Multi-sensor large-scale dataset for multi-view 3D reconstruction Supplementary material

Oleg Voynov[1,2] Gleb Bobrovskikh[1] Pavel Karpyshev[1] Saveliy Galochkin[1]
Andrei-Timotei Ardelean[1] Arseniy Bozhenko[1] Ekaterina Karmanova[1] Pavel Kopanev[1]
Yaroslav Labutin-Rymsho[3] Ruslan Rakhimov[1] Aleksandr Safin[1] Valerii Serpiva[1]
Alexey Artemov[4*] Evgeny Burnaev[1,2] Dzmitry Tsetserukou[1] Denis Zorin[5]

[1]Skolkovo Institute of Science and Technology [2]Artificial Intelligence Research Institute
[3]Moscow Engineering Physics Institute [4]Technical University of Munich [5]New York University

In Table 1 we show an extended comparison of our dataset to a number of relevant datasets. In Appendices A and B we discuss the choice of lighting variations in our dataset, and the choice of sensors. In Appendices C and D we provide details of data acquisition and camera calibration processes. In Appendices E and F we provide details of testing and evaluation of 3D reconstruction methods on our dataset. In Appendix G we show complete evaluation results. Finally, in Appendix H we describe and discuss the variability of key surface reflection parameters in our dataset.

A. Lighting variability

In Figure 1 we illustrate all 14 lighting setups in our dataset. *Ambient Low* and *Flash Low* correspond to *real-time / high-noise* camera settings for the ambient diffuse lighting and the phone flashlight. We aimed to provide a broad range of realistic lighting conditions: directional light sources and flashlights of the phones provide eight samples of "hard" light, typical, for example, for streetlight; soft-boxes provide three samples of diffuse light, typical for indoor illumination; LED strips imitate ambient light, typical for cloudy weather.

B. Sensors

We aimed to include commodity RGB-D sensors with different properties. Smartphones are ubiquitous, and are increasingly commonly augmented with a depth sensor, while Kinect and RealSense devices represent dedicated RGB-D cameras. We included smartphones that capture depth with a time-of-flight sensor, Kinect v2 also uses a time-of-flight sensor but with a higher resolution and accuracy, and the RealSense device uses stereo-matching of infra-red images (a different technology). These devices capture depth maps with different resolution, level of noise, and different artefacts, as briefly illustrated in Figure 4. The structured-light scanner provides the reference 3D data for these devices.

These devices include commodity RGB sensors with different resolutions; we supplemented them with industrial RGB cameras with high-quality optics and low-noise sensors. The pair of industrial RGB cameras can serve as yet another source of depth maps based on stereo-matching of RGB images, in contrast to IR images in RealSense.

The data captured in the same environment with different sensors can be used to test generalization ability of computer vision methods, or to train a generator of synthetic data to reproduce a specific sensor, etc.

C. Data acquisition details

Lens settings. We set the focal distance for the industrial RGB cameras and for the cameras of the phones to the expected average distance from the cameras to the surface of the scanned object, namely 62.5 cm. To extend the depth of field for the industrial cameras to the whole scanning area, we set the aperture to the minimal value that does not cause any visual blur due to diffraction. We kept the lens parameters for Kinect and RealSense at their factory settings.

White balance. We set the white balance for all RGB sensors fixed at the start of each scanning session (*i.e.*, daily), using a black-and-white calibration pattern. Most of our light sources have the same light temperature so that their light appears white under this setting, except for the ambient illumination which has a somewhat higher green component, and for the flashlight of the phones which has a yellow tint. We did not fix the white balance setting for Kinect, for which this control is not available.

Exposure and gain. For Kinect, it is not possible to control camera exposure and gain directly. Instead, the built-in auto-exposure function is permanently turned on, which sets the exposure and gain so that the mean pixel value over the whole image is 50% gray. Since in our setup a large portion of the image is the black background, the built-in algorithm produces over-exposed images. To minimize this effect, we added a dimming light filter mounted on a servomotor to the

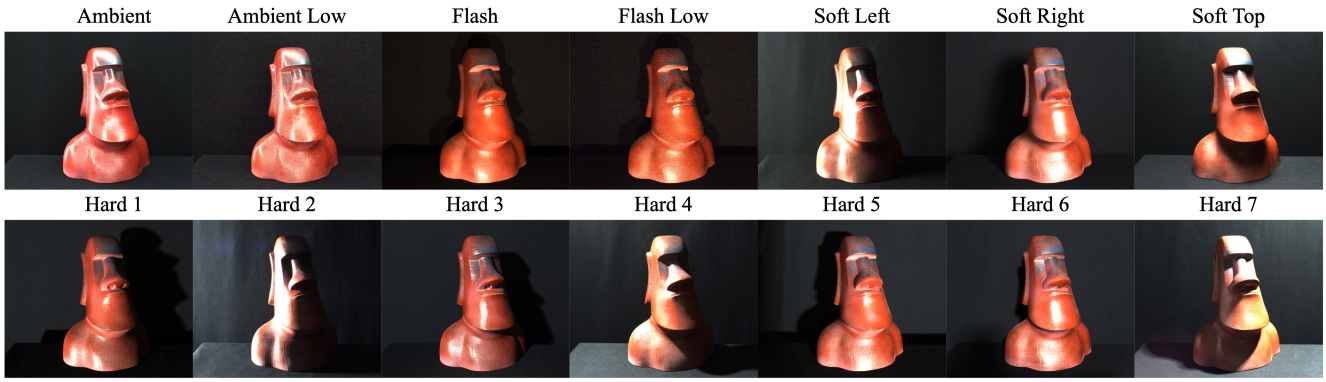

Figure 1. **Lighting setups in our dataset:** ambient lighting, a flashlight attached to the camera, three variants of soft light coming from different directions, and seven variants of hard light. For the ambient and the flash lighting there is an additional low exposure variant: notice a higher noise level in Ambient Low and Flash Low.

rig, which is placed in front of the RGB camera of Kinect automatically when the bright light sources are activated, and removed for the dim ones.

