Alignment of rendered images with photographs for testing appearance models

MORTEN HANNEMOSE[1], MADS EMIL BRIX DOEST[1], ANDREA LUONGO[1], SØREN KIMMER SCHOU GREGERSEN[1], JAKOB WILM[2], AND JEPPE REVALL FRISVAD[1,*]

[1] *Technical University of Denmark, Richard Petersens Plads, Building 321, 2800 Kongens Lyngby, Denmark*
[2] *University of Southern Denmark, Campusvej 55, 5230 Odense M, Denmark*
[*] *Corresponding author: jerf@dtu.dk*

We propose a method for direct comparison of rendered images with a corresponding photograph in order to analyze the optical properties of physical objects and test the appropriateness of appearance models. To this end, we provide a practical method for aligning a known object and a point-like light source with the configuration observed in a photograph. Our method is based on projective transformation of object edges and silhouette matching in the image plane. To improve the similarity between rendered and photographed objects, we introduce models for spatially varying roughness and a model where the distribution of light transmitted by a rough surface influences direction-dependent subsurface scattering. Our goal is to support development toward progressive refinement of appearance models through quantitative validation. © 2020 Optical Society of America. One print or electronic copy may be made for personal use only. Systematic reproduction and distribution, duplication of any material in this paper for a fee or for commercial purposes, or modifications of the content of this paper are prohibited.

This is the authors' version of the work. The definitive version is available at https://doi.org/10.1364/AO.398055

1. INTRODUCTION

Photorealistic rendering has many applications: product appearance prediction, digital prototyping, inverse rendering to acquire optical properties, 3D soft proofing, etc. In most of these applications, it is important to validate the photorealism of the employed rendering technique. In graphics, side-by-side visual comparison of rendered and photographed images has traditionally been the validation method of choice. Phong [1], for example, qualitatively compared a rendered sphere with a photographed sphere as a final evaluation of his shading and lighting models. Similarly, the Cornell box [2, 3] was presented as a test scene for qualitative comparison of photographs and rendered images. Rushmeier [4] was seemingly the first to discuss quantitative comparison of photographed and rendered images, and Pattanaik et al. [5] then presented a difference image for rendering versus photograph of a version of the Cornell box. Differences in scene geometry and the view-light configuration tend to be the main difficulty in setting up such pixel-by-pixel comparisons [4, 6].

Alignment of rendered and photographed images has reached good precision in controlled setups for geometry and reflectance acquisition [7]. For images captured in less controlled settings, the main difficulties are pose estimation of an object from a given computer-aided design (CAD) model and light source estimation. These are most often considered two separate problems. For pose estimation, a large dataset is usually employed to train a statistical model [8, 9]. A multitude of techniques exist for light source estimation [10, 11]. However, as we estimate the object pose, we may as well use the pose for light source estimation. Moreover, if we use the cast shadow for estimating the light position, we can use it to improve the estimate of the object pose as well.

Inverse rendering [12] enables recovery of both lighting and reflectance properties but often assumes a known object with a known pose. More recent inverse rendering techniques [13–15] allow pose estimation and deformation of object geometry too. These techniques are based on differentiable rendering, where per pixel derivatives are computed as part of the rendering. While this is a powerful approach for estimating surface displacements and spatially varying reflectance [13], it is also a gradient-based optimization based on per pixel derivatives that requires careful initialization to avoid local minima [14]. In this landscape, we missed a practical method for estimation of both object pose and light source position to enable pixel-by-pixel comparison of a photograph with a rendering. We propose such a method and find that it delivers a good starting point for validating rendering techniques, estimating optical properties, and testing appearance models. In addition, our method is useful for initialization of inverse rendering techniques.

Our outset is a photograph of a single object of known geometry that has been captured with a known camera. We assume that the object is placed on a diffuse planar surface and illuminated by a point-like light source. We let the term *point-like* refer to a small source with a uniform far-field radiant intensity distribution within the part of the scene observed by the camera. In this scene configuration, we let the user approximately initialize the orientation of the object relative to the planar surface

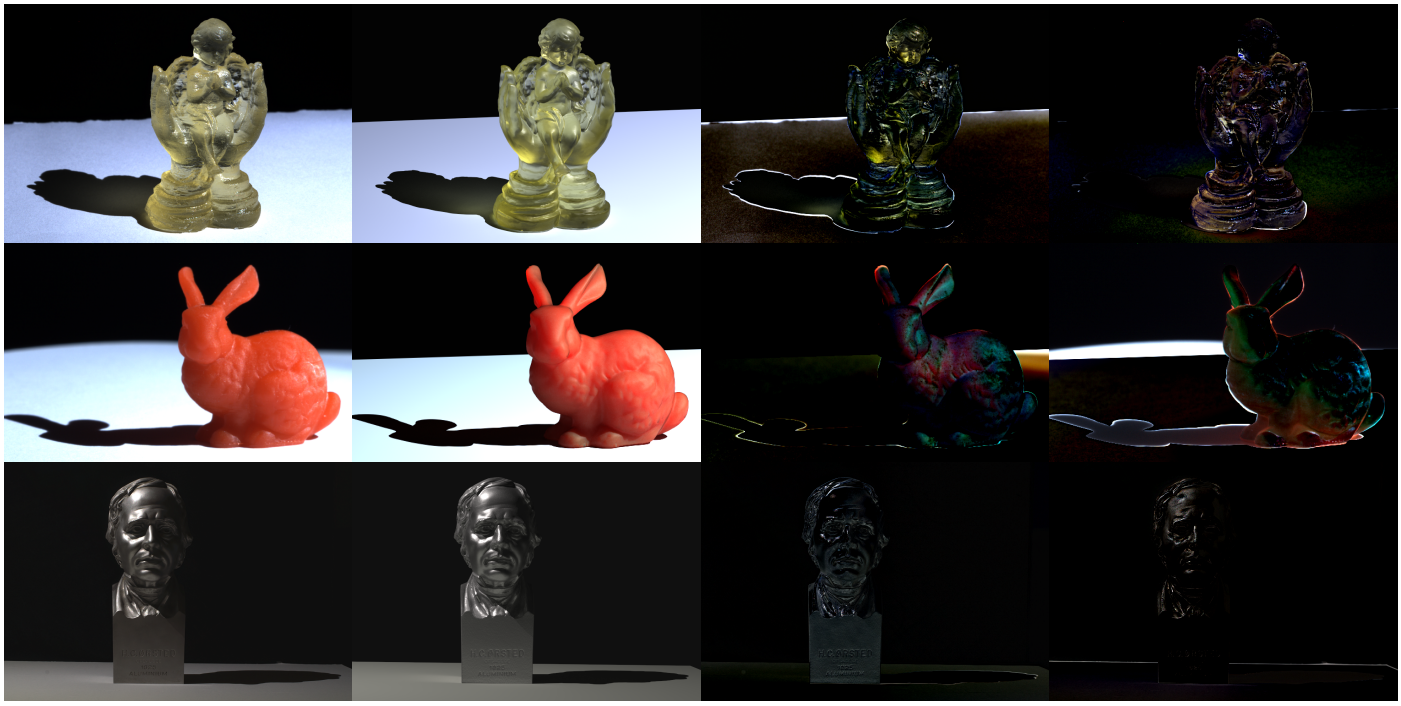


Fig. 1. Pixel-by-pixel comparison of renderings with a photograph enables a detailed investigation of the virtues and deficiencies of an appearance model. Our practical alignment technique is here used for testing different models: rough transparent (top), rough translucent (middle), and metallic (bottom). The signed difference images to the right have been scaled by a factor of 2.

(this could be done using a physics engine), or we use a camera calibration. Our method then estimates the light source position and the camera and object poses. We do this by segmenting the photograph and matching the object and the shadow silhouettes to the silhouettes of the virtual object found by projective transformation of the edges.

We exemplify our method using three scanned objects (see Fig. 1): the Stanford bunny [16], a cupped angel figurine, and an aluminium bust of H. C. Ørsted (the scientist who discovered electromagnetism and who was also the first to isolate aluminium). The Stanford bunny was scanned by Greg Turk using a technique for zippering several range scans [17], and we 3D scanned the other two objects using structured light and stereo vision [18]. We use a translucent 3D printed version of the Stanford bunny, the angel figurine was 3D printed using an almost transparent photopolymer, while we used the aluminium bust as is. This enabled us to take photographs and test appearance models for both subsurface scattering, rough refraction, and metallic rough reflection. We quantitatively test the ability of such models to match the appearance of object samples from the real world (Fig. 1), and we suggest improved models based on our findings. Notably, we for the first time integrate rough surface scattering [19] with the directional dipole model for subsurface scattering [20].

