Imaging Behind Occluders Using Two-Bounce Light

Supplementary Information

Connor Henley, Tomohiro Maeda, Tristan Swedish, and Ramesh Raskar

Massachusetts Institute of Technology, Cambridge MA 02139, USA
{co24401, tomotomo, tswedish, raskar}@mit.edu

1 Reconstruction of Stationary Hidden Objects and Hidden Objects that are Moving

To validate our method we constructed a simple testbed consisting of three walls. Photographs of the testbed are shown in the main paper. This simple testbed allows us to avoid solving secondary problems–such as determining the 3D geometry of the visible scene, or separating shadows from the textures of visible surfaces–that may have otherwise confounded the assessment of our method. One of the walls, which we refer to as the *occluding wall*, was constructed from black foam-board, and placed between the scene-to-be-imaged and all imaging equipment. The other two walls, referred to as *observation walls*, were constructed from white posterboard and placed to either side of the hidden area, oriented parallel to one another and perpendicular to the occluding wall. The observation walls were 61 cm $\times$ 76 cm rectangles and were spaced 76 cm apart. The floor and back wall of the testbed were typically covered with black cloth. However, during our video reconstructions, the floor of the testbed was uncovered to reveal a stainless steel optical breadboard, and doing so had no observable effect on our results.

1.1 Laser Scanning and Image Acquisition

We illuminate the two observation walls at a series of points using a green CW laser with a power of $\sim$5 mW (Thorlabs CPS532), scanned with a two-axis scanning galvo mirror system (Thorlabs GVS012). For each laser position we used a single Point Grey Blackfly camera to capture two images: a short exposure that is used to estimate the 3D position of the laser spot, and a longer exposure to capture shadows. We loop through the scan pattern two times. On the first pass we acquire short exposure images, and on the second pass we acquire long exposure images. With this approach the camera's exposure setting is only changed once, between scans. This allows us to acquire frames more rapidly. In our setup, we achieve a frame rate of 15 FPS. After all frames have been acquired, the 3D position of the laser spots are estimated from the short exposure images using a homography, and the long exposure frames are converted to binary shadow

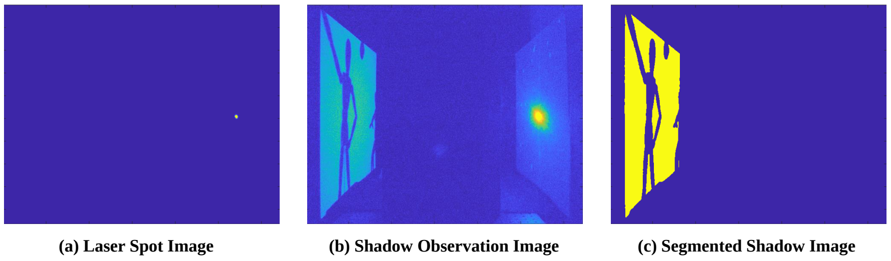

Fig. 1. Acquisition Process. For each laser position we acquire two frames: a short exposure (left) that is used to estimate the 3D position of the laser spot, and a longer exposure (middle) to capture shadows. A shadow segmentation procedure converts the long exposure frame to a binary shadow image (right).

images using a shadow segmentation procedure. These post-processing steps are described in greater detail in the main text.

We chose to use the galvo system to produce rapid and repeatable scan sequences, however we note that this is not required. Although we do not include the results in this document, we were also able to obtain good results aiming the laser by hand.

1.2 Algorithm

Algorithm 1 Imaging Behind Occluders using Planar Cast Shadows

```
input $\mathbf{L}, \mathbf{S}$
define $\mathbf{X}$, checked = false
define $\mathcal{P}_c$                                   ▷ Camera transform
for $\mathbf{l}_i$ in $\mathbf{L}$ do
    define $\mathcal{P}_{\ell_i}$                        ▷ Projection to obs. plane
    for $\mathbf{x}_j$ in $\mathbf{X}$ do
        $\mathbf{c}_j^w = \mathcal{P}_{\ell_i}(\mathbf{x}_j)$
        $\mathbf{c}_j^p = \mathcal{P}_c(\mathbf{c}_j^w)$
        if $\mathbf{c}_j^p$ is observable then
            $\text{checked}_j = \text{true}$
            if $\mathbf{c}_j^p$ is illuminated then
                remove $\mathbf{x}_j$, $\text{checked}_j$
            end if
        end if
    end for
end for
return $\mathbf{X}$, checked
```