D. Camera calibration details

The original implementation of the calibration pipeline of [18] supports calibration of rigid camera rigs, however, a straightforward application of this pipeline for our camera rig in its entirety proved to be numerically unstable due to the properties of the setup. Firstly, the included sensors have a large variation in the field of view (from 30° for the SLS camera to 90° for the IR sensors of RealSense) and resolution (from 0.04 MPix for the IR sensors of the phones to 40 MPix for their RGB sensors). Secondly, the focus of the cameras of the phones, being fixed programmatically, fluctuates slightly over time, which we relate to thermal deformations of the device (see [4] for a study of such an effect). Finally, the camera rig deforms slightly depending on its tilt in different scanning positions. To avoid the loss of accuracy we split the calibration procedure into several steps.

First, we obtained intrinsic camera models for each sensor independently. Then, for the sensors with a relatively high resolution and stable focus, namely the SLS and all RGB sensors except the sensors of the phones, we estimated their relative position within the rig for its vertical orientation. Next, we estimated the relative position of RGB sensors of the phones within the rig, w.r.t. the other sensors. After that, we estimated the position of each IR (depth) sensor w.r.t. the RGB sensor of the respective device, assuming that the sensors are fixed rigidly to the frame of the device. Finally, we estimated the position of each RGB sensor individually for different scanning positions of the robotic arm, and then linked the positions of all sensors together through a scanning position with the vertical orientation of the rig.

This whole procedure required capturing thousands of images of the calibration pattern, which we almost fully automated with the use of the robotic arm, except for several manual reorientations of the pattern.

In Figure 2 we show distributions of calibration error for each sensor at different stages of the calibration procedure. Note that the resolution of the RGB sensors of the phones is relatively high compared to the other RGB sensors so a higher value of the calibration error measured in pixels is expected.

At the bottom of Figure 2 we show calibration errors for the straightforward application of the calibration pipeline to our camera rig. Here, we estimated the relative positions for a subset of sensors within the rig simultaneously. The mean error of the straightforward approach is 1.8-5.7 times higher depending on the sensor compared to ours.

E. Parameters of tested methods

We tested all methods using ground-truth camera poses and intrinsic camera models, after the refinements of camera poses and depth camera calibration. We used RGB images from the right industrial camera and the depth maps from Kinect at full resolution after removing distortion (which resulted in a small amount of cropping at the boundaries), specifically, 2368×1952 for RGB and 496×400 for depth.

For each method, we tried to pick the values of the method parameters which resulted in the best reconstructions on our dataset, based on 5-10 typical scenes. We describe the parameters using the original notations from the respective works.

We tested **COLMAP** [14–16] with parameters set to their "performance" values recommended in the software documentation. Specifically, for feature extraction, we enabled estimation of affine shape of SIFT features, and enabled the more discriminative DSP-SIFT features instead of plain SIFT; for feature matching, we disabled estimation of multiple geometric models per image pair; for patch-match stereo, we enabled the regularized geometric consistency term; finally, for stereo fusion we set the minimum number of fused pixels to produce a point to 3, the maximum relative difference between measured and projected pixels to 1, and the maximum depth error to 1 mm.