2. RELATED WORK

In many side-by-side comparisons of renderings with photographs [1–3, 6, 12, 21], alignment is done manually. This is usually a time-consuming process with an imprecise result. When a comparison is done in the context of 3D acquisition, alignment is given with good accuracy because the object geometry was acquired in a calibrated setup [22, 23]. We are however looking for an alignment method that does not require concurrent 3D scanning of the object. Differentiable rendering [13, 15, 24, 25] is an option, but the aim of such a technique is usually more than alignment. We think of our technique as an enabler for an inverse (differentiable) rendering system, which is then free to focus on estimation of parameters not related to alignment. In Sec. 6.B, we compare our object pose estimation with that of a differentiable rendering method [15] to demonstrate the advantages of our specialized technique.

Our work is related to CAD-based vision [26], where the CAD model of a 3D object is used to recognise the physical version of the object in an image. An important part of such recognition is pose estimation of the object. In a view-based approach [27, 28], multiple views of the object are used for the training of a statistical model to recognise the object and suggest an initial pose. The views can be obtained from photographs captured in a calibrated robot setup [27] or from rendered images of object edges [8, 28, 29]. After estimating an initial pose using a statistical model, the pose is typically refined using iterative shape matching [28, 30]. We combine some of these ideas. Petit et al. [29] suggest a method based on foreground/background segmentation in the case of a moving object. Our method is also based on such a segmentation but for a static object. As in the discussed previous work, we use the edges of the CAD model for pose estimation, specifically the silhouette [8], but we avoid the training of a statistical model based on a dataset with many views.

Iterative methods for pose estimation [30] are good for pose refinement but also prone to local minima if not carefully initialised. An exhaustive search for initial parameters is then needed if we want to avoid the training of a statistical model,

but such a search is infeasible for the full 6D pose of an object. An option is then to limit the dimensionality of the search space using invariants [31, 32]. Hu's moment invariants [33] are for example invariant to scale, rotation, and translation. For a 2D shape, this reduces the search space in pose estimation to two angular dimensions [31]. We use this concept for 3D shapes by applying it to the object silhouette found in the image plane.

If one is willing to generate a dataset of object silhouettes (for example) as observed across a view sphere, the pose estimation can be accomplished using shape descriptors even for cluttered scenes [34]. After image segmentation and initial pose estimation, refinement is still required using an iterative method. Several other learning-based techniques are available as well [35–38]. These all require a large dataset for training and pose refinement after estimating the initial pose. Interestingly, Tekin et al. [39] report a fast learning-based method that does not require pose refinement, but then Li et al. [40] present an iterative learning-based method for pose refinement with improvements over Tekin et al. Peng et al. [9] present an improved method inspired by Tekin and others that indeed seems not to require *a posteriori* pose refinement. This is based on an extensive dataset augmented with 20,000 synthetic images of each object. These learning-based techniques contribute robustness with respect to object detection. This is however not important for our scenes which must, in any case, be uncluttered to enable photorealistic rendering of a corresponding digital scene.

A distinctive advantage of our silhouette matching approach is that we can estimate the light source position too. In this way, we avoid the traditional calibration of a point light by observing highlights in mirroring spheres [7]. Our method employs the shadow silhouette, which we find using Blinn's projection shadows [41]. In some related work [42], the shadow silhouette was detected in an input image with depth information (RGB-D) and used for estimating the position of one or more light sources. However, since we estimate the pose of a known object together with the position of the light, we do not need the depth information. In addition, our treatment of pose and light as a joint problem enables us to refine the estimation of both.

3. ALIGNMENT METHOD

Our method is based on the following input:

- image of an object on a uniform ground plane illuminated by a point-like light source
- segmentation of the image into object, shadow, and background
- 3D model of the object
- camera intrinsics (focal length / camera constant / field of view)
- approximate rotation of the object relative to the ground plane.

Any camera can be used to capture the input image, but we need to know the field of view. If this is not known for a given camera, we can obtain it through camera calibration, but we exclude images captured with an unknown and unavailable camera. In most cases, the segmentation can be accomplished by appropriate thresholds of the input image. In harder cases, such as transparent objects, a good segmentation can be obtained through background subtraction based on one image with and one without the object.

Although we work with one light source per view, we also illuminate a static object with multiple light sources in different positions one at a time. In this case, we use the additional information to improve the object pose and light source positions in a final refinement step.

Algorithm 1. Computing a silhouette from edges of a mesh projected to a plane. Each edge exists once in each direction.

$\mathbf{p} := \mathbf{p}_0$ (the leftmost point)
$\mathbf{e}$:= edge from $\mathbf{p}$ with the largest slope
repeat
  from $\mathbf{p}$ follow $\mathbf{e}$ until next intersection, $\mathbf{p}_{\text{new}}$
  $\mathbf{e}_{\text{new}}$:= choose from edges intersecting $\mathbf{p}_{\text{new}}$
    such that angle$(\mathbf{e}_{\text{new}}, \mathbf{e})$ is minimized
  $\mathbf{p} := \mathbf{p}_{\text{new}}$, $\mathbf{e} := \mathbf{e}_{\text{new}}$
until $\mathbf{p} = \mathbf{p}_0$

To obtain object pose and light source position (in $\mathbb{R}^3$), we project the 3D model into the image plane of the camera and extract the silhouette. Our method aligns the silhouette in this plane with the corresponding silhouette in the input image. We obtain the latter from the segmentation of the input image. The silhouette is a useful representation that enables different comparisons of two silhouettes with options for being either exact or invariant to various measures such as rotation and translation, all while being differentiable.

We define a silhouette as a list of 2D point pairs each representing an edge with a direction. In analogy with a triangle mesh, we can use an indexed edge set to represent a silhouette or a set of lists of 2D points, where the points in each list are connected by edges. This works in general, as we can describe objects with holes (nonzero genus) by having both outer and inner perimeters. An inner perimeter should then be in the opposite direction.

A. Silhouette Computation

To compute the silhouette of the real object, we enlarge the segmentation resolution by a factor of two using nearest neighbor sampling. We then use the algorithm by Suzuki and Abe [43, 44] to trace the perimeter of the object. We downscale the traced perimeter and round the coordinates so that they lie exactly on the border between object and background. After tracing the perimeter, we have an optional step to simplify the perimeter to accelerate computations later on. The optional simplification is done using the Ramer-Douglas-Peucker algorithm [45, 46]. If the lens distortion of the camera that captured the ground truth image is known, the silhouette points can be undistorted, removing the need to undistort the segmentation itself.

We compute silhouettes of the 3D models without rasterization. This makes the silhouettes directly differentiable with respect to scene parameters, which is an advantage in a gradient-based optimization. Given a CAD model, we extract a polygonal mesh and build a half-edge representation of this for easy queries. For a given view matrix, we project the vertex positions to the image plane and connect them using the edges of the mesh polygons. To compute the silhouette, we traverse these edges using Algorithm 1. This algorithm assumes a fully connected object silhouette without holes. Extension to objects with holes is done by restarting the algorithm inside each hole.

For the silhouette computation in Algorithm 1, we find the signed angle between two vectors in 2D using

$$\text{angle}\left(\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}, \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}\right) = \text{atan2}\left(a_1 b_2 - a_2 b_1,\ a_1 b_1 + a_2 b_2\right). \quad \textbf{(1)}$$

The majority of time in Algorithm 1 is spent computing edge intersections [47]. Computational complexity thus depends on the number of edges. We significantly reduce this number by exploiting that an edge can only be part of the silhouette if it is shared by one face facing the camera and another facing away [48]. If we let $\vec{n}_{e,1}$ and $\vec{n}_{e,2}$ denote the 3D surface normals of the faces bordering an edge e, the edge e can only be part of the silhouette if

$$\left(\vec{n}_{e,1} \cdot (\mathbf{v}_e - \mathbf{c})\right)\left(\vec{n}_{e,2} \cdot (\mathbf{v}_e - \mathbf{c})\right) \leq 0\,, \tag{2}$$

where $\mathbf{v}_e$ is any point on the edge and $\mathbf{c}$ is the position of the camera. After removing all edges that cannot be part of the silhouette and building a bounding volume hierarchy for the remaining edges, intersection testing is inexpensive.

Another way to reduce the computation time of this algorithm is to use a mesh with a lower polygon count for the silhouette while retaining the original mesh for rendering. A modest mesh simplification often has a negligible influence on the silhouette.

We have several options when computing silhouette derivatives. For simplicity, we use finite differences. Exact derivatives can be obtained with automatic differentiation.

B. Shadow Contours

To include the shadow of an object when considering its silhouette, we assume that the object is placed on a planar surface and use projection shadows [41]. This is also done without rasterization to keep our method valid for the entire image plane. We project the edges of the mesh to the ground plane to generate shadow edges. We then project both object and shadow edges to the image plane of the camera. After this, we use Algorithm 1 to compute the silhouette of the object including its shadow. The number of edges in the shadow that we need to consider is reduced early in the procedure by substituting $\mathbf{c}$ with the light position in Eq. (2).