Our algorithm is presented in Algorithm 1. The algorithm's input consists of the acquired set of laser spot locations $\mathbf{L} = \{\mathbf{l}_i\}$ and a set of binary shadow images $\mathbf{S} = \{S_i\}$. To start, our algorithm initializes an array of points $\mathbf{X} = \{\mathbf{x_j}\}$ lying on an evenly spaced 3D grid. The extent of the grid and the spacing between elements is specified by the user. Given the set of illumination points and the areas observed it is possible that some of these points may lie outside of the portion of the hidden space which can be imaged. To account for this we also initialize a second, binary array **checked** to record which points in memory have been subjected to an inside/outside test.

The algorithm then proceeds to loop through the acquired measurements. Each measured laser spot position $\mathbf{l}_i$ is used to define a perspective transform $\mathcal{P}_{\ell_i}$. We apply this transform to all points in $\mathbf{X}$ to calculate the points $\mathbf{c}_j^w = \mathcal{P}_{\ell_i}(\mathbf{x}_j)$ at which rays drawn from $\mathbf{l}_i$, through each point $\mathbf{x}_j$, intersect the plane of the observation wall. We then apply a second perspective transform $\mathcal{P}_c$ to convert these points of intersection from world coordinates to pixel coordinates associated with the camera that acquired the shadow images.

We compare each of these transformed points to the shadow image S_i. If a projected point lies inside the bounds of the observation wall, the point is marked as checked. If the projection of $\mathbf{x}_j$ lies within an illuminated region of the shadow image then $\mathbf{x}_j$ is judged to be outside the hull of all hidden objects, it is removed from $\mathbf{X}$, and is not considered in future inside/outside tests. However, if the point is found to lie within the shadowed region of the observation plane, then it is kept for future tests.

As each shadow image is processed, more hidden points are carved out of $\mathbf{X}$, until all that remains are points lying inside the visual hull of the hidden objects, and unchecked points lying in regions that could not be imaged. We take the array of all checked points in $\mathbf{X}$ and convert it to a 3D occupancy grid. Finally, we can use a marching cubes algorithm [2] to convert this occupancy grid into a surface mesh which can be conveniently displayed.

1.3 Additional Stationary Object Reconstructions

We were able to image a large number of stationary objects in the planar testbed. Three of these results are presented in the primary text. Our remaining results are shown in Figures 2 and 3. These results were collected on the same day as the results reported in the main paper, but were excluded from the main paper for the sake of brevity.

1.4 Video Reconstruction Methods

We've shown that our method is capable of producing detailed reconstructions of hidden objects when a large number of shadows is observed. However, in some situations a coarse reconstruction that can be produced using fewer shadows is desirable–if one can capture a few shadows very quickly, one might be able to observe motion in the scene that would be washed out by a longer acquisition time.

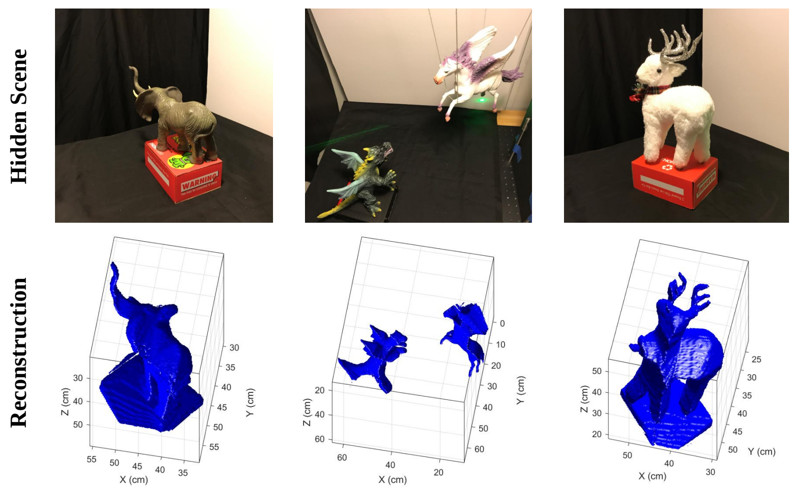

Fig. 2. Additional reconstruction results for stationary hidden scenes. Our method recovers the complex geometry of an elephant (left), a pegasus and dragon (center), and a reindeer (right).

