Adversarial Training on Point Clouds for Sim-to-Real 3D Object Detection

Robert DeBortoli[1*], Li Fuxin[1], Ashish Kapoor[2], Geoffrey A. Hollinger[1]

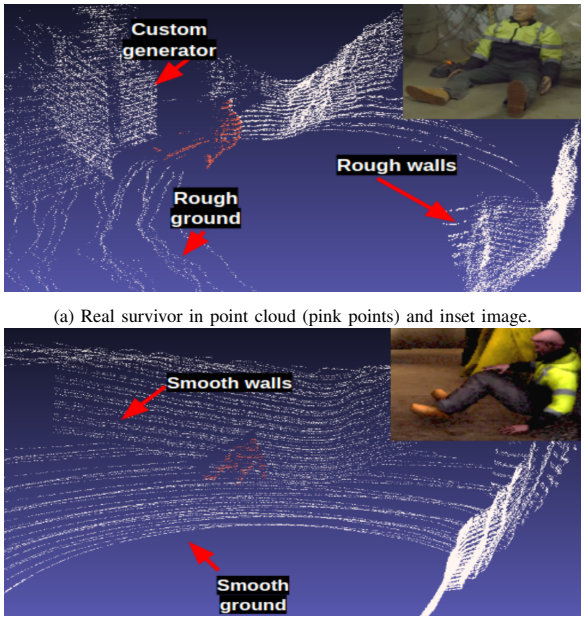
***Abstract*—In this work we address the problem of 3D object detection from point clouds in data-limited environments. Training with simulated data is a common approach in such scenarios; however a *sim-to-real* gap exists between clean and crisp simulated clouds and noisy real clouds. Previous sim-to-real approaches for processing point cloud scenes have compressed clouds into 2D and used 2D transfer techniques. However, this may compress useful 3D information and does not effectively reason about the unstructured nature of point cloud data. We thus propose a 3D adversarial training architecture that leverages an adaptive sampling module to reason about the unstructured nature of point cloud data. Our approach encourages the 3D feature encoder to extract features that are invariant across simulated and real scenes. We validate our approach in the context of the DARPA Subterranean Challenge and demonstrate that our 3D adversarial training architecture improves 3D object detection performance by up to 15% depending on the data representation.**

***Index Terms*—Deep Learning Methods, Field Robots, Object Detection, Segmentation and Categorization**

I. INTRODUCTION

SCENARIOS such as search and rescue in poorly-lit underground environments [1] and autonomous driving in poor weather [2] can cause the same object or environment to appear differently in camera imagery depending on the conditions. Light Detection and Ranging (LiDARs) are increasingly being used as a complementary sensor to cameras because of their ability to better capture clear sensor data in such low-light or poor weather conditions. In this work we address 3D object detection in point clouds from a LiDAR.

Recently, deep learning approaches have achieved state-of-the-art performance for object detection from point clouds [3]. Oftentimes such methods rely on having large amounts of real data for training. In this work we target applications where a large amount of training data do not exist; such as those in agricultural robotics [4] or those motivating the DARPA Subterranean (SubT) Challenge [5].



(a) Real survivor in point cloud (pink points) and inset image.

(b) Synthetic survivor in point cloud (pink points) and inset image.

Fig. 1: Example of the sim-to-real gap for point cloud object detection. The real point cloud (top) has rough walls and features as well as objects such as the generator that are not found in simulation. The synthetic point cloud (bottom) has relatively smooth surfaces. Points are colored for visualization purposes only and are not used for object detection.

Simulators can be used to generate large amounts of training data, however, as shown in Fig. 1, a *sim-to-real gap* often exists. This gap limits the performance of networks that naively incorporate simulated data for training [6], [7].

Addressing the sim-to-real gap for point clouds has received limited research attention, often focusing on enforcing similarity constraints between real and synthetic feature extraction. For example, SqueezeSegV2 is a feature comparison approach which uses a geodesic distance function between real and synthetic feature vectors [8]. However, in [8] point clouds were compressed into 2D first which can cause the loss of detailed 3D information [9]. While fully 3D methods exist, these approaches often focus on small-scale applications such as single-object classification, which we will show is not always scalable to larger robotic scenes [9].

Adversarial training is an alternative method for feature comparison which has seen success for comparing features from 2D camera images. For example, Ganin et al. [6] used

Manuscript received: February 24, 2021; Revised May 26, 2021; Accepted June 18, 2021.

