8. Appendix

The full MatSim dataset and benchmark, as well as generation code and trained models have been made available in these URLs:

GIT
https://e1.pcloud.link/publink/show?code=kZIiSQZCYU5M4HC
https://icedrive.net/s/A13FWzZ8V2aP9T4ufGQ1N3fBZxDF
https://zenodo.org/record/7390166.Y$_6$cNIBBxH4

9. Building the MatSim Dataset

9.1. Setting objects in the scene

Objects for the dataset were taken from the ShapeNet core2 dataset [12], which has 56000 3D model objects with over 200 categories. Each object is loaded, randomly scaled and rotated, and its original materials are removed. In addition, overlapping faces on the surface are removed.

9.2. UV mapping

Since the SVBRDF (PBR) materials are given as 2D texture maps, assigning a material to a 3D object involves wrapping these 2D maps around the object's surface. Wrapping 2D maps around complex 3D objects (UV mapping) is a rather complex problem. Blender 3.1 contains some automatic wrapping techniques, which we use. A major problem that we encounter is that many objects contain overlapping faces (surfaces), which causes the textures to be wrapped around the same areas several times, causing strong unrealistic visual artifacts. We managed to solve this by removing vertices and faces that were too close to each other using the remove overlapping vertexes tool of Blender. This led to a slight deformation of the shapes of some objects. The wrapping itself depends on various parameters, such as the relative scale, orientation, and translation of the texture map relative to the object, as well as the coordinate system. We randomize all of these parameters between each image to achieve maximum variability.

9.3. Background Illumination and HDRI

The environment and illumination greatly affect the appearance of a material. The main method used by the CGI community to control the background and illumination involves high dynamic range images (HDRIs). These are panoramic images that wrap around the scene and provide 360 illumination. The range of light intensities is from 0 to 6550, compared to 0 to 255 in standard images. This allows HDRIs to represent a much wider range of illumination. We downloaded more than 500 HDRIs from the polyhaven repository [3]. These HDRIs are highly diverse scenes that cover both daytime and nighttime conditions in a large number of indoor and outdoor environments. These HDRIs were used to add both illumination and background

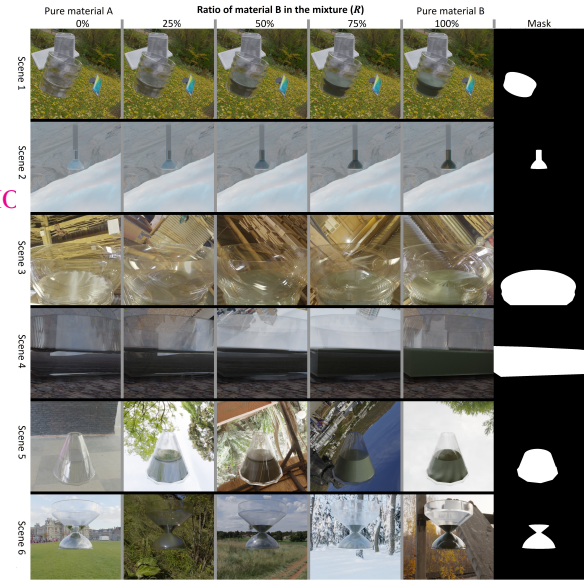

Figure 8. A Set of the MatSim dataset with materials inside transparent containers. The set structure is described in Section 11.2 and Figure 12. The vessel generation is described in Sec. 11.4

to the scenes. To increase diversity, the HDRIs were randomly rotated and scaled, and their intensities were randomly increased or decreased for each scene.

9.4. Adding Ground Plane and Background Objects

Random objects were scattered in the scene to make the environment more diverse and add shadows and back reflections. This was done by loading random objects from the ShapeNet data set and randomly scaling, rotating, and positioning them in the scenes. In addition, a ground plane was generated by adding a flat plane below the object and assigning a random PBR material to it.

9.5. Limitation of the Synthetic Dataset

A major limitation of the dataset is that phase transitions between materials are often not the average between the two states; instead, they are either completely different states (the transition between wood and coal is fire) or they are not uniformly distributed, like the nucleation of ice in water. Both cases are not completely addressed by the dataset and require further work. In addition, the natural image benchmark contains relatively few near-field examples.

