Algorithm 1 Rendering loss

Require: Ground Truth material(M_{gt}), Inferred material (M_{I}), Number of specular renderings (N_S), Number of diffuse renderings (N_D)

for n in N_S **do**
  $P_{\text{S}} \leftarrow \text{Uniform}(-1.0, 1.0)$ ▷ Draw a position shift from a uniform distribution between -1 and 1
  $\omega_{\text{V}} \leftarrow \text{Cosine}()$ ▷ Draw a random cosine-distributed 3D view vector
  $P_{\text{V}} \leftarrow (\omega_{\text{V}} e^{\text{Normal}(\mu=0.5,\sigma=0.75)}) + P_{\text{S}}$ ▷ Shifted view position
  $P_{\text{L}} \leftarrow (\omega_{\text{V}} * [-1, -1, 1]) e^{\text{Normal}(\mu=0.5,\sigma^2=0.75)}) + P_{\text{S}}$ ▷ Shifted light position, mirror from view position
  $Ls \leftarrow \text{linspace}(-1, 1, 256)$
  $C \leftarrow \text{concatenate}(\text{meshgrid}(Ls, Ls, 0), \text{axis} = 2)$ ▷ Create a plane coordinate grid to calculate view/light direction
  $\omega_{\text{V}} \leftarrow P_{\text{V}} - C$ ▷ Near-field view direction
  $\omega_{\text{L}} \leftarrow P_{\text{L}} - C$ ▷ Near-field light direction
  $\omega_{\text{LV}}.\text{store}([\omega_{\text{V}}, \omega_{\text{L}}])$ ▷ Store Light/View directions for future renderings
end for
for n in N_D **do**
  $\omega_{\text{V}} \leftarrow \text{Cosine}()$ ▷ Distant view direction
  $\omega_{\text{L}} \leftarrow \text{Cosine}()$ ▷ Distant light direction
  $\omega_{\text{LV}}.store([\omega_{\text{V}}, \omega_{\text{L}}])$ ▷ Store Light/View directions for future renderings
end for
for $\omega_{\text{V}}, \omega_{\text{L}}$ in ω_{LV} **do**
  $R_{\text{gt}}.\text{store}(\text{Render}(GT_M, \omega_{\text{V}}, \omega_{\text{L}}))$ ▷ Render the ground truths using computed directions
  $R_{\text{I}}.\text{store}(\text{Render}(I_M, \omega_{\text{V}}, \omega_{\text{L}})$ ▷ Render the inferred materials using computed directions
end for
return $\text{mean}(|\log(R_{\text{gt}} + 0.01) - \log(R_{\text{I}} + 0.01)|)$ ▷ Compute the mean distance between all stored renderings

Algorithm 2 Cosine-distributed random vector generation

$u_1 \leftarrow \text{Uniform}(0.001, 0.95)$
$u_2 \leftarrow \text{Uniform}(0, 1)$
$r \leftarrow \sqrt{u_1}$
$\phi \leftarrow 2\pi u_2$
$x \leftarrow r\cos(\phi)$
$y \leftarrow r\sin(\phi)$
$z \leftarrow \sqrt{1 - r^2}$
return $[x, y, z]$

Algorithm 3 Rendering Algorithm

Require: Incident lighting direction vectors (ω_{L}), View direction vector (ω_{V}), Material M = [Diffuse map (d), Normal map (n), Specular map (s), Roughness map (r)]

Performed at each surface point separately

$h \leftarrow \text{Normalize}((\omega_{\mathrm{L}} + \omega_{\mathrm{V}})/2)$ ▷ Half-vector computation

$d \leftarrow \frac{d(1-s)}{\pi}$ ▷ Diffuse contribution scaling

$D \leftarrow \frac{1}{\pi}\left[\frac{r^2}{(n\cdot h)^2(r^4-1)+1}\right]^2$ ▷ Cook-Torrance micro-facet normal distribution

$G \leftarrow \frac{1}{(n\cdot\omega_{\mathrm{L}})(1-\frac{r^2}{2})+\frac{r^2}{2}} \frac{1}{(n\cdot\omega_{\mathrm{V}})(1-\frac{r^2}{2})+\frac{r^2}{2}}$ ▷ Cook-Torrance shadowing and masking term

$F \leftarrow s + (1-s)2^{((-5.55473(\omega_{\mathrm{V}}\cdot h))-6.98316)(\omega_{\mathrm{V}}\cdot h)}$ ▷ Fresnel effect approximation

return $\frac{(\frac{FGD}{4}+d)(n\cdot\omega_{\mathrm{L}})}{\omega_{\mathrm{L}}\cdot Z}$ ▷ Final rendering equation compensated for low angles lighting directions regarding ω_{L} and the upright normal direction