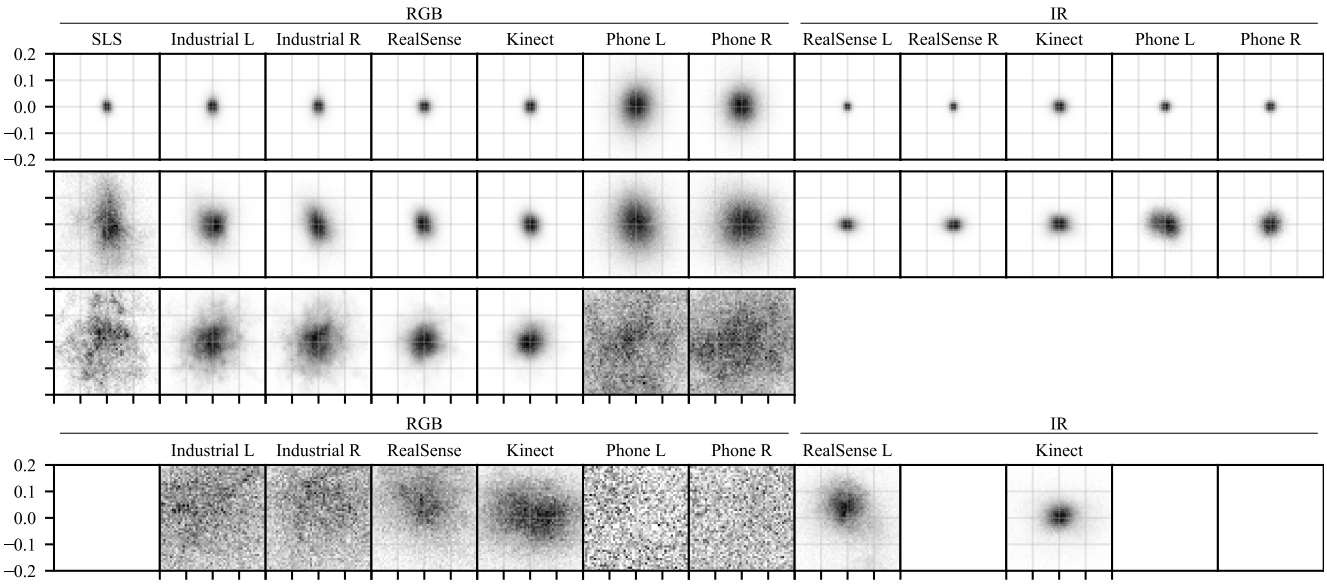

Figure 2. **Error distributions for camera calibration,** as histograms of 2D differences between the detection of a feature on the calibration pattern and its projection from 3D to the image. The range of each histogram is $[-0.2, 0.2]$ pixels in each dimension. Each histogram represents one sensor at one of the three steps of the calibration procedure: estimation of intrinsic camera models (first row), estimation of relative position of the sensor within the rig (second row), estimation of position of the sensor on the scanning trajectory (third row). The histograms for the IR sensors in the third row are not shown, since the trajectory of the IR sensors is calculated directly from the trajectory of their RGB companions. The fourth row shows the errors for the baseline approach of estimation of relative positions of the sensors within the rig (compare with the second row).

We tested **ACMP** [27, 28] with the default parameters used in the source code. For this MVS method and the two methods below we sampled the depth hypotheses uniformly from 473 mm to 983 mm with a 2 mm resolution, and for view selection used the strategy proposed in [30], applied to the reference SL scan instead of a sparse reconstruction of the scene.

To test **VisMVSNet** [32, 33] we trained it from scratch on BlendedMVG dataset [29], which is an extended version of the BlendedMVS dataset [31], originally used by the authors of the method. We used the original training parameters and trained the network on a single Nvidia GTX 1080Ti GPU for 2 epochs with a batch size of 2 (342K iterations), using Adam optimizer [7] with a learning rate of 10^{-4}.

We tested VisMVSNet with the number of neighboring source images $N_v = 7$, the numbers of depth hypotheses $N_{d,1}, N_{d,2}, N_{d,3} = 64, 32, 16$, the minimal number of consistent views during fusion $N_f = 4$, and the fusion probability thresholds $p_{t,1}, p_{t,2}, p_{t,3} = 0.8, 0.7, 0.8$.

We tested **UniMVSNet** [11, 12] using the published model trained on DTU dataset [6] and fine-tuned on BlendedMVS dataset. We used the number of neighboring source images $N = 11$, the numbers of depth hypotheses $M_1, M_2, M_3 = 64, 32, 16$, and the fusion probability thresholds $\phi_1, \phi_2, \phi_3 = 0.1, 0.15, 0.9$.

We tested **NeuS** [22,23] with the original hyperparameters, optimizing the network for 300K iterations. To accelerate convergence and prevent oversmoothing of reconstruction we cropped the input images for this method to the bounding box of the object expanded by ~ 10 pixels.

To test **TSDF Fusion** [3, 5] we used an implementation of this algorithm from Open3D library [34], with a voxel size of 3 mm, and a TSDF truncation distance of 2 cm. We additionally tested an implementation of this algorithm from [8, 9] and obtained very similar results; we do not report them.

We tested **SurfelMeshing** [17, 19] with the number of inliers for depth filtering set to 1, the depth map erosion radius set to 0.1, and with 1 iteration of median filtering.

To test **RoutedFusion** [24, 25] we trained it from scratch on ModelNet dataset [26]. We used the original training parameters for ShapeNet dataset [2], with an increased level of synthetic noise $\sigma = 0.01$, as suggested by the authors. We tested RoutedFusion with a grid resolution of 384^3, which corresponds to a voxel size of 2.6 mm.

We tested **Neural RGB-D Reconstruction** [1, 13] with the original hyperparameters, optimizing the network until convergence for 200K iterations, and sampling the coarse points on the ray for every 2 mm.

F. Evaluation details

We evaluated 3D reconstruction methods using *precision* $P(\tau)$ defined as the percentage of reconstruction points which are closer to the reference surface than a distance threshold τ; *recall* $R(\tau)$ defined as the percentage of reference points which are closer to the reconstruction than the distance threshold; and *F-score* $F(\tau)$ defined as the

harmonic mean of precision and recall. Additionally, we calculated the mean distance from the reconstruction to the reference, and the mean distance from reference to reconstruction, which we report in Appendix G.

To calculate the measures based on the distance from the reference to the reconstruction, we used vertices of the full SL scan; their average distance from the nearest neighbor is around 0.15 mm. To evaluate the methods which reconstruct the surface in the form of a triangular mesh, we sampled points from the mesh uniformly at a sampling distance of 0.1 mm.

The reference SL scans are incomplete, so the distance from some reconstruction points to the SL scan does not represent the distance to the real surface of the object, specifically, if the point lies near the missing part of the surface. To calculate the measures using only the points for which the distance to the real surface is reliable we used the approach of [20], and extended it using our new occluding surface.

First, we only kept the points which lie in the free space between the SL scanner and the object or near the surface of the object: we checked if the depth of the point w.r.t. SL scanner for any of its scanning positions is less than the depth given by the SL scan, plus a small tolerance $t_{\text{subsurf}} = 3$ mm to keep the points just below the surface for evaluation.

Next, to evaluate the precision metric, we checked if a point is closer to the real surface of the object than a distance threshold. For every reconstructed point, we calculated the distance to the SL scan and the distance to the occluding surface. If the distance to the SL scan was below the threshold, we considered the point to be closer to the real surface than the threshold. If the distance to the occluding surface was above the threshold, we considered the point to be farther from the real surface than the threshold. In any other case (in which the point can only be closer to the occluding surface and farther from the SL scan than the threshold, since the occluding surface encloses the SL scan by definition), we considered the distance from the point to the real surface to be unknown and excluded the point from the calculation of the measure.

