INVERSE RENDERING FOR ARTISTIC PAINTINGS

Shinya Kitaoka[†], Tsukasa Noma[‡], Yoshifumi Kitamura[†] and Kunio Yamamoto[‡]
[†]*Graduate School of Information Science and Technology, Osaka University*
2-1 Yamadaoka, Suita, Osaka 565-0871, Japan
[‡]*Department of Artificial Intelligence, Kyushu Institute of Technology*
680-4 Kawazu, Iizuka, Fukuoka 820-8502, Japan

Keywords: Non-Photorealistic Rendering, Inverse Rendering, Relighting.

Abstract: It is difficult to apply inverse rendering to artistic paintings than photographs of real scenes because (1) shapes and shadings in paintings are physically incorrect due to artistic effects and (2) brush strokes disturb other factors. To overcome this difficulty of non-photorealistic rendering, we make some reasonable assumptions and then factorize the image into factors of shape, (color- and texture-independent) shading, object texture, and brush stroke texture. By transforming and combining these factors, we can manipulate grate paintings, such as relighting them and/or obtaining new views, and render new paintings, e.g., ones with Cézanne's shading and Renoir's brush strokes.

1 INTRODUCTION

One of the goals of Non-Photorealistic Rendering (NPR) (Gooch, 2001) is to render images in the style of great painters or find new representations. To do this, we need to analyze their paintings, resolve their pixel values into modeling, lighting, and artistic factors, and then clarify the maestros' secrets. Most previous work on artistic rendering has focused on stroke-based rendering and textural features (Drori, 2003; Hertzmann, 2003). However, the textural features of brush strokes alone cannot reproduce great painters' work. For example, Paul Cézanne's shading differs from Pierre-Auguste Renoir's. Sloan et al. captured NPR shading from paintings (Sloan, 2001), but their method needs manual fitting of surface patches, and more seriously, lighting information and texture are not separated in their model. Kulla et al. extract a shading function from an actual paint sample which is created by the user (Kulla, 2003). Sato et al. superimposed synthetic objects onto oil painting images (Sato, 2003). Their method, however, relies on 3D shape recovery by photo-modeling (Debevec, 1996), and scenes with natural objects and/or irregular shapes are hard to handle. Other studies on NPR shading did not capture shading from sample images (Gooch, 1998; Lake, 2000).

In this paper, we apply inverse rendering (Ramamoorthi, 2001b) to the estimation of factors in paintings including shading, object textures, and brush strokes. For inverse rendering for photographs, we can model the image generation process as follows (Marschner, 1997; Ramamoorthi, 2001b) using BRDF (Bidirectional Reflectance Distribution Function):

Model + Lighting + BRDF
+ Texture + Camera = Image

But a model for generating artistic paintings is more complex:

Model + Lighting + BRDF
+ Object texture + Brush stroke texture
+ Camera (painter's eye) = Image

The difficulty in inverse rendering for paintings lies in 3 problems: (1) Shapes and shadings in paintings are physically incorrect due to artistic effects, (2) the brush stroke texture is appended as 2D texture and disturbs other factors, and (3) we cannot control/observe the factors (e.g., Cézanne drew his paintings about 100 years ago.). The factorization is thus ambiguous as is, and under some assumptions, we factorize the above from a single painting. Our approach enables us to render artistic objects with different lightings, different textures, and different brush strokes from the original.

Kitaoka S., Noma T., Kitamura Y. and Yamamoto K. (2007).
INVERSE RENDERING FOR ARTISTIC PAINTINGS.
In *Proceedings of the Second International Conference on Computer Graphics Theory and Applications - GM/R*, pages 293-298
DOI: 10.5220/0002079902930298

2 ASSUMPTIONS FOR INVERSE RENDERING

Factorization, the separation of illumination and albedo, has received a lot of attention for many years (e.g., (Brainard, 1986)), and such problems are ambiguous in general (Ramamoorthi, 2001b). We thus make the following assumptions:

A1: Shape from Contour. Object shape is reconstructed by its contour using radial basis interpolation. For example, a cube shape cannot be represented with its contour because obtaining its shading information introduces cracks.