This paper was recommended for publication by Editor Markus Vincze upon evaluation of the Associate Editor and Reviewers' comments. Approved for public release; distribution is unlimited. This work was in part sponsored by DARPA under agreement #HR00111820044. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor

Li Fuxin was partially supported by the National Science Foundation (NSF) under Project #1751402 and #2040735, as well as USDA National Institute of Food and Agriculture (USDA-NIFA) under Award 2019-67019-29462

[1]Robert DeBortoli, Li Fuxin, and Geoffrey A. Hollinger are with the Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University, Corvallis, Oregon 97331 {debortor, fuxin.li, geoff.hollinger}@oregonstate.edu

[2]Ashish Kapoor is with Microsoft, One Microsoft Way, Redmond, WA 98052 akapoor@microsoft.com

*Robert DeBortoli partially completed this work as an intern at Microsoft.

Digital Object Identifier (DOI): see top of this page.

adversarial training with synthetic road signs to assist with training a network that classifies real road signs. This is accomplished by using an adversarial discriminator to promote extraction of similar features from real and synthetic data.

While such approaches are effective for 2D images, we note that 3D point cloud data is quite different than images particularly with respect to the highly-unstructured nature of point cloud data. Towards developing effective solutions for real-world point cloud object detection, we develop a fully 3D adversarial discriminator which does not compress information into 2D, and is lightweight enough to scale to large robotics scenes. Our adversarial discriminator is not specific to a fixed 3D representation and we analyze its performance with both point and voxel-based detectors [3], [10]. Additionally, for point-based backbones we develop an adaptive sampling module which maintains important data throughout the discriminator.

In summary, in this work we:

- Develop a flexible and lightweight 3D adversarial training scheme that leverages an adaptive sampling module to reason over large robotic scenes.
- Demonstrate the utility of this scheme across voxelized and point-based 3D object detection architectures.
- Release our 3D object detection dataset and code as a training resource and benchmark for future work.

We validate our approach on real data collected in a variety of environments as part of the DARPA Subterranean Challenge and demonstrate that our approach increases performance by up to 15%.

II. RELATED WORK

Limited work has been done for closing the sim-to-real gap for 3D point clouds. Therefore, we first discuss previous work for 2D camera images. We then discuss object detection and sim-to-real approaches for 3D point clouds. Finally, we discuss adaptive sampling methods for 3D point clouds.

A. Using simulated training data with camera images

Addressing the sim-to-real data gap for 2D images has received much research attention. One common technique is to use Generative Adversarial Networks (GANs) for generating realistic images [11]. However GANs are notably difficult to train for point cloud generation due to the increased dimensionality [12]. Therefore such works often focus on small-scale generation instead of large scenes [12], [13].

Adversarial training is a lightweight alternative which uses a discriminator loss as an additional training signal for the feature extractor, encouraging it to extract similar features from both simulated and real data [6], [7]. Ganin et al. demonstrated that this training approach improved real-world performance when using synthetic data for handwriting and road sign recognition [6]. Inspired by these increases in performance for 2D images, we introduce a fully 3D adversarial training approach which can handle the high-dimensonality and lack of structure found in 3D point clouds.

B. 3D object detection in point clouds

3D object detection from point clouds has received increasing attention from the research community [3], [14]. For example, Qi et al. [3] demonstrated accurate 3D object detection in indoor environments from only point clouds. Methods such as ComplexYOLO have also been proposed which project the point cloud into a 2D view and complete object detection from that view [14]. Such techniques work fairly well in autonomous driving scenarios, where large objects such as cars can be seen easily from top-down views. However, they are not appropriate for applications such as underground exploration, where clutter and large overhangs preclude the use of a top-down projection. Additionally, a 2D projection inherently loses valuable 3D information that could be used in downstream tasks and thus many state-of-the-art detectors do not compress clouds [9], [10].

C. Closing the sim to real gap with point clouds

Reducing the domain shift between simulated and real point clouds has received increasing research attention. Wu et al. proposed SqueezeSegV2 which projects point clouds into 2D and uses a geodesic distance function to compare synthetic and real features [8]. This geodesic distance function works well in 2D, however we show it does not scale as well as our adversarial approach to fully 3D processing. Alternatively, PointDAN is a method that operates in 3D: by using attention networks on low-level features it improves 3D object classification performance between different domains [9]. We will show that focusing on local features cannot reason over large 3D scenes as well as our adversarial discriminator.

Recently, much work on reducing the domain shift for point cloud processing has focused on autonomous driving. For example, Wang et al. demonstrated that by leveraging knowledge of test dataset car sizes, great improvements could be made in detection performance when testing on a dataset different than training dataset [15]. This method works well for autonomous driving scenarios, however the size of our objects of interest varied considerably less than cars across different continents. Targeting applications without target domain labels, Yang et al. developed a pseudo-labelling scheme for unsupervised domain adaptation and while they achieved great results, we are targeting applications where some target domain labels exist and can be directly leveraged for training [16]. This is especially important in the SubT application, where the domain gap is very large, leading to difficulties for detectors trained on just source data to produce accurate detections or pseudo-labels.

