Ignorance is bliss: flawed assumptions in simulated ground truth

Andrew R. Buck, Derek T. Anderson, Joshua Fraser, Jeffrey Kerley, and Kannappan Palaniappan

Department of Electrical Engineering and Computer Science, University of Missouri, Columbia MO, USA

ABSTRACT

Current generation artificial intelligence (AI) is heavily reliant on data and supervised learning (SL). However, dense and accurate truth for SL is often a bottleneck and any imperfections can negatively impact performance and/or result in biases. As a result, several corrective lines of research are being explored, including simulation (SIM). In this article, we discuss fundamental limitations in obtaining truth, both in the physical universe and SIM, and different truth uncertainty modeling strategies are explored. A case study from data-driven monocular vision is provided. These experiments demonstrate performance variability with respect to different truth uncertainty strategies in training and evaluating AI algorithms.

Keywords: Ground truth, simulation, artificial intelligence, machine learning, deep learning, depth estimation

1. INTRODUCTION

Generating ground truth is tedious, error-prone, and introduces human bias to training data making synthetic data from simulated environments a promising approach to alleviating these difficulties. Many recent synthetic ground truth data sets make similar promises in describing their ground truth: "pixel-perfect",[1] "pixel-accurate",[2] "sub pixel accurate",[3] "pixel-level".[4] This demonstrates a prevalence of thought where each pixel is treated as a *little square*.[5] We argue that per-pixel ground truth in synthetic data is not a panacea due to the uncertainty in assigning a single discrete depth or class label to each sample of a complex 3D environment.

Herein, we look at the meaning of a pixel and compare sampling methods for ground truth representations of training data then introduce a novel bundle approach to allow our models to train in a way robust to the ambiguity of a pixel that may be a mix of class labels or depths. Rather than attempt to improve a state-of-the-art model in light of confounding factors, we demonstrate our ideas on a controlled toy monocular vision problem to show how a failure to consider the betweenity of training samples limits the robustness of the resulting model. We present our results showing that making the model ambiguity-aware improves over the prevailing single-sample representations. Finally we provide suggestions for future work to extend this to class labels and to evaluate our methods against real world test cases.

2. THE MEANING OF A PIXEL

The representation of analog data in a digital domain has always been limited by the density and distribution of the discrete sampling elements, or "pixels". In 5 and 6, Smith argues that pixels are the fundamental building blocks of all modern digital media, but are often misunderstood as pure digital objects and not representations of a true underlying analog phenomenon. When using digital images to represent real-world objects, it is important to recognize the limitations of a pixel and understand what the data truly represents.

In computer rendering software, an image is generated as a collection of regularly spaced pixels on a grid. Each pixel has an associated value, or array of values, that define various image properties at that location. By far, the most common data to store with a pixel is light intensity or value. An array of pixels representing

Send correspondence to Andrew R. Buck
E-mail: buckar@missouri.edu

value produces a grayscale image, where large values represent bright regions and dark values represent dark areas. Three of these images can be combined as red, green, and blue "channels" to give a color (RGB) image. Many more wavelengths can be stored to give a high-dimensional multispectral or hyperspectral image. Other properties can be stored with each pixel in addition to light value, which may not be available to real-world sensors but can be produced in simulation, such as depth, object ID, and object normals.

While it is common to think of an image as a pixel array that fills 2D space, a more accurate description (at least in the case of digital photography) would be a collection of sampled points on a grid. Each pixel represents a sampling of the environment at a discrete location. In real-world optics, an image sensor collects light rays from throughout the environment, aggregating the information into discrete bins (pixels). Modern computer rendering techniques strive to replicate this realism by ray-tracing many beams of light in reverse, from the pixel out into the scene and aggregating the attributes of the objects they hit. While this can improve photo-realism, it has unintended consequences with regard to representing ground truth. If a pixel is made to represent the aggregation of multiple sampled points in the environment, then it can be difficult or even impossible to define a single truth value for certain properties such as depth or object ID.

Simulated imagery can avoid this issue in some cases by not performing any anti-aliasing or ray-tracing techniques. By only projecting a single ray for each pixel, the rendering engine can explicitly decide what object properties each pixel represents. However, this leads to aliasing artifacts, often observed as a stair-step pattern on straight lines at an angle such as object boundaries. The trade-off is that while image quality suffers for aliased imagery, the ground truth is precise and accurate. We explore the implications of using aliased and anti-aliased imagery in our experiments in Section 4.

3. ISSUES WITH SIMULATED DATA

No sensor is perfect. There is always some measurement uncertainty. This uncertainty extends even to simulated sensors under what might be considered "perfect" conditions. Specifically, synthetically generated imagery is by nature generated as a discrete array of pixels. The process of rendering these images and interpreting them as ground truth introduces subtle biases that need to be accounted for when using this data for AI applications.

This problem is not unique to image-based ground truth. Other forms of labeled data can contain flaws. For instance, hand-labeled bounding boxes or segmentation maps can only ever be as accurate as the human labeler. Even full image classification labels can be ambiguous if the class labels are not descriptive enough. For depth estimation and 3D reconstruction, LIDAR is often viewed as ground truth, but it suffers from the same issues of accuracy and reliability.

Many researchers have turned to simulated data and imagery (SIM) to train ML algorithms. There are many advantages of using SIM, such as repeatability, the volume of data, the ability to collect data in a variety of conditions, and the accuracy of the data. However, as with human labelers, the accuracy of a simulated scene is only as good as the rendering engine that produces the data. Our objective in this article is to present some of the issues with using simulated imagery to train AI algorithms and to offer some mitigation approaches.

We focus mainly on the problem of single image depth estimation (SIDE). This is a problem that clearly demonstrates the issues that are present with SIM and provides a way to visualize and understand the different methods. Our conclusions are not limited to only this problem domain, but have themes that can be broadly applied in many aspects of using SIM for AI.

4. EXPERIMENTS

To demonstrate our ideas, we performed a series of experiments using simulated data for monocular depth estimation. The primary goal was to create a simple example with as few confounding variables as possible. The focus is then on a single, straightforward problem of estimating the depth of a pixel in a color image, given some training data. The ground truth is flawed in several ways, and we show the issues involved with each approach.

For each experiment, we trained a depth estimation network to learn image features such as color, size, and shape of simple objects and evaluated this network on a nearly identical test set. In this way, this represents a simple experiment where we expect the model to perform well so that we can more clearly identify the cases where it breaks down.