A2: Normal dependence. The lighting effect depends only on a surface normal, and the surface reflectance cannot have view-dependent effects. This assumption is similar to (Sloan, 2001).

A3: Chrominance independence of shading. The lighting effect called *shading* in this paper, is independent of chrominance channels and can be represented in greyscale. It is supposed to be sufficiently smooth for the change of normals. Such smoothness assumptions are also found in existing reports (Kimmel, 2003; Oh, 2001).

A4: Illuminance independence of object texture. Object texture has the same reflection coefficient for each chrominance channel. It is expected to be smooth compared with brush stroke textures.

A5: Brush stroke texture. Brush stroke texture is a perturbation onto a shaded image and is independent of the shape, shading and texture of a rendered object. Typically a painting's shading was created as spatial density by brush stroke. In computer graphics, this effect is used as dithering which is called the pulse-surface-area modulation when reducing the number of colors. Dithering is realized by the random dither method which is adding/subtracting pixel value. So we assume brush stroke texture is obtained as a residual term of our inverse rendering model.

3 ALGORITHM

Our inverse rendering algorithm is illustrated in Figure 1. As input, it takes a contour of an object specified by the user in a source painting image (Figure 2). The contour is specified as shown in Figure 3a and then the masked image (Figure 3b) is derived from the contour. The output is its (color- and texture-independent) shading in greyscale (Figure 1c), object texture (Figure 1d), and brush stroke texture (Figure 1e).

Our algorithm has three steps: the first step is to get the object shape and normals from the contour of the object. We assume that the shapes of target objects in paintings can be approximated by radial basis interpolation. The second step is to factorize the source image into factors (shading, object texture, and brush stroke texture). This is the core of our inverse rendering. The third step to render an artistic image using evaluated factors such as 3D object shapes, 3D shading information, and 2D brush strokes. These steps are discussed in sections 3.2-3.4.

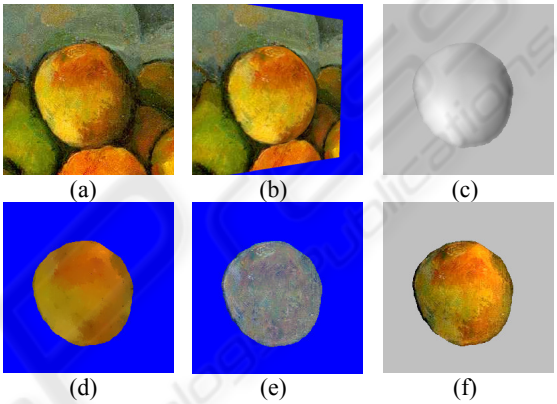


Figure 1: Inverse rendering for artistic paintings: (a) A target object in a source painting image is specified and then (b) an object shape is reproduced. Our factorization method resolves the image into: (c) (color- and texture-independent) greyscale shading (lighting) information (which is tone-mapped), (d) object texture, and (e) brush stroke texture. (f) Relighting is achieved by changing the direction of (c).

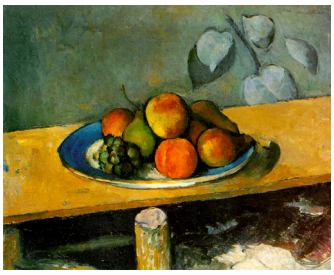


Figure 2: A still life by Paul Cézanne.

Figure 3: Contour and masked image of a target object: (a) specifying a contour, and (b) masked image of a target.

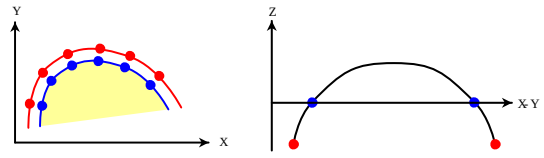


Figure 4: An object contour (blue points and curve) and an anchor outline (red points and curve).