D. Adaptive sampling for 3D point clouds

Due to the increased dimensionality of 3D point clouds, uniform sampling techniques can remove useful information too early in the processing pipeline [17]. For the point clouds found in SubT a very small number of points represent the object of interest and a large amount of the points represent the walls and floors and therefore a uniform sampling method may not preserve useful object information.

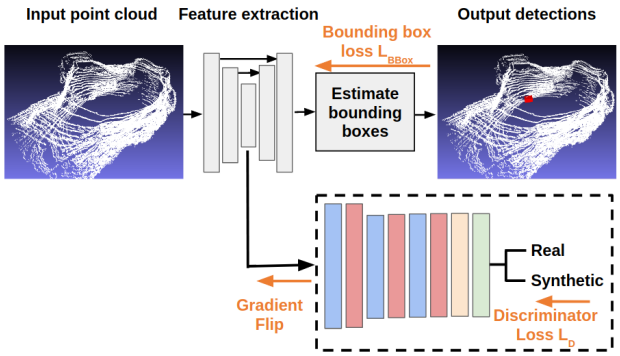


Fig. 2: Overview of our 3D adversarial training architecture. The top gray pipeline is the backbone network, which can be any standard model for point-based or voxel-based representations. Inside of the dashed box is our adversarial discriminator which leverages intelligent adaptive sampling layers (red) to maintain detailed information throughout the pipeline. By flipping the gradient ("Gradient Flip" in orange above) the feature extractor is encouraged to fool the discriminator by extracting similar features shared across synthetic and real data.

One example of adaptive sampling for point clouds is PointASNL which uses a learned adaptive sampling module to improve sampling of local neighborhoods [18]. However, because this module is learned, it potentially will require a lot of data to train. Alternatively, Nezhadarya et al. propose an adaptive global sampling technique, similar to global max pooling, that does not require any additional training [17]. However, standard architectures such as VoteNet require a fixed number of points after each layer. To address this [17] copies the downsampled set, which works well for their object classification application, but may destroy too much information in large robotic scenes.

Our method starts similar to [17] and generates a downsampled set based on the maximally activated features from the previous layer. However, this downsampled set may be too small and sparse for entry into the next layer, and therefore we next upsample using nearest neighbor upsampling around the downsampled set. This allows for structural information to be kept and does not require any additional training, an attractive feature in our data-limited application.

III. METHODS

In this section we first describe the 3D object detection problem and our overall adversarial training architecture. Next, we describe our point-based adversarial training architecture and our adaptive sampling module. Finally, we discuss our voxel-based adversarial training architecture.

A. Problem description

Given a raw 3D point cloud, in this work we seek to train a 3D object detector that outputs 3D bounding boxes with center *(x, y, z)*, size *(length, width, height)*, and yaw (θ).

We validate our approach on an application with a particular lack of data: the subterranean environments which motivate the DARPA Subterranean Challenge [5]. Competitors in this challenge must develop a robotic system to venture into unexplored underground environments and detect certain objects of interest. In this work we target three object classes that are large enough to appear in point cloud data: survivor, backpack, and fire extinguisher. Examples of these objects and their point cloud representations are shown in Fig. 5.

Due to the limited availability of real data, Microsoft's AirSim robot simulator is used to generate large amounts of labelled real data [19]. While the simulator provides a myriad of data, there still exists a sim-to-real gap. Our adversarial training approach reduces the effect of this gap by better leveraging the synthetic data during training.

Algorithm 1 Training algorithm for point-based adversarial discriminator architecture

Input: Gradient penalty coefficient λ, ratio of discriminator to backbone updates n_{critic}, dataset S of real and synthetic data, learning rate l, pretrained backbone parameters θ_0, and initial discriminator parameters w_0.