4.1 Dataset

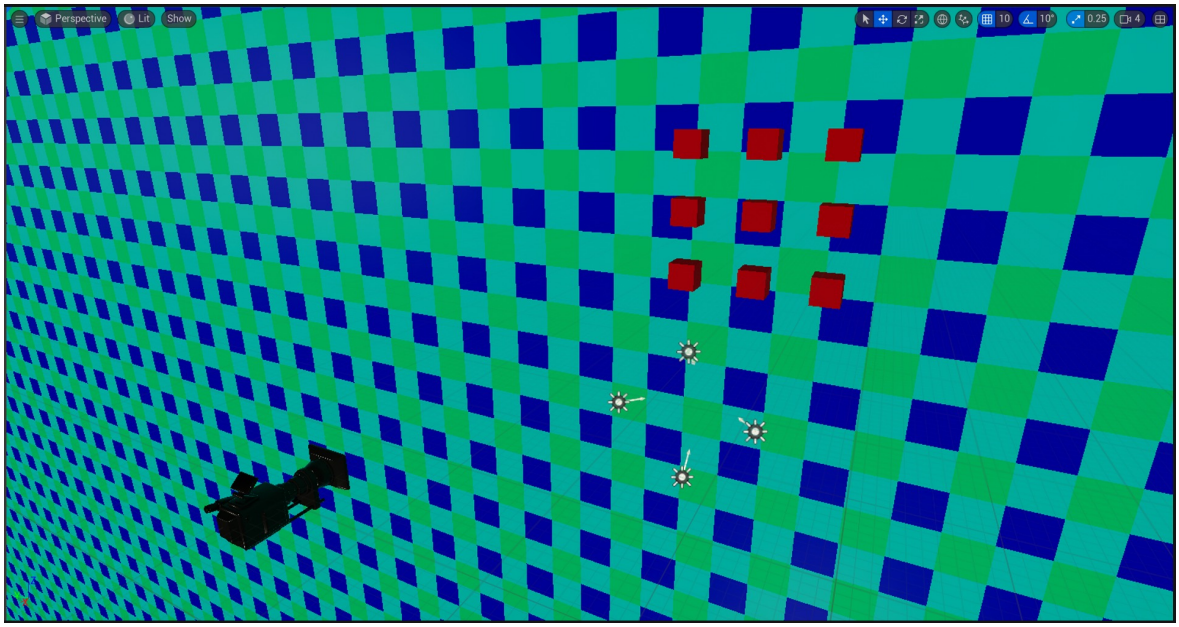

Figure 1. The simulated dataset used in our experiments was generated with Unreal Engine. It consists of a flat plane with a checkerboard pattern at various depths and a set of nine rotating cubes in the foreground.

The dataset for our experiments was generated with Unreal Engine 5, using the Movie Render Queue Pipeline. The scene consists of a fixed camera looking down the X axis, a flat checkerboard background in the YZ plane, and a set of nine rotating cubes in the foreground, shown in Fig. 1. We collected several 512×512 images in a rendered sequence, using a Blueprint script to continuously rotate the cubes and move the objects back at regular intervals. The background texture also receives an offset every frame to ensure that no two frames are identical. Every 40 frames, the cubes and the background are moved away from the camera, with the background moving farther each time. This continues for 24 iterations, resulting in 960 total image samples with the background ranging from 24 to 70 meters away, and the foreground cubes ranging from 16.7 to 29.0 meters away. Note that because the background is a flat plane, each background pixel has a constant depth value for each of the 24 sets, whereas the depth corresponding to pixels on the foreground cubes falls within some closer fixed range. Fig. 2 shows some examples of the generated data.

Our intention with this dataset is to isolate as many variables as possible and focus on producing features that are relevant to depth estimation. The textures used are intentionally sparse, and the lighting is front-lit with flat shading. The cubes are colored red and appear brightest when a face is perpendicular to the camera. The background uses a green and blue repeating pattern that stays at a fixed size in world space, but appears to grow or shrink in image space, depending on the distance of the background to the camera. In this way, it should be possible for the network to learn that features in the red channel correspond to the foreground cubes and features in the green and blue channels correspond to the background plane. The size of the background checkerboard pattern is a cue to the net that enables it to learn the distance to the background plane.

We collect both aliased and anti-aliased data sets. A comparison of the two approaches is shown in Fig. 3. For the aliased data, no smoothing is applied by the rendering engine, resulting in crisp edge boundaries. In this case, the attributes of each pixel are computed by casting a ray from the camera focal point out into the scene and determining the first object it intersects. The ray can intersect either a cube or the background but never both.* The object properties at the point of intersection determine the rendered color, depth, and object ID.

*In the rare case that a ray perfectly touches the edge of a cube without intersecting the body, the rendering engine must decide whether to interpret this pixel as a foreground or background point.

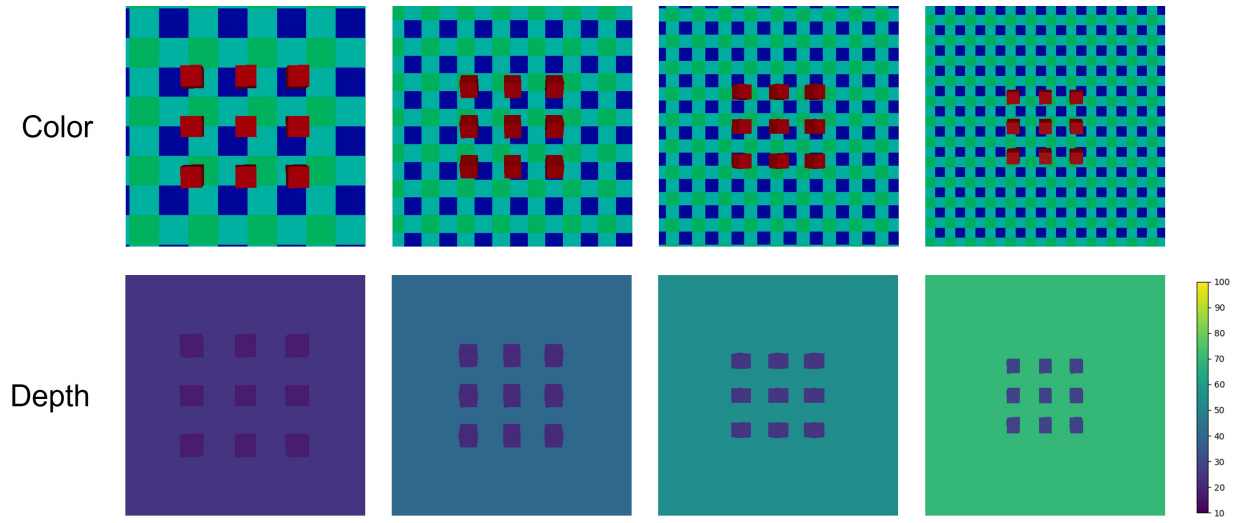

Figure 2. Examples of simulated color and depth images used in the experiments.

Because there is no mixing between objects in the aliased case, each pixel represents a "pure" sampling of the environment, although this leads to the well-known stair-step pattern when rendering sharp edges at an angle.