To demonstrate this we produced a video reconstruction of a moving object, which is included in the supplemental material folder along with a line-of-sight video taken with a smartphone. The hidden object was a mannequin that was slowly moved around the hidden area. Although we do not recover the fine details of the mannequin, we are able to recover its approximate size, shape, and position of using only 4 observed shadows per frame.

To achieve this, we continuously loop over a scan pattern consisting of four points. At each point our camera observes a shadow. For each shadow, we perform a fresh space carving operation (see Algorithm 1), resulting in a sequence of single-shadow estimates of the object's shape. We convert each single-shadow reconstruction into a binary occupancy grid, where each element corresponds to a single hidden point in the scene. Inside points are assigned a value of 1, and outside points are set to zero. Each video frame is simply the binary multiplication of the four occupancy grids corresponding to the previous four shadows.

We acquire 15 shadows per second, such that the four point scan pattern can be completed in 0.27 seconds. Space carving was executed on a 55 cm $\times$ 45 cm $\times$ 55 cm grid of hidden points, with a grid spacing of 1 cm. Because video frames are produced using a four-frame sliding window, the result is a 15 FPS video.

The frame rate of our video was limited by the frame rate of the camera used for this experiment. If we had used a camera with a higher frame rate, we would have been able to capture faster motion within the scene. Alternatively,

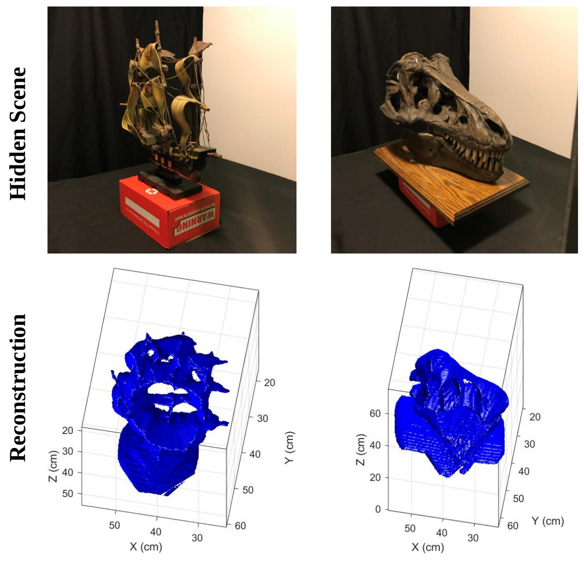

Fig. 3. Additional reconstruction results for stationary hidden scenes. Our method recovers the complex geometry of a model ship (left), and a model of a Tyrannosaurus Rex skull (right). In the case of the skull, occlusions due to the upper skull and the wooden platform prevent accurate reconstruction of the mandible.

we could have increased the number of points in our scan pattern in order to produce more detailed reconstructions of the moving object.

2 Overcoming Geometric and Photometric Challenges of Imaging Behind Occluders

2.1 Derivation of Probabilistic Carving Criterion

We model each voxel in the hidden scene as a binary variable v_i which is either *empty* or *occupied*. An *empty* voxel is assumed to consist of empty space which does not scatter, occlude, or otherwise interact with light that travels through it. A voxel that is *occupied* is assumed to be entirely filled with opaque material such that any light path which passes through an occupied voxel will be occluded. The prior probabilities for each voxel's state are defined as

$$p_e = \mathbb{P}(v_i = \mathit{empty}), \quad p_o = \mathbb{P}(v_i = \mathit{occupied}), \tag{1}$$

and $p_e + p_o = 1$.