9.6. Data Augmentation

A major problem with using only simulated data for learning is the fact that modern cameras significantly modify an image's appearance by smoothing and adjusting colors. To account for this, we use extensive augmentation for the image during training. This includes random Gaussian smoothing, the darkening or brightening of the image,

partial or complete decoloring (to grayscale), and noise addition. Each of these augmentations was applied for about 10%-20% of the image, regardless of whether other augmentations were used. Resizing, cropping, and flipping were performed on almost every image. Combined, this led to an improvement in accuracy of more than 10%-30% (compared to the nonaugmented version).

9.7. Data sampling

Sampling is done by choosing one set for each batch and, from this set, choosing 12 random images (without repetition). Sets with materials inside and outside transparent vessels were selected with equal probability.

9.8. CGI Assets From Artist Repositories

One of the main challenges in creating the MatSim dataset was collecting enough large and diverse examples of backgrounds, objects, and materials. Large-scale repositories of assets that serve the CGI artist community have been available for a while, but remain almost unutilized by the machine-learning community. Utilizing these repositories made it possible for us to create a wide range of scenes on a large scale and with sufficient diversity; this would have been very difficult if we had created these data ourselves. The main three repositories used for this work were CGBookCases [38] and FreePBR [2] for materials, HDRI Haven for environments and light, and ShapeNet for objects.

9.9. Blender Render Setting and Hardware.

The dataset was rendered using Blender 3.1, with CYCLEs rendering, 120 rendering cycles per frame per, images were created with 800X800 resolution and noise reduction mode (smoothing). Nvidia RTX 3090

9.10. Realism vs. Generality in the Dataset Creation

There are two ways to go about generating a synthetic dataset. One approach is to make the dataset as realistic as possible by maximizing the similarity of the generated data to the real world. The other approach is to maximize variability and make the scenes as diverse as possible, even at the cost of realism. In this work, we prefer variability over realism, which means that we assign random materials to random objects and set them in random environments. For example, an image in the dataset may contain a car made of wood on the ground made of metal in a forest surrounded by random objects. This makes the dataset easier to generate and helps the network achieve a much higher level of generalization and identify materials regardless of the environment or the object on which they appear.