To alleviate aliasing artifacts present in the imagery, it is common to apply various spatial and temporal sampling techniques to produce a more visually appealing image. This can have unintended consequences, however, when using the image for machine learning applications. Although anti-aliasing can smooth edges in color images, other attributes, such as depth, are not well-preserved by the anti-aliasing operation. To collect anti-aliased data, we render the image at a much larger resolution (5120×5120) and use bilinear resampling to reduce the image back down to the original size of 512×512. This results in smooth edges for the color image, but the same method cannot be applied to the depth image. Doing so results in interpolated depths along object boundaries that do not reflect the true state of the world. If these interpolated depth values are projected out into 3D points, we observe "trails" that span the gap between the depths of the foreground and background objects. As an alternative, we propose (for the sake of analysis) storing each of the depth values of the larger image as an array for each pixel. For our 10x upsampling, this effectively gives 100 bundled depth values for each pixel. We note that in addition to depth, we also store the object ID. This allows us to determine edges by identifying pixels that contain more than one ID value.

All together, this gives two different variations of color imagery (aliased and anti-aliased) and three different ways to represent depth: aliased (precise) depth, anti-aliased (average) depth, and the bundle approach. We compare and contrast these different methods in our experiments below.

4.2 Depth Estimation Model

SIDE is a challenging computer vision problem that seeks to learn a model that can predict the depth (distance from the camera) of each pixel in an input image without any additional context. Typically, in real-world situations, the ground truth required to train such a model is either unavailable or sparsely generated (e.g. from a LiDAR system). Even when available, the provided depth may not be perfectly accurate, leading to questions about the reliability of such approaches. Nevertheless, a variety of methods have been proposed to tackle this problem and are in many cases able to produce impressive results.

One strategy that is often employed to overcome the lack of ground truth depth for training is to use a self-supervised approach. Using this idea, one can use an image sequence from a moving camera platform to generate image pair samples to perform stereo reconstruction. It is then possible to train a pair of deep neural networks to simultaneously learn the pose transformation between image pairs and the projected depths of the pixels in the images. This results in two networks: a pose network and a depth network.

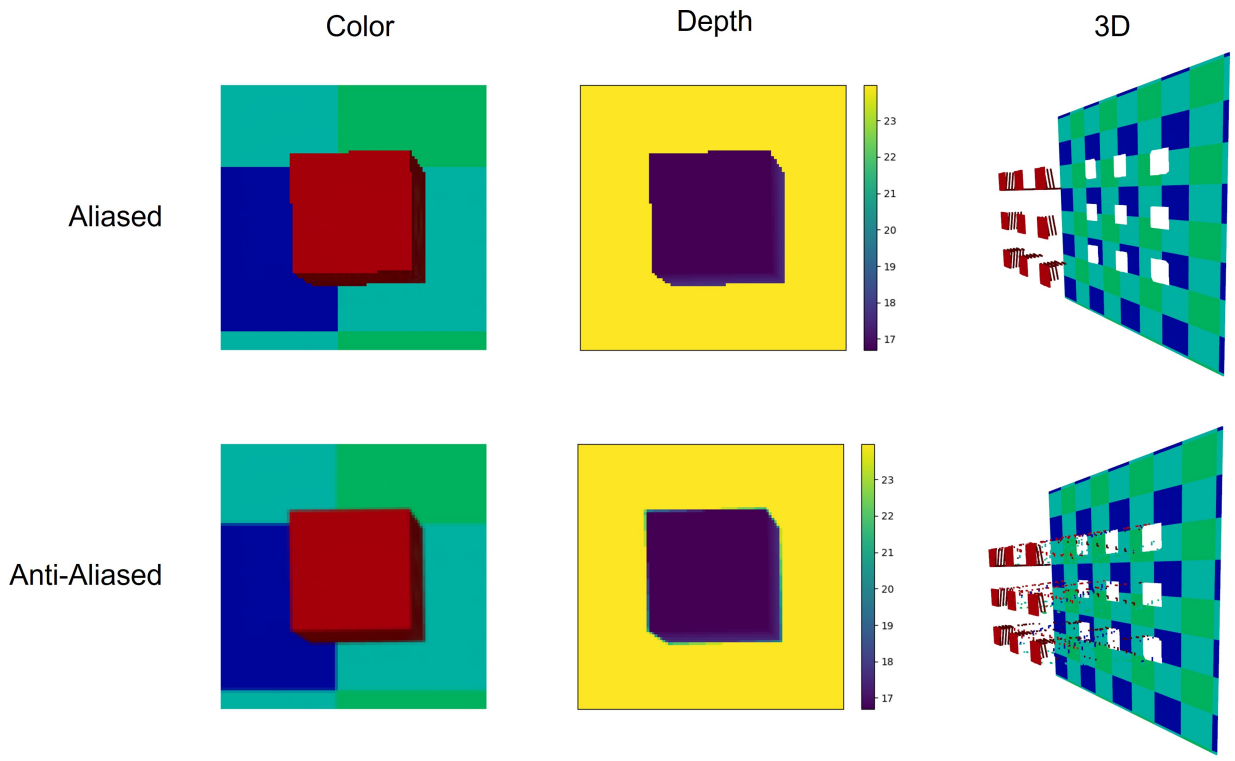

Figure 3. A comparison of aliased and anti-aliased data generation methods. The top row shows aliased color and depth images, which have sharp edge boundaries and no trailing artifacts in 3D. The bottom row shows anti-aliased color and depth, which smooths the edges but introduces artifacts in the depth resulting in trails in 3D.

If the training data is computer-generated with dense per-pixel ground truth depth, then we can utilize just the depth network from a SIDE model and train it directly. This is the approach we use in our experiments. We use the depth prediction network implementation from Monodepth2,[7] which consists of a Resnet[8] encoder that maps an input RGB image into a latent variable space and a depth decoder that reconstructs a depth image. The final layer of the decoder network is mapped to a sigmoid activation function and scaled to a predefined output range, which in our experiments is 10 to 100 meters.

4.3 Results

We performed several experiments with this simulated data set. For each of the experiments reported below, we trained a model using the Monodepth2 Resnet18 depth network architecture for 30 epochs. We partitioned the data into even and odd frames and used one half for training and the other for testing. This is about as close to a resubstitution experiment as possible without actually mixing the training and testing sets. The first experiment looks at a common issue in synthetically generated data by using anti-aliased color imagery and anti-aliased (average) depth. The next experiment considers a simple solution to this problem by using aliased color and depth imagery. Finally, the last set of experiments propose an alternative way of learning depth by using the bundle approach.

4.3.1 Experiment 1: Anti-aliased Color with Anti-aliased GT Depth (AARGB-AAGT)

In this first experiment, we consider a common issue that plagues many synthetically generated data sets: anti-aliasing[†]. Although anti-aliasing can smooth images to make them appear more natural, the process is not

[†]Note, while we say anti-aliasing, we are actually referring to a multitude of techniques that aggregate data. This can include ray tracing at a single pixel, spatial and temporal anti-aliasing functions, and spatial/temporal multi sampling techniques (e.g., the UE MRQ). The point is, a single pixel is assigned a value that is an aggregate of multiple observations.