Each voxel is subjected to a sequence of inside/outside tests. The result of each test y_j is either *outside* or *inside*. An *outside* or an *inside* result is obtained when a voxel projects to an illuminated or a shadowed region on the visible surface, respectively. We do not consider test results that are inconclusive, such as when a voxel does not project to the visible surface at all, or projects to a shadow boundary.

We assume that the result of all tests are independent when conditioned upon the true state v_i of a voxel. Under this assumption, if a voxel is probed by a sequence of tests $\mathbf{y} = y_1, ...y_N$ then we have

$$p(y_1, ...y_N | v_i) = \prod_{j=1}^{N} p(y_j | v_i). \tag{2}$$

We define η as the *miss probability*—that is, the probability that an occupied voxel erroneously produces an *outside* result—and ξ as the *probability of false alarm*—the probability that an empty voxel erroneously produces an *inside* result. In summary,

$$\eta = p(y_j = \mathit{outside} | v_i = \mathit{occupied}), \quad \xi = p(y_j = \mathit{inside} | v_i = \mathit{empty}). \tag{3}$$

Finally, we use Bayes' rule to calculate the probability that a voxel is occupied given m out N tests produce *outside* results and $n = N - m$ tests produce *inside* results:

$$\mathbb{P}(v_i = o | y_1, ...y_N) = \frac{\eta^m (1-\eta)^n p_o}{(1-\xi)^m \xi^n p_e + \eta^m (1-\eta)^n p_o}. \tag{4}$$

The model that we have presented here has limits. In particular, the assumption stated in Eq. (2) ignores the possibility that a subset of test results could potentially remain co-dependent after they are conditioned on the state of v_i. This might occur if the test results depend on the states of multiple voxels. Despite this assumption we found that the result given in Eq. 4 still proved to be a useful filter that modulated the presence of false carving and false negatives as expected as the parameters η, ξ, p_o, and p_e were tuned.

2.2 Description of Robust Carving Algorithm

In this section we provide a more detailed description of the robust carving reconstruction algorithm used in Sections 5 and 6 of the main text. Our algorithm is presented in Algorithm 2. We represent the hidden scene as a grid of *voxels* $\mathbf{X}$, rather than a grid of points. Our reconstruction algorithm takes a stack of segmented shadow images $\mathbf{S}$ and a set of illumination spot positions $\mathbf{L}$ as input. The 3D positions of all pixels $\mathbf{s}_j$ in each shadow image S_i are also provided (these positions are acquired during data capture using the depth channel of our RGB-D camera). In each frame we determine which pixels $\mathbf{s_j}$ in shadow image S_i are in shadow, and which are lit. For each shadowed or lit pixel, we use Amanatides and Woo's ray tracing method [1] to determine which voxels lie on the line segment that connects that pixel to the illumination spot. We maintain separate counts of the number of rays-to-shadow and rays-to-light that pass through each voxel for a single frame's worth of measurements. If at least one ray-to-shadow and exactly zero rays-to-light pass through a voxel, that voxel is declared to project *inside* the hidden scene's shadow for that frame. Likewise, if at least one ray-to-light and exactly zero rays-to-shadow pass through a voxel, that voxel is declared to project *outside* of the hidden scene's shadow (to an illuminated region) for that frame. Mixed results are not used.

We maintain separate counts of the number of *inside* and *outside* results that are accrued by each voxel. Once all frames have been processed, we use these counts to determine the probability that each voxel is occupied. This probability is determined using Eq. 4. We visualize this occupancy probability grid using the maximum intensity projection functionality in MATLAB's volume viewer application. We can alternatively choose some probability threshold, and simply view a scatter plot of all voxels with a calculated occupancy probability that lies above this threshold.

2.3 Reconstruction without Background Subtraction

In Section 5.3 of the main paper we presented results produced using our robust carving method when shadow-less background observations of visible surfaces could be used to aid in shadow segmentation. Although this strategy might be effective for imaging moving objects or for long-term surveillance applications, there are many application scenarios in which background measurements will not be available.

