixel p and its neighbor q; λ is a constant set to 1000.

$C(p, l_p)$ is a data cost term, which is computed as follows:

$$\mathrm{C}(p, l_p) = \begin{cases} -\log(f_{bg}(p)) & \text{if } l_p = \text{background} \\ -\log(f_{fg}(p)) & \text{if } l_p = \text{foreground} \end{cases}$$

where $f_{bg}(p)$ is the probability that a pixel p belongs to the background. It is set to 0 if p has an initial foreground label, or to 1 for background. If p has no initial label, it is set to the result of the background *GMM* probability distribution for the color values of p. The reverse applies for $f_{fg}(p)$.

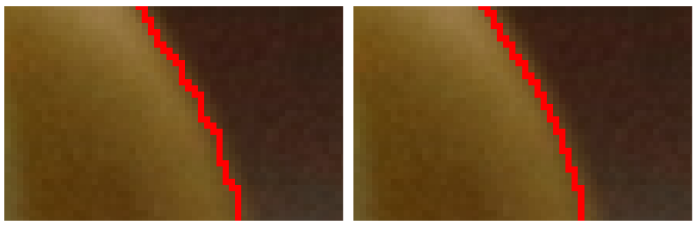
$S(p, q, l_p, l_q)$ is a smoothing term that penalizes transitions from one label to the other. It decreases exponentially with the color difference between p and q:

$$\mathrm{S}(p, q, l_p, l_q) = \begin{cases} 0 & \text{if } l_p = l_q \\ \exp\left(-\beta \left\| x_p - x_q \right\|^2\right) & \text{if } l_p \neq l_q \end{cases}$$

where β is a constant set to 50.

In order to compute a good approximation of the solution of this NP-complete equation, the image is considered as a graph, with each node representing a pixel. Furthermore, weights on the edges between each node are set to the smoothing term values (assuming different labels for every pixel). Two more nodes representing the foreground and background seeds are added. Both connect every pixel with weights equal to the data cost term with the implied label. The solution of Eq. 1 can then be approximated by computing a minimum cut between both seeds.

The result of this algorithm often leads to serrated results when edges are blurry and spread among several pixels in the image (see Figure 3). These translate into less accurate results during the inpainting process. Therefore, a smoothing step is needed, for which we use local regression lines, as proposed by Cleveland.[18] For each point on the contour line, a second-order curve is fitted using this point and the N previous and next ones (N is the mean of the width and the height of the image divided by 150; if $N < 5$, we set $N = 5$). The new coordinates are defined as the projection of the point onto the curve. In order to give less importance to the points far away from the point of interest, weighted orthogonal least squares are considered, with exponentially decreasing weights.



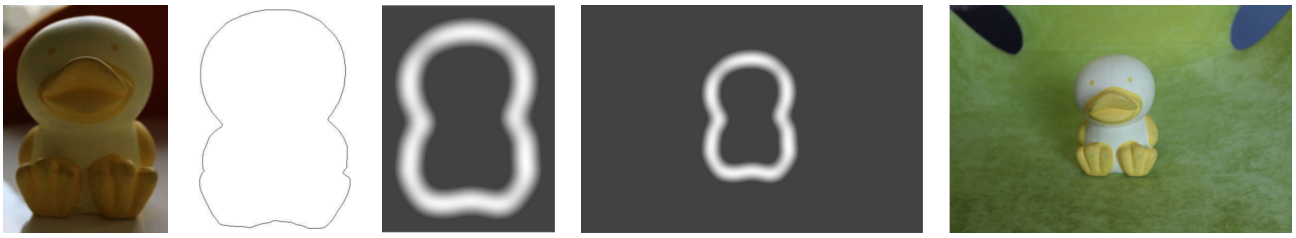
(a) (b)

Figure 3: Segmentation results without (a) and with (b) smoothed borders.

3.1.2 Segmentation of the object in the reference images

We use a modified version of the *GraphCut* algorithm for the segmentation in the reference images. It incorporates a prior reflecting the shape of the source object in the energy minimization equation. The core of the method is based on the work of Freedman and Zhang.[19]

The shape prior is a probability mask of the same size as a reference image, indicating the probability that a pixel belongs to the object boundary. The main issue is to determine the optimal position and scale for the source shape. To address this problem, we ask the user to draw a rectangle inside the reference object. A first *GraphCut* segmentation is then performed using the rectangle as a foreground seed and the region of the image which is not near this rectangle as a background seed. The resulting mask is then used to align the shape prior on the reference image considering the centers of mass and the diagonal lengths. Furthermore, we might encounter some variations of the shape as the viewpoint will be slightly different. In order to crate a mask that takes into account this variation, we give a high probability to the pixels at the border locations. A Gaussian filter is then applied to produce probabilities that exponentially decrease with the distance to the original shape. We call this resulting mask $f_{shape}(p)$. Figure 4 illustrates the process.