C. Silhouette Matching

To be able to align silhouettes, we introduce a silhouette similarity metric. We refer to the silhouette of the real object observed by camera c as $\mathbf{R}_c$, and the union of object and shadow silhouettes as $\mathbf{R}_{c,\ell}$, where ℓ is the light source causing the shadow. Equivalently, we define for the virtual object $\mathbf{V}_c$ and $\mathbf{V}_{c,\ell}$. We now let $\mathrm{P}(\mathbf{X}, t)$ denote a parameterization of the silhouette $\mathbf{X}$ with $t \in [0, 1]$. We measure the similarity of two silhouettes by using a function (d) that finds the shortest distance from a point to a silhouette. Taking n equidistantly sampled points on the silhouettes, we find the shortest distance to the other silhouette and take the sum. The similarity is then computed by

$$\mathrm{sim}(\mathbf{R}, \mathbf{V}, n) = \sum_{i=1}^{n} \left(\mathrm{d}\left(\mathbf{R}, \mathrm{P}\left(\mathbf{V}, \tfrac{i}{n}\right)\right)^2 + \mathrm{d}\left(\mathbf{V}, \mathrm{P}\left(\mathbf{R}, \tfrac{i}{n}\right)\right)^2 \right), \tag{3}$$

A visualization of what sim computes is in Fig. 2. We can again use a spatial data structure to obtain an efficient implementation of the d function [49]. Our similarity metric (sim) has the advantage that it has a nonzero gradient even for non-intersecting silhouettes, which enables the use of our method with a poor initial guess.

Our final goal is to minimize the difference between the silhouettes of the real and virtual objects. For a silhouette without shadow, we measure the similarity by

$$E_c = \mathrm{sim}(\mathbf{R}_c, \mathbf{V}_c, \lceil \|\mathbf{R}_c\| \rceil)\,, \tag{4}$$

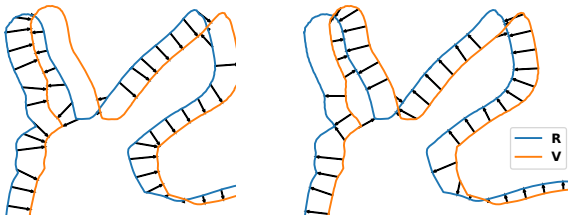


Fig. 2. Illustration of how $\mathrm{sim}(\mathbf{R}, \mathbf{V}, n)$ is computed for a small value of n. The arrows illustrate evaluations of $\mathrm{d}(\cdot, \cdot)$.

where $\|\cdot\|$ denotes the length of a silhouette in pixels. Ideally, we would like to sample as many points as possible. In this performance vs. accuracy trade-off, we have chosen $n = \lceil \|\mathbf{R}_c\| \rceil$ to place the sampled points approximately one pixel apart.

To compare silhouettes including shadows, we introduce a similarity measurement $E_{c,\ell}$. As mentioned previously, we would like to refine estimates using multiple cameras and light sources as long as only one is active per image. We compute the sum of comparisons of silhouettes over one or more configurations as follows:

$$E_s = \sum_{\ell} \sum_{c} \Big(\underbrace{\mathrm{sim}(\mathbf{R}_{c,\ell}, \mathbf{V}_{c,\ell}, \lceil \|\mathbf{R}_{c,\ell}\| \rceil)}_{E_{c,\ell}} + E_c \Big). \tag{5}$$

In the following, we describe how we estimate object pose and light source position using these silhouette similarity measurements.

D. Pose Estimation

We compute the pose of the object independently for each camera. We do this in camera space, where the camera is fixed at the origin. In the end, we can then use the known relation between object and ground plane to position each camera in world space.

Starting in camera space, the first step of the pose estimation is to find an initial guess for the position of the object. We do this by minimizing E_c with respect to the position, which places the virtual object approximately in the same position as the real object.

To find a good initial guess of the rotation, we randomly sample rotations. For each rotation, we compare the silhouette of the digital object to the real object using Hu's moment invariants [33]. These are calculated from image moments but are invariant to scale, rotation, and translation. For an image of pixel values $I(x, y)$, the image moments are defined by

$$M_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q I(x, y)\, dx\, dy, \tag{6}$$

where the p and q exponents are the moment orders and integration is across the image plane. Since the silhouette can be considered a polygon, the image moments can be computed efficiently by applying Green's theorem [50]. Hu's moment invariants are seven polynomial combinations of image moments that we store in a vector and compare using the sum of squared differences. Using the Hu moment invariants, the search space of the rotation is practically reduced to two dimensions. The rotation giving the silhouette that best matches the Hu moment invariants of the real silhouette is chosen as the initial guess of the rotation. We parameterize the rotation using quaternions and use the centroid of the object as the rotation centre.

With these initial guesses for position and rotation, we minimize E_c, which gives an object pose for each view. We use

Levenberg-Marquardt [51, 52] for the minimization. This is possible as E_c is a sum of squares. As part of our input, we know the object pose in relation to the ground plane. We use this to convert the per camera object poses into camera poses in world space.

E. Light Positions and Final Refinement

To estimate the position of each light, we randomly sample positions and then choose the one with the lowest $E_{c,\ell}$ for each light. Following this, $E_{c,\ell}$ is minimized using Levenberg-Marquardt.

The last step of our method is a joint optimization where we minimize E_s with respect to object pose, camera pose(s), light position(s), and a non-uniform scaling of the mesh. The non-uniform scaling of the mesh is to compensate for some of the shrinkage that may occur during 3D printing. The final optimization of the object pose is beneficial as the inclusion of the shadow silhouette(s) enables us to use more information from the input image.

F. Known Camera Poses

If camera poses are known in advance, for example from a stereo calibration of the camera rig, we can use the same steps as in Sec. 3.D to find the pose for all cameras jointly. When finding the rotation, it is then no longer desirable to have rotational invariance for all cameras. Instead, we propose to rotate the object to align the normalized image moments of the virtual and digital objects in the best way possible along a randomly chosen camera's viewing direction. The rotation is found by aligning the principal components of the two silhouettes [53]. We choose the rotation that best matches the normalized image moments across all cameras as the best rotation.

An initial guess of the object's scale is required, but if the camera poses are known in relation to the ground plane, we need not know the rotation of the object relative to the ground plane. The method for light source estimation is as with unknown camera poses.

4. APPEARANCE MODELS FOR REAL OBJECTS

Rendering systems provide a multitude of rendering techniques that we need to choose among when composing an appearance model for a real object. We start from a very approximate model at the most macroscopic scale. We then gradually increase complexity by reconsidering the involved optical properties [54] and what types of materials and visual effects that they can model.

At the most macroscopic scale, we have the bidirectional reflectance/transmittance distribution function (BRDF/BTDF) and the simplest models at this scale are the ones for perfectly diffuse and perfectly specular materials [55]. To cover a broad spectrum of different material types, we consider three different starting points: (a) diffuse, (b) metallic, or (c) transparent. In the following, we describe existing appearance models for these material types as well as model extensions (Secs. 4.A–4.C).

The perfectly diffuse (or Lambertian) material is a good starting point for objects that exhibit a significant amount of subsurface scattering (a). The BRDF of a perfectly diffuse material is $f_{r,d} = \rho_d/\pi$, where ρ_d is the bihemispherical diffuse reflectance, which we can set in an RGB renderer using a color vector in $[0,1]^3$. This reflectance represents the subsurface scattering of the material. We can then add an interface to model highlights and switch to a bidirectional scattering-surface reflectance distribution function (BSSRDF) to model translucency. The Fresnel equations for reflection are an excellent starting point for metallic and transparent objects (b-c).

The BRDF/BTDF of a perfectly smooth or a rough interface are available from Walter et al. [19]. The BRDFs presented by these authors work just as well for metals as long as we use the complex index of refraction of the metals to find the Fresnel factor. The key difficulty in use of the Fresnel equations is that indices of refraction are physical parameters that are defined as a spectrum rather than colors. We can convert a spectrum to a representative RGB vector using weighted averages based on RGB color matching functions [56, 57]. Assuming known (complex) index of refraction, the key parameter for metallic and transparent objects is the surface roughness (which is different for different surface microfacet distributions [19, 58]).

A natural extension of the diffuse model (a) is to introduce a refractive interface. The BRDF then becomes a sum of a specular and a diffuse component [59]. We can think of the specular term as in-surface scattering and of the diffuse term as subsurface scattering. The Fresnel equations are then useful for ensuring energy conservation (and reciprocity) both for smooth surfaces [60] and for rough surfaces [61, 62]. The trick is to sample the BRDF/BTDF of a transparent surface [19] and then let incident light that refracts into the material reflect diffusely before it refracts back out of the material using the BTDF of the surface again but this time for the outgoing direction. This enables addition of glossy reflections and highlights to an object with an otherwise matte appearance.

A natural extension of the transparent model (b) is to account for absorption based on the distance d that a ray travels through the interior of the object. This is done using an (RGB) absorption coefficient σ_a and Bouguer's law of exponential attenuation of light (attenuation factor $e^{-\sigma_a d}$). The absorption coefficient is directly linked to the imaginary part of the index of refraction [63]. The index of refraction was assumed known, and for metals σ_a is very large. We can thus assume that all light transmitted into a metal is absorbed. However, for transparent objects, σ_a is often very small and may need some adjustment to account for dissolved substances [56] or impurities [57]. The absorption coefficient then becomes an RGB parameter in the model that controls the color of transmitted light.