Here we demonstrate that our method still produces results of similar quality *without* the aid of background measurements. Instead of background subtraction, we use a simple shadow segmentation method that exploits our knowledge of

Algorithm 2 Robust Carving Reconstruction Algorithm

```
input $\mathbf{L}, \mathbf{S}$                    ▷ Laser positions, shadow images
define $\mathbf{X}$                               ▷ Hidden voxel grid
define $\mathbf{V}^{(in)} = \mathbf{0}$, $\mathbf{V}^{(out)} = \mathbf{0}$   ▷ Grids to count number of inside, outside results
define $\mathbf{W}^{(s)} = \mathbf{0}$, $\mathbf{W}^{(l)} = \mathbf{0}$ ▷ Count rays-to-shadow and rays-to-light through each voxel

for $\mathbf{l}_i$ in $\mathbf{L}$ do
    for $\mathbf{s}_j$ in $S_i$ do
        if $s_j$ is in shadow then
            for $x_k \in \mathbf{X}$ that intersects line segment $l_i$-$c_j$ do
                $w_k^{(s)} = w_k^{(s)} + 1.$          ▷ Increment rays-to-shadow
            end for
        end if
        if $s_j$ is lit then
            for $x_k \in \mathbf{X}$ that intersects line segment $l_i$-$c_j$ do
                $w_k^{(l)} = w_k^{(l)} + 1$           ▷ Increment rays-to-light
            end for
        end if
    end for
    for $x_k \in \mathbf{X}$ do
        if $(w_k^{(s)} \geq 1)$ and $(w_k^{(l)} = 0)$ then
            $v_k^{in} = v_k^{in} + 1$                 ▷ Increment num. inside results
        end if
        if $(w_k^{(s)} = 0)$ and $(w_k^{(l)} \geq 1)$ then
            $v_k^{out} = v_k^{out} + 1$               ▷ Increment num. outside results
        end if
    end for
    $\mathbf{W}^{(s)} = \mathbf{0}$                   ▷ Reset ray counting arrays
    $\mathbf{W}^{(l)} = \mathbf{0}$
end for

define $\mathbf{P}$                               ▷ Probability of occupancy grid
define $\mathbb{P}$                               ▷ Voxel occupancy probability function

for $x_k \in \mathbf{X}$ do
    $p_k = \mathbb{P}(v_k^{in}, v_k^{out})$           ▷ Calculate occupancy prob. for each voxel
end for

return $\mathbf{P}$
```

visible surface geometry. We begin by denoising the acquired intensity images using a bilateral filter. Then, for each frame, we use depth channel information to calculate the distance between the illumination spot and each pixel in the region of interest on the opposing surface. We scale the intensity value of each pixel by the square of this distance. We then apply a hand-tuned binary threshold to determine which pixels are lit, and which pixels lie in shadow. A single threshold value is used to process all frames in the image stack. We also choose to ignore pixels that are classified as in-shadow in almost every frame. It is assumed that these pixels have a particularly dark albedo, or lie in a shadow that is cast by the visible surface itself. These pixels are not used in any subsequent inside/outside tests.

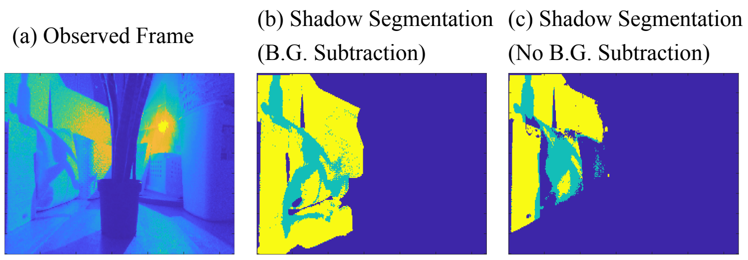
(a) Observed Frame
(b) Shadow Segmentation (B.G. Subtraction)
(c) Shadow Segmentation (No B.G. Subtraction)

Fig. 4. Shadow segmentation comparison. Pixels on left-hand side of occluder in the observed frame (a) are segmented into lit (yellow) and shadowed (teal) regions by our (b) background-assisted and (c) threshold-based shadow segmentation methods.