(a) Source image (b) Object shape (c) Probability mask (d) Positioning and scaling (e) Reference image

Figure 4: Overview of the creation of the shape prior. (b) represents the shape of the object in (a), which is Gaussian filtered to obtain (c). This probability mask is then positioned and scaled to get (d) and matches the boundaries of the reference instance of the object in (e).

While our positioning and scaling algorithm is different from the one proposed by Freedman and Zhang, the remaining steps of the process follow their paper exactly,[19] as summarized here. The energy minimization equation (1) is modified as follows:

$$\min_{labels} \sum_{p \in \text{pixels of I}} \left(\mathrm{C}(p, l_p)) + \lambda_1 \cdot \sum_{q \in \text{neighbors of } p} \left[\mathrm{S}(p, q, l_p, l_q))) + \underbrace{\lambda_2 \cdot \mathrm{P}\,(p, q, l_p, l_q)}_{\text{new term}} \right] \right) \tag{2}$$

where λ_1 and λ_2 are constants, set to 1000 and 0.3 respectively. In order to still be able to translate the equation into a graph problem, its structure should not be modified. This is why the shape prior mask is incorporated in the same way as the smoothing term; it is simply added in the weights between pixel nodes.

$\mathrm{P}(p, q, l_p, l_q)$ is the shape prior:

$$\mathrm{P}(p, q, l_p, l_q) = \begin{cases} 0 & \text{if } l_p = l_q \\ f_{shape}(p) + f_{shape}(q) & \text{if } l_p \neq l_q \end{cases}$$

Eq. (2) is then minimized as before. Smoothing by local regression lines is also applied.

This part of the application is computationally expensive, especially when the resolution of the images is high. It is also the only part which requires some user intervention. The user has to define more accurate seeds or to adjust the position and scale of the shape prior if the result of the algorithm is inaccurate. A more general discussion about those issues can be found in Section 4.

3.2 Texture transfer

The purpose of this step is to integrate the texture of the texture reference image into the source object. We first compute interest points along the borders of both instances of the object. Using those we compute a transformation function describing the relation between the boundaries of both objects. We apply this function on the whole texture reference instance and incorporate the result into the source image using alpha blending techniques to provide natural transitions at the borders.

3.2.1 Interest points location detector

The only common information between both original and texture reference images are the boundaries of their object instances. Their backgrounds can indeed be completely different, and the foreground of the source image is clipped. Our matching process thus only relies on the shapes resulting from the segmentation of both objects. In order to extract useful information from those shapes, we have devised a new interest point detector.

A good way to describe a boundary is via its corners, which can be located with high precision. Furthermore, two boundaries of object instances taken from more or less the same viewpoints will exhibit similar corners. In order to detect corners, we consider each pixel along the boundary and place a circle on that pixel. The two

points where the boundary crosses that circle are used to compute an angle (see Figure 5). This computation is performed with varying circle radii (from 1 to 20), and the mean of the resulting angles is considered. In order to detect corners, we consider the absolute value of the difference between those angles and π. We perform a peak detection along the sequence formed by concatenating those values and only keep the peaks above a certain threshold, as illustrated in Figure 6.

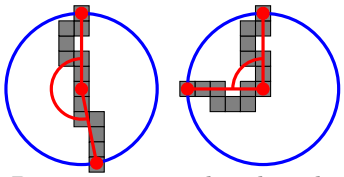
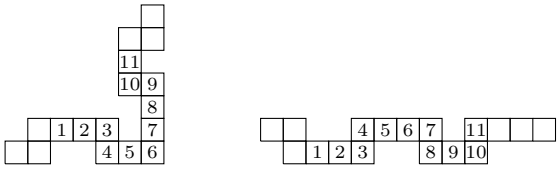
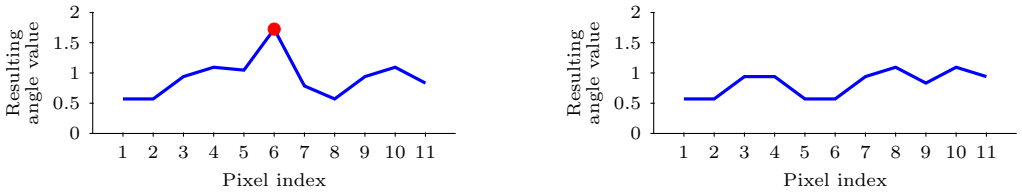


Figure 5: Determining angles along boundaries.