A further extension of the diffuse model (a) is to replace $f_{r,d}$ with proper subsurface scattering, where light may be incident at one surface position and observed at another. In terms of input parameters, this requires knowledge of the (RGB) scattering coefficient σ_s and the phase function. The latter is the distribution of the scattered light, which is often represented by an analytical model taking an (RGB) asymmetry parameter (g) as input. Several rendering techniques are available for evaluating the volumetric light transport between two surface positions [64]. For highly scattering materials, however, a full-fledged unbiased path tracing technique [65] is unpractical due to long rendering times. We need faster rendering when tuning parameters based on comparison of renderings with a reference photograph. A more practical rendering technique for subsurface scattering is then to use an analytical approximation of the BSSRDF [20, 21].

The standard dipole approximation for subsurface scattering [21] does not model how the direction of the incident light influences the subsurface scattering. To include this component, we can use a directional dipole approximation [20]. However, these models use Fresnel terms that assume a perfectly smooth interface. Donner and Jensen [66] explained how to account for a rough surface with a distribution of microfacet normals [58, 59]. In the following, we describe how to account for a rough surface

in the case of a model that accounts for the directional dependency of the subsurface scattering. We also describe simplistic models that we use to account for spatial variation in the surface roughness of our example objects.

A. Directional Subsurface Scattering for Rough Surfaces

The BSSRDF depends on the object geometry X, the position $\mathbf{x}_i$ and the direction $\vec{\omega}_i$ of the incident light as well as the position $\mathbf{x}_o$ and the direction $\vec{\omega}_o$ of the observed light. The normals at the points of incidence and observation $\vec{n}_i$ and $\vec{n}_o$ are known from the object geometry. An analytic BSSRDF model developed for a material with a smooth surface then usually has the form

$$S(X;\mathbf{x}_i,\vec{\omega}_i;\mathbf{x}_o,\vec{\omega}_o) = F_t(\vec{\omega}_o\cdot\vec{n}_o)(S_d+S^*)F_t(\vec{\omega}_i\cdot\vec{n}_i)\,, \quad \textbf{(7)}$$

where F_t is Fresnel transmittance, S_d is the diffusive part, which is typically modeled by a dipole, and S^* is the remaining light transport. The number of arguments used with S_d and S^* is different for different models.

To incorporate a rough surface in a BSSRDF model of this kind, we add a BRDF in the special case where the point of incidence equals the point of emergence, and we insert hemispherical transmittance integrals in place of the Fresnel terms:

$$\begin{aligned} S(X;\mathbf{x}_i,\vec{\omega}_i;\mathbf{x}_o,\vec{\omega}_o) &= \delta(\mathbf{x}_o-\mathbf{x}_i)f_r(\mathbf{x}_o,\vec{\omega}_i,\vec{\omega}_o) \\ &+ \int_{2\pi}\int_{2\pi} f_t(\mathbf{x}_o,\vec{\omega}_{21},\vec{\omega}_o)(-\vec{n}_o\cdot\vec{\omega}_{21})(S_d+S^*)\,d\omega_{21} \\ &\quad f_t(\mathbf{x}_i,\vec{\omega}_i,\vec{\omega}_{12})(-\vec{n}_i\cdot\vec{\omega}_{12})\,d\omega_{12}\,, \end{aligned} \quad \textbf{(8)}$$

where f_r is the BRDF and f_t is the BTDF of the surface, δ is a Dirac delta function, $\vec{\omega}_{12}$ is the direction of a ray transmitted into the volume, and $\vec{\omega}_{21}$ is the direction of a ray to be transmitted out of the volume. The directions $\vec{\omega}_{12}$ and $\vec{\omega}_{21}$ would thus be the ones to use as arguments for the S-functions.

The S^* term is usually fully directional, and the integrations over BTDFs at $\mathbf{x}_i$ and $\mathbf{x}_o$ are evaluated using regular volume path tracing with rough refraction at the interfaces. In the case of the standard dipole [21], $S^* = S^{(1)}$ includes evaluation of single scattering in the volume. In the case of the directional dipole [20], $S^* = S_{\delta E}$ is evaluated in the same way as absorption in a transparent material, but with a modified coefficient in the exponential attenuation. One should note that analytic expressions are available for the Fresnel transmittance integrals in cases where S_d is independent of $\vec{\omega}_i$ and/or $\vec{\omega}_o$ [66, 67]. Some care must be taken as some models [20, 67] assume a diffuse distribution of the light at $\mathbf{x}_o$ and then include the integration over $\vec{\omega}_{21}$ in their formulation. In the case of the directional dipole, our expression becomes

$$\begin{aligned} S(X;\mathbf{x}_i,\vec{\omega}_i;\mathbf{x}_o,\vec{\omega}_o) &= \delta(\mathbf{x}_o-\mathbf{x}_i)f_r(\mathbf{x}_o,\vec{\omega}_i,\vec{\omega}_o) + S^*_{\delta E} \\ &+ \int_{2\pi} S_{d,\mathrm{dir}}(\mathbf{x}_i,\vec{\omega}_{12};\mathbf{x}_o)f_t(\mathbf{x}_i,\vec{\omega}_i,\vec{\omega}_{12})(-\vec{n}_i\cdot\vec{\omega}_{12})\,d\omega_{12}\,, \end{aligned} \quad \textbf{(9)}$$

where $S_{d,\mathrm{dir}}$ is the diffusive part of the BSSRDF in the directional dipole model, but taking the transmitted direction directly as input instead of $\vec{\omega}_i$, and $S^*_{\delta E}$ is the modified reduced intensity term appearing in this model, but here including the BTDF integrations (rough refractions at the interfaces).

Comparing Eq. (8) to common illumination models [1, 59], the first term corresponds to the specular term and the second term corresponds to the diffuse term. The BRDF f_r to be used for the first term should therefore not include an added diffuse term. The BSDF (collective name for BRDF and BTDF) used in Eq. (8) should rather depend only on surface properties, such as a distribution of microfacet normals, see the work of Walter et al. [19] for examples. In particular, we use the so-called GGX distribution developed by these authors. This distribution has a width parameter α_g that we refer to as the GGX roughness.

B. Surface Roughness of a 3D Printed Object

Since most 3D printers print in layers, the surface of a printed object is usually rougher when the intended surface normal points in a direction aligned with layer edges in the voxel cubes of the print volume. If the z-axis is the print direction, we can use the following function to control the GGX roughness (α_g) based on the z-component of the surface normal (n_z):

$$\alpha_g = \rho + (1-\rho)\frac{|\sin(2\theta)|^s}{s} = \rho + (1-\rho)\frac{\left(2|n_z|\sqrt{1-n_z^2}\right)^s}{s}, \quad \textbf{(10)}$$

where θ is the angle of the surface normal $\vec{n}$ with the z-axis. We can think of the user parameters as follows: $\rho \in [0,1]$ is the minimum roughness and $s > 0$ is the shininess, which controls the height and width of the bumps in the curve around angles of $\pm 45^\circ$, $\pm 135^\circ$.

C. Surface Roughness of a Polished Metal Object

Quick hand polishing of a metallic object can result in an object with a rougher surface in curved areas and a smoother surface in flat areas. One way to specify the curvature of an object is using the mean curvature normal $\mathbf{H}$ [68]. This is a quantity that we can precompute for a triangle mesh using vertex circulators and store as a vertex attribute. The dot product of the outward-pointing surface normal $\vec{n}$ and the mean curvature normal $\mathbf{H}$ provides a signed measure of the curvature, where positive is a concavity and negative is a convexity. We use the absolute value of this dot product as an indicator of areas that were maybe not as easy to polish. To reduce noise from the surface scan and set a high roughness for curved areas, we employ a sigmoid function. Our use of the mean curvature normal is demonstrated in Fig. 3, and the formula is

$$\alpha_g = \rho + \frac{1-\rho}{1+\exp(s\,(1-30\,|\mathbf{H}\cdot\vec{n}|))}\,, \quad \textbf{(11)}$$

where ρ is again minimum roughness and s is a sort of shininess while $\mathbf{H}$ is the mean curvature normal after division by the length of the longest mean curvature normal in the triangle mesh.

5. RENDERING

We implemented a progressive unidirectional path tracer using OptiX [69]. To include subsurface scattering, we sample a new set of surface positions for each progressive update. For each update and within each pixel, the ray tracer generates a random position $\mathbf{x}_p$ in pixel coordinates. With the rotation of the camera relative to the object $\mathbf{R}$ and the camera intrinsic matrix $\mathbf{K}$, we get the direction of the corresponding ray using

$$\vec{\omega} = (\mathbf{KR})^{-1}\mathbf{S}\,\mathbf{x}_p = \mathbf{R}^T\mathbf{K}^{-1}\mathbf{S}\,\mathbf{x}_p\,. \quad \textbf{(12)}$$

Since the intrinsic matrix $\mathbf{K}$ is locked to the resolution of the camera ($W_c \times H_c$), which is usually very high, we use the scaling matrix $\mathbf{S} = \mathrm{diag}(W_c/W_r, H_c/H_r, 1)$ to enable rendering in a different resolution ($W_r \times H_r$).