To visualize the distance from the reconstruction to the reference and to calculate the mean distance we used a similar strategy and considered the distance to the real surface to be unknown whenever the point was closer to the occluding surface than to the SL scan. To account for the approximate nature of our occluding surface and to prevent computational instabilities at points where it must coincide with the SL scan exactly in case of exact calculations, we replaced the distance to the occluding surface in all calculations by its value increased by $\varepsilon_{\text{occ}} = 0.1$ mm.

G. Complete evaluation results

In Figures 5 to 111[1] we show qualitative evaluation of 3D reconstruction methods. For Neural RGB-D surface reconstruction we show the results only for four scenes: two relatively easy ones, based on performance of all the methods, `dragon` and `small_wooden_chessboard`, and two relatively hard ones `green_flower_pot` and `white_box`.

In each figure in the first column titled *Reconstruction*, we show the reconstruction produced by each method, and at the bottom of this column we show the reference surface from the SL scanner. In the column *Accuracy on reconstruction*, we show the reconstructed surface with color-coded distance to the reference surface; at the bottom of this column we show a photo of the scene. In the column *Accuracy on reference*, we show the reference surface, with the color showing the distance from the reconstruction to the reference. For each vertex of the reference surface, the distance is averaged over the reconstructed points for which this vertex is the closest one. Thanks to this projection of the distance values from the reconstructed points to the reference surface, these images show the accuracy of all points not just the ones closest to the camera. In the column *Completeness on reference*, we show the reference surface with color-coded distance to the reconstructed surface.

We use two colormaps for two different scales of error. In the last three columns, the points with no definite value of the distance are grey.

Since the methods TSDF Fusion, RoutedFusion, SurfelMeshing, and Neural RGB-D surface reconstruction produce the surface with a significant error, and in particular deep below the reference surface, we increased the value of sub-surface tolerance t_{subsurf} for these methods from 3 mm to 20 mm.

In Figures 112 to 218[1] we show the recall, precision, and F-score curves for reconstructed surfaces produced by the methods for each scene. Each curve represents the measure in percent for different values of the distance threshold τ in millimeters. Additionally, we mark the mean distance from the reference to the reconstruction for each method with a vertical dashed line on the recall plot, and the mean distance from the reconstruction to the reference with a vertical dashed line on the precision plot. Note that for some methods these lines may be out of the plot range.

Finally, in Figures 3 and 4 we show the values of the recall and precision metrics calculated with the distance threshold $\tau = 0.5$ mm for the reconstructions produced by the RGB-based methods COLMAP, ACMP, VisMVSNet, UniMVSNet, and NeuS. The value of the measure for each method is represented by the right edge of the bar with the respective color. The scenes in each figure are sorted by the best result on the scene.

[1] See at skoltech3d.appliedai.tech/data/skoltech3d_supp_results.pdf

H. Material properties in our dataset

We provide more information on the qualitative descriptors of surface reflection parameters assigned to each object and their relation to performance indicators for various reconstruction methods.

Identification of surface properties. As outlined in the main text, all labels were assigned to objects by visually inspecting photos of each object, to provide a preliminary assessment of the importance of various factors for reconstruction quality. Multiple photos of each object were used, with backlight proving particularly useful to establish the degree of translucency. The labels refer to the *dominant* material or materials of the objects, which have most influence on integral metrics of reconstruction quality. While this classification is imprecise and not a substitute for photometric measurements, as we use only rough estimation for each property, typically 3 buckets, we expect that visually assigned labels are highly correlated with the actual physical parameters. Where necessary, more than one label was assigned (*e.g.*, for objects consisting of multiple parts with distinct reflectivity properties, with no single dominant material); in this case several labels for some properties are used simultaneously.

Most of our labels are aligned with typical parameters of simple reflectance models: diffuse and specular reflection coefficient, shininess/roughness, measuring the reflection peak width, and translucency.

- *Translucency.* We assess the degree of translucency for the materials of the large parts of the object, from *none* (the default), through *low, medium, high,* all the way to completely *transparent.*
- *Reflection sharpness.* We characterize how sharp the reflectance function peak is, if highlights are present on the object (i.e., it is not purely diffuse), grouping the objects into four categories, from low to very high.
- *Specularity.* We visually estimate the ratio of specular to diffuse reflection for the dominant object materials, resulting in a degree of manifestation of view-dependent highlights, complementing *mirror-like* tag with a weaker label. We distinguish between no view dependent highlights *(diffuse),* largely diffuse highlights *(low),* somewhat diffuse light reflections and wide highlights *(medium),* and narrow highlights for partially mirror-like surfaces where one can only see sharp reflections of lights *(high).*
- *Mirror-like.* A binary tag for surfaces for which, in addition to a near-perfect reflection peak, the diffusive component is low relative to specular. This label overlaps very high reflection sharpness labels.
- *Metallic.* A binary tag for surfaces made of metal; compared to dielectric surfaces, these surfaces are always completely non-transparent, and the specular component of reflected light tend to have the same color as diffuse, while for dielectric materials it is closer to the color of the incident light.
- *Geometric features.* We visually assess the dominant qualitative scale of geometric structures of the surface, if any. We seek to distinguish between fine 3D structure with characteristic scale close but above SLS 3D resolution *(small),* a larger, discernable geometry with feature size equal to a small fraction of the object size *(medium).* Additionally, among surfaces lacking a dominant feature scale, we identified those with flat areas of sufficiently large size to make an impact on overall reconstruction quality (*flat* label), in the absence of 3D texture.
- *Texture type.* We differentiate between several types of textured surfaces: most of the surface covered by high-frequency image texture *(color),* sufficiently small scale (below or close to what the scanner can resolve), high-frequency displacement variation *(3D),* or other texture-like imperfections such as dirt, speckles, or visible roughness *(imperfections).*

Correlating properties and reconstruction performance. Most surface reflectance features mentioned above are expected to influence performance of common reconstruction methods.