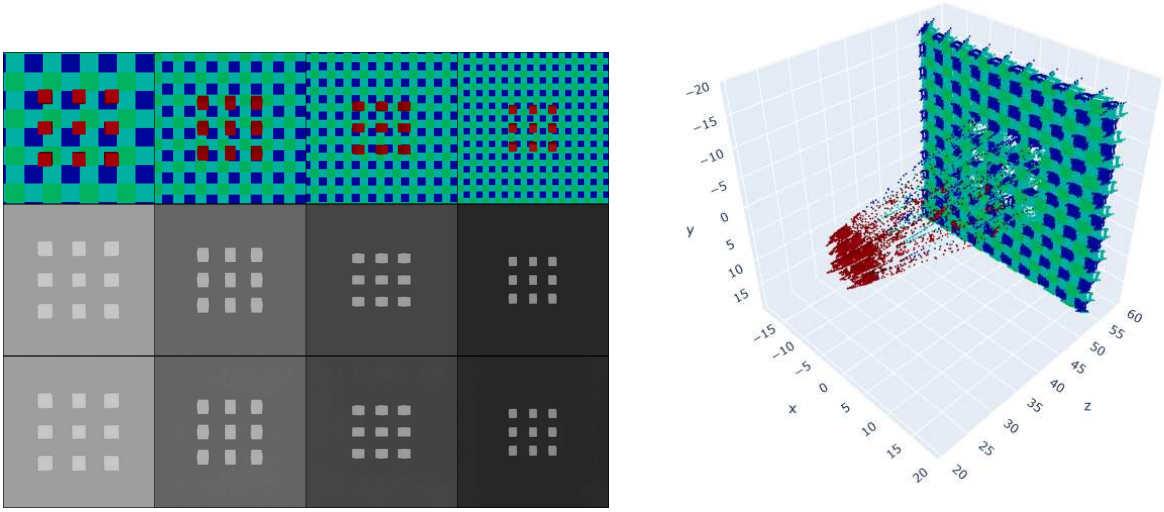

Figure 4. Left: Example outputs of the anti-aliased color to anti-aliased ground truth depth model (AARGB-AAGT). Each column is a different sample from the dataset with a receding background. The top row shows the color input, the middle row shows the target ground truth depth, and the bottom row shows the predicted output. Right: A 3D point cloud projection of one of the model predictions.

appropriate for generating ground truth. Consider what happens along the edge of an object that becomes smoothed by anti-aliasing. The pixel on an edge or corner now represents a mixture of both foreground and background objects. When assigning an object ID, is the index that of the foreground or background? Which depth is used? In many instances, the values are simply averaged over all sources that contribute to the pixel.

For this experiment, we used anti-aliased color imagery as input to the algorithm, and trained using the average depth of a pixel. This is shown as the bottom row in Fig. 3. We used the RMSE-log loss function, averaged over all pixels in an image, given as

$$L(X,Y) = \frac{1}{N}\sum_i \big(\log(Y_i) - \log(X_i)\big)^2, \tag{1}$$

where X_i and Y_i are the predicted and ground truth depths of pixel i respectively, and N is the total number of pixels in the image. Using this loss function, a model was trained on half of the generated data (odd frames) and evaluated on the other half (even frames).

Some of the outputs of this model are shown in Fig. 4. In the figure, we see that the model appears to do very well qualitatively when generating grayscale depth images. However, the 3D point cloud projection shows quite a few trail artifacts between the foreground cubes and the background plane, where the edge pixels are projected into a range somewhere between the two. Clearly, even though the general concepts of the foreground cubes existing in front of a background plane are captured by the model, many of the details are obscured by the simple analysis.

To dig deeper into these results, we look at each set of background depths independently and separate our analysis based on foreground, background, and edge pixel sets. To determine these sets, we use the object IDs stored with the bundled depth ground truth. Any pixel that is assigned more than one object ID is labeled as an edge pixel. The results are reported in Fig. 5 as histograms of depth values. These histograms reveal several interesting observations about the results of this experiment.

First, we notice that when the background is at its closest range, the model overestimates depth, and when the background is at its farthest range, the model underestimates depth. It seems to perform best when the background is somewhere in the middle of what was presented in the training data. This suggests that the model has difficulty extrapolating and fully utilizing the examples with the most extreme depth values.

Next, we see that the ground truth depth is distributed between the foreground and background for all edge pixels. This is because the anti-aliased depth performs an averaging over all scene objects that contribute to a given pixel. The spikes that are observed are due to limited precision in the generated ground truth depth (another potential source of error).

Last, the histogram plots show that the predicted depth for edge pixels tends to be distributed widely between the foreground and background depths. This is to be expected, since this is also how the ground truth appears.

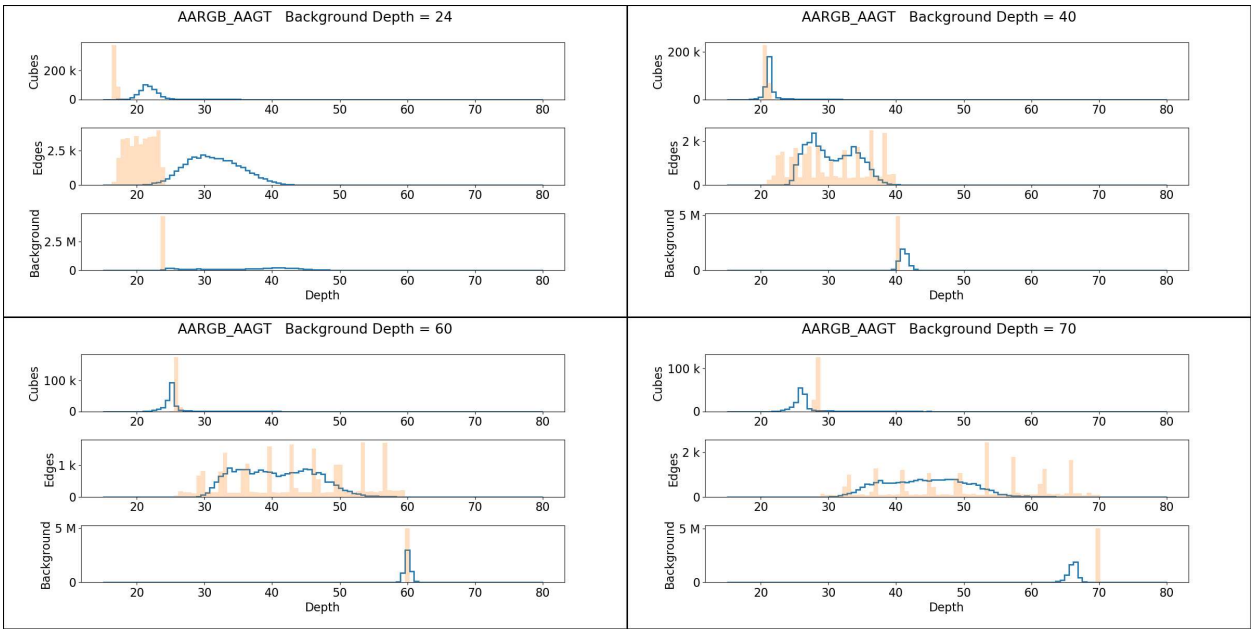

Figure 5. Depth prediction histograms of the anti-aliased color to anti-aliased ground truth depth model (AARGB-AAGT). Results are shown for 4 different background depths. For each depth, the per-pixel predictions of 20 test images are aggregated, separated into Cube, Edge, and Background sets. Ground truth depths are shown with solid orange bars and the predictions are shown as a stair-step plot in blue.