Fig. 3. Model of spatially varying roughness α_g for an aluminium bust that has from time to time been subjected to hand polishing. We use the dot product of the mean curvature normal $\mathbf{H}$ and the surface normal $\vec{n}$. The model correctly marks eyes, hair, nostrils, and engraved letters as rough, but also incorrectly marks edges along the box-like base of the bust as being very rough.

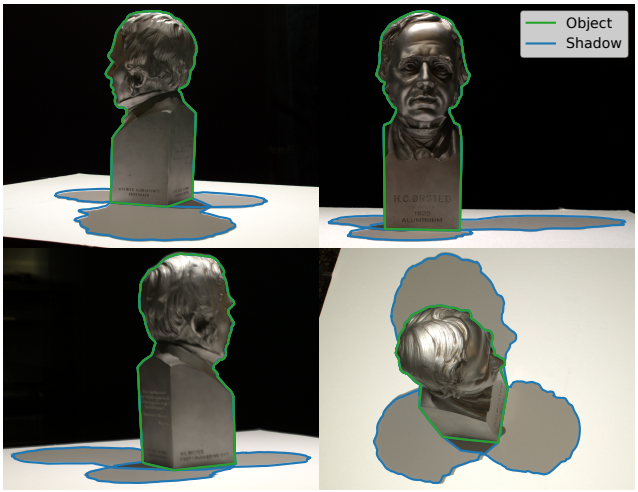


Fig. 4. Each image is an additive blend of three photos of the bust illuminated by the light source at different positions and overlaid with aligned silhouettes of the digital object.

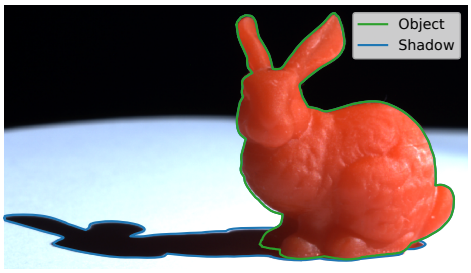


Fig. 5. Photo of the bunny overlaid with the aligned silhouette of the digital object.

6. RESULTS

The three objects of interest are (a) a translucent 3D print of the Stanford bunny, (b) an aluminium bust, and (c) a cupped angel figurine 3D scanned and printed using almost transparent resin. Two of our test objects (b-c) were 3D scanned using structured light based on phase shifting [18]. The employed 3D scanner has a precision of around 100 μm [70]. Our 3D printed objects (a, c) were produced using vat photopolymerization additive manufacturing processes. In our pose estimation and renderings, we used the geometry of these objects without correction for print artifacts. The Stanford bunny was printed by Luongo et al. [71] using red Industrial Blend resin (manufactured by Fun To Do) and a digital light processing (DLP) printer developed for research. The vertical resolution of this printer is 18 μm and the horizontal resolution is 15.08 μm. The angel was printed using general-purpose resin IM2.0 GP1 (manufactured by AddiFab) and a Peopoly Moai stereolithography (SLA) printer. The laser spot size (horizontal resolution) of this printer is 70 μm, and we used a vertical resolution of 50 μm. In simulation, we use a real index of refraction of 1.54 for the printed objects as this is in the middle of the range of commercial acrylic resins with low shrinkage after photopolymerization [72]. Our three objects all have a rough surface and exhibit different types of spatial variation in this roughness.

We used our method to align renderings of the objects of interest with their photographs. We then tested different appearance models following the presented guidelines, where we started from a simplistic model and gradually added complexity. In each case, our end result is an appearance model and a rendering paired with a photograph for validation that would serve as a suitable starting point for an inverse rendering technique. The optical properties that we estimated for our different objects are in Table 1. The reference photographs and the associated CAD files and relative camera and light source alignments will be available as a supplement (at https://eco3d.compute.dtu.dk/pages/appearance). We encourage the reader to use this dataset for testing preferred appearance models and rendering software and for finding better optical properties including better spatial variation of surface roughness by means of inverse rendering.

A. Acquisition

The objects were placed on a flat piece of paper and illuminated by a Thorlabs MNWHL4 LED light source. This source is neutral white with a point-like radiation distribution within an angular diameter of 10°. The bunny (a) and the angel (c) were captured using a FLIR Grasshopper3 GS3-U3-60QS6C-C camera, while the bust (b) was captured using D3200, D7000, D7500 and D750 cameras from Nikon. We used four cameras to cover all angles of the object while also taking multiple images from the same positions with different light positions. As different cameras were used, the images of the bust were color calibrated using a ColorChecker from X-Rite. All images were captured with a small aperture so that all parts of the object and shadow were in focus. We performed camera calibration [44, 73] using a ChArUco board which is a checkerboard with ArUco markers [74]. For the bunny (a), we did not use the estimated extrinsics and only used the estimated focal length from the intrinsics. Lens distortion from the camera calibrations were used to undistort the reference photographs and ground truth silhouettes.

B. Alignment

To segment the photographs as required by our alignment method, we used thresholding followed by hole closing and selected the largest connected component. For the images of the bunny and the angel, some manual cleaning of the segmentation was necessary due to caustics.

Our test cases span different setups to showcase the flexibility of our alignment method. For the bunny (a), we use just a single

Table 1. Estimated optical properties.

Material	n	σ_a	σ_s	ρ	s
Bunny (FTD, red Industrial Blend)	1.54	$(0.33, 25, 67) \cdot 10^3$ m^{-1}	$(10, 21, 0.083) \cdot 10^3$ m^{-1}	0.20	2.4
Angel (AddiFab, IM2.0 GP1)	1.54	$(0.032, 32, 640)$ m^{-1}	0	0.15	5.0
Bust (aluminium)	$(1.04, 0.76, 0.49) + i\,(6.45, 5.73, 4.76)$	$1.3 \cdot 10^8$	n/a	0.22	4.5

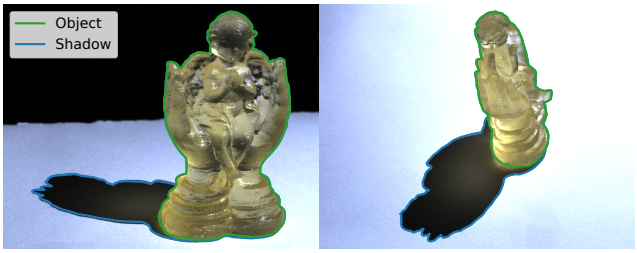


Fig. 6. Photos of the angel overlaid with the aligned silhouette of the digital object.

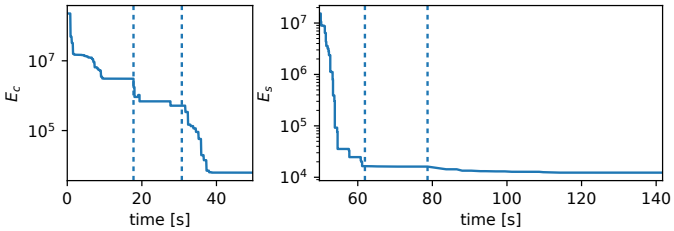


Fig. 7. Convergence plots of our pose estimation method applied to the bunny, showing the total time elapsed, with vertical dashed lines separating the different steps of our method. Left: as described in Sec. 3.D, steps: translation optimization, random rotation search, pose optimization. Right: as described in Sec. 3.E, steps: random light search, light position optimization, joint optimization.

picture with unknown camera pose to align the scene. For the angel (c), we use two camera poses and a single light position to do the estimation. Finally, the bust (b) was captured from four camera poses, each with four different light source positions, yielding a total of 16 images that we used to do the alignment. The more light source positions, the more information we have available for the pose estimation. This comes at the small cost of increasing the dimensionality of the optimization problem. If we again consider our method an enabler for inverse rendering, it is an advantage to have multiple light positions as these provide additional samples for estimation of BRDFs, for example.

Outputs from our alignment method are in Figs. 4 to 6. We achieve good alignment of the outlines of the bust, which makes sense as this is the only object in our collection for which the geometry is directly from the photographed object. Both the angel and the bunny have a quite good alignment, but especially the bunny has noticeable differences between the rendered silhouette and the object. We presume these mostly stem from non-linear shrinkage during printing that our method cannot account for. For the angel, our method estimated shrinkage of 3%, 6%, 1% in the x, y, z directions as compared to the size of an ideal 3D print.