(a) Example 1 (with corner)

(b) Example 2 (straight)



(c) Example 1: clear peak at pixel #6.

(d) Example 2: no distinct peak.

Figure 6: Corner detection along two sample boundaries. The plots show the absolute difference between the angles (average from circle radii varying from 1 to 3) and π for the indexed pixels of both examples.

The distribution of corner points along a boundary can be highly irregular. Extended tests have shown that the matching algorithm (presented in the next section) performs better with more uniformly distributed locations. We thus add a hundred points which are equally spaced along the boundary, reflecting its smoother parts. A fixed number has been chosen in order to achieve invariance to size differences.

3.2.2 Matching the shape of the texture reference object with the source instance

We now compute the optimal deformation to apply to the boundary of the texture reference object instance to match that of the source instance. We use the Thin Plate Spline Robust Point Matching (*TPS-RPM*) algorithm introduced by Chui and Rangarajan.[20] It produces a deformation function describing the distortion to apply to a set of interest point locations to closely match them with those of a second set. This transformation function is computed considering a smoothness characteristic. We thus apply it on all the pixels of the object region in the reference. The result then fits inside the boundary of source instance of the object. A detailed and mathematical description of the algorithm can be found in their paper.[20]

Since this technique only relies on feature points, the computational requirements do not increase with image resolution. It thus provides an efficient and fully automatic way to match the two instances of the object.

3.2.3 Composition with smooth borders

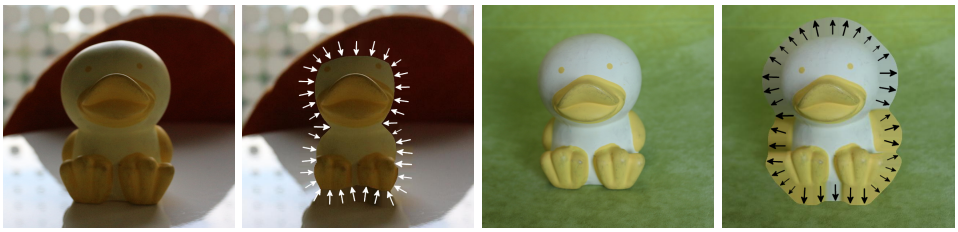
Having deformed the texture reference to fit inside the source object boundaries, we need to copy these pixels into the source image. However the result of the segmentation step does not always exactly match the true border of the object. Furthermore, the warping function (which includes a smoothness constraint) sometimes

cannot deform the pixels in a way that there is an exact match along the complete boundary. In those cases a simple replacement of the pixels following the sharp boundaries would lead to artefacts.

In order to address this problem, we replicate the pixels of the object or the background (outwards and inwards respectively) before the composition. There are two regions that we want to avoid in the final image:

- We should not see the background of the texture reference image. This could happen if the segmentation of the reference image includes areas outside the object.
- We should not see the foreground of the source image, i.e. the clipped instance of the object. This could happen if the segmentation of the source image misses areas inside object.

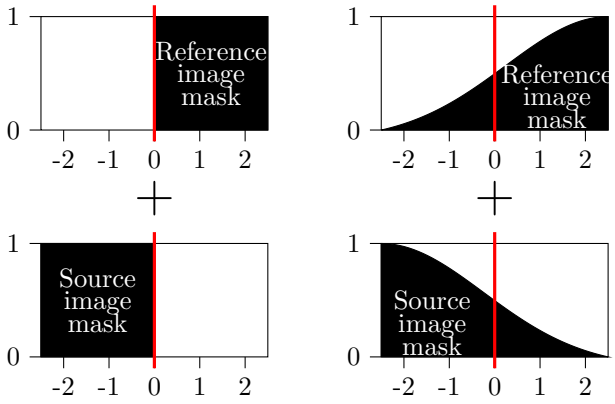
We thus mask those regions by copying pixels that are either inside the object in the texture reference image or outside in the source image, as shown in Figure 7.



(a) Source (b) Compression of source boundary (c) Reference (d) Expansion of reference boundary

Figure 7: Replication of object and background pixels.