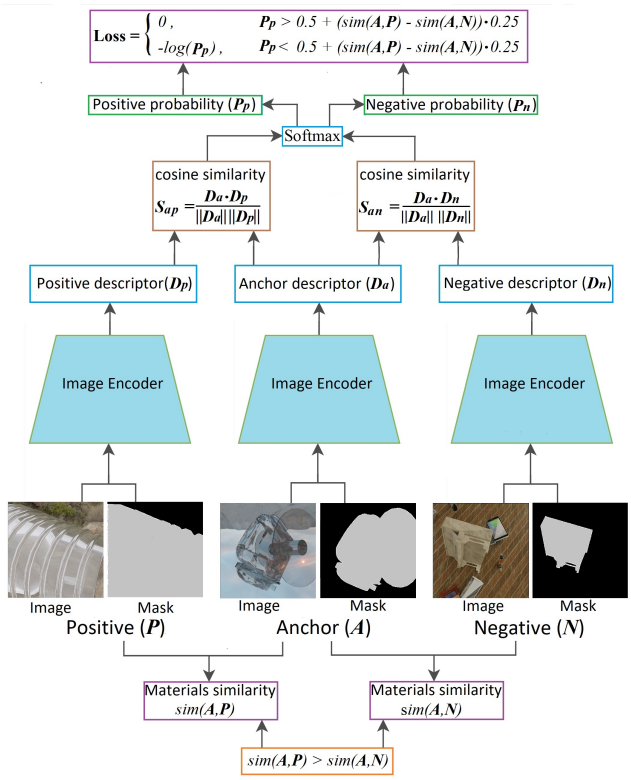

Figure 9. Training and loss function. The loss function is based on cosine similarity with cross-entropy loss. For every three images in the batch, one image is defined as an anchor (A). A second image, specifically the image that has material more similar to the anchor (Section 4.2), is defined as positive (P), and the third image is defined as negative (N). All images and material masks are passed through the neural network to produce descriptors (D_a, D_n, D_p). The cosine similarity is calculated between the descriptor of the anchor and the positive and negative examples. These cosine similarities are the input for the softmax function, which returns the probability of a match between the anchor and the positive image (P_p). If this probability is below the threshold defined as $0.5 + (sim(A,P) - sim(A,N)) \cdot 0.25$, we calculate the cross-entropy loss ($Loss = (-\log(P_p))$; otherwise, the loss is set to zero. $sim(A,P) = 1 - |R(A) - R(P)|$ is the similarity between the anchor (A) material and the positive material P, $R(A)$ is the mixture ratio in material A (Section 11.3).

10. Training and Architecture

We examined several standard image encoder architectures and found that ConvNeXt [20] gave the best results. Two modifications were made in ConvNeXt: First, the network input was changed from a standard three-channel image (RGB) to an image plus the mask of the region containing the material. This was done by simply adding the ROI mask as another layer to the RGB image and changing the first layer of the network so that it could receive a four-layer image (R, G, B, mask) instead of a three-layer im-

age (RGB). In addition, the final linear layer of ConvNeXt was changed to produce a 512-value vector (instead of 1000 ImageNet classes). This vector was then normalized using L2 normalization and used as the output descriptor. The training itself was done using the AdamW optimizer for ten epochs (200000 steps with a batch size of 12) using a single RTX3090. This took around three days per training session. We used the ConvNeXt base model trained on ImageNet.

10.1. General Loss Function

The loss function is described in Figure 9. Similar to recent works, we use cosine similarity combined with cross-entropy loss for training [13]. An important aspect of our data set is that the levels of similarity between materials are continuous and can have any value between zero and one (depending on the mixture; Figure 12). This was accounted for by using a semi-hard loss with a threshold depending on the similarity level. We found that a semi-hard loss, in which only loss terms that are below some threshold are used, works better than a hard loss (in which all loss terms are used). In this case, the threshold was controlled by the difference in similarities between the anchor material and the positive and negative examples.

10.2. Material Similarity

We define the real similarity of the two materials as the difference between their mixture ratios (R, Figure 12). As described in Section 11.3, each material in a set is a linear mixture between two materials A and B. We define this ratio as $R(0 < R < 1)$. The similarity between the materials in the two images (i_1 and i_2) is defined as:

$$sim(i_1, i_2) = 1 - |R(i_1) - R(i_2)|$$

Where $R(i_1)$ is the mixture ratio in image i_1 (Section 11.3). Assuming both images are from the same set.

10.3. Predicted Material Similarity

The predicted material similarity between two images, A and N, is calculated by first passing the images and their corresponding masks through the neural network and predicting the descriptors D_a and D_n, which are then normalized using L2 normalization. The predicted similarity is the cosine similarity between the two descriptors (S_{an}, Figure 9).

10.4. Loss Calculation

The training was performed as shown in Figure 9: For each batch, we randomly selected a single set (Section 11.2) and sampled 12 random images from this set. The loss for every three images in the batch was calculated separately using the procedure shown in Figure 9. One of the three images was chosen as an anchor (A). The similarity between the material in the anchor image and the other two images was calculated as described in Section 10.2. The image that was more similar to the anchor was defined as positive (P), and the other image was defined as negative (N); hence, $sim(A, P) > sim(A, N)$. If the similarities of the anchor to the two images were equal ($sim(A, P) = sim(A, N)$), the loss for this triplet was set to zero. Next, each of the three images and their material masks were passed through the neural net to predict the material descriptors (Section 10, Figure 9). The cosine similarities between the descriptors of the anchor and the negative and positive images were calculated as described in Section 10.3. These similarities were then passed to a softmax function to calculate the probability of a match between the anchor and the positive image:

$$P_p = \frac{e^{S_{ap}/t}}{e^{S_{ap}/t} + e^{S_{an}/t}}. \quad (1)$$

S_{ap} and S_{an} are the cosine similarity between the anchor (A) and the negative(N) and positive(P) descriptors, respectively, P_p is the probability for a match between the anchor and the positive image, and t is a constant (0.2 in our case). We then calculate the semi-hard loss using the following condition: If $P_p > 0.5 + (sim(A, P) - sim(A, N)) \cdot 0.25$, the loss is set to zero; otherwise, we calculate the standard cross-entropy loss ($Loss = -log(P_p)$).