To gain a better insight into how the model performs over all background ranges, Fig. 6 shows prediction accuracy as correct, under, and over ratios. Here, we see that performance is indeed best for the non-extrema background ranges, and there is a general tendency to over-predict for close backgrounds and under-predict for far backgrounds. We also see that there are some ranges where the cubes and background are generally high, although the edge pixels are never predicted very accurately.

4.3.2 Experiment 2: Aliased Color with Aliased GT Depth (ARGB-AGT)

For the second experiment, we consider the effects of using aliased rather than anti-aliased imagery for both the input color and depth. In this case, the edges of objects are well-defined and each pixel has only a single source of truth. This is shown in the top row of Fig. 3. Although each individual pixel now maps to a single real depth value rather than an average, over many training examples similar looking edge features can map to both foreground and background depths. This leads to another ambiguity. Should an edge pixel be rendered by the algorithm as foreground or background?

We use the same loss function as before (Eq. 1) and train using the same data splits. Some example outputs are shown in Fig. 7. Again, we see that the model appears to do quite well qualitatively when generating grayscale

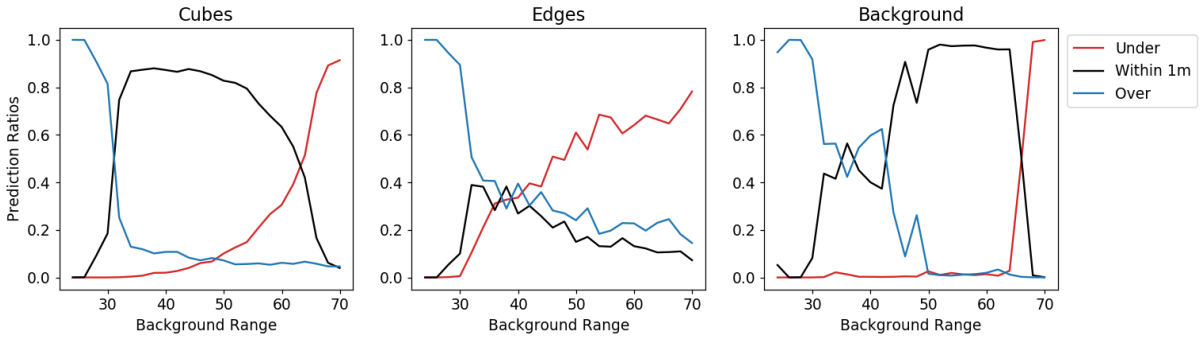

Figure 6. Depth prediction accuracy ratios for the anti-aliased color to anti-aliased ground truth depth model (AARGB-AAGT). For each pixel type (Cube, Edge, Background), the percentage of pixels that were predicted to be within 1 meter of the ground truth depth are shown alongside the ratios of pixels that were over and under-predicted.

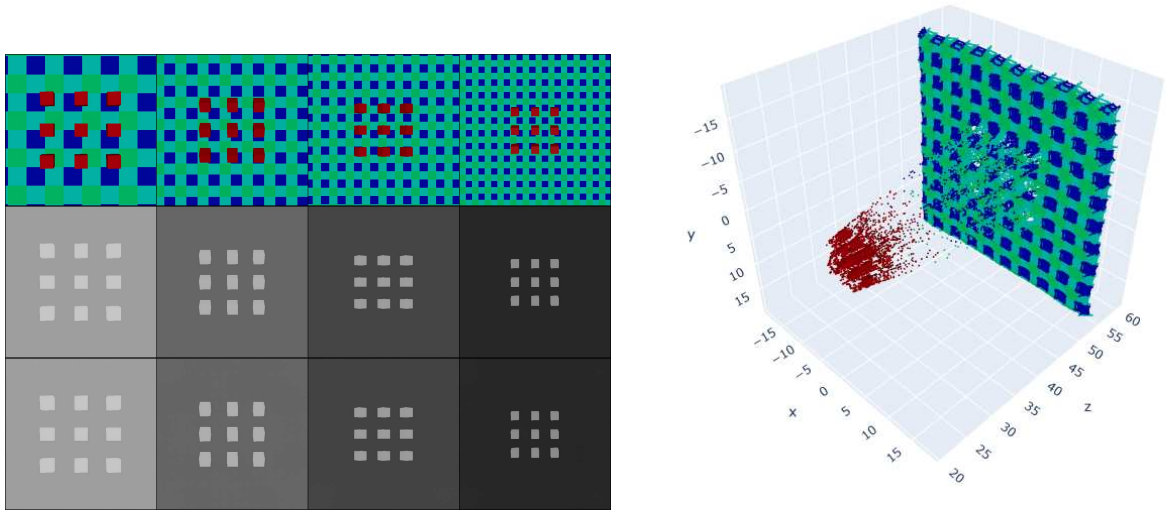

Figure 7. Left: Example outputs of the aliased color to aliased ground truth depth model (ARGB-AGT). Each column is a different sample from the dataset with a receding background. The top row shows the color input, the middle row shows the target ground truth depth, and the bottom row shows the predicted output. Right: A 3D point cloud projection of one of the model predictions.

depth images. The 3D point cloud also appears to have fewer trailing points directly in the middle between the foreground and background. Most points are closer to either the foreground or background.

The depth prediction histograms for this aliased case are shown in Fig. 8. We see from these plots that the aliased ground truth depth causes a bimodal distribution in the depth prediction for the edge pixels. This confirms what we observe with the 3D point cloud in Fig. 7. Besides this trend, we note that many of the previous observations hold, such as over-predicting when the background is close and under-predicting when the background is far.

