The Visual Turing Test for Scene Reconstruction Supplementary Material

Qi Shan[†], Riley Adams[†], Brian Curless[†], Yasutaka Furukawa[*], and Steven M. Seitz[*†]

[†]University of Washington [*]Google

1. Performing the Visual Turing Test with Amazon Mechanical Turk

Here we provide additional details and results on the Visual Turing Test. Figure 1 shows a screen shot of the test user interface. We randomized the order in which the reference photo and rendered result were shown to avoid position bias (e.g., 50% of the time, the reference photo appears above the rendered image). Figure 2 illustrates the four resolution levels.

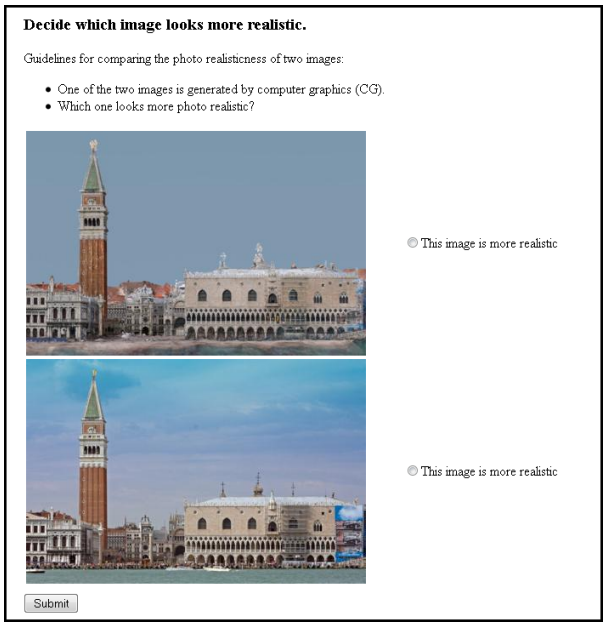
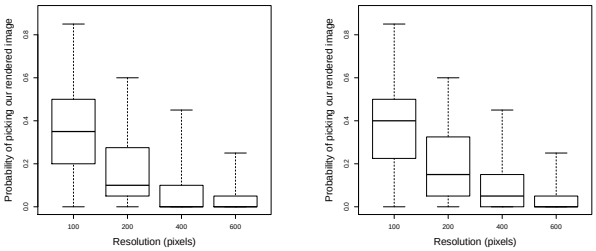


Figure 1. The UI for the Visual Turing Test shown to workers on Amazon Mechanical Turk. In this case, the top image is our rendering, while the bottom one is a real photo.

We show additional typical good and bad image results in Figures 3 and 4. Even with good geometry rendering, subjects are still more likely to choose the reference photo if people are present (e.g., the first example in Figure 4). Figure 6(b) shows the increase in performance if we omit photos containing people; observe that the scores are significantly increased, compared to 6(a). In some cases, the photos themselves may appear unrealistic; for example, 2 of the 100 randomly selected photos are black-and-white. Subjects rate them as less realistic than the (colored) renderings (Figure 5).

Figure 5. Black and white photos are anomalous (left real, right rendered). 10% and 20% test subjects choose the rendered image as more photo-realistic at the 600 resolution level. The numbers increase to 50% and 60% at the 100 resolution level.



(a) On all 100 images. (b) On 75 images with no people.

Figure 6. Performance increase from omitting photos with people.

100px 200px 400px 600px

Figure 2. Test images at different resolutions. For each pair at a given resolution, the reference photo is on the left, and the rendered image is on the right.

70%, 5% 60%, 25% 65%, 15% 85%, 10%

Figure 3. Additional typical good results (top real, bottom rendered). The numbers are the probabilities that our rendered image fools test subjects at resolution levels 100 and 600, respectively.

Figure 4. Additional typical bad results (left real, right rendered). More than 90% of test subjects pick the reference photos as more realistic in every resolution level.

2. Regularization Weight on Per-point Albedo Estimation

For per-point albedo estimation in Section 4.3, we optimize Equation 7 in the paper, reproduced here:

$$\begin{aligned}\underset{\{N_i,a_i,\delta_{i,j}\}}{\operatorname{argmin}} \sum_{j\in V_i} \sqrt{\tilde{R}_{i,j}}\|R_{i,j}(\Theta)-\tilde{R}_{i,j}\|_2^2 \\ +\lambda_1 \sum_{i\in\mathbf{P^C}} \|a_i f(N_i)-\tilde{a}_i f(\tilde{N}_i)\|_2^2 \\ +w_i\|N_i-\tilde{N}_i\|_2^2.\end{aligned}$$

The first term is the data term measuring the image discrepancy between observed pixel intensities and what is predicted by our model. The second term is a regularizer based on cloudy images; it keeps the estimated albedos for points seen in cloudy images close to the estimates recovered by optimizing Equation 5. Reference normals come from the Poisson reconstruction.

In this section, we focus on the third term, a regularizer that encourages adherence to the Poisson normals when the weight w_i is high. Ideally, we would make w_i depend on the first, image discrepancy term in the objective; i.e., when the discrepancy is high, the normal estimation is unreliable, and the weight should be high. Of course, we do not know the magnitude of the image discrepancy term before actually solving Equation (7). Nonetheless, for a subset of points $\mathbf{P^L}$, we have already solved Equation 6 to estimate lighting and thus have computed image discrepancies for those points. Our approach is to use that information to construct per-point weights based on just these image discrepancies.

More precisely, we first compute an average image discrepancy measure d_j^{image} for each image I_j by taking the average of the discrepancy term over points $\mathbf{P_j^L}$ that are in $\mathbf{P^L}$ and visible in I_j:

$$d_j^{image} = \frac{1}{|\mathbf{P_j^L}|}\sum_{P_i\in\mathbf{P_j^L}} \|R_{i,j}(\Theta)-\tilde{R}_{i,j}\|_2^2.$$

Then, we define the reliability r_i^{point} of imagery information at each point P_i by aggregating d_j^{image} over P_i's visible images V_i:

$$r_i^{point} = \sum_{I_j\in V_i} \sqrt{|\mathbf{P_j^L}|}\exp(-\kappa\, d_j^{image}).$$

The exponentiated discrepancy measure from each image is weighted by the square root of the number of contributing points $\mathbf{P_j^L}$. Note that the reliability of a point should increase when it is visible in more images, which is also modeled by the formula above. We used $\kappa = 20$ in our experiments.

Finally, we normalize r_i^{point} so that its mean is 1.0 over the entire point set to give a normalized reliability measure $\hat{r}_i^{point}$, and we then define the regularization weight w_i to be inversely proportional to $\hat{r}_i^{point}$:

$$w_i = \frac{\lambda_2}{\hat{r}_i^{point}},$$

where $\lambda_2 = 0.01$ in our experiments.

3. Additional Constraints on the Optimization

Here we describe additional physical, non-negativity constraints that are imposed on our optimization problems (Equations 3, 5, 6, 7).

- Light intensities (k_j^{sky}, k_j^{sun}) and surface albedos (a_i) must be non-negative. Thus, we add the following constraints to each optimization problem:

$$k_j^{sky} \geq 0, \quad k_j^{sun} \geq 0, \quad a_i \geq 0.$$

- Each lighting direction L_j must be in the upper hemisphere, i.e., must satisfy $L_i \cdot U \geq 0$. We add this as a constraint to the optimization. Note that U is the "up" direction, which is estimated by simply taking the average of the "y-axes" of the input cameras.