We also introduce an alpha-blended transition at the boundaries. The fusion is implemented using binary masks. The source and the reference images are multiplied by their respective masks and then added. In order to smoothen the transition, we simply modify the masks along the boundaries such that they have the form of a one-sided Gaussian, as shown in Figure 8. The width of the transition is inversely proportional to the difference of luminosity between inner and outer parts.



(a) Without alpha-blending (b) With alpha-blending

Figure 8: Masks used to fuse the images. The red line indicates the location of the segmentation/boundary.

3.3 Color transfer

The last step of our method deals with the color characteristics. Our aim is to transfer those from the object in the color reference image to the texture recovered source image.

We follow the work of Reinhard et al.[21] First we need to determine a good color space to work in. When operations are applied on a color channel, we want the impact on the other channels to be minimal. This means that the color space should minimize the correlation between channels. Ruderman et al.[22] introduce a color space called $l\alpha\beta$, which has been created by orthogonally decorrelating the human cone response of a set of natural images in a logarithmic space. This is a reasonable solution, since a perfectly decorrelated color space is not achievable in practice. Furthermore a logarithmic scale has the advantage of bypassing the problem of gamma correction. Indeed, due to the property $\log(x^\gamma) = \gamma \cdot \log(x)$, we achieve invariance to color shifting and scaling.

The transfer of the "look and feel" of the object in the color reference image towards the instance in the texture reference is achieved by a modification of its color statistics. The images are first converted to the $l\alpha\beta$ color space using the equations described in Reinhard et al.[21] We then replace the mean and the standard deviation of the color of the target image by those computed from the color reference:

$$\begin{aligned} l'_{\text{texture}} &= \langle l_{\text{color}} \rangle + \frac{\sigma^l_{\text{color}}}{\sigma^l_{\text{texture}}} \cdot (l_{\text{texture}} - \langle l_{\text{texture}} \rangle) \\ \alpha'_{\text{texture}} &= \langle \alpha_{\text{color}} \rangle + \frac{\sigma^\alpha_{\text{color}}}{\sigma^\alpha_{\text{texture}}} \cdot (\alpha_{\text{texture}} - \langle \alpha_{\text{texture}} \rangle) \\ \beta'_{\text{texture}} &= \langle \beta_{\text{color}} \rangle + \frac{\sigma^\beta_{\text{color}}}{\sigma^\beta_{\text{texture}}} (\beta_{\text{texture}} - \langle \beta_{\text{texture}} \rangle) \end{aligned}$$

where $\langle \cdots \rangle$ represents the mean. Mean and standard deviation are computed using only the pixels inside the object. Finally we convert back to *RGB* space.

4. EVALUATION

Figures 10 and 11 show our results over 16 pairs of source/reference images.

4.1 User study

In order to evaluate the performance of our method, a common approach would be to use a visual similarity measure. Among the results presented later, some source images were artificially clipped. We could have compared their original non-clipped images with the destination images. However our goal is not to generate an image which is as close as possible to the original scene, but rather to obtain a natural and plausible result that will be appealing to users. Consequently, a similarity measure is not suitable for this purpose.

Therefore we conducted a user study. 16 pairs of source and destination images (Figures 10 and 11) were presented to 15 subjects, both experts and non-experts in photography. Male and female were equally represented, with ages varying between 22 and 75 years. They were shown original and corrected images side-by-side. Each user viewed the images on his personal screen, providing a wide range of viewing conditions. Participants were asked which one (the original or the corrected) they would prefer to use in their photo album. They were allowed to choose neither of them. They also had to rate the "naturalness" of the enhanced image (on a scale from 1 to 10).

The detailed results are shown in Table 1. We see that the majority of participants preferred the enhanced image in all cases except two (*Brandenburg* and *Liberty*). The naturalness value for *Brandenburg* is however better than the one for *Liberty*. This might be due to the less visible clipped regions of the *Brandenburg* source image compared to those of *Liberty*, resulting in a more appealing source image. The overall mean naturalness rating is 6.88 (95%-confidence interval of 1.01), which is better than the middle grade of 5.5. Several sets with poorly chosen reference images were incorporated in the study, whose low grades contribute to a lower overall result.

4.2 Discussion

As evidenced by the user study, the idea of deforming one instance to fit the boundaries of the badly exposed one shows promising results, particularly for *Duck*, *Merlion*, *Mermaid*, *Moai*, *Pig*, *Redeemer*, *Tea*, and *Tiger* pictures (see Figures 10 and 11). Some cases are more challenging (refer to Figure 9 for a magnification of the problematic areas).