We use data-driven approach to model the variability in a given measure of reconstruction performance caused bydifferent reflectance properties. Viewing all measures described in Appendix F as target variables, we constructed a numerical feature representation for each scene in our dataset; and used sparse L_1 regularised (Lasso) regression [21] to determine how each surface reflectance feature modulates reconstruction performance in each scene.

As we wanted to determine which features had statistically the most influence on reconstruction performance, we used the automatic feature selection mechanism built into the Lasso algorithm.

Our target variables were *precision* $P\,(0.5\,\text{mm})$, *recall* $R\,(0.5\,\text{mm})$, and *F-score* $F\,(0.5\,\text{mm})$ for the image-based methods ACMP, COLMAP, VisMVSNet, and UniMVSNet (Appendix E). For constructing the feature representation of each scene, we formed 25 binary features from the 7 properties mentioned above by encoding the presence of each surface property as 1, and its absense as 0. For fitting the model, we used the implementation of Lasso available in `scikit-learn` [10]; we z-scored all features, and, for each target variable, sought to find an optimal value of the regularisation parameter α (a minimizer of RMSE measure) using leave-one-out cross-validation by varying it over a range $[0.01, 2.0]$, and record the values of the R^2 statistic, the regularization weight α as well as the regression weights of the final fit.

Table 2 displays (normalized) values of coefficients weighting the (normalized) value of each feature in each constructed linear model. Most regression problems have demonstrated a consistent value of the regularization weight $\alpha \in [0.18, 0.57]$ with an average of 0.36; the values of coefficient of determination R^2 vary from 0.24 to 0.54

with a mean of 0.43, indicating that a reasonable performance has been achieved.

Factors affecting algorithms performance. Using the process described above, we have identified five labels having the most pronounced *negative effect* on reconstruction quality for image-based MVS methods, according to a mean F-score across methods: very high, high, or medium reflection sharpness, high specularity, medium-scale geometric features; additionally, a negative effect from high translucency, low specularity is also expected, unlike moderate but notable contribution of non-metallic and diffuse materials. Conversely, having any type of texture (either color, 3d, or texture-like surface imperfections) *contributes positively* to reconstruction performance as one expect for MVS-type algorithms; the same can be said about non-translucent objects.

We consider this study very preliminary, given the approximate nature of our labels. Its results are encouraging as these show clear correlation between surface reflectance properties and algorithm performance.

References

[1] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6290–6301, 2022. 3

[2] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 3

[3] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96*, pages 303–312, Not Known, 1996. ACM Press. 3

[4] Melanie Elias, Anette Eltner, Frank Liebold, and Hans-Gerd Maas. Assessing the influence of temperature changes on the geometric stability of smartphone-and raspberry pi cameras. *Sensors*, 20(3):643, 2020. 2

[5] Shahram Izadi, Andrew Davison, Andrew Fitzgibbon, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Dustin Freeman. KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, page 559, Santa Barbara, California, USA, 2011. ACM Press. 3

[6] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 406–413. IEEE, 2014. 3

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 3

[8] Matthias Nießner. Voxelhashing. https://github.com/niessner/VoxelHashing. 3

[9] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):1–11, 2013. 3

[10] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011. 5

[11] Rui Peng. Unimvsnet. https://github.com/prstrive/UniMVSNet. 3

[12] Rui Peng, Rongjie Wang, Zhenyu Wang, Yawen Lai, and Ronggang Wang. Rethinking depth estimation for multi-view stereo: A unified representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8645–8654, 2022. 3

[13] Neural RGB-D Surface Reconstruction. Neural rgb-d surface reconstruction. https://github.com/dazinovic/neural-rgbd-surface-reconstruction. 3

[14] Johannes Lutz Schönberger. Colmap. https://github.com/colmap/colmap. 2

[15] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[16] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[17] Thomas Schops. Surfelmeshing. https://github.com/puzzlepaint/surfelmeshing. 3

[18] Thomas Schops, Viktor Larsson, Marc Pollefeys, and Torsten Sattler. Why having 10,000 parameters in your camera model is better than twelve. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2535–2544, 2020. 2

[19] Thomas Schops, Torsten Sattler, and Marc Pollefeys. SurfelMeshing: Online Surfel-Based Mesh Reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10):2494–2507, Oct. 2020. 3