As the bunny (a) is the more difficult case (with only one view and light source position to constrain the problem), we have analyzed the performance of our method more closely for this case. Convergence plots in Fig. 7 show that each step improves



RMSE: 0.1315
SSIM: 0.8065
Liu et al. [15]
for object pose

RMSE: 0.1304
SSIM: 0.7405
ours
w/o joint estimation

RMSE: 0.1124
SSIM: 0.8181
ours
final

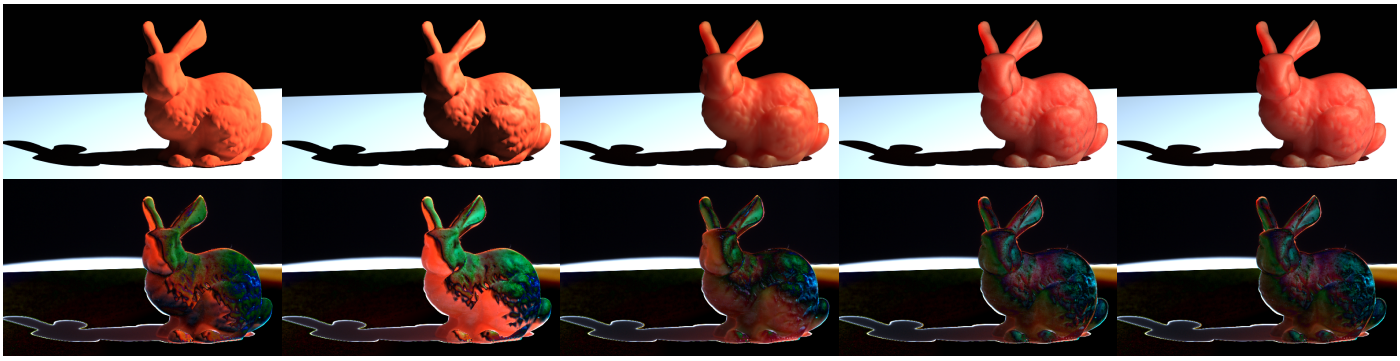
Fig. 8. Ablation study shown with signed difference images $\times 2$. Blue and red indicate positive and negative differences (for rendering minus photograph): average of the color bands in the third and the fourth column of Fig. 1, respectively.

the similarity (reduces E_c and E_s). While the joint optimization in the last step gives a smaller improvement of E_s than other steps, the improvement of the final rendered result is significant as seen in Fig. 8. We also compare our alignment result with an object pose obtained using the differentiable rendering method of Liu et al. [15]. We observe that the performance of this related work is similar to ours without joint optimization and we needed many random initial guesses with this method too in order for it to converge to a good solution. With other methods than ours, we do not get the advantages of jointly estimating light source position and mesh shrinkage. In the result found using the method of Liu et al. (Fig. 8, left), we used the camera pose and light source position from our final result. The key benefit of our work is thus collective extraction of information available in projected silhouettes (object pose, light source position, mesh shrinkage), and that we can use joint optimization to collectively improve each part of the result.

C. Appearance

Since our objects are placed on a piece of paper assumed to be flat, we place a quad in the ground plane and resize it manually to approximately fit the paper observed in the photograph. Precise alignment of the paper could be part of the object alignment, but we find that it is not so important with respect to testing the appearance model applied to the object. To start simple, we consider the paper to be a diffuse surface. More complexity could easily be added to the paper appearance model [75], but we focus our attention on the objects of interest.

We initialise the diffuse reflectance of the paper to $\rho_d = (0.8, 0.8, 0.8)$ and select the simplest shading model for the material category of the object in question. We then use the intensity of the light reflected from the paper to estimate the intensity of the point light. Since our source is neutral white, we use the same intensity in all color bands. An easy way to do a comparison is using two colored difference images: one for positive difference and one for negative difference (see examples in Fig. 1). Once the light intensity has been set, we modify the reflectance values until each color appears equally in the positive and the negative difference image. We also evaluate our results



RMSE: 0.1152	RMSE: 0.1237	RMSE: 0.1136	RMSE: 0.1127	RMSE: 0.1124
SSIM: 0.8108	SSIM: 0.7931	SSIM: 0.8145	SSIM: 0.8177	SSIM: 0.8181
Lambertian	interfaced	standard SSS	directional SSS	SV roughness

Fig. 9. Renderings (top) and absolute difference images $\times 2$ (bottom) to test appearance models for the rough translucent bunny. The interfaced model adds a rough surface with a GGX microfacet normal distribution [19, 60, 61]. The standard subsurface scattering (SSS) model is the standard dipole including path traced single scattering and a rough surface [21, 66]. The directional SSS model uses the directional dipole [20] and incorporates a rough surface (Sec. 4.A). The model with spatially varying (SV) roughness uses Eq. (10). Further comparison of the input image with the end result is in Fig. 1. The not quite so flat paper worsens both RMSE and SSIM by approximately 0.05.

quantitatively using root-mean-squared error (RMSE) (lower is better) and structural similarity (SSIM) index [76] (higher is better). The initial results for each of our three test cases are leftmost in Figs. 9 to 11.

To estimate absorption and scattering coefficients (σ_a and σ_s), we need the physical size of the object as these optical properties are measured per distance unit that a ray has travelled through the material. Using the physical dimensions of the object, we get the coefficients in Table 1. We decided to leave the phase function as isotropic ($g = 0$) since the analytic BSSRDF models mostly use the reduced scattering coefficient $\sigma_s' = \sigma_s(1 - g)$ and thus do not distinguish much between a reduction in σ_s and an increase of g. The directional dipole is not exclusively based on the reduced scattering coefficient, but the role of g seems limited. When estimating the coefficients, 10 over the length of the bounding box diagonal is usually a good value to start with for the absorption or the scattering to have a reasonable effect.

Refinement of the model for the rough translucent bunny (a), see Fig. 9. We first add an interface to the model [60, 61] to enable rendering of highlights. However, this also directs a lot of energy into a glossy reflection lobe meaning that the missing transport of light from the point of incidence to a different point of emergence becomes apparent and RMSE and SSIM both worsen. As soon as we switch to a model that accounts for this subsurface light transport [21], the result becomes better than the Lambertian model. This is true even without single scattering and assuming that the surface is perfectly smooth. The directional dipole [20] and our spatially varying roughness from Sec. 4.B further improve the result. However, the models cannot fully represent the scattering process. This is probably due to limiting assumptions such as diffuse emergent light and a locally flat, convex object. It should be mentioned that the bunny was printed using greyscale values to reduce staircasing artefacts [71]. These staircasing artefacts due to layered printing are significantly less pronounced for the bunny as compared with the angel (which was not printed using greyscale values). Nevertheless, the bunny object still exhibits some spatial variation in its roughness that we have modelled.

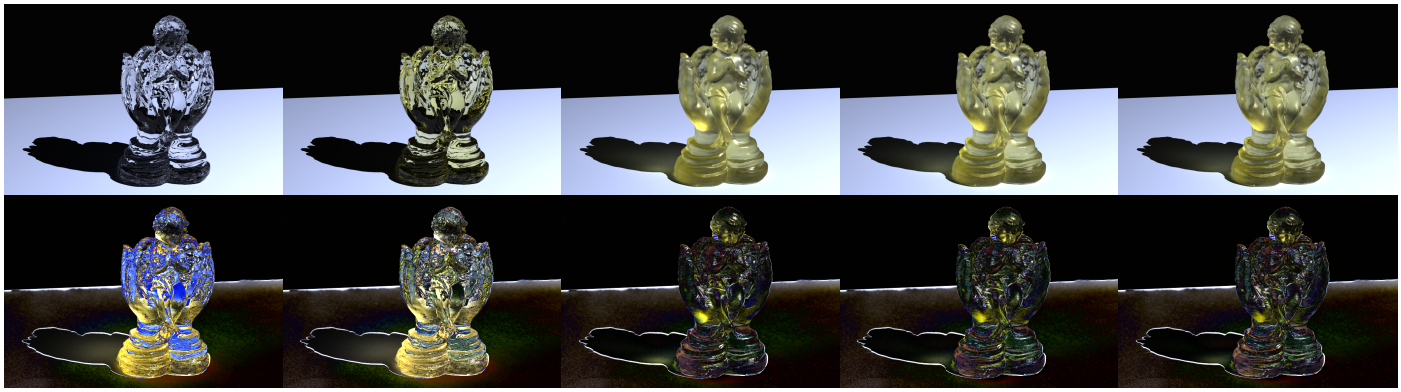
Refinement of the model for the aluminium bust (b), see Fig. 11. We use the complex index of refraction of aluminium from McPeak et al. [77] (this is available for download at refractiveindex.info). Since we have a dark scene with a point light, the appearance is off without surface roughness (as highlights then disappear). Adding a microfacet normal distribution was thus essential for this case, and we found that the GGX distribution [19] provided a good result. When adding spatially varying roughness based on the curvature, we found that SSIM would improve for a larger shininess s at the cost of a poorer RMSE. The SSIM-improved result is in Fig. 1. The RMSE probably suffers from a slight misplacement of the highlight peak in the forehead of the bust.