11. The MatSim Synthetic Dataset Generation, UNCUT

Note that this section is the same as Section 3 in the main paper, but a little longer with some paragraphs and images that were removed from the original section appearing here. The goal of the MatSim dataset is to train a computer vision system capable of recognizing any visually distinguished material state and texture on any surface in any reasonable environment. The visual appearance of a material depends not only on its physical properties but also on the object's shape, environment, and illumination. If during training, any of these properties are restricted in some way (for example, only indoor scenes are used) or correlated with other properties (for example, wood materials only appear on trees), the net is unlikely to achieve a true generalization for material recognition [5, 18].

11.1. Generation Procedure

In order to achieve the above goals, the dataset needs to be extremely diverse in terms of objects, environments, and materials. To achieve this, we utilize large-scale CGI artist repositories [1, 38] used for animation, and computer games. We use thousands of highly diverse physics-based rendering materials (SVBRDF / PBR) repositories to simulate realistic materials [25]. We overlay the textures materials on 3D objects taken from the ShapeNet dataset with

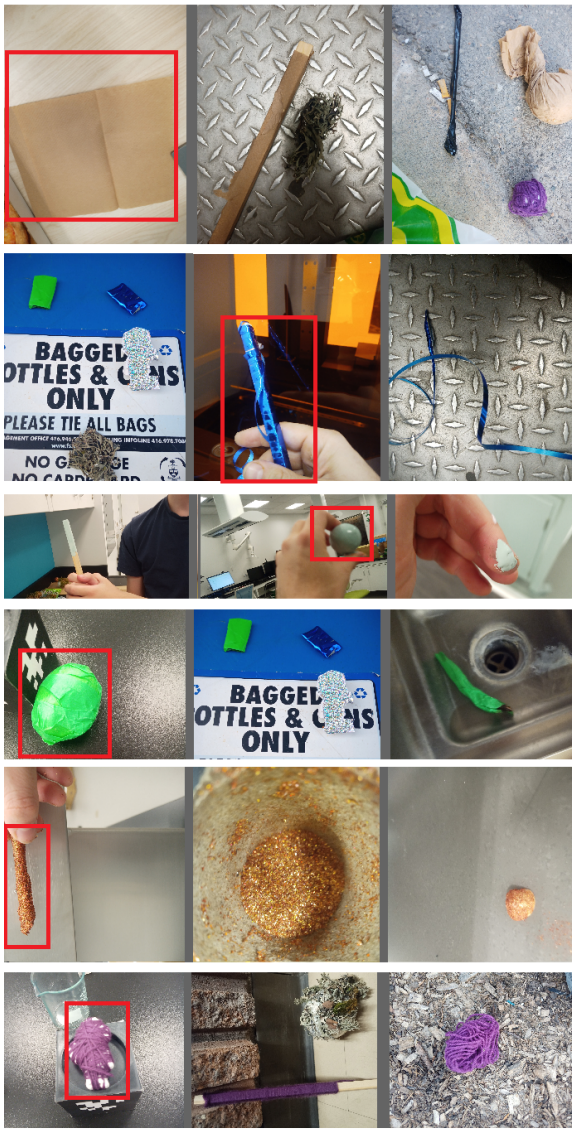

Figure 10. Samples from the second benchmark: uncorrelated materials and objects. Random materials cover random objects to avoid material and object correlation. Each row corresponds to one type of material. For clarity, a red square was used to mark the material in each row (This doesn't appear in the actual benchmark). Material masks are not shown but are available in the benchmark.

tens of thousands of different objects [12] and hundreds of categories; these objects are then placed in scenes with random illumination and backgrounds utilizing the Polyh-Haven repository for HDRI images [3]. The HDRI image is wrapped around the scene and provides a realistic 360-degree background and illumination to the scene (Figure 11). Combining these repositories allows us to generate a large-scale, highly diverse dataset (Figure 11). The large number of materials forces the network to generalize instead of identifying only specific classes. The fact that ev-