Figure 9 shows a breakdown of prediction accuracy using the aliased model over all ranges for cubes, edges, and background pixels. Overall, the trends are similar to the previous anti-aliased model shown in Fig. 6. One notable difference is the relatively equal likelihood of the edge pixels to be over or under-predicted at far

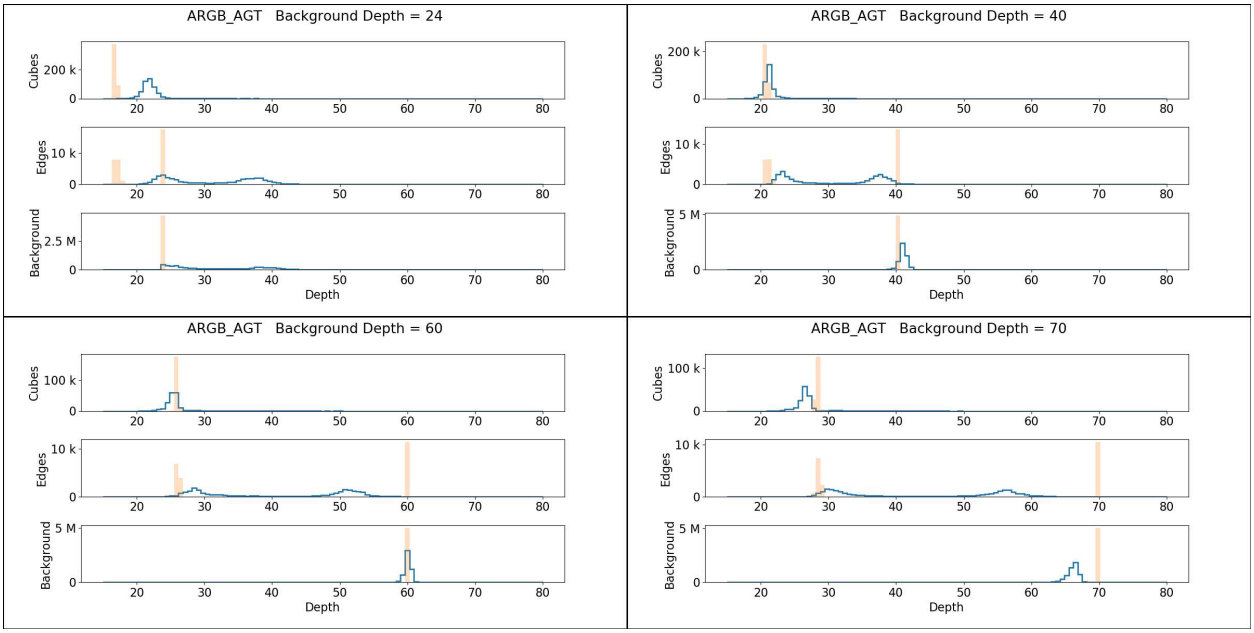

Figure 8. Depth prediction histograms of the aliased color to aliased ground truth depth model (ARGB-AGT). Results are shown for 4 different background depths. For each depth, the per-pixel predictions of 20 test images are aggregated, separated into Cube, Edge, and Background sets. Ground truth depths are shown with solid orange bars and the predictions are shown as a stair-step plot in blue.

background ranges rather than exclusively under-predicted. This can be explained by the presence of two modes in the edge pixel depth distributions, which balances both the over and under-predictions.

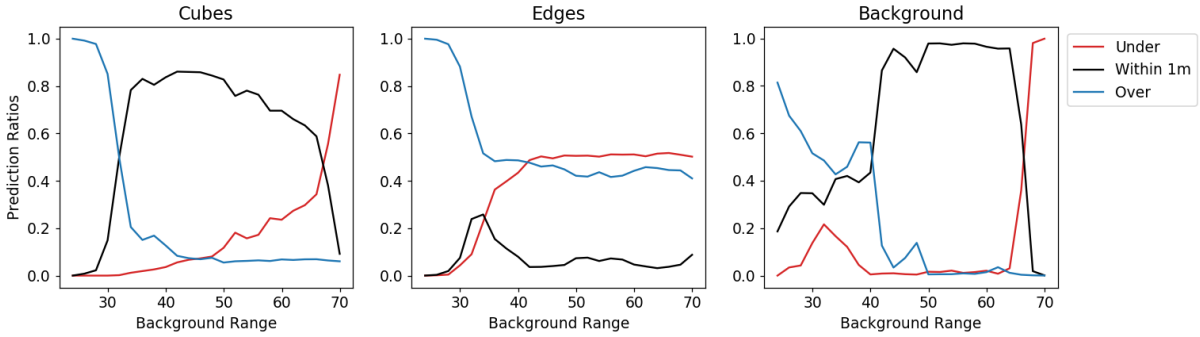

Figure 9. Depth prediction accuracy ratios for the aliased color to aliased ground truth depth model (ARGB-AGT). For each pixel type (Cube, Edge, Background), the percentage of pixels that were predicted to be within 1 meter of the ground truth depth are shown alongside the ratios of pixels that were over and under-predicted.

4.3.3 Experiment 3: Anti-aliased Color with Bundled GT Depth (AARGB-BundleGT)

The next experiment proposes an alternative to using either aliased or anti-aliased imagery exclusively. In this method, we use the anti-aliased color imagery and the bundled depth values as ground truth. This allows the model to learn using the more natural imagery, but still have access to the original source of each depth contribution.

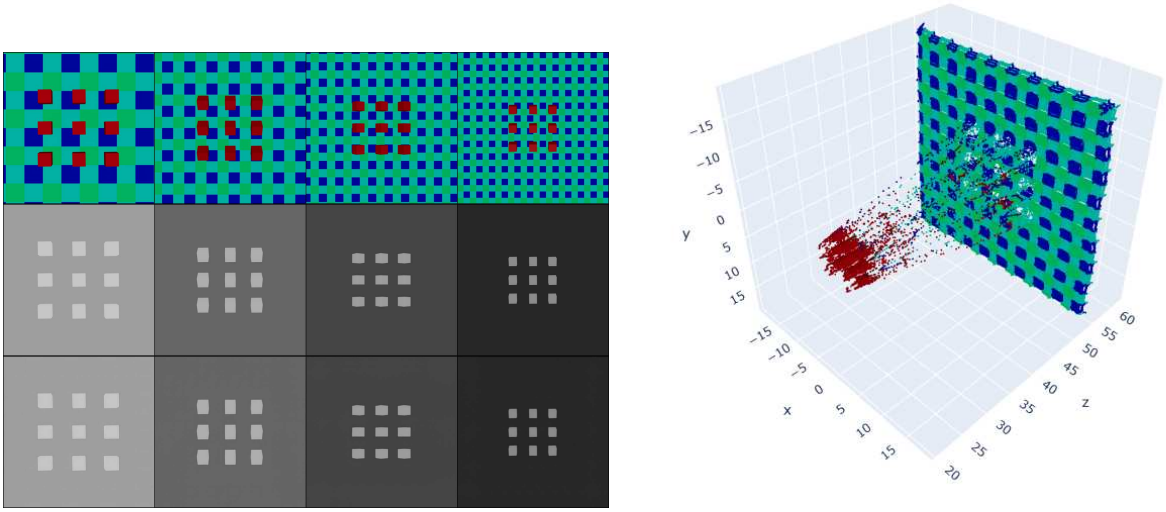

Figure 10. Left: Example outputs of the anti-aliased color to bundled ground truth depth model (AARGB-BundleGT). Each column is a different sample from the dataset with a receding background. The top row shows the color input, the middle row shows the target ground truth depth, and the bottom row shows the predicted output. Right: A 3D point cloud projection of one of the model predictions.