while θ not converged **do**
  for j = 0,1, ... n_{critic} **do**
    $f_{real}, f_{syn}, f_{mixed} \leftarrow GetFeatures()$
    $L_D \leftarrow D_w(f_{real}) - D_w(f_{syn}) + \lambda(||\nabla_{f_{mixed}} D_w(f_{mixed})||_2 - 1)^2$
    $w \leftarrow Adam(\nabla_w L_D, w, l)$
  $f_{real}, f_{syn}, f_{mixed} \leftarrow GetFeatures()$
  $L_{bboxreal} \leftarrow L_{bbox}(bboxes_{real}, target_{real})$
  $L_{bboxsyn} \leftarrow L_{bbox}(bboxes_{syn}, target_{syn})$
  $L_{bboxmixed} \leftarrow L_{bboxreal} + L_{bboxsyn}$
  $L_D \leftarrow D_w(f_{real}) - D_w(f_{syn}) + \lambda(||\nabla_{f_{mixed}} D_w(f_{mixed})||_2 - 1)^2$
  $\theta \leftarrow Adam(\nabla_\theta L_{bboxmixed} - L_D, w, l)$
function GETFEATURES
  $inp_{real} \leftarrow SampleRealBatch(S)$
  $bboxes_{real}, f_{real} \leftarrow Backbone(inp_{real})$
  $inp_{syn} \leftarrow SampleSynBatch(S)$
  $bboxes_{syn}, f_{syn} \leftarrow Backbone(inp_{syn})$
  $f_{mixed} \leftarrow \epsilon f_{real} + (1-\epsilon) f_{syn}$
  return $f_{real}, f_{syn}, f_{mixed}$

B. Training overview

An overview of our training approach can be found in Fig. 2 and Algorithm 1; we first start by training the discriminator. We feed input data into the backbone network, take the feature encoding produced and use it as input into our adversarial discriminator which completes a binary classification, either synthetic or real. The Wasserstein-GAN Gradient Penalty Loss (WGAN-GP)[1] is then computed on this output:

$$L_D = D_w(f_{real}) - D_w(f_{syn}) + \lambda(||\nabla_{f_{mixed}} D_w(f_{mixed})||_2 - 1)^2, \quad (1)$$

where D and w are the discriminator and its weights, f_{real} and f_{syn} are feature encodings from real and synthetic data respectively, and f_{mixed} is defined as:

$$f_{mixed} = \epsilon f_{real} + (1-\epsilon) f_{syn}, \quad (2)$$

where ϵ is a random number between 0 and 1 [20].

[1] https://github.com/caogang/wgan-gp

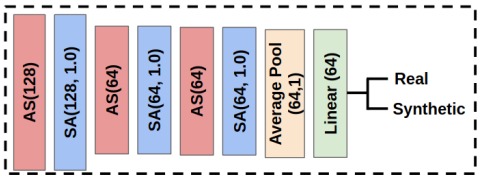


Fig. 3: Our novel point-based adversarial discriminator which leverages an adaptive sampling layer to maintain object information throughout the processing pipeline. SA(x,y) = VoteNet Set Abstraction layer with x points and y radius. AS(x) = adaptive sampling to x number of points. Average pooling is used to avoid destroying detailed information.

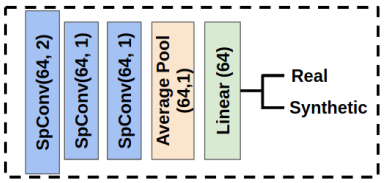


Fig. 4: Our novel adversarial discriminator used for voxel-based experiments. This discriminator works with the PartA2 backbone which leverages sparse convolutions to operate over voxelized inputs. SpConv(x,y) is a sparse convolution layer with x number of filters and y stride.

We note that using the WGAN-GP loss function is a departure from the standard adversarial training paradigm [6]. WGAN-GP leverages a gradient penalty term to provide a more stable loss signal than standard distance measures and has typically been used for applications that *generate* data such as 2D images or small single-object 3D point clouds [21]. Towards deployment on large 3D robotic scenes, in this work, we adapt this loss function to the application of adversarial training on point clouds, thereby enabling stable training on high-dimensional and unstructured point clouds.

Next, the weights of the discriminator are updated according to L_D. This loop is run n_{critic} times to ensure the discriminator is trained enough with respect to the backbone. Next, we update the weights of the backbone network. Once again, data is input and now training utilizes both the output bounding boxes as well as the output of the discriminator. To compute the bounding box loss, L_{BBox}, we use the standard loss function found in [3] which balances objectness, bounding box location, and bounding box size compared to the ground truth bounding box parameters. During this phase we again compute the discriminator loss L_D, however we now pass the gradients through the discriminator. After the gradients get through the discriminator, they are flipped, which trains the feature extractor layers in the backbone to maximize L_D. The final loss for the backbone is thus:

$$L_{Backbone} = L_{BBox} - \alpha L_D, \tag{3}$$

where α is a balancing parameter between the two losses.

By training the backbone network to predict accurate bounding boxes *and* fool the discriminator, we enforce similar feature extraction from synthetic and real data, thereby enabling better usage of synthetic data during training. At inference time, the adversarial discriminator is removed and the backbone architecture produces 3D bounding boxes.