[20] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3260–3269, 2017. 4

[21] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996. 5

[22] Peng Wang. Neus. https://github.com/Totoro97/NeuS. 3

[23] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*, 2021. 3

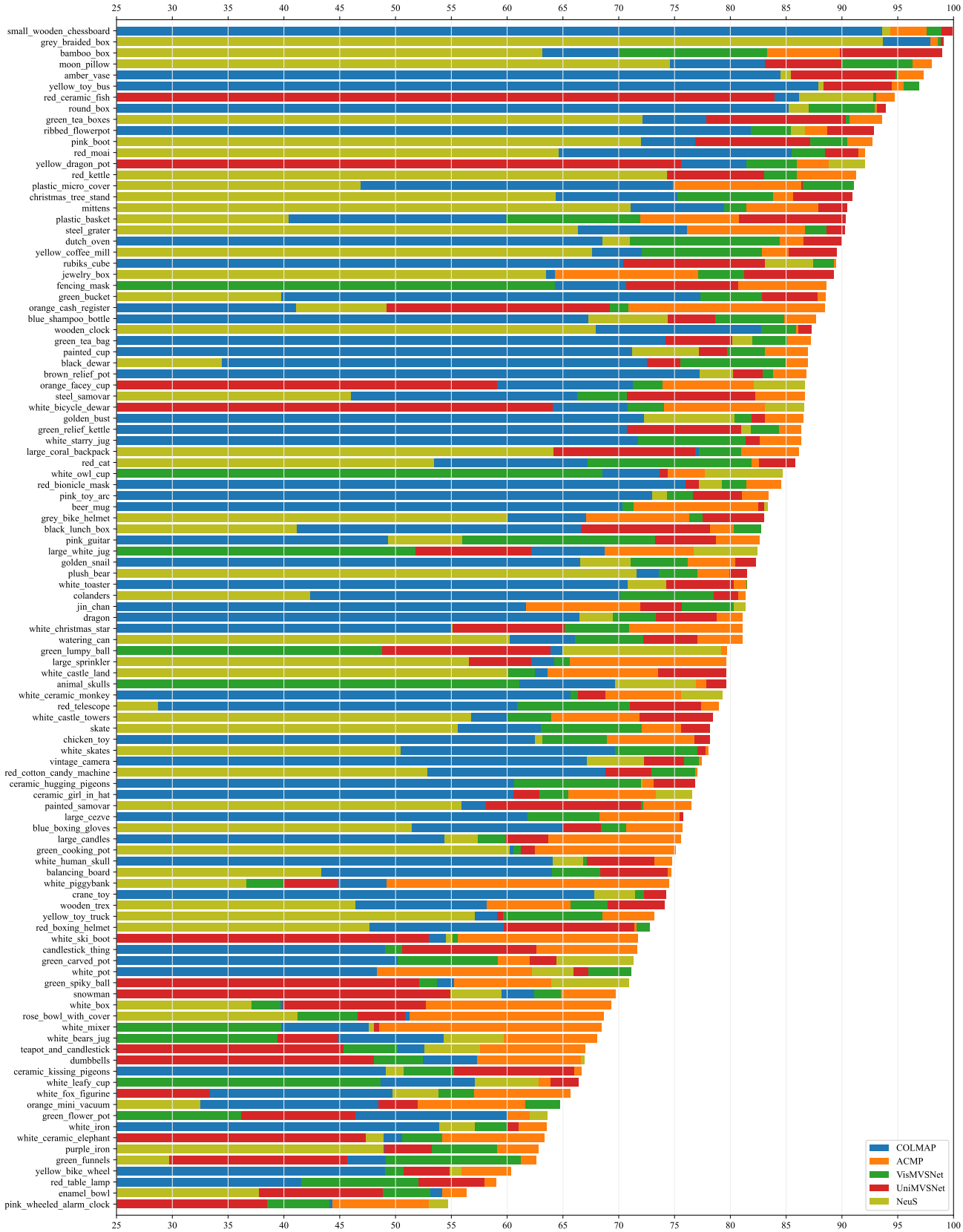

Figure 3. **Recall for the RGB-based methods for all scenes** with the distance threshold $\tau = 0.5$ mm.