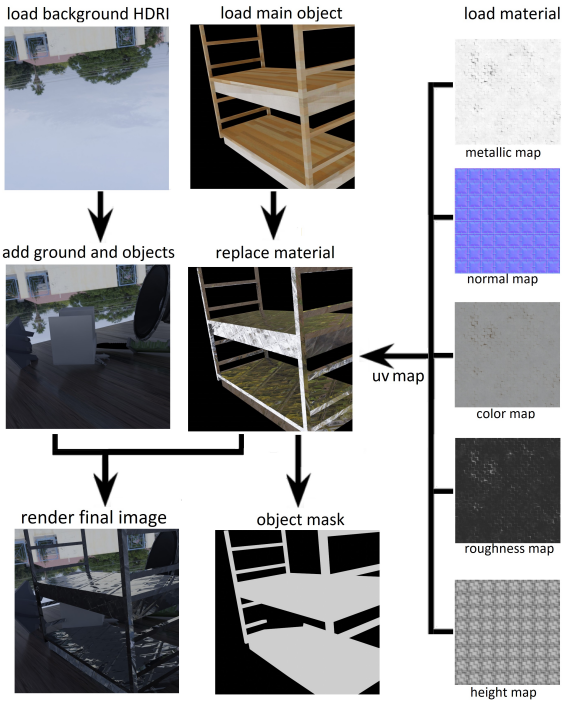

Figure 11. Dataset creation: 1) CGI Materials have been randomly created or downloaded from large-scale artist repositories (CG-BookCases [38], FreePBR [2]). 2) The material is UV mapped on the surface of a random object loaded from the ShapeNet dataset [12].3) Random background and illumination are loaded from the HDRI Haven repository [3]. 5) Ground plane and background objects are added. 6) and the scene is rendered.

ery material can be used on any object in any environment means that the net has to identify the material everywhere and prevents the net from associating the material with specific objects or environments. Gradual transformations between materials in the dataset allow the net to detect gradual transitions between materials. Additionally, rendering some of the materials inside transparent containers allows the network to learn to recognize materials stored inside glass vessels.

11.2. General Dataset Structure

The dataset is divided into sets; each set contains two random materials and six scenes involving a gradual transition between these two materials (Figure 12). The objects, backgrounds, and environment are randomly selected for each scene separately (Figure 11). Each scene involves one static main object and a static camera, which are both positioned randomly, and both remain unchanged for the scene (Appendix 10.1). The object's material is gradually changed between images in the scene (Figure 12). For each scene, we rendered five images with different mixtures of the two materials. The mixture ratios (R) of the

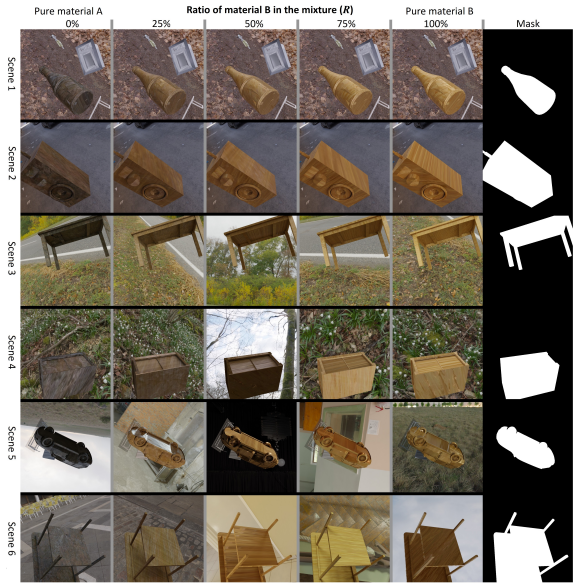

Figure 12. Dataset structure. The dataset is composed of sets. Each set involves two materials and six scenes with a gradual transition between the two materials. Each image column corresponds to a different mixing ratio (R) of the two materials. The ratio of the mixture of the two materials is given in the top column. All images in the same column involve the same material mixture on different objects and in different environments. Each of the six scenes involves one main object. The material on this object gradually transitions from one material to another. The mask of the object is given in the right column. For scenes 1–2, the background remains exactly the same for all images in the scene. For scenes 3–4, the background HDRI is randomly rotated between images, leading to small changes in illumination. For scenes 5–6, the background HDRI is completely replaced between images, leading to large changes in illumination.