C. Point-based architecture

We next discuss our point-based adversarial architecture and our adaptive sampling module. The VoteNet model is used as a backbone for feature extraction and bounding-box generation [3]. VoteNet operates on the raw point clouds without compression into 2D or discretization by voxelization and is a top approach for 3D object detection on the ScanNetV2 dataset [22]. It is thus well-suited for dense environments such as tight underground tunnels.

An overview of our point-based discriminator can be seen in Fig. 3. For feature extraction, we leverage 3D Set Abstraction layers for seamless integration with VoteNet as well as to support a powerful 3D-aware discriminator. Additionally, we develop an adaptive sampling module that maintains important information throughout the discriminator. This module is a drop-in replacement for the Farthest Point Sampling used in vanilla Set Abstraction layers. To start, similar to [17], we downsample to the set of features that are maximally activated. This downsampled set may be too small for input into the next layer, so to bring it up to the fixed size required by VoteNet, we sample the nearest neighbors of the points in the downsampled set. This is distinctly different than [17] which copies points in the downsampled set to upsample, potentially losing out on valuable neighborhood information which could be leveraged by the discriminator later on for classification.

D. Voxel-based architecture

To demonstrate the flexibility of our 3D adversarial training approach, we also develop a voxel-based adversarial training architecture. For our backbone we leverage the PartA2 architecture; a 3D voxelized model which uses sparse convolution to efficiently extract features [10]. PartA2 has shown high-quality 3D object detection capabilities in outdoor scenes and is one of the best publicly available methods for pedestrian and cyclist detection on the KITTI dataset [23]. PartA2 is thus appropriate for cavernous open-spaces underground such as natural caves.

To integrate effectively with the PartA2 backbone, we use the same type of sparse convolutional feature extractors in our discriminator as in the backbone. Once again, we leverage 3D-aware layers to enable a powerful discriminator which can reason about the incoming features fully in 3D. The discriminator is shown in Fig. 4.

For training the voxel-based architecture we use the same overall training structure as the point-based architecture in Algorithm 1. Due to the voxelization process, different batches had different amounts of voxels which made the computation of a gradient penalty term between synthetic and real feature vectors difficult in PyTorch. Therefore, we used the WGAN optimization technique which is similar to WGAN-GP, but does not include the gradient penalty term (i.e. $(||\nabla_{f_{mixed}} D_w(f_{mixed})||_2 - 1)^2$ in Equation 1) [24].

IV. EXPERIMENTS AND RESULTS

In this section, we start with a description of our robot system and point cloud dataset. Next, we demonstrate the benefits of our adversarial training approach on real-world

TABLE I: Real dataset composition (# clouds)

	Backpack	Fire Ext.	Survivor	No object
Train	49	88	67	1522
Validation	78	29	26	125
Test	54	53	16	568

TABLE II: Synthetic dataset composition (# clouds)

	Backpack	Fire Ext.	Survivor	No object
Train	185	225	383	5434
Validation	354	454	180	5229
Test	118	30	163	7069

detection datasets. Finally, we explore the effects of feature encoding levels as input to our adversarial discriminator.

A. System overview

Real data used for training and testing was captured by Team Explorer's custom-built ground vehicles designed for the DARPA SubT Competition. These vehicles typically travel with speed up to 2 m/s and have a variety of onboard sensors including rgb and thermal cameras as well as a Velodyne VLP-16 used to generate the point clouds for this work. We spin our Velodyne at 10 Hz and use the LOAM SLAM system to concatenate sets of about 5 individual scans for generating high-resolution inputs to our model [25].

To generate synthetic data, we use Microsoft AirSim with a synthetic underground tunnel environment rendered in the Unreal engine [19]. High-resolution surface meshes enable accurate LiDAR simulation. The simulated vehicle has realistic size, physical capabilities, and LiDAR parameters.