	Which image would you use in your photo album?			How natural does the image look (scale of 1-10)?	
Images	Original	Enhanced	Neither	Mean	Confidence interval
Brandenburg	53%	40%	7%	6.86	1.34
Chillon	13%	73%	13%	6.53	0.95
Duck	40%	60%	0%	7.20	0.72
Liberty	27%	40%	33%	4.50	1.28
Mercury	13%	80%	7%	7.00	1.05
Merlion	0%	80%	20%	7.40	1.32
Mermaid	27%	53%	20%	6.93	1.17
Moai	7%	93%	0%	8.07	0.91
Mug	13%	67%	20%	6.53	0.81
Pig	7%	87%	7%	7.33	1.04
Pisa	27%	53%	20%	6.13	1.48
Redeemer	13%	73%	13%	7.13	0.85
Rivella	33%	53%	13%	7.27	0.91
Sirup	0%	80%	20%	7.00	0.98
Tea	0%	80%	20%	6.80	0.61
Tiger	13%	80%	7%	7.47	0.87
Mean	18%	68%	14%	6.88	1.01

Table 1: Results of the user study.

For example, *Brandenburg*, *Pisa*, and *Chillon* show that human perception is less tolerant to deformations in buildings with strong parallel and straight lines. Incorporating gradient information in the deformation algorithm could lead to better results.

The deformation can result in other geometrical problems. In *Mercury*, the cane in the destination image is broken. In *Sirup*, the rectangles and the texts are not perpendicular to the border of the object.

Occlusions and cropped objects also introduces unnatural results. In *Mercury*, flowers occlude the basement of the statue in the reference image, introducing a yellow spot in the destination image.

In *Brandenburg*, the color of the gate in the source image is the same as the two neighboring buildings. Our algorithm breaks this color continuity. This problem could be avoided with an additional color adjustment step reproducing in the resulting image the color constancy between object and background of the source image.

More generally, artifacts are also introduced when the reference images are not well chosen. For *Mug* (Figure 9h), the color reference image is much darker than the source. As a consequence, the transferred object does not go well with the background, giving the impression that the mug levitates on the table. For *Liberty* (Figure 9i), the deformation algorithm is not able to deal with the large viewpoint change (compare the distances between the torch and the spikes of the crown), resulting in an unsatisfactory output image.

The segmentation of objects with a complex shape can be laborious in terms of user interaction. This happens particularly when the background colors do not strongly differ from those of the object. This part of the method also becomes exponentially expensive with increasing resolution, as the accuracy of the segmentation is key in the success of the matching algorithm. Furthermore the result of the color transfer process depends on the complexity of the color distribution. The higher the number of patches with different colors, the worse the result. Lastly the acceptable viewpoint variation strongly depends on the shape of the object. The more complex the shape, the smaller the permitted variation.

The entire application has been implemented in *MATLAB* and was not optimized for speed. Typical running times on a Intel Xeon W3505 (2×2.53GHz) dual-core Linux-based system with images of 1089×726 (0.79 megapixels)are as follows:

(a) *Brandenburg:* Building deformation

(b) *Pisa:* Building deformation

(c) *Chillon:* Building deformation

(d) *Mercury:* Broken cane

(e) *Sirup:* Loss of perpendicularity

(f) *Mercury:* Yellow spot

(g) *Brandenburg:* Color differences

(h) *Mug:* Poor color reference

(i) *Liberty:* Poor texture reference

Figure 9: Magnification of the artefacts for some of the more challenging cases from Figures 10 and 11.

- Segmentation: the segmentation of the source image takes 18 seconds. If the result is not accurate enough, the user has to modify the seeds manually and to perform a new segmentation (again 18 seconds). The segmentation of the reference image takes 25 seconds, with an additional 17 seconds for guessing the size and location of the shape prior. Once again, the user will have to rerun the algorithm if the result is not satisfactory.
- Texture inpainting: 45 seconds, including 15 seconds spent on pixel replication (see Section 3.2.3).
- Color inpainting: less than a second.

Implementing the algorithm in *C++* with the *OpenCV* library, or other code-level optimizations could lead to substantial speedups.

5. CONCLUSIONS AND FUTURE WORK

We presented a method to enhance images where an object in the scene is clipped due to under- or over-exposure. We make use of a well-exposed image of this object taken at approximately the same viewpoint to transfer texture. We deform the latter such that the object instance fits into the boundaries of the source. Another image of the same object, taken under similar lighting conditions, is used to recover its color by transferring the color statistics. A user study has shown that this approach is able to produce images which are subjectively better than the originals in the majority of cases.