3.1 Step 1: Shape from Contour

We generate an object shape from its contour (Igarashi, 1999). For this purpose, we use radial basis interpolation for scattered data (Powell, 1987). Our method is based on (Turk, 2002).

First, an anchor outline is generated as an expanded contour along contour normals. In Figure 4, blue points (and the curve) represent a contour specified by a user, and red points are anchor outline points.

The height (z coordinate) is treated as a real-valued function on the *xy*-plane, expressed in the form:

$$z(\mathbf{x}) = \sum_{j=1}^{n} d_j \cdot \Phi\left(\left\|\mathbf{x} - \mathbf{c}_j\right\|\right) + P(\mathbf{x})$$

where $\Phi(\cdot)$ is a radial basis function, $\mathbf{c}_j$ are the positions of contour points and anchor outline points, d_j are the weights, and $P(\mathbf{x})$ is a degree-one polynomial. We currently use $\Phi(x) = \sqrt{x}\log(1+x)$ as a basis function which was adjusted so that a hemisphere could be constructed from a circle.

The system is solved for value d_j such that $z(\mathbf{x})$ represents the given pose at the maker locations, supposing that $h_i = z(\mathbf{c}_i)$, the constraint is represented as

$$h_i = \sum_{j=1}^{n} d_j \cdot \Phi\left(\left\|\mathbf{c}_i - \mathbf{c}_j\right\|\right) + P(\mathbf{c}_i)$$

where $h_i = 0$ if $\mathbf{c}_i$ is a contour point and $h_i = -\delta z$ if $\mathbf{c}_i$ is an anchor point.

Since this equation is linear with respect to the unknowns, d_j and the coefficients of $P(\mathbf{x})$, it can be formulated as the following linear system:

$$\begin{bmatrix} \Phi_{11} & \cdots & \Phi_{1n} & 1 & c_1^x & c_1^y \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ \Phi_{n1} & \cdots & \Phi_{nn} & 1 & c_n^x & c_n^y \\ 1 & \cdots & 1 & 0 & 0 & 0 \\ c_1^x & \cdots & c_n^x & 0 & 0 & 0 \\ c_n^y & \cdots & c_n^y & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} d_1 \\ \vdots \\ d_n \\ p_0 \\ p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_n \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

where $\mathbf{c}_i = \begin{pmatrix} c_i^x & c_i^y \end{pmatrix}$, $\Phi_{ij} = \Phi\left(\left\|\mathbf{c}_i - \mathbf{c}_j\right\|\right)$,

$P(\mathbf{x}) = p_0 + p_1 x + p_2 y$.

Figure 5: An object shape generated from the contour.

We can obtain the interpolation function $z(\mathbf{x})$ by solving the above linear system and then obtain object normals from the following equation:

$$\mathbf{n} = k \cdot \begin{pmatrix} \partial \mathbf{z}(x)/\partial x & \partial \mathbf{z}(x)/\partial y & 1 \end{pmatrix},$$

where k is a normalization term.

Finally, the system makes the object shape as a height map and then normals as a normal map. Figure 5 shows an object shape generated from the contour.

3.2 Step 2: Factorization

This section describes how to factorize a source image into shading, object texture, and brush stroke texture. Let $C(p) \in R^3$ be the R, G, B values at pixel p in the source image, and let its normal $\mathbf{n}_p$ on restored object be determined as shown in section 3.2. Now we obtain greyscale shading $R(p) \in R$, object texture $T(p) \in R^3$, and brush stroke texture $B(p) \in R^3$ at pixel p. The shading $R(p)$ is dependent on the normal $\mathbf{n}_p$. From assumption A2 in Section 2, if two pixels p and q have the same normal, then $R(p) = R(q)$. Therefore, we represent shading value $R(\mathbf{n})$ as a function of normal $\mathbf{n}$.