In Figure 4 we compare the performance of this threshold-based shadow segmentation method with the background-assisted method used to produce the results in Section 5.3 of the main text. Although the theshold-based method classifies more pixels as always-in-shadow than the background-assisted method does, many illuminated pixels are classified correctly, and very few shadowed pixels are misclassified as illuminated. This is important, because an excess of false "lit" classifications (or *misses*) can result in significant false carving.

In Figure 5 we compare reconstruction results produced using each shadow segmentation method. Each result successfully reproduces the form of the mannequin, although there appear to be more false positives when the threshold based segmentation method is used.

Visible Surfaces with Varying Albedo Thus far we have only considered visible scenes consisting of mostly white surfaces with flat albedo. Most scenes in the real world will not share this characteristic. In scenes with varying albedo, observed pixel intensities will be scaled by the reflectivities of the illuminated *and* observed points in addition to the varying distances between these points. This prevents us from using a simple binary threshold to segment shadows. Here

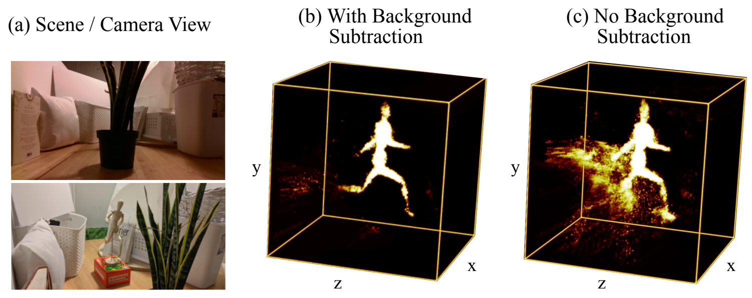
(a) Scene / Camera View
(b) With Background Subtraction
(c) No Background Subtraction
y
x
z
y
x
z

Fig. 5. Effect of shadow segmentation on reconstruction quality. (a) Scene setup and view from the camera. A mannequin is hidden behind an occluder. Voxel occupancy probability maps are produced via robust carving after (b) background-assisted shadow segmentation or (c) threshold-based shadow segmentation methods are used.

we demonstrate that a minor modification to our data capture routine enables robust shadow segmentation when visible surfaces have varying albedo, and this in turn allows us to recover hidden object shapes.

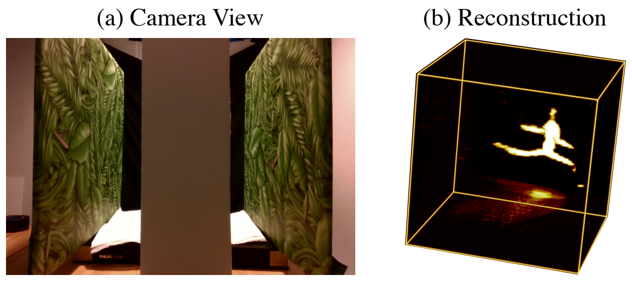

Fig. 6. Visible surfaces with varying albedo. Our method recovers the shape of a mannequin (b) that casts shadows onto visible surfaces with a varying (leafy) albedo pattern, as seen in (a).

We reconstruct the shape of a mannequin that is hidden behind an occluding wall but flanked by two walls covered with a leafy pattern. This scene is shown in Figure 6 (a), and our reconstruction is shown in Figure 6 (b). We make two modifications to our method to enable reconstruction in this setting. First, we observe the brightness of each illuminated point as well as its position. Second, we use this brightness measurement to calculate the expected irradiance received at each pixel in each frame. This calculation relies on a Lambertian light transport model. We scale observed pixel values by the calculated received irradiance, and then observe how this scaled intensity fluctuates over time for each pixel in frame.

We declare a pixel to be in shadow for frames in which the scaled intensity drops below 25% of the maximum scaled intensity value observed for that pixel.

References

1. Amanatides, J., Woo, A.: A fast voxel traversal algorithm for ray tracing. Proceedings of EuroGraphics **87** (08 1987)
2. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. ACM siggraph computer graphics **21**(4), 163–169 (1987)