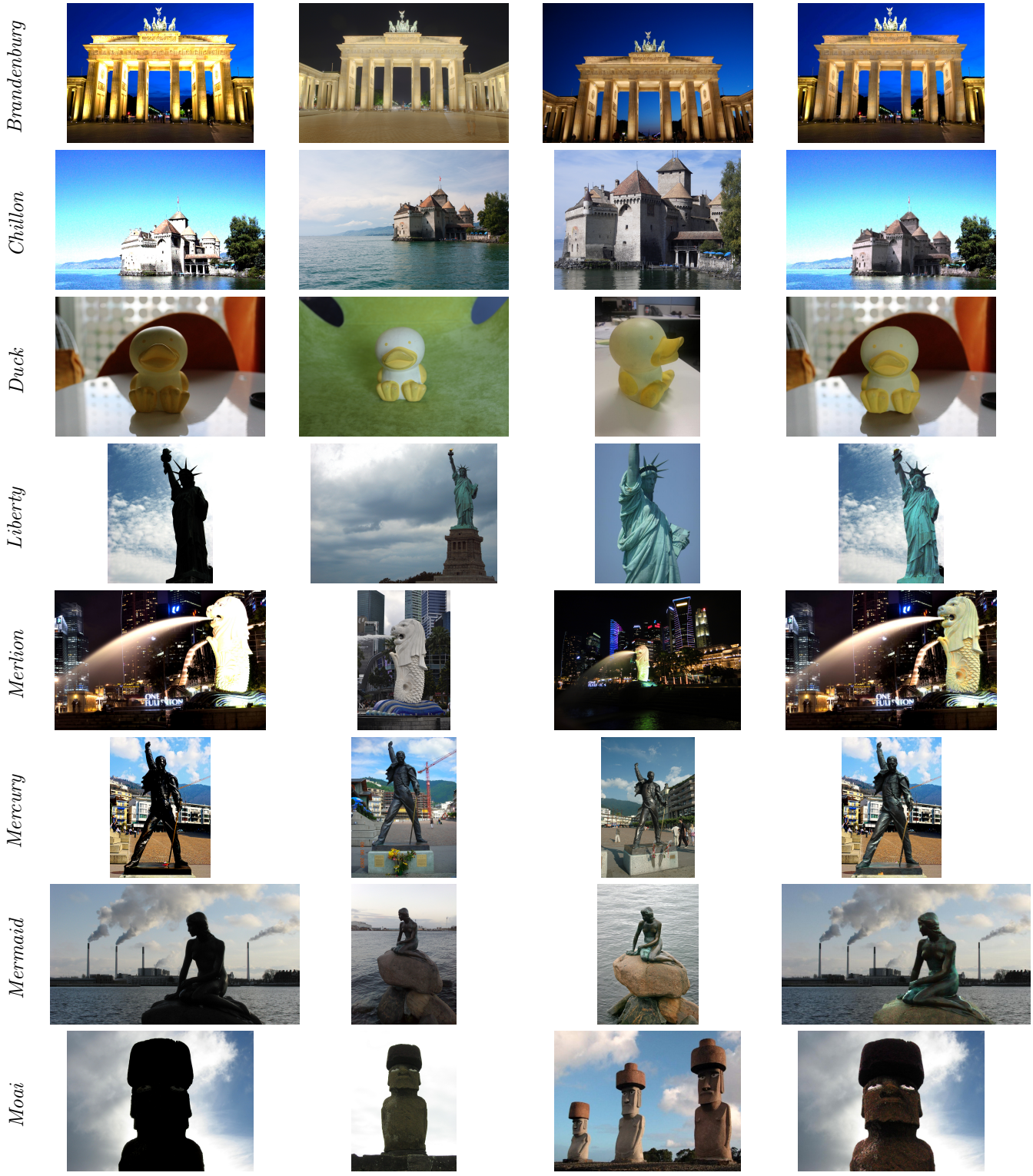
The proposed technique serves as a proof of concept that such an approach can give promising results. One of the main issues with our approach is the segmentation process, which requires interactive input from the user and can be tedious in some cases. If the algorithms involved can be made more robust, one could consider a system which uses the coefficients of the thin plate spline to automatically choose the best reference image among a set. A background color comparison could also help in automatically choosing a good color reference. To improve the segmentation process, one could make use of the whole set of images of the same object we retrieve. Under those assumptions we envisage a system which can enhance an image only by giving the name of the clipped object as a keyword, assuming it is connected to an online image hosting website.