We assume that painters create paintings based on a physical model (observed illumination) and then add artistic effects with brush strokes. The pixel value $C(p)$ at pixel p is thus modelled as

$$C(p) = \int_{\Omega} f_r(\mathbf{x}, \mathbf{e}, \mathbf{l}) L(\mathbf{x}, \mathbf{l}) |\mathbf{n} \cdot \mathbf{l}| d\mathbf{l} + B(p)$$

where $f_r(\mathbf{x}, \mathbf{e}, \mathbf{l})$ is a BRDF at $\mathbf{x}$ in direction $\mathbf{l}$ to $\mathbf{e}$, $L(\mathbf{x}, \mathbf{l})$ be incoming radiance from direction $\mathbf{l}$ at $\mathbf{x}$, and let $\mathbf{n}$ be a normal at $\mathbf{x}$.

Now, assuming that the surface is completely diffuse, we have

$$C(p) = (T(p)/\pi) \cdot \int_{\Omega} L(\mathbf{x}, \mathbf{l}) |\mathbf{n} \cdot \mathbf{l}| d\mathbf{l} + B(p) \cdot$$

From assumption A3, we assume that

$$R(\mathbf{n}_p) \sim (1/\pi) \cdot \int_{\Omega} L(\mathbf{x}, \mathbf{l}) |\mathbf{n} \cdot \mathbf{l}| d\mathbf{l} \qquad (1)$$

and

$$C(p) = R(\mathbf{n}_p) \cdot T(p) + B(p) \cdot \qquad (2)$$

3.2.1 Shading

From Equation 1 and (Ramamoorthi, 2001a), chrominance-independent shading $R(n)$ can be represented as a low-frequency signal on a sphere: where $\widetilde{R}$ is the low-frequency-part of spherical wavelets from the input image and α is a constant (we discuss how to get it in Section 3.2.2).

3.2.2 Object Texture

Without brush strokes, object texture is obtained as

$$\widetilde{T}(p) = C(p) / \widetilde{R}(\mathbf{n}_p).$$

From assumption A3, we suppose that object texture changes as intensity (Y value in YUV color space) changes. Then, the object texture at pixel p is represented by a linear model:

$$T(p) = \sum_{q \in Adj(p)} w_{pq} T(q).$$

where w_{pq} is a weighting function, whose sum is 1, $w_{pq}^{-1} \sim 10^{-6} + (Y(p) - Y(q))^2$, and $Adj(p)$ is a set of neighbouring pixels of p.

Now we formulate the problem as a stochastic optimization problem with the following evaluation function:

$$J(T) = \sum_p \left(T(p) - \sum_{q \in Adj(p)} w_{pq} T(q) \right)^2.$$

We minimize $J(T)$ using the least squares method. This $J(T)$ minimization is represented by the $\exp(-J(T))$ minimization which is the maximum likelihood estimation. Then, we set w_{pq} as

$$w_{pq}^{-1} \sim 10^{-6} + \left(\widetilde{T}^y(p) - \widetilde{T}^y(q) \right)^2.$$

The problem then becomes one of solving a linear system:

$$T(p) = \sum_{q \in Adj(p)} w_{pq} T(q).$$

We also apply constraints $T(p) = \widetilde{T}(p)$ at random points determined by a 2D Halton sequence. The obtained value is scaled so that $T(p) \le 1$. The result is the texture $T(p)$. The scaled value is α which introduced above. Finally, we obtain brush stroke texture $B(p)$ by simple subtraction (Equation 2).

3.2.3 Interpolation on Spherical Wavelets

We use spherical wavelet transform to handle shading values. We recursively divide the initial polyhedron for spherical wavelets. As the spherical wavelet bases, we use a spherical Haar basis, which is described by

$$\phi_k^{(j)} = T_k^{(j)},\ \psi_{k,m}^{(j)} = \sum_{i=0}^{3} q_{m,l} \phi_{4k+i}^{(j+1)},$$

where $\phi_k^{(j)}$ is a scaling function, $\psi_{k,m}^{(j)}$ is a wavelet function, k is a spherical coordinate, and $T_k^{(j)}$ is 1 (if it is included in the k-th triangular region of level j) or 0 (otherwise).