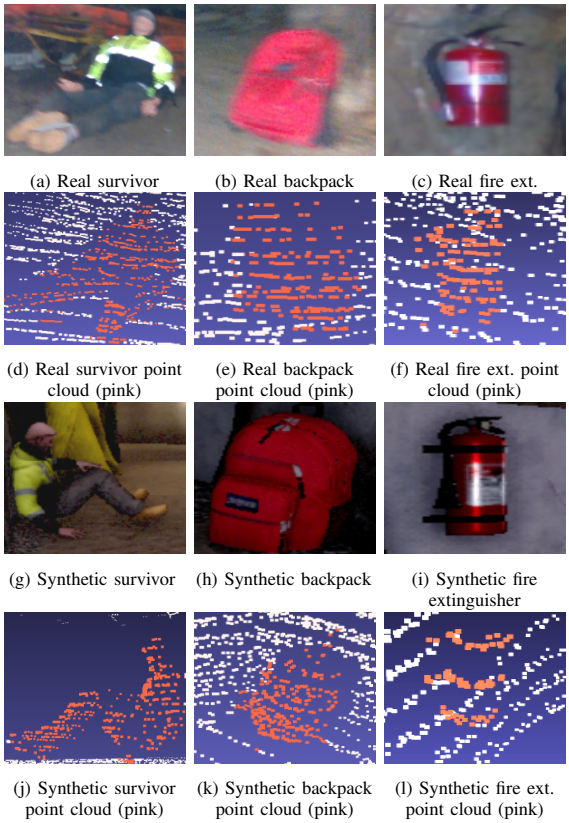
B. Dataset overview

Fig. 5a - 5f and Table I contain an overview of our real dataset. Training data was captured during the DARPA SubT STIX event at the Edgar Experimental Mine, CO, the SubT Tunnel Circuit at the NIOSH Mine, PA, and on the Carnegie Mellon University (CMU) Campus, PA in representative urban environments such as a parking garage. Validation data was captured at Tour-Ed mine, PA. Test data is from the Edgar Mine, the Tour-Ed mine, and the CMU campus in parts of the environments different than those used for capturing training/validation data. After collection, the data was hand-labelled by a human expert.

Fig. 5g - 5l and Table II contain an overview of our synthetic dataset. Data was gathered by randomly moving the robot around, collecting scans, and using AirSim's automatic label generation which enabled a large amount of data to be generated cheaply. Training, validation, and test sets were all gathered from different locations in the simulation. Additionally, due to the realistic nature of the SubT Challenge, often only 10-15% of the clouds in the dataset contain objects.

C. 3D object detection performance

We next discuss pre-processing and comparison methods for both the point and voxel-based experiments. We then present point and voxel-based object detection results.



(a) Real survivor (b) Real backpack (c) Real fire ext.

(d) Real survivor point cloud (pink) (e) Real backpack point cloud (pink) (f) Real fire ext. point cloud (pink)

(g) Synthetic survivor (h) Synthetic backpack (i) Synthetic fire extinguisher

(j) Synthetic survivor point cloud (pink) (k) Synthetic backpack point cloud (pink) (l) Synthetic fire ext. point cloud (pink)

Fig. 5: Dataset overview. (a)-(c) are the objects in camera images taken from the robot. Notice the motion blur that can occur in camera imagery underground. (d)-(f) are the objects in point cloud form. (g)-(i) and (j)-(l) show the camera images and point clouds from AirSim. Note that while the real and synthetic point cloud representations are similar, the simulated clouds are often cleaner and more regularly sampled.

1) Point cloud preprocessing: Before point clouds are input to the model, there are a number of preprocessing steps. Clouds are first downsampled about 20% to 80,000 points to make training and inference faster. Keeping the original number of points did not significantly improve performance. Because of the sparsity of objects to be detected in the environment, clouds are first cropped to 5 meter cubes to make learning easier. We note that this brings our clouds to about the same physical size as other point cloud datasets like ScanNet [22] and that cropping has been used in previous works [26]. To ensure objects were actually observable in the clouds, we did not label objects that did not meet certain size and number of point constraints: The backpack had a minimum (length, height, width) of (10, 10, 10) cm and a minimum of 40 points, fire extinguishers had a minimum size of (5, 20, 5) cm and 60 points, and the survivor had a minimum size of (25, 25, 25) cm and minimum of 80 points.

2) Comparison methods: We compare our adversarial approach against a number of baselines. The first three methods only use the VoteNet architecture (for point-based experiments) or the PartA2 architecture (for voxel-based experiments) and are dataset composition approaches:

- *OnlySyn* uses only synthetic data for training/validation.
- *OnlyReal* uses only the small amount of real data.
- *SynAndReal* mixes the synthetic and real data for train-

TABLE III: Point-based 3D object detection results

Train method	Average precision on real test data			mAP
	Backpack	Survivor	Fire Ext.	
OnlySyn	0.29±0.14	0.04±0.07	0.00±0.01	0.11±0.03
OnlyReal	0.56±0.02	0.49±0.09	0.05±0.01	0.37±0.04
SynAndReal	0.56±0.03	0.61±0.13	0.03±0.03	0.40±0.02
GeoLoss [8]	0.61±0.02	0.54±0.05	**0.07**±0.02	0.41±0.03
PointDAN [9]	0.57±0.03	0.45±0.10	0.04±0.03	0.35±0.03
AT (Ours)	0.62±0.04	0.64±0.06	0.04±0.01	0.43±0.04
AT-AS (Ours)	**0.63**±0.02	**0.71**±0.04	**0.07**±0.04	**0.47**±0.01

ing/validation. To ensure real data is emphasized during learning, it is sampled at a 2:1 rate with synthetic data.