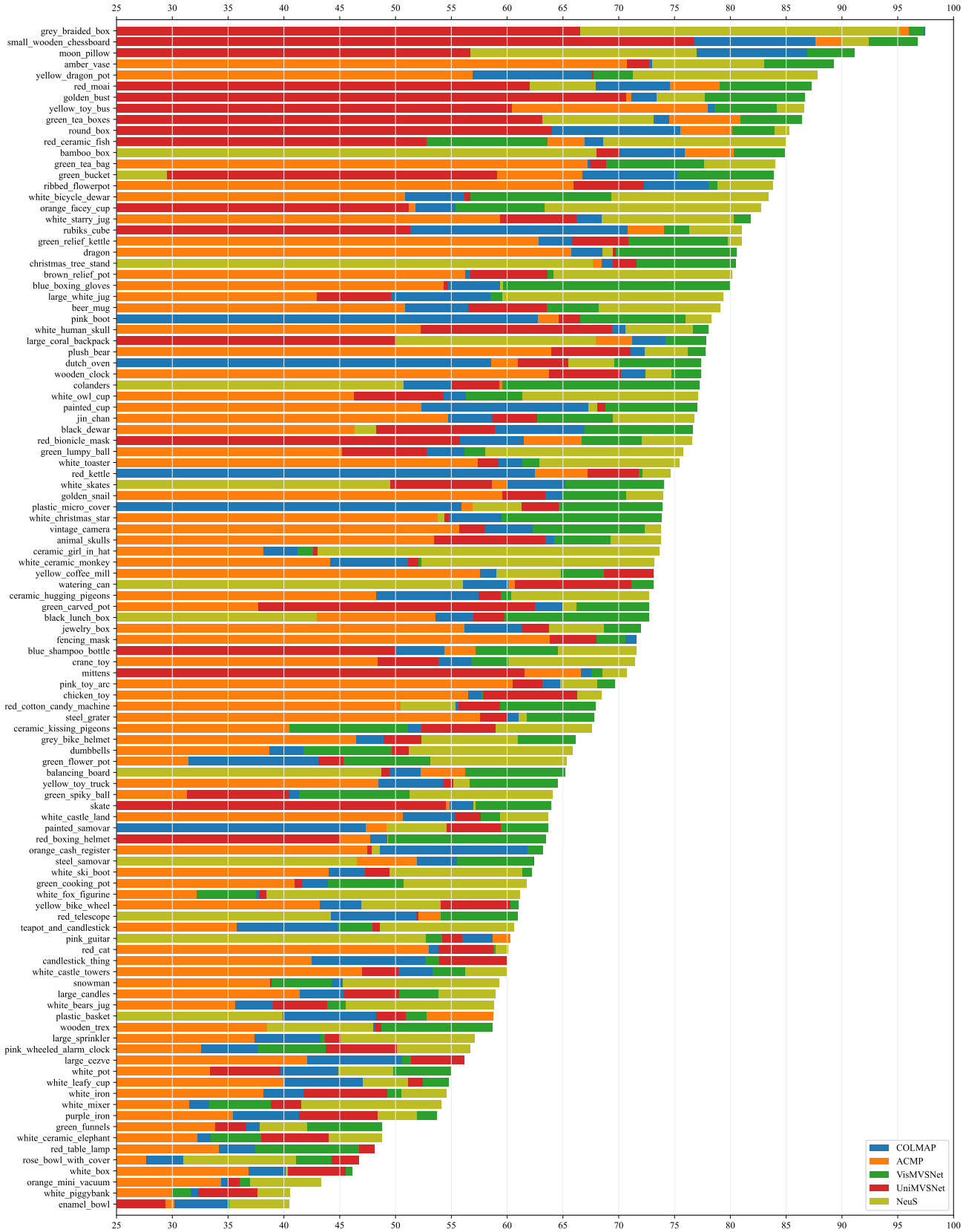

Figure 4. **Precision for the RGB-based methods for all scenes** with the distance threshold $\tau = 0.5$ mm.

[24] Silvan Weder. Routedfusion. https://github.com/weders/RoutedFusion. 3

[25] Silvan Weder, Johannes Schonberger, Marc Pollefeys, and Martin R. Oswald. RoutedFusion: Learning Real-Time Depth Map Fusion. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4886–4896, Seattle, WA, USA, June 2020. IEEE. 3

[26] Zhirong Wu, Shuran Song, Aditya Khosla, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shape modeling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, USA, June 2015. 3

[27] Qingshan Xu. Acmp. https://github.com/GhiXu/ACMP. 3

[28] Qingshan Xu and Wenbing Tao. Planar prior assisted patchmatch multi-view stereo. *AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 3

[29] Yao Yao. Blendedmvg. https://github.com/YoYo000/BlendedMVS#upgrade-to-blendedmvg. 3

[30] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. *European Conference on Computer Vision (ECCV)*, 2018. 3

[31] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. *Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[32] Jingyang Zhang. Vismvsnet. https://github.com/jzhangbs/Vis-MVSNet. 3

[33] Jingyang Zhang, Yao Yao, Shiwei Li, Zixin Luo, and Tian Fang. Visibility-aware multi-view stereo network. *British Machine Vision Conference (BMVC)*, 2020. 3

[34] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 3

Dataset	Devices	RGB res.	Depth res.	Hi-res. geom.	Video	Camera positioning	Poses/scene	Lighting	# Scenes	#points in high-res pointclouds	#Frames	Scene type	Data provided	Target tasks
DTU	2 RGB unknown model	1200×1600	—	✓	—	6DOF robotic	49 or 64	8	80	13.4M	27K	small objects	• intrinsics, extrinsics • raw RGB • undistorted RGB • surfel clouds • voxels where GT is known	MVS
ETH3D (hi)	Nikon D3X, Faro Focus X 330	6048×4032	—	✓	—	tripod	10-70	U	13 train 12 test	28M	11K	in/outdoor large scenes	• intrinsics, extrinsics • raw RGB • undistorted RGB • raw 3D scans • scans with outliers removed • rendered depth maps	MVS
ETH3D (lo)	4 global shutter, Faro Focus X 330	752×480	—	✓	Mono	handheld	160-300	U	5 train 5 test	28M	10K	in/outdoor large scenes	same as ETH3D hi-res	MVS, stereo
TnT	DJI Zenmuse X5R, Sony a7S II, Faro Focus X 330	3840×2160	—	✓	RGB	handheld + gimbal	150-500	U	7 train 14 test	5.5M-53.4M	148K	large objects, in/outdoor large scenes	• raw videos • sets of extracted RGB • raw 3D scans • point clouds • camera params (extracted with COLMAP)	SfM and MVS pipelines, view synthesis
BlendedMVS / MVG	unkown RGB cameras	1536×2048 (rescaled)	—		—	unknown	20-1000	U	113 in MVS 502 in MVG	—	18K/ 110K	MVS:large outdoor, small scale objects, sculptures MVG: mostly aerial imaging	MVS and MVG: • intrinsics and extrinsics • 768x576 rendered RGBD • masked rendered RGB MVS only: • 2048x1536 rendered RGBD • reconstructed meshes	MVS (including training)
Redwood	PrimeSense Carmine	1280×1024	640×480		RGBD	handheld	—	U	10K	—	144K	medium objects	• RGBD • for 398 scenes, meshes	
SUN RGBD	RealSense, Xtion, Kinect v1 Kinect v2	1920×1080 640×480	628×428 512×424 640×480		—	handheld	—	U	unknown	—	10K	indoor scenes	• 2D polygon object segmentation; • 3D object bounding boxes, • 3D polygon room layouts.	classification, segmentation
BigBIRD	5x Canon Rebel T3, 5x PrimeSense Carmine 1.08	4272×2848	640×480		—	fixed 5 cameras+ turntable	600	1	120	—	75K	small objects	• hi-res RGB • low-res RGBD • point clouds for each scan • merged point clouds	instance recognition, category recognition, 3D reconstruction
CORBS	Kinect, 3Digify	1920×1080	512×424	✓	RGBD	handheld (tracked)	5 trajectories, 700-7000 frames each	U	4	unknown	47K	indoor scenes	• RGBD • IR • camera trajectories • reconstructed mesh	SLAM
ScanNet	Structure	1296×968	640×480		RGBD	handheld	—	U	1513	—	2.5M	indoor scenes	• intrinsics, camera trajectories • reconstructed meshes (semantically labeled) • aligned CAD objects	classification, retrieval, voxel level annotation
ARKitScenes	Faro Focus S70 Apple iPad Pro	4032×3024 3680×2760	256×192	✓	RGB	handheld	—	U	5047	—	450K	indoor scenes	• intrinsics, camera trajectories • reconstructed meshes (semantically labeled)	classification, segmentation, depth upsampling
RGB-D-D	LUCID Helios ToF Huawei P30 Pro	3648×2736	640×480 240×180		—	unknown	—	U	4811	—	4811	portraits, models, plants, lights	• intrinsics, camera trajectories	depth upsampling
Ours	2x DFK 33ux250 cameras 2x Huawei Mate 30 Pro RealSense D435 Kinect V2 RangeVision Spectrum	2448×2048 7296×5472 1920×1080	240×180 512×424 1280×720	✓	—	6DOF robotic	100	14	107	9.5M-66.5M	877K	small objects	• intrinsics, extrinsics • RGB: cameras • undistorted RGB • RGBD: phones, Kinect, RealSense • undistorted RGBD • raw IR: phones, Kinect, RealSense • raw 3D scans • reconstructed mesh • occluding mesh	MVS, 3D reconstruction, depth fusion, depth upsampling