If a domain contains no data due to the object shape, then we approximate the value by averaging the values in adjacent triangles.

3.3 Step 3: Forward Rendering

After we have obtained greyscale shading, object texture, and brush stroke texture, the system can render a new scene by (re)using them. Shading data is stored as a texture to accelerate rendering for interactivity using paraboloid texture mapping (Heidrich, 1990). We need a single shading texture because shading data is only on the hemisphere of the viewpoint direction.

The texture coordinates of paraboloid texture mapping can be calculated as

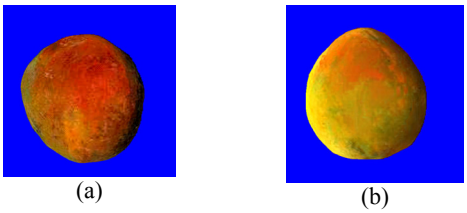
$$u = \frac{1}{2}\left(1 + \frac{n^x}{1+n^z}\right),\ v = \frac{1}{2}\left(1 + \frac{n^y}{1+n^z}\right).$$

where $\mathbf{n} = (n^x \quad n^y \quad n^z)$ is an object normal in viewing coordinates.

4 RESULTS

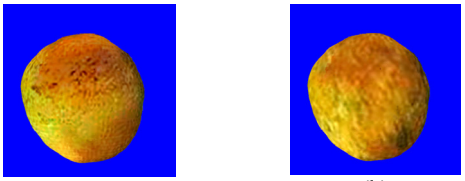
Factorization results are shown in Figure 10a-d (From left to right: original image, target object, shading (which is tone-mapped by (Reinhard, 2002)), object texture, brush stroke texture, and relighted object). The obtained shading, which is a low-frequency signal, cannot handle high-frequency highlights, but it can express low-frequency shadings of a source object. Extracted object textures are missing lighting effects and the appearance of shape. The result of brush stroke textures represents a reduced term, which is brush stroke effects (for instance, that for Renoir is different from those of the others.). Relighted results are very natural and do not have artefacts.

The results of changing the object texture are shown in Figure 6. The object textures were swapped using factorized results (Figure 1 and Figure 10a). Those results changed the object texture, but kept the shapes, shadings, and brush stroke effects. Composed results do not have any visual artefacts observed as unnatural noise.



(a) (b)

Figure 6: Examples of changing the object texture: (a) The object texture of Figure 10a applied to the object in Figure 1. (b) The object texture of Figure 1 applied to the object in Figure 10a.



(a) (b)

Figure 7: Examples of changing the brush stroke texture: (a) Chardin's brush stroke (Figure 10b) applied to Cézanne. (b) Renoir's brush stroke (Figure 10c) applied to Cézanne.

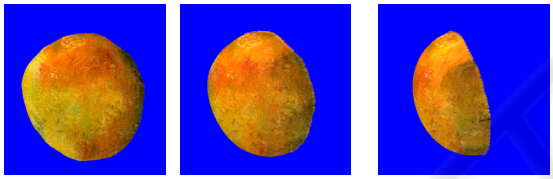


Figure 8: Example of a new view.

Figure 9: Example of relighting a still life by Paul Cézanne.

The results of changing brush stroke are shown in Figure 7. The effect of the brush stroke peculiar to each painter can be added. Comparing Figure 7 with Figure 1, we can observe Chardin's brush stroke features of Figure 10b in Figure 7a and Renoir's brush stroke features of Figure 10c in Figure 7b. Composed results do not have any visual artefacts observed as unnatural noise.

The results of a new view are shown in Figure 8. The brush stroke effects were fixed to the image-plane and the viewpoint was changed, because the effects are independent of the viewpoint. Comparing Figure 8 with Figure 5, whose brush stroke effect is not fixed, we can observe the brush stroke effects on the image plane. The results of relighting a painting are shown in Figure 9.