two materials are 0%, 25%, 50%, 75%, and 100%. With 0% means that the object is only made of material A, while 100% means that the object is completely made of material B, and 50% indicates an equal mixture of the two materials (Section 11.3). For scenes 1–2, all aspects of the scene remain the same. Only the object material is gradually changed between images. Furthermore, the translation and rotation of the texture UV mapping to the object were randomly changed between each render (Appendix 10.2). For scenes 3–4, the procedure is the same, except that the HDRI background is randomly rotated between images (Appendix 10.3), leading to a small variation in illumination. For scenes 5–6, the HDRI background for each image is randomly replaced, leading to a large variation in illumination. This provides almost any possible variation of the appearance of each material. Since all the scenes in a set are composed of the same two materials, it is possible to compare the appearance of the materials between two scenes with different objects and backgrounds, and light. The gradual transition allows the network to learn to distinguish between highly similar materials. Some scenes contain several random objects and a ground plane for variability (Appendix 10.4). Since a scene can contain many background objects, for each scene, we provide the mask (region) of the object on which the material is used (Figure 12, right). If the material is uniform (BSDF), the values of the material properties (color, transparency, etc.) are given in the dataset. Altogether, 30k sets, with about 1 million images, were rendered.

11.3. Materials Representation, Mixtures and Gradual Transformations

The appearance of materials is mostly controlled by their surface-scattering properties. These properties are often referred to as bidirectional reflectance (BRDF) and the more general bidirectional scattering function (BSDF) Figure 13a) [4, 6]). Each surface point has a set of properties, such as roughness, transmittance, and color, that determine the surface appearance. If the material is uniform, it could be represented as a set of values for each property across the entire surface. In Blender3D [9], this is done using the BSDF node (Figure 13a). Creating a random material could be done by setting a random value for each property. Mixing two such materials can be achieved by using a weighted average of the values of each property from each material: $P_{\text{mix}} = R \cdot P_a + (1 - R) \cdot P_b$ where P is the value of a property in materials a,b and R the mix ratio. Gradual transition between materials is achieved by creating different mixture ratios ($R = 0, 0.25, 0.5, 0.75, 1$ Figure 12). Most materials in the world are not uniform and have unique textures, which means different properties for each point on the surface Figure 13b. Such materials can be represented as spatially variable BRDF or SVBRDF (often called PBR materials [25]). This means that instead of a single value for each property, we have a 2D texture map that represents the spatial distribution of this property on the surface (Figure 13b). This 2D map is then wrapped around the object using a process called UV mapping to give each surface point its property. Mixing two textured materials is achieved by a pixel-wise weighted average of two texture maps into a new texture map. In other words, each pixel in the texture map of a given property is the average of the corresponding pixels in the two materials that are mixed: $P_{\text{mix}}(u, v) = R \cdot P_a(u, v) + (1-R) \cdot P_b(u, v)$. Where $P(u, v)$ is the property in surface position u,v. Gradual transition between materials A and B is again achieved by different mixtures ratios(R). In order to increase variability, texture maps of different materials can be rotated and rescaled relative to the other material before mixing (but not relative to maps of the same material). Unlike uniform BSDF textures, it's not possible to create textures by assigning random values, as this will just create noise. We, therefore, got

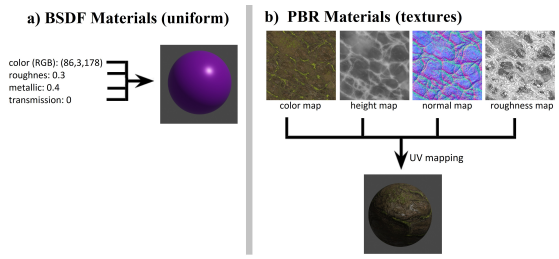

Figure 13. A material's visual appearance is controlled by several main properties (color, transparency, etc.). For uniform materials (BSDF), each property has a single value across the surface. Textured materials are represented by texture maps for each property. These maps are wrapped around the object (UV mapping) and provide properties for each point on the object's surface.