The loss function is modified as

$$L(X, \hat{Y}) = \frac{1}{N} \sum_{i} \min_{y_i \in Y_i} \left\{ \big(\log(y_i) - \log(X_i) \big)^2 \right\}, \tag{2}$$

where X_i is the predicted ground truth depth of pixel i, Y_i is the bundle of depths at pixel i, and each y_i is a different depth value in the bundle. $\hat{Y}$ is the set of all depth bundles for all N pixels in the image. Again, a model was trained on half of the generated data (odd frames) and evaluated on the other half (even frames) using this updated loss function.

Some example outputs are shown in Fig. 10. Again, the grayscale depth predictions look very good, but we see a return of the trailing points in the 3D projection. A look at the depth prediction histograms in Fig. 11 shows that the model has again returned to a wide and flat distribution for edge pixels between the foreground and background. The same trends of over and under-predicting can be observed here as well. One explanation for this is that the bundle loss treats all distance values equally. This means that any intermediate depth value can be chosen without preference by the algorithm.

In terms of accuracy rates at various background ranges, only the correct predictions (those within 1 meter of at least one of the ground truth depths in the bundle) can be shown in Fig. 12. This is because when the ground truth spans a range instead of a single value, it becomes hard to classify a point as over or under-predicted. Nevertheless, we observe similar trends in the correctly classified pixels as with the previous two experiments.

4.3.4 Experiment 4: Anti-aliased Color with Bundled GT Depth and Minimum Preference (AARGB-BundleGT-min)

The final experiment is a variation on the previous bundle experiment to demonstrate the effect of changing the loss function. Whereas the loss function presented in Eq. 2 takes the minimum of the squared difference between the ground truth and the predicted depth, for this experiment we perform the squaring operation after taking the min. Effectively, this results in introducing a bias towards favoring predicting closer depths.

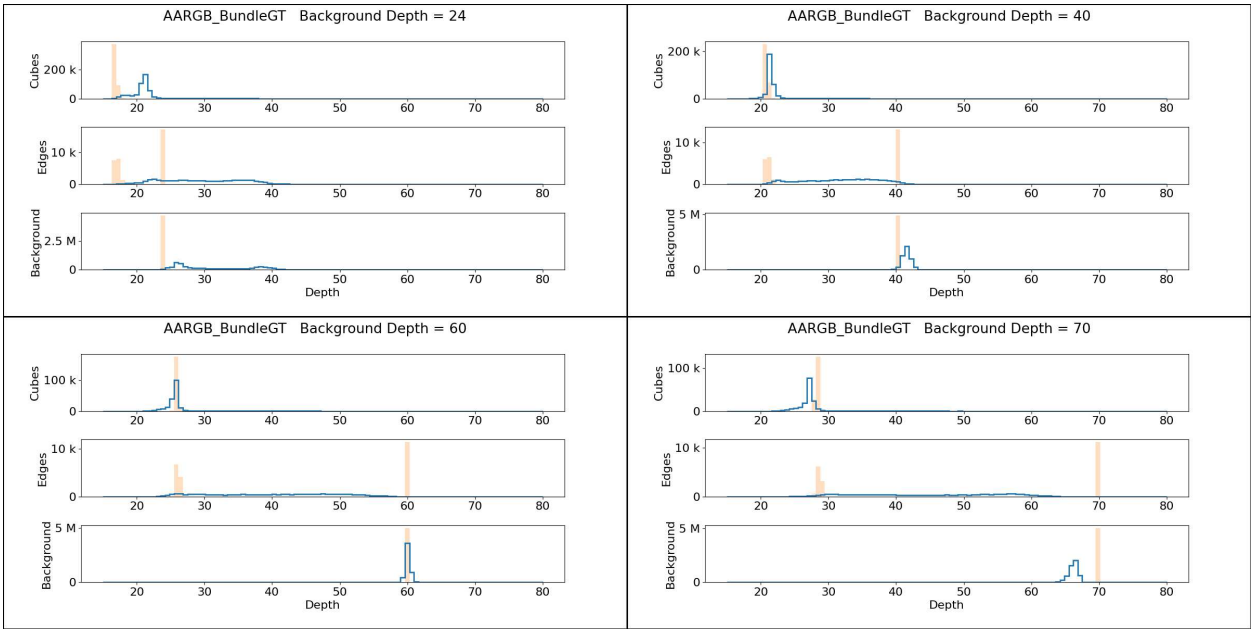

Figure 11. Depth prediction histograms of the anti-aliased color to bundled ground truth depth model (AARGB-BundleGT). Results are shown for 4 different background depths. For each depth, the per-pixel predictions of 20 test images are aggregated, separated into Cube, Edge, and Background sets. Ground truth depths are shown with solid orange bars and the predictions are shown as a stair-step plot in blue.

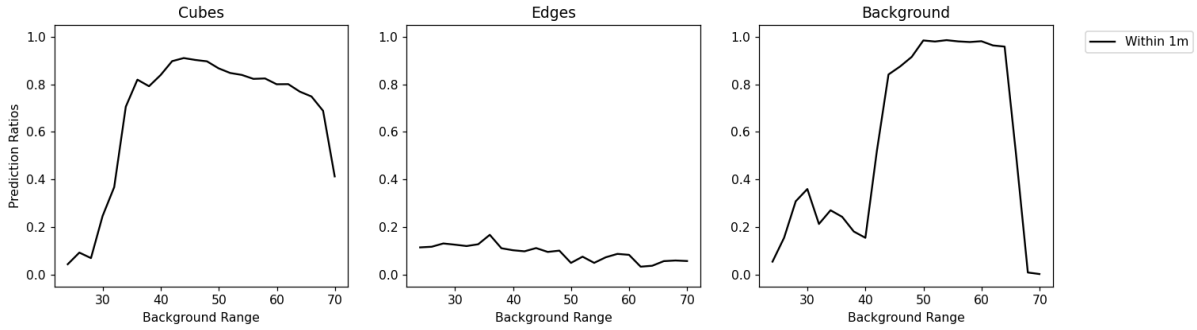

Figure 12. Depth prediction accuracy ratios for the anti-aliased color to bundled ground truth depth model (AARGB-BundleGT). For each pixel type (Cube, Edge, Background), the percentage of pixels that were predicted to be within 1 meter of the ground truth depth are shown.