The remaining methods, including our adversarial training approach, all have access to synthetic and real data.

- *GeodesicLoss* (GeoLoss) is based on the work by [8] which utilizes the geodesic distance between the covariance matrices of the real and synthetic feature vectors from 2D point cloud images[2]. While this method is similar to ours in that it encourages synthetic and real feature vectors to be similar, our learned adversarial discriminator is able to scale better to 3D environments.
- *PointDAN* uses Maximum Mean Discrepancy to align local features for point cloud object classification[3] [9]. Focusing on fine-grained features works well for object classification, however we show it does not scale as well as adversarial training to large scenes.
- *Adversarial architecture* (AT) is our proposed approach which uses a lightweight and powerful 3D adversarial discriminator to encourage similar feature extraction from real and synthetic data in a fully 3D manner.
- *Adversarial architecture with adaptive sampling* (AT-AS) uses the same architecture as AT but leverages our adaptive sampling module in the discriminator to avoid removing important object information before completing classification.

Given the small amount of real training data available, it is relevant to explore if non-deep learning methods could outperform deep learned methods. For example, Wang et al. propose a method which extracts shape-factor features from point clouds and uses an SVM to classify these features as an object of interest or not [27]. While their code is proprietary, we implemented this method, ran it on our SubT data, and found it was unable to achieve higher than 0.01 average precision on the test set survivor. We hypothesize this is due to the shape-factor features not being powerful enough to detect objects with high appearance change, as is the case in our application. Specifically, the objects for SubT are significantly smaller than other applications such as the KITTI dataset and therefore common challenges with point clouds such as occlusions and varying point densities can be much more degrading to our objects of interest.

3) Point-based object detection: To train our adversarial discriminator for point-based object detection, we used a learning rate of 0.0005, weight decay of 0.1, batch size of 8, and class sampling weightings of 1, 2, and 2 for the backpack, fire extinguisher and survivor respectively. We used an n_{critic} of 2 because the discriminator trained quickly and a λ value of 10. To emphasize useful feature extraction early on, we varied the α balancing term in a linear manner from 0 to 1 starting at epoch 0 until epoch 300.

All methods were trained to convergence based on validation mAP, which is usually up to 200 epochs and takes 13 hours on a single GTX 1080 GPU. For the point-based object detection experiment, all methods used a VoteNet backbone pretrained on the ScanNet dataset[4]. For data augmentation we rotated/translated the clouds, scaled the clouds by $\pm$ 10%, applied $\pm$ 2 cm of jitter, and moved objects between clouds.

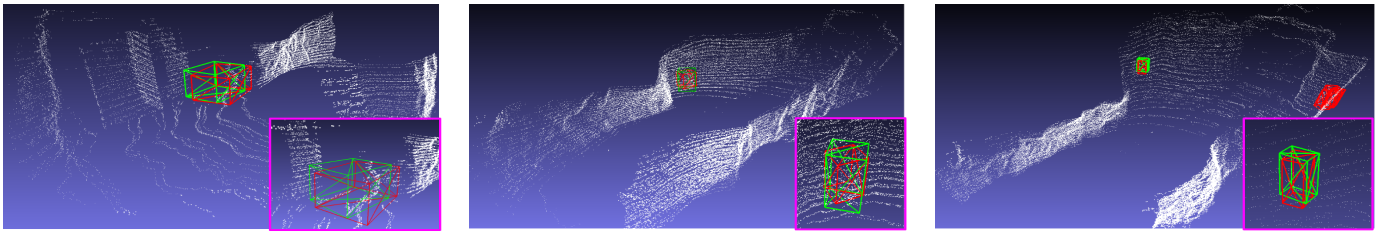
Methods are evaluated using the VoteNet evaluation pipeline and average precision metrics are computed at 0.25 3D IOU. The mean and standard deviation results averaged over three training runs are shown in Table III. At a high-level, the results are intuitive: many methods do the best on the large survivor and perform relatively poorly on the fire extinguisher. Given the rough walls and overhangs that often accompanied the fire extinguisher, this was a particularly challenging environment for such detections.

Investigating the results in detail, the performance difference between OnlySyn and OnlyReal demonstrates the sim-to-real-gap. The difference between OnlyReal and SynAndReal demonstrates the utility of using the synthetic data for increasing the amount of data we have. Due to the use of attention on low-level features, PointDAN struggles to scale to larger 3D scenes. The GeodesicLoss provides improvement over SynAndReal, but due to the inflexible nature of its loss function, it is not able to beat our adversarial approach in many cases. Our adversarial architecture (AT) improves performance on the backpack, survivor, and on the Mean Average Precision (mAP) metric by not compressing information and learning the discriminator while training. Finally, our adversarial architecture with adaptive sampling (AT-AS) achieves the best performance on all three objects by maintaining important information throughout the discriminator.