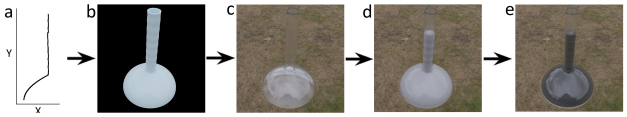

Figure 14. Procedurally generating material inside transparent containers. a) A random 2D curve is generated by combining random polynomial and trigonometric functions. b) The curve is used as a profile for the symmetric 3D object. c,d) The object is assigned random transparent materials and content objects. e) The content object is assigned a material.

a large number of textured materials by downloading 4400 textures from free large-scale artists' repositories. These are highly diverse realistic materials scanned or generated for CGI purposes. To further increase this set, we mix two or more materials as described above.

11.4. Materials in Transparent Vessels

Liquids and other materials in kitchens, hospitals, and labs are usually handled inside transparent containers (glassware, flasks, tubes, etc.). To support these applications, we generated scenes in which we put the material inside a transparent container. The vessel object was procedurally generated. The curvature of the transparent vessel was generated by creating a random 2D curve (by randomly combining linear, polynomial, and trigonometric functions; Figure 14). This curve was used as the profile of a symmetric vessel by creating a cylindrical (or other symmetrical) shape with the curve as the vertical profile (Figure 14). In some cases, the shape was also randomly stretched to create more variability. The vessel object was assigned a random transparent material and a random thickness. The content of the vessel was either a random object loaded from ShapeNet or a mesh that filled the bottom part of the vessel (similar to static liquid or powder). As before, the content was assigned a random material. Otherwise, the creation of the data set was the same as in Section 11.2. Examples can be seen in Figure 8 (Appendix).

References

[1] Ambientcg, free pbr textures repositories. https://ambientcg.com/. 3, 4, 11

[2] freepbr, free pbr textures repositories. https://freepbr.com/. 3, 4, 10, 12

[3] Polyhaven free hdri repository. https://polyhaven.com. 3, 4, 9, 12

[4] Clara Asmail. Bidirectional scattering distribution function (bsdf): a systematized bibliography. *Journal of research of the National Institute of Standards and Technology*, 96(2):215, 1991. 5, 13

[5] Laura Alexandra Daza Barragan, Jordi Pont-Tuset, and Pablo Arbelaez. Adversarially robust panoptic segmentation (arpas) benchmark. 2022. 3, 6, 11

[6] Frederick O Bartell, Eustace L Dereniak, and William L Wolfe. The theory and measurement of bidirectional reflectance distribution function (brdf) and bidirectional transmittance distribution function (btdf). In *Radiation scattering in optical systems*, volume 257, pages 154–160. SPIE, 1981. 5, 13

[7] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Opensurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on graphics (TOG)*, 32(4):1–17, 2013. 2, 3, 7, 8

[8] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Material recognition in the wild with the materials in context database. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3479–3487, 2015. 2, 3, 7, 8

[9] blender.org Community. Blender: A 3d modelling and rendering. https://www.blender.org/. 5, 13

[10] Ioannis K Brilakis, Lucio Soibelman, and Yoshihisa Shinagawa. Construction site image retrieval based on material cluster recognition. *Advanced Engineering Informatics*, 20(4):443–452, 2006. 3

[11] Alice C Chadwick and RW Kentridge. The perception of gloss: A review. *Vision research*, 109:221–235, 2015. 3

[12] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 3, 4, 9, 12

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 3, 6, 11

[14] Kristin J Dana, Bram Van Ginneken, Shree K Nayar, and Jan J Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions On Graphics (TOG)*, 18(1):1–34, 1999. 2, 3

[15] Sagi Eppel, Haoping Xu, Mor Bismuth, and Alan Aspuru-Guzik. Computer vision for recognition of materials and vessels in chemistry lab settings and the vector-labpics data set. *ACS central science*, 6(10):1743–1752, 2020. 2, 3