The depth prediction histograms for this experiment can be seen in Fig. 13. Compared to the plots of the previous experiment in Fig. 11, this modification to the loss function shifts the predictions of edge pixels towards the foreground cubes. This demonstrates that the bundle approach can be tailored to suit the needs of a specific problem by changing the way that the loss is computed.

5. CONCLUSIONS

To summarize, simulated ground truth can be a useful way to train AI algorithms, but there are several shortcomings that need to be addressed. When a pixel in an image represents only a single value, it aggregates or discards information about the true underlying environment that may be important. For this reason, we have

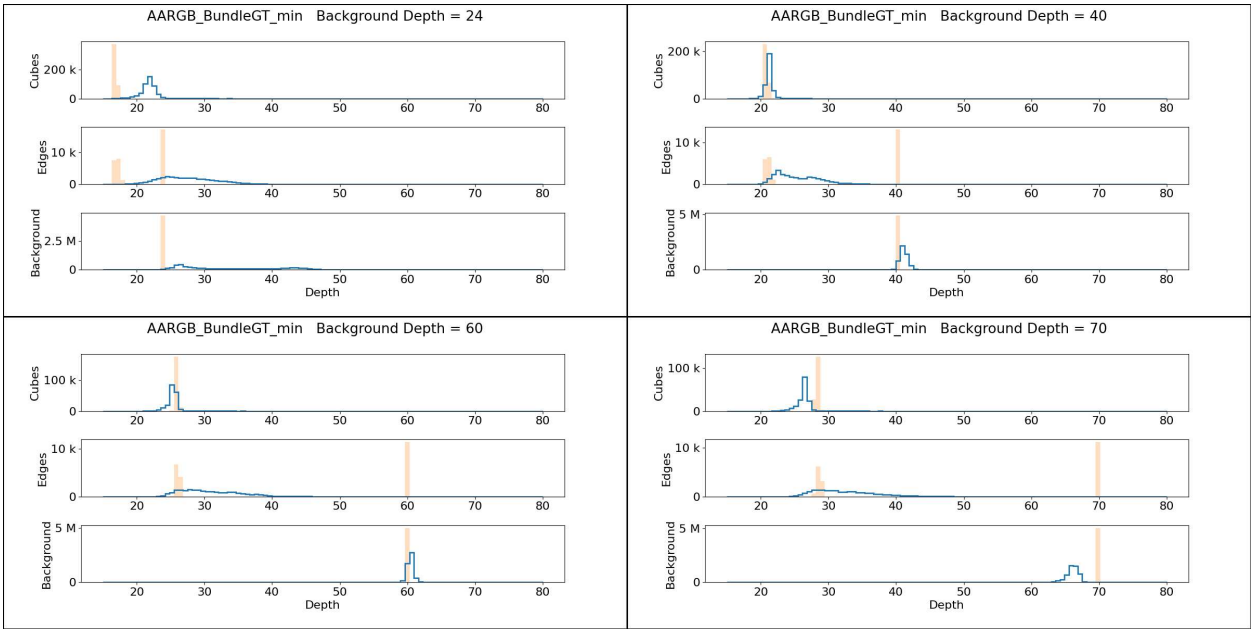

Figure 13. Depth prediction histograms of the anti-aliased color to bundled ground truth depth model with minimum preference (AARGB-BundleGT-min). Results are shown for 4 different background depths. For each depth, the per-pixel predictions of 20 test images are aggregated, separated into Cube, Edge, and Background sets. Ground truth depths are shown with solid orange bars and the predictions are shown as a stair-step plot in blue.

come to prefer the idea of a "gold standard" rather than a ground truth, since the real truth cannot be known by imaging alone.

There are ways to mitigate this issue. Aliased and anti-aliased imagery are often used, but should be done so with care, and with an awareness of the data types being used. For instance, a color image may be anti-aliased to look more lifelike, but at the cost of making other features, such as pixel depth, less well-defined. Choosing a particular depth (by making the depth aliased) introduces bias into the model. An alternative is to represent the pixel as a bundle of values with a distribution that can be used to guide a loss function during training. Using a bundle instead of a single truth can allow the machine to learn that a single input feature may map to many different possible truth values. This is the primary motivation for this approach.

Our experiments demonstrate some of the different ways to manage ground truth. We showed the use of aliased, anti-aliased, and bundle approaches with subtly different results for each. This is not an exhaustive list of ways to manage ground truth issues. For example, one could attempt to model the underlying geometry of a scene with a triangle mesh, but this leads to practical complications and issues that may arise from overlapping or missing data. Another factor we did not explore is the temporal aspect of a scene that changes over time and how the bundle ideas could be extended to that case.

We focused here on the use of various ground truth representations to train a network, but did not look deeply at how to evaluate these approaches more broadly. Quantitative metrics and more realistic data sets could help advance this area of research. Our current bundle implementation could be optimized by using a compressed representation to save space and computing time. There may also be other loss functions that can better utilize the bundled ground truth data. Ultimately, the issues presented in this article should be considered whenever using synthetically generated ground truth to avoid unintentional bias.

REFERENCES

[1] McCormac, J., Handa, A., Leutenegger, S., and Davison, A. J., "SceneNet RGB-D: Can 5M Synthetic Images Beat Generic ImageNet Pre-training on Indoor Segmentation?," in [*2017 IEEE International Conference on Computer Vision (ICCV)*], 2697–2706, IEEE, Venice (Oct. 2017).

[2] Richter, S. R., Vineet, V., Roth, S., and Koltun, V., "Playing for Data: Ground Truth from Computer Games," in [*Computer Vision – ECCV 2016*], Leibe, B., Matas, J., Sebe, N., and Welling, M., eds., *Lecture Notes in Computer Science*, 102–118, Springer International Publishing, Cham (2016).

[3] Reyes, M. F., D'Angelo, P., and Fraundorfer, F., "SyntCities: A Large Synthetic Remote Sensing Dataset for Disparity Estimation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **15**, 10087–10098 (2022). Conference Name: IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing.

[4] Gaidon, A., Wang, Q., Cabon, Y., and Vig, E., "VirtualWorlds as Proxy for Multi-object Tracking Analysis," in [*2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*], 4340–4349 (June 2016). ISSN: 1063-6919.

[5] Smith, A. R., "A Pixel Is Not A Little Square, A Pixel Is Not A Little Square, A Pixel Is Not A Little Square! (And a Voxel is Not a Little Cube)," Tech. Rep. Technical Memo 6, Microsoft Research (1995).

[6] Smith, A. R., [*A Biography of the Pixel*], MIT Press (Aug. 2021).

[7] Godard, C., Mac Aodha, O., Firman, M., and Brostow, G., "Digging Into Self-Supervised Monocular Depth Estimation," *arXiv:1806.01260 [cs, stat]* (Aug. 2019).

[8] He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," (Dec. 2015).