5 CONCLUSIONS

We have developed an approach to inverse rendering for artistic paintings. Compared with a previous work (Sloan, 2001), we have achieved great improvements: (1) contour-based object specification liberates users from manual surface sampling in (Sloan, 2001) and (2) our method factorizes the pixels of the image of a painting into reusable artistic factors: (color- and texture-independent) greyscale shading, object texture, and brush stroke texture. We reconstruct an object shape to obtain its normals by its contour. You can use any other existing approach to obtain an object's normals, however, you note that the normals which can represent the shading can usually not be obtained from a shape which is not reconstructed by spherical approximation.

Our approach, however, has some limitations and problems: The first problem is that if object texture is too biased, its greyscale shading cannot be obtained correctly. We are now investigating more sophisticated optimization. The second problem is that relighting animation is not so smooth because shading for an artistic/static painting is not necessarily appropriate for animation. Currently, we avoid smoothing so as not to decrease artistic effects. In the future, we should develop another approach so that artistic shading is compatible with animation. The third problem is that we treat brush strokes as 2D texture independent of shape and shading. For natural animation of painting-like images, inter-frame continuity with shape and shading should be considered (Meier, 1996). And as suggested in (Haeberli, 1990), the independence assumption of brush stroke texture should be reconsidered. The fourth problem is that we ignored global illumination effects like shadows. In our relighted results, shadows were not moved. For actual relighting of paintings, we must consider these effects in future.

REFERENCES

Brainard, D.H., Wandell, B.A., 1986. Analysis of the retinex theory of color vision. *J. Opt. Soc. Am. A*, 3(10):1651-1661.

Debevec, P., Taylor, C., Malik, J., 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proceedings of SIGGRAPH '96*, pages 11-20.

Drori, I., Cohen-Or, D., Yeshurun, H., 2003. Example-based style synthesis. In *Proceedings of CVPR 2003*, pages 143-150.

Gooch, A., Gooch, B., Shirley, P., Cohen, E., 1998. A nonphotorealistic lighting model for automatic

technical illustration. In *Proceedings of SIGGRAPH 1998*, pages 447-452.

Gooch, B., Gooch, A., 2001. Non-Photorealistic Rendering. A. K. Peters.

Haeberli, P., 1990. Paint by numbers: Abstract image representation. In *Proceedings of SIGGRAPH 1990*, pages 207-214.

Heidrich, W., Seidel, H.-P., 1998. View-independent environment maps. In *Proceedings of ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, pages 39-45.

Hertzmann, A., 2003. A survey of stroke-based rendering. *IEEE Computer Graphics and Applications*, 23(4):117-128.

Igarashi, T., Matsuoka, S., Tanaka, H., 1999. Teddy: A sketching interface for 3D freeform design. In *Proceedings of SIGGRAPH 1999*, pages 409-416.

Kimmel, F., Elad, M., Shaked, D., Keshet, R., Sobel, I., 2003. A variational framework for retinex. *Int'l J. Comput. Vision*, 52(1):7-23.

Kulla, C.D., Tucek, J.D., Bailey, R.J., Grimm, C.M., 2003, Using texture synthesis for Non-Photorealistic shading from paint samples. In *Proceedings of 11th Pacific Conference on Computer Graphics and Application 2003*, pages 477-481.

Lake, A., Marshall, C., Harris, M., Blackstein, M., 2000. Stylized rendering techniques for scalable real-time 3D animation. In *Proceedings of NPAR 2000*, pages 13-20.

Marschner, S.R., Greenberg, D.P., 1997. Inverse lighting for photography. In *Proceedings of IS&T/SID 5th Color Imaging Conference*.

Meier, B.J., 1996. Painterly rendering for animation. In *Proceedings of SIGGRAPH 1996*, pages 477-484.

Oh, B.M., Chen, M., Dorsey, J., and Durand, F., 2001. Image-based modeling and photo editing. In *Proceedings of SIGGRAPH 2001*, pages 433-442.

Powell, M. J. D., 1987. Radial basis functions for multivariable interpolation: A review. pages 143-167 Oxford Clarendon Press.