Refinement of the model for the rough transparent angel (c), see Fig. 10. Using the convention that surface normals always point outwards, absorption is easily included by applying Bouguer's law of exponential attenuation to all rays that hit the surface from the inside. Accounting for absorption and a rough interface is highly important when modelling the appearance of the angel. Apart from this, the print layers are visually obvious, especially in highlights. We, therefore, tried to model the layers by calculating a layer index based on the point of intersection and using an increased roughness for every second layer. This represents the rougher layer edges more explicitly. Visually, we find this layered result more convincing and it also has lower RMSE, but SSIM disagrees. We tried adding single scattering to the material, but this only seemed to worsen RMSE and SSIM. Thus, it seems that the remaining deviations from the reference are mostly due to geometric print artefacts and inaccuracies in the spatial variation of the surface roughness.

7. DISCUSSION

Although our method is able to quantify the differences between a rendering and a photograph, it does not provide a direct way of determining what the source of these differences are. However, when a change of reflectance model leads to a smaller error, it is very likely that the previous model was a source of error.

While we use a pinhole camera model, one should note that our method can also work for more advanced camera models as



RMSE: 0.1267	RMSE: 0.1135	RMSE: 0.0741	RMSE: 0.0733	RMSE: 0.0730
SSIM: 0.7699	SSIM: 0.8051	SSIM: 0.8876	SSIM: 0.8870	SSIM: 0.8858
glass	absorbing	rough	SV roughness	layered

Fig. 10. Renderings (top) and absolute difference images ×2 (bottom) to test appearance models for the rough transparent angel. We use the GGX microfacet normal distribution [19] and add absorption through analysis by synthesis [57] and spatially varying (SV) roughness (Sec. 4.B). We also tested a layered variation of the roughness in the print direction (every second layer is rougher to model a staircase). SSIM is sensitive to structure and takes a hit because the layers do not perfectly match the real layers. Further comparison of the input image with the end result is in Fig. 1.



RMSE: 0.0311	RMSE: 0.0159	RMSE: 0.0134
SSIM: 0.9339	SSIM: 0.9693	SSIM: 0.9713
metallic	rough	SV roughness

Fig. 11. Renderings (top) and absolute difference images ×2 (bottom) to test appearance models for the aluminium bust. We test spatially varying (SV) roughness as depicted in Fig. 3 and use high dynamic range when computing differences. The input image is in Fig. 1, where it is compared with an SSIM-improved result (RMSE: 0.0148, SSIM: 0.9725).

we can apply the necessary transformations to the object edges before computing the silhouette. Extending to area lights is however challenging and left for future work.

A disadvantage of using silhouettes is their simplicity. In some cases, they describe the features of an object inadequately, which can cause ambiguities in the pose estimation. An example of this could be a bowl with contents, where the silhouette only contains information enough to pose estimate the bowl. To have more information, some methods [29] also use features on the object itself. In cases where the segmentation has inaccuracies and our pose may have small errors, our method is still useful for obtaining a good initial guess that can be refined by other methods (such as differentiable rendering).

8. CONCLUSION

We presented a practical method for aligning photographs with rendered images. Our method is based on silhouette matching and estimates both object pose and the position of a point-like light source. If multiple images have been captured from different views and/or with light sources in different positions, our method can include this added information in the pose estimation. As opposed to differentiable rendering techniques, our method works not only in pixel space but in the entire image plane. This means that we can estimate a pose from a very poor initial guess. Thus we find our work a practical enabling technique for inverse rendering that could be based on differentiable rendering.

Given an alignment, we proposed a procedure for composing an appearance model suitable for the photographed object. The concept is to start from a simplistic model and gradually increase the complexity of appearance models guided by difference images and quantitative metrics such as RMSE and SSIM. As a consequence of this approach, we presented extensions of existing models providing improved photorealism. One extension was the combination of a rough surface with directional subsurface scattering. We believe that practical alignment of photographs with renderings is an important step in furthering the predictive abilities of appearance models.

Funding. Innovationsfonden (6151-00006B, 6151-00005B); Poul Due Jensen Foundation (2018-017).

Acknowledgments. Thanks to Macarena Mendez Ribo for 3D printing the transparent angel figurine and to Andreas Bærentzen for useful discussions on finding a silhouette given projected polygon edges.

Disclosures. Mads Emil Brix Doest: LEGO Group (F). Jakob Wilm: Euler3D ApS (I), Calib.io I/S (I).

The authors declare no conflicts of interest.

REFERENCES

1. B. T. Phong, "Illumination for computer generated pictures," Commun. ACM **18**, 311–317 (1975).
2. C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile, "Modeling the interaction of light between diffuse surfaces," Comput. Graph. (SIGGRAPH '84) **18**, 213–222 (1984).

3. G. W. Meyer, H. E. Rushmeier, M. F. Cohen, D. P. Greenberg, and K. E. Torrance, "An experimental evaluation of computer graphics imagery," ACM Transactions on Graph. **5**, 30–50 (1986).
4. H. Rushmeier, G. Ward, C. Piatko, P. Sanders, and B. Rust, "Comparing real and synthetic images: Some ideas about metrics," in *Rendering Techniques '95,* (Springer, 1995), pp. 82–91.
5. S. N. Pattanaik, J. A. Ferwerda, K. E. Torrance, and D. P. Greenberg, "Validation of global illumination solutions through CCD camera measurements," in *Proceedings of Color Imaging Conference (CIC 1997),* (1997), pp. 250–253.
6. C. Ulbricht, A. Wilkie, and W. Purgathofer, "Verification of physically based rendering algorithms," Comput. Graph. Forum **25**, 237–255 (2006).
7. M. Weinmann and R. Klein, "Advances in geometry and reflectance acquisition (course notes)," in *Proceedings of SIGGRAPH Asia 2015 Courses,* (ACM, 2015).
8. C. Reinbacher, M. Ruther, and H. Bischof, "Pose estimation of known objects by efficient silhouette matching," in *Proceedings of International Conference on Pattern Recognition (ICPR 2010),* (IEEE, 2010), pp. 1080–1083.
9. S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise voting network for 6DoF pose estimation," in *Proceedings of CVPR 2019,* (2019), pp. 4561–4570.
10. A. Panagopoulos, C. Wang, D. Samaras, and N. Paragios, "Illumination estimation and cast shadow detection through a higher-order graphical model," in *Proceedings of CVPR 2011,* (IEEE, 2011), pp. 673–680.
11. J. Lopez-Moreno, E. Garces, S. Hadap, E. Reinhard, and D. Gutierrez, "Multiple light source estimation in a single image," Comput. Graph. Forum **32**, 170–182 (2013).
12. R. Ramamoorthi and P. Hanrahan, "A signal-processing framework for inverse rendering," in *Proceedings of SIGGRAPH 2001,* (ACM, 2001), pp. 117—128.
13. G. Loubet, N. Holzschuch, and W. Jakob, "Reparameterizing discontinuous integrands for differentiable rendering," ACM Transactions on Graph. **38**, 228:1–228:14 (2019).
14. M. Nimier-David, D. Vicini, T. Zeltner, and W. Jakob, "Mitsuba 2: a retargetable forward and inverse renderer," ACM Transactions on Graph. **38**, 203:1–203:17 (2019).
15. S. Liu, W. Chen, T. Li, and H. Li, "Soft rasterizer: Differentiable rendering for unsupervised single-view mesh reconstruction," in *Proceedings of ICCV 2019,* (2019), pp. 7708–7717.
16. G. Turk, "The Stanford bunny," https://www.cc.gatech.edu/~turk/bunny/bunny.html (2000).
17. G. Turk and M. Levoy, "Zippered polygon meshes from range images," in *Proceedings of SIGGRAPH '94,* (1994), pp. 311–318.
18. J. Geng, "Structured-light 3D surface imaging: a tutorial," Adv. Opt. Photonics **3**, 128–160 (2011).
19. B. Walter, S. R. Marschner, H. Li, and K. E. Torrance, "Microfacet models for refraction through rough surfaces," in *Proceedings of Eurographics Symposium on Rendering (EGSR 2007),* (The Eurographics Association, 2007), pp. 195–206.
20. J. R. Frisvad, T. Hachisuka, and T. K. Kjeldsen, "Directional dipole model for subsurface scattering," ACM Transactions on Graph. **34**, 5:1–5:12 (2014).
21. H. W. Jensen, S. R. Marschner, M. Levoy, and P. Hanrahan, "A practical model for subsurface light transport," in *Proceedings of SIGGRAPH 2001,* (ACM, 2001), pp. 511–518.
22. H. Lensch, J. Kautz, M. Goesele, W. Heidrich, and H.-P. Seidel, "Image-based reconstruction of spatial appearance and geometric detail," ACM Transactions on Graph. **22**, 234–257 (2003).
23. M. Holroyd, J. Lawrence, and T. Zickler, "A coaxial optical scanner for synchronous acquisition of 3D geometry and surface reflectance," ACM Transactions on Graph. **29**, 99:1–99:12 (2010).
24. M. M. Loper and M. J. Black, "OpenDR: An approximate differentiable renderer," in *Proceedings of ECCV 2014,* (Springer, 2014), pp. 154–169.
25. T.-M. Li, M. Aittala, F. Durand, and J. Lehtinen, "Differentiable Monte Carlo ray tracing through edge sampling," ACM Transactions on Graph. **37**, 222:1–222:11 (2018).
26. B. Bhanu, "CAD-based robot vision," Computer **20**, 13–16 (1987).
27. J. Byne and J. A. D. W. Anderson, "A CAD-based computer vision system," Image Vis. Comput. **16**, 533–539 (1998).
28. M. Ulrich, C. Wiedemann, and C. Steger, "CAD-based recognition of 3D objects in monocular images," in *Proceedings of ICRA 2009,* (IEEE, 2009), pp. 2090–2097.
29. A. Petit, E. Marchand, R. Sekkal, and K. Kanani, "3D object pose detection using foreground/background segmentation," in *Proceedings of ICRA 2015,* (IEEE, 2015), pp. 1858–1865.
30. B. Rosenhahn, T. Brox, D. Cremers, and H.-P. Seidel, "A comparison of shape matching methods for contour based pose estimation," in *International Workshop on Combinatorial Image Analysis,* (Springer, 2006), pp. 263–276.
31. O. Tahri and F. Chaumette, "Complex objects pose estimation based on image moment invariants," in *Proceedings of ICRA 2005,* (IEEE, 2005), pp. 436–441.
32. O. Tahri, H. Araujo, Y. Mezouar, and F. Chaumette, "Efficient iterative pose estimation using an invariant to rotations," IEEE Transactions on Cybern. **44**, 199–207 (2013).
33. M.-K. Hu, "Visual pattern recognition by moment invariants," IRE transactions on information theory **8**, 179–187 (1962).
34. M. Zhu, K. G. Derpanis, Y. Yang, S. Brahmbhatt, M. Zhang, C. Phillips, M. Lecce, and K. Daniilidis, "Single image 3D object detection and pose estimation for grasping," in *Proceedings of ICRA 2014,* (IEEE, 2014), pp. 3936–3943.
35. Z. Cao, Y. Sheikh, and N. K. Banerjee, "Real-time scalable 6DOF pose estimation for textureless objects," in *Proceedings of ICRA 2016,* (IEEE, 2016), pp. 2441–2448.
36. E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold *et al.*, "Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image," in *Proceedings CVPR 2016,* (2016), pp. 3364–3372.
37. W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "SDD-6D: Making RGB-based 3D detection and 6D pose estimation great again," in *Proceedings of ICCV 2017,* (2017), pp. 1521–1529.
38. M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth," in *Proceedings of ICCV 2017,* (2017), pp. 3828–3836.
39. B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," in *Proceedings CVPR 2018,* (2018), pp. 292–301.
40. Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "DeepIM: Deep iterative matching for 6D pose estimation," in *Proceedings of ECCV 2018,* (2018), pp. 683–698.
41. J. Blinn, "Me and my (fake) shadow," IEEE Comput. Graph. Appl. **8**, 82–86 (1988).
42. N. Chotikakamthorn, "Near point light source location estimation from shadow edge correspondence," in *Proceedings of Cybernetics and Intelligent Systems (CIS) and Robotics, Automation and Mechatronics (RAM),* (IEEE, 2015), pp. 30–35.
43. S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," Comput. Vision, Graph. Image Process. **30**, 32–46 (1985).
44. G. Bradski, "The OpenCV library," Dr. Dobb's J. Softw. Tools **25**, 120–125 (2000).
45. U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," Comput. Graph. Image Process. **1**, 244–256 (1972).
46. D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," Cartogr. The Int. J. for Geogr. Inf. Geovisualization **10**, 112–122 (1973).
47. F. Antonio, "Faster line segment intersection," in *Graphics Gems III,* D. Kirk, ed. (Academic Press, 1992), pp. 199–202.
48. P. Bénard and A. Hertzmann, "Line drawings from 3d models: a tutorial," Foundations Trends Comput. Graph. Vis. **11**, 1–159 (2019).
49. P. Alliez, S. Tayeb, and C. Wormser, "3D fast intersection and distance computation," CGAL user reference manual **3** (2016).