6. ACKNOWLEDGMENTS

This study is supported by the research grant for ADSC's Human Sixth Sense Programme from Singapore's Agency for Science, Technology and Research (A*STAR).

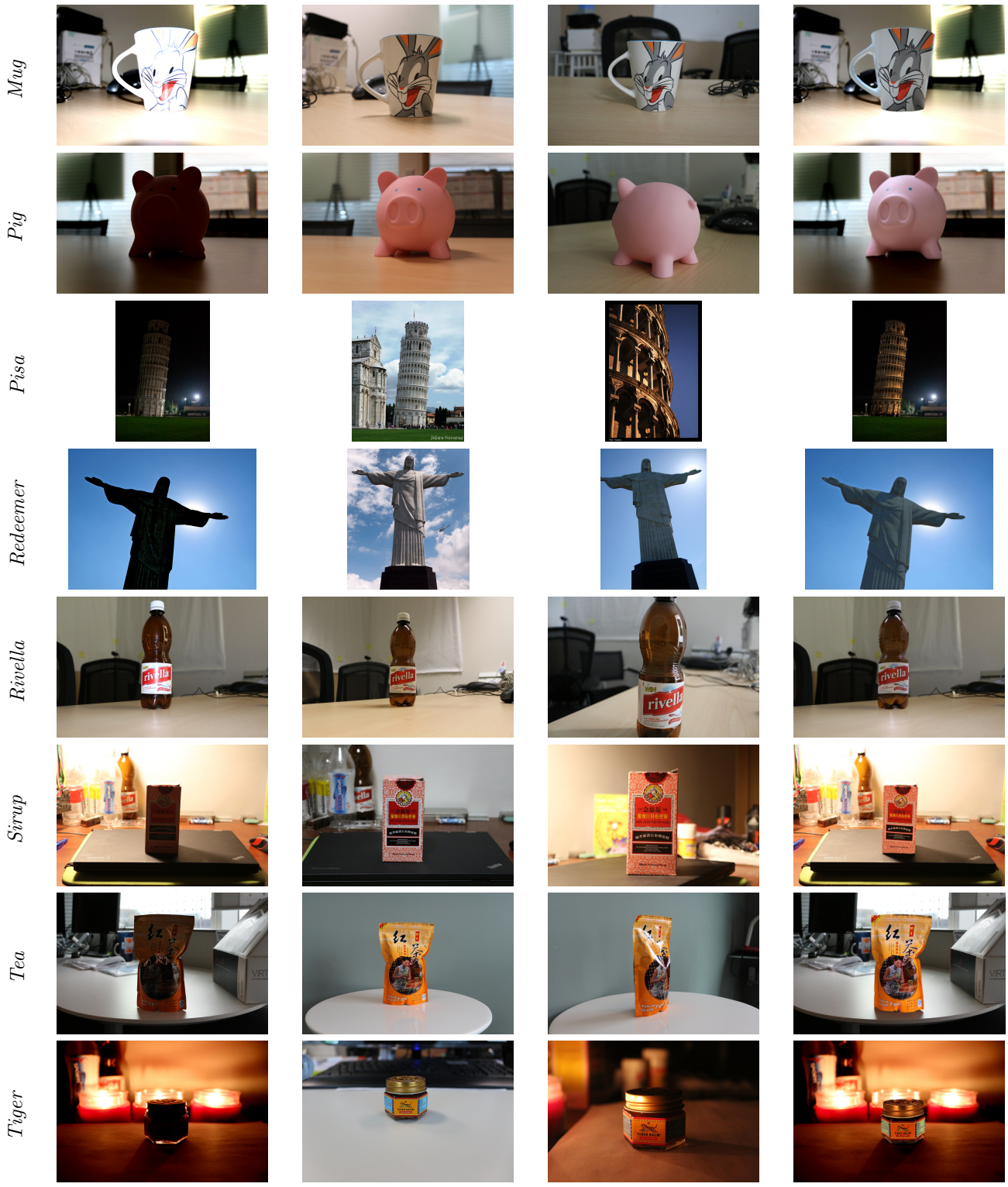
REFERENCES

[1] Cohen, M. F. and Szeliski, R., "The moment camera," *Computer* **39**(8), 40–45 (2006).
[2] Zhang, X. and Brainard, D. H., "Estimation of saturated pixel values in digital color imaging," *Journal of the Optical Society of America A* **21**, 2301–2310 (December 2004).
[3] Masood, S., Zhu, J., and Tappen, M., "Automatic correction of saturated regions in photographs using cross-channel correlation," *Computer Graphics Forum* **28**, 1861–1869 (October 2009).
[4] Guo, D., Cheng, Y., Zhuo, S., and Sim, T., "Correcting over-exposure in photographs," in [*Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*], 515–521 (June 2010).
[5] Wang, L., Wei, L.-Y., Zhou, K., Guo, B., and Shum, H.-Y., "High dynamic range image hallucination," in [*Proceedings of the 18th Eurographics Conference on Rendering Techniques*], 321–326 (2007).
[6] Didyk, P., Mantiuk, R., Hein, M., and Seidel, H.-P., "Enhancement of bright video features for HDR displays," *Computer Graphics Forum* **27**(4), 1265–1274 (2008).
[7] HaCohen, Y., Shechtman, E., Goldman, D. B., and Lischinski, D., "Non-rigid dense correspondence with applications for image enhancement," *ACM Transactions on Graphics* **30**(4), 70:1–70:9 (2011).
[8] Bychkovsky, V., Paris, S., Chan, E., and Durand, F., "Learning photographic global tonal adjustment with a database of input/output image pairs," in [*Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*], 97–104 (2011).
[9] Liu, X., Wan, L., Qu, Y., Wong, T.-T., Lin, S., Leung, C.-S., and Heng, P.-A., "Intrinsic colorization," *ACM Transactions on Graphics* **27**(5), 152 (2008).
[10] Chia, A. Y.-S., Zhuo, S., Gupta, R. K., Tai, Y.-W., Cho, S.-Y., Tan, P., and Lin, S., "Semantic colorization with Internet images," *ACM Transactions on Graphics* **30**(6), 156 (2011).
[11] Hays, J. and Efros, A. A., "Scene completion using millions of photographs," *ACM Transactions on Graphics* **26**(3), 4 (2007).