Ramamoorthi, R., Hanrahan, P., 2001a. On the relationship between radiance and irradiance: Determining the illumination from images of a convex lambertian object. *J. Opt. Soc. Am. A*, 18(10):2448-2459.

Ramamoorthi, R., Hanrahan, P., 2001b. A signal-processing framework for inverse rendering. In *Proceedings of SIGGRAPH 2001*, pages 117-128.

Reinhard, E., Stark, M., Shirley, P., Ferwerda, J., 2002. Photographic tone reproduction for digital images, In *Proceedings of SIGGRAPH 2002*, pages 267-276.

Sato, I., Sato, Y., Ikeuchi, K., 2003. Superimposing synthetic objects into oil paintings with artistic shadings. *IPSJ Transactions on Computer Vision and Image Media*, 44(SIG 9(CVIM 7)):132-142. (In Japanese)

Sloan, P.-P. J., Martin, W., Gooch, A., Gooch, B., 2001. The lit sphere: A model for capturing NPR shading from art. In *Proceedings of Graphics Interface 2001*, pages 143-150.

Turk, G., O'Brien, J.F., 2002. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics*, 21(4):855-873.

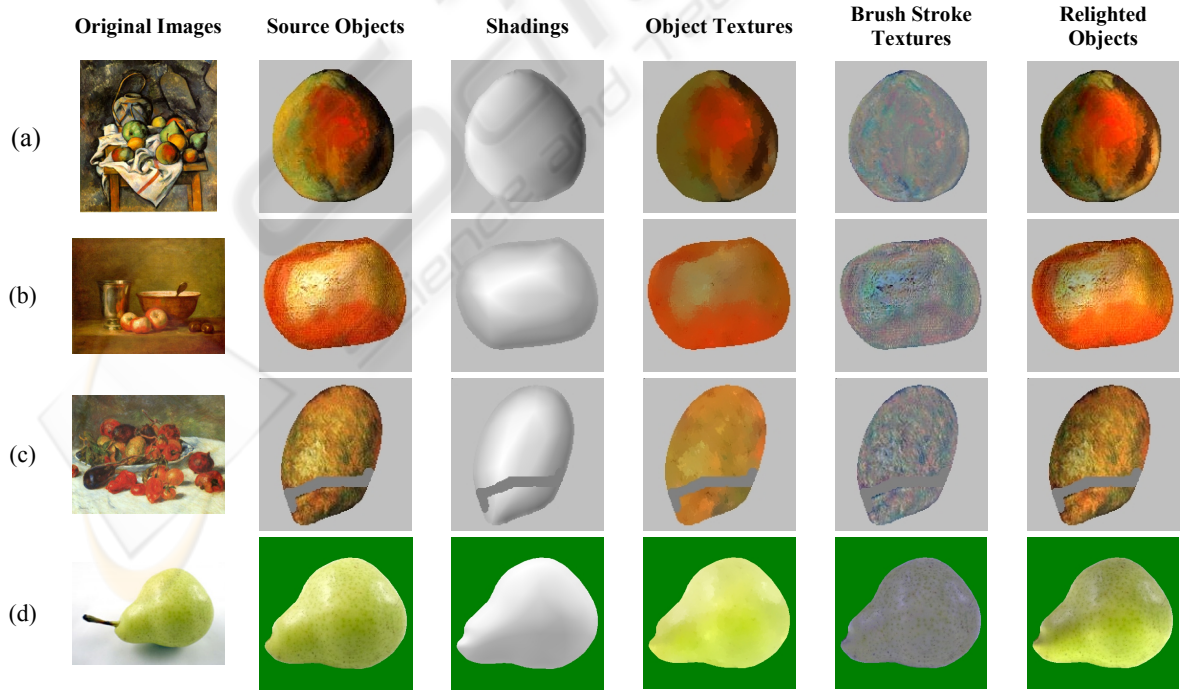


Figure 10: Factorization results and relighting: (a) Paul Cézanne, (b) Jean-Baptiste-Sim'eon Chardin, (c) Pierre-Auguste Renoir, (d) real photo.