50. X. Y. Jiang and H. Bunke, "Simple and fast computation of moments," Pattern Recognit. **24**, 801–806 (1991).
51. K. Levenberg, "A method for the solution of certain non-linear problems in least squares," Q. Appl. Math. **2**, 164–168 (1944).
52. D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," J. Soc. for Ind. Appl. Math. **11**, 431–441 (1963).
53. R. Candelier, "Tracking object orientation with image moments," http://raphael.candelier.fr/?blog=Image%20Moments (2016).
54. J. R. Frisvad, S. A. Jensen, J. S. Madsen, A. Correia, L. Yang, S. K. S. Gregersen, Y. Meuret, and P.-E. Hansen, "Survey of models for acquiring the optical properties of translucent materials," Comput. Graph. Forum **39** (2020). To appear.
55. F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis, "Geometrical considerations and nomenclature for reflectance," Tech. Rep. NBS MN-160, National Bureau of Standards (1977).
56. J. R. Frisvad, N. J. Christensen, and H. W. Jensen, "Computing the scattering properties of participating media using Lorenz-Mie theory," ACM Transactions on Graph. **26**, 60:1–60:10 (2007).
57. J. D. Stets, A. Dal Corso, J. B. Nielsen, R. A. Lyngby, S. H. N. Jensen, J. Wilm, M. B. Doest, C. Gundlach, E. R. Eiriksson, K. Conradsen, A. B. Dahl, J. A. Bærentzen, J. R. Frisvad, and H. Aanæs, "Scene reassembly after multimodal digitization and pipeline evaluation using photorealistic rendering," Appl. Opt. **56**, 7679–7690 (2017).
58. R. L. Cook and K. E. Torrance, "A reflectance model for computer graphics," ACM Transactions on Graph. **1**, 7–24 (1982).
59. K. E. Torrance and E. M. Sparrow, "Theory for off-specular reflection from roughened surfaces," J. Opt. Soc. Am. **57**, 1105–1114 (1967).
60. P. Shirley, B. Smits, H. Hu, and E. Lafortune, "A practitioners' assessment of light reflection models," in *Proceedings Pacific Graphics 97,* (1997), pp. 40–49.
61. M. Ashikmin, S. Premože, and P. Shirley, "A microfacet-based BRDF generator," in *Proceedings of SIGGRAPH 2000,* (ACM/Addison-Wesley, 2000), pp. 65–74.
62. L. Simonot, "Photometric model of diffuse surfaces described as a distribution of interfaced Lambertian facets," Appl. Opt. **48**, 5793–5801 (2009).
63. M. Born and E. Wolf, *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light* (Cambridge University Press, 1999), seventh (expanded) ed.
64. M. Pharr, W. Jakob, and G. Humphreys, *Physically Based Rendering: From Theory to Implementation* (Morgan Kaufmann/Elsevier, 2017), 3rd ed.
65. M. Raab, D. Seibert, and A. Keller, "Unbiased global illumination with participating media," in *Monte Carlo and Quasi-Monte Carlo Methods 2006,* (Springer, 2008), pp. 591–605.
66. C. Donner and H. W. Jensen, "Light diffusion in multi-layered translucent materials," ACM Transactions on Graph. **24**, 1032–1039 (2005).
67. E. d'Eon and G. Irving, "A quantized-diffusion model for rendering translucent materials," ACM Transactions on Graph. **30**, 56:1–56:13 (2011).
68. J. A. Bærentzen, J. Gravesen, F. Anton, and H. Aanæs, *Guide to Computational Geometry Processing: Foundations, Algorithms, and Methods* (Springer, 2012).
69. S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison, and M. Stich, "OptiX: A general purpose ray tracing engine," ACM Transactions on Graph. **29**, 66:1–66:13 (2010).
70. E. R. Eiríksson, J. Wilm, D. B. Pedersen, and H. Aanæs, "Precision and accuracy parameters in structured light 3-D scanning," The Int. Arch. Photogramm. Remote. Sens. Spatial Inf. Sci. **40**, 7–15 (2016).
71. A. Luongo, V. Falster, M. B. Doest, M. M. Ribo, E. R. Eiriksson, D. B. Pedersen, and J. R. Frisvad, "Microstructure control in 3D printing with digital light processing," Comput. Graph. Forum **39**, 347–359 (2020).
72. F. Aloui, L. Lecamp, P. Lebaudy, and F. Burel, "Refractive index evolution of various commercial acrylic resins during photopolymerization," eXPRESS Polym. Lett. **12**, 966–971 (2018).
73. Z. Zhang, "A flexible new technique for camera calibration," IEEE Transactions on pattern analysis machine intelligence **22**, 1330–1334 (2000).
74. S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," Pattern Recognit. **47**, 2280–2292 (2014).
75. M. Papas, K. de Mesa, and H. W. Jensen, "A physically-based BSDF for modeling the appearance of paper," Comput. Graph. Forum **33**, 133–142 (2014).
76. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Transactions on Image Process. **13**, 600–612 (2004).
77. K. M. McPeak, S. V. Jayanti, S. J. Kress, S. Meyer, S. Iotti, A. Rossinelli, and D. J. Norris, "Plasmonic films can easily be better: rules and recipes," ACS Photonics **2**, 326–333 (2015).