[12] Whyte, O., Sivic, J., and Zisserman, A., "Get out of my picture! Internet-based inpainting," in [*British Machine Vision Conference (BMVC)*], 1–11 (2009).
[13] Garg, R., Du, H., Seitz, S. M., and Snavely, N., "The dimensionality of scene appearance," in [*Computer Vision (ICCV), IEEE International Conference on*], 1917–1924 (2009).
[14] Zhang, C., Gao, J., Wang, O., Georgel, P., Yang, R., Davis, J., Frahm, J.-M., and Pollefeys, M., "Personal photograph enhancement using Internet photo collections," *IEEE Transactions on Visualization and Computer Graphics* **20**(2), 262–275 (2014).
[15] Joshi, N., Matusik, W., Adelson, E. H., and Kriegman, D. J., "Personal photo enhancement using example images," *ACM Transactions on Graphics* **29**(2), 1–15 (2010).
[16] Boykov, Y. Y. and Jolly, M.-P., "Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images," in [*Computer Vision (ICCV), IEEE International Conference on*], **1**, 105–112 (2001).
[17] Boykov, Y., Veksler, O., and Zabih, R., "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **23**(11), 1222–1239 (2001).
[18] Cleveland, W. S., "Robust locally weighted regression and smoothing scatterplots," *Journal of the American Statistical Association* **74**(368), 829–836 (1979).
[19] Freedman, D. and Zhang, T., "Interactive graph cut based segmentation with shape priors," in [*Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*], **1**, 755–762 (2005).
[20] Chui, H. and Rangarajan, A., "A new point matching algorithm for non-rigid registration," *Computer Vision and Image Understanding* **89**(2), 114–141 (2003).
[21] Reinhard, E., Adhikhmin, M., Gooch, B., and Shirley, P., "Color transfer between images," *IEEE Computer Graphics and Applications* **21**(5), 34–41 (2001).
[22] Ruderman, D. L., Cronin, T. W., and Chiao, C.-C., "Statistics of cone responses to natural images: Implications for visual coding," *Journal of the Optical Society of America A* **15**(8), 2036–2045 (1998).



(a) Original image (b) Texture reference (c) Color reference (d) Enhanced image

Figure 10: Source and reference images with final result (set 1).



(a) Original image (b) Texture reference (c) Color reference (d) Enhanced image

Figure 11: Source and reference images with final result (set 2).