Table 1. **Comparison of our dataset to the most widely used related datasets.** U indicates uncontrolled lighting; frames are counted per sensor, *i.e.*, all data from an RGB-D sensor are counted as a single frame. The number of separate images acquired may be considerably larger (1.4 M for our dataset). All scenes, from both training and testing sets, were counted.

		Recall (0.5 mm)				Precision (0.5 mm)				F-score, thres=0.5mm				
Parameter	Value	ACMP	COL.	Vis.	Uni.	ACMP	COL.	Vis.	Uni.	ACMP	COL.	Vis.	Uni.	Mean
Translucency	none	0	0	0.91	1.45	0	1.3	0.48	1.57	0	1.04	0	1.85	0.72
	low	-0.23	0	0.3	0	0	0	0	0	0	0	0	0	0
	medium	0.67	0	0.62	0.83	0	0	0	0	0	0	0	0.38	0.09
	high	-1.72	-1.15	-1.98	-1.81	-1.93	-1.36	0	-0.56	-1.88	-1.53	-0.48	-1.19	-1.27
Reflection sharpness	no refl.	0	0	-0.17	0	0	0	0	0	0	0	0	0	0
	low	-0.83	-0.8	-0.64	-1.82	-0.56	0	1.13	0	-0.52	-0.57	0	-0.62	-0.43
	medium	-2.05	-0.58	-3.87	-3.37	-2.6	-2.45	-1.57	-3.26	-2.3	-2.18	-2.14	-3.32	-2.49
	high	-1.59	-0.12	-3.22	-1.46	-2.41	-2.93	-1.03	-3.66	-2.04	-2.34	-1.34	-2.71	-2.11
	very high	-2.24	-1.47	-4.8	-3.47	-3.99	-3.46	-0.91	-4.21	-3.4	-3.37	-1.51	-4.11	-3.1
Specularity	diffuse	-1.81	0	-3.66	-2.15	-1.15	0	0	0	-1.21	-0.5	0	-1.32	-0.76
	low	-0.75	-0.28	-4	-2.32	-1.39	-0.97	0	-1.02	-1.28	-1.32	0	-2.08	-1.17
	medium	0	0	-2.07	0	0	0	0	0.76	0	0	0	0	0
	high	-1.66	-0.64	-4.33	-2.79	-2.02	-1.69	-0.91	-1.08	-1.91	-1.35	-1.56	-2.29	-1.77
Geometric features	small	0	0.06	2.49	1.05	0	0	0.47	0	0.06	0.29	1.15	0.59	0.52
	medium	-1.58	-1.1	-0.79	-2.31	-1.43	-1.25	0	-2.08	-1.5	-1.42	0	-2.42	-1.34
	flat	0.23	0	1.15	0.59	2.08	0.27	0	0	1.58	0	0.3	0.32	0.55
	NA	0	0	0	0	0	0	0	0	0	0	0	0	0
Mirror-like	no	0	0	0	0	0	0	0	0	0	0	0	0	0
	yes	0.42	0.72	0.85	0.23	0	1.13	1	1.38	0	1.3	0.8	0.83	0.73
Metallic	no	-0.62	-0.21	-0.23	-1.3	-1.25	-0.34	-1.24	-1.16	-1.14	-0.24	-1.21	-1.27	-0.97
	yes	0	0	0.67	0	0	0	0	0	0	0	0	0	0
Texture type	color	2.61	2.41	3.9	3.41	3.65	2.48	2.31	2.97	3.36	2.59	2.73	3.34	3.01
	imperf.	2.06	1.96	5.1	4.14	2.61	2.01	1.24	3.4	2.59	2.43	2.32	3.92	2.81
	3d	1.23	1.89	2.42	1.51	3.43	4.07	3.21	4.14	2.8	3.13	2.88	3.01	2.96
Coef. of det.	$R2$	0.39	0.24	0.54	0.41	0.46	0.47	0.39	0.49	0.44	0.4	0.44	0.48	
Reg. coef.	α	0.26	0.57	0.18	0.3	0.37	0.4	0.39	0.38	0.37	0.28	0.54	0.27	

Table 2. **Data-driven estimates of surface reflection features in our datasets.** Note the *negative effect* on reconstruction quality (rightmost column, negative numbers highlighted in red) of very high, high, or medium reflection sharpness, high specularity, medium-scale geometric features. Conversely, note the *positive contribution* to reconstruction performance of color textures, 3d textures, or texture-like surface imperfections (rightmost column, large positive values highlighted in blue). *COL.*, *Vis.*, and *Uni.* denote COLMAP, VisMVSNet, and UniMVSNet respectively.