[16] Leon Eversberg and Jens Lambrecht. Generating images with physics-based rendering for an industrial object detec-

tion task: Realism versus domain randomization. *Sensors*, 21(23):7901, 2021. 2, 3

[17] Eric Hayman, Barbara Caputo, Mario Fritz, and Jan-Olof Eklundh. On the significance of real-world conditions for material classification. In *European conference on computer vision*, pages 253–266. Springer, 2004. 2, 3

[18] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021. 2, 3, 6, 11

[19] Manuel Lagunas, Sandra Malpica, Ana Serrano, Elena Garces, Diego Gutierrez, and Belen Masia. A similarity measure for material appearance. *arXiv preprint arXiv:1905.01562*, 2019. 3, 7, 8

[20] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022. 10

[21] Abhimitra Meka, Maxim Maximov, Michael Zollhoefer, Avishek Chatterjee, Hans-Peter Seidel, Christian Richardt, and Christian Theobalt. Lime: Live intrinsic material estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6315–6324, 2018. 3

[22] Perseverança Mungofa, Arnold Schumann, and Laura Waldo. Chemical crystal identification with deep learning machine vision. *BMC Research Notes*, 11(1):1–6, 2018. 3

[23] Diego Inácio Patrício and Rafael Rieder. Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review. *Computers and electronics in agriculture*, 153:69–81, 2018. 2, 3

[24] Maxine Perroni-Scharf, Kalyan Sunkavalli, Jonathan Eisenmann, and Yannick Hold-Geoffroy. Material swapping for 3d scenes using a learnt material similarity measure. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2034–2043, 2022. 3

[25] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016. 3, 5, 11, 13

[26] Gabriella Pizzuto, Jacopo De Berardinis, Louis Longley, Hatem Fakhruldeen, and Andrew I Cooper. Solis: Autonomous solubility screening using deep neural networks. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2022. 3

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 3, 6, 7, 8

[28] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022. 3, 6, 7, 8

[29] Gabriel Schwartz and Ko Nishino. Recognizing material properties from images. *IEEE transactions on pattern analysis and machine intelligence*, 42(8):1981–1995, 2019. 3

[30] Lavanya Sharan, Ruth Rosenholtz, and Edward Adelson. Material perception: What can you see in a brief glance? *Journal of Vision*, 9(8):784–784, 2009. 2, 3, 7, 8

[31] Pallavi Srivastava, Aasheesh Shukla, and Atul Bansal. A comprehensive review on soil classification using deep learning and computer vision techniques. *Multimedia Tools and Applications*, 80:14887–14914, 2021. 3

[32] Ying Sun and Zhaolin Gu. Using computer vision to recognize construction material: A trustworthy dataset perspective. *Resources, Conservation and Recycling*, 183:106362, 2022. 2, 3

[33] Paul Upchurch and Ransen Niu. A dense material segmentation dataset for indoor and outdoor scene parsing. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VIII*, pages 450–466. Springer, 2022. 3, 7, 8

[34] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X*, pages 268–285. Springer, 2020. 7

[35] Raquel Vidaurre, Dan Casas, Elena Garces, and Jorge Lopez-Moreno. Brdf estimation of complex materials with nested learning. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1347–1356. IEEE, 2019. 3

[36] Sebastian Weiss, Robert Maier, Daniel Cremers, Rudiger Westermann, and Nils Thuerey. Correspondence-free material reconstruction using sparse surface constraints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4686–4695, 2020. 3

[37] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European conference on computer vision (ECCV)*, pages 418–434, 2018. 2

[38] Dorian Zgraggen. Cgbookcase free pbr textures library. https://www.cgbookcase.com/. 3, 4, 10, 11, 12

[39] Song Zhang, Yumiao Chen, Zhongliang Yang, and Hugh Gong. Computer vision based two-stage waste recognition-retrieval algorithm for waste classification. *Resources, Conservation and Recycling*, 169:105543, 2021. 2, 3

[40] Junwei Zheng, Jiaming Zhang, Kailun Yang, Kunyu Peng, and Rainer Stiefelhagen. Materobot: Material recognition in wearable robotics for people with visual impairments. *arXiv preprint arXiv:2302.14595*, 2023. 2