Qualitative results for our point-based adversarial architecture on real test data from Edgar Mine, CO are shown in Fig. 6. Overall, we are able to detect objects well in this challenging real environment. However, one area for future improvement is the angular estimation of the bounding boxes. In Fig. 6c it is interesting to note that on the right side of the scene we have a number of false positives which were caused by the DARPA organizers covering up another fire extinguisher with a large form-fitting covering. This was not labelled as a ground truth object location in our dataset, and is demonstrative of the difficulty of our task.

4) Voxel-based object detection: For the voxel-based architecture experiments, data pre-processing and data augmentation techniques were all similar to the point-based architecture. All methods use a PartA2 anchor-free backbone[5] pretrained on the KITTI dataset and are trained to convergence based on validation set mAP.

[2]https://github.com/pmorerio/minimal-entropy-correlation-alignment
[3]https://github.com/canqin001/PointDAN
[4]https://github.com/facebookresearch/votenet
[5]https://github.com/open-mmlab/OpenPCDet



(a) Survivor detection (b) Backpack detection (c) Fire extinguisher detection

Fig. 6: Qualitative results for our point-based adversarial architecture. Green boxes are ground truth, red boxes are predicted bounding boxes. The pink inset images are zoomed in on the detection shown in the larger scene. All clouds were captured in the Edgar Experimental Mine, CO as part of the DARPA SubT STIX Integration Exercise. Additional results with 3D manipulation of detections can be found at: https://youtu.be/oXk88gmsx8g

TABLE IV: Voxel-based 3D object detection results

	Average precision on real test data			
Train method	Backpack	Survivor	Fire Ext.	mAP
OnlySyn	0.31±0.06	0.0±0.01	0.07±0.05	0.13±0.01
OnlyReal	0.39±0.06	0.13±0.06	0.18±0.08	0.23±0.01
SynAndReal	**0.57**±0.05	0.09±0.03	0.13±0.08	0.26±0.02
GeoLoss [8]	0.52±0.08	**0.16**±0.07	0.12±0.01	0.27±0.04
PointDAN [9]	0.44±0.07	0.15±0.02	0.15±0.08	0.25±0.01
AT (Ours)	0.46±0.12	0.15±0.03	**0.24**±0.03	**0.28**±0.05

We use a learning rate of 0.0005, weight decay of 0.1, and class sampling weightings of 1, 2, and 2 for the backpack, fire extinguisher and survivor respectively. The same n_{critic} and linear increase to α from the point-based experiment are used. Due to the dense cluttered environments in our dataset we set a very small voxel size of 1.8 cm which allowed clouds to fit onto the GPU. Due to the computational expense of these small voxels, the model took 25 hours to train on 4 Tesla K80 GPUs. We only changed two significant model parameters. To reduce pooling in these dense environments we increased the output pool size from 12 to 20. To encourage the detection of the sparse objects, we decreased the foreground threshold from 0.65 to 0.25.

Voxel-based methods are evaluated in the same manner as the point-based experiment. The results averaged over 3 training runs are shown in Table IV. Overall, the results are intuitive and follow the same trend as the point-based methods: SynAndReal is able to leverage the quantity of the synthetic data and the small amount of real data to perform better than OnlySyn and OnlyReal. PointDAN is unable to provide much improvement due to its focus on small fine-grained features. Different than for the point-based methods, our adversarial approach performs similarly to the GeodesicLoss approach. This could be because a structured voxelized representation does not require the increased flexibility of the adversarial discriminator. Additionally, due to difficulties implementing the WGAN-GP loss function for our voxel-based architecture, the use of a less-stable WGAN loss function could be to blame. In spite of this, our method still does the best at detecting the fire extinguisher, and in terms of Mean Average Precision, if only by a small margin.

While the trends between the voxel-based and point-based architectures are similar, the performance of the voxel-based architecture is not as high overall. This could be because the dense environments VoteNet was designed for, more closely resemble the tight underground tunnels in our dataset.

TABLE V: Effect of feature encoding level on detection performance

	Mean Average Precision (mAP)		
Architecture	Level2	Level3	Level4
Point-based	0.38	0.41	0.47
Voxel-based	0.25	0.31	0.21

Qualitative results for our voxel-based architecture can be found in Fig. 7. Overall, the detector once again is adept at detecting objects in this challenging environment. It is interesting to note that in Fig. 7a that there are few points on the survivor's legs and even though the label does not extend over the legs, the detector is able to use contextual information to extend outward in that direction.

D. Utility of leveraging feature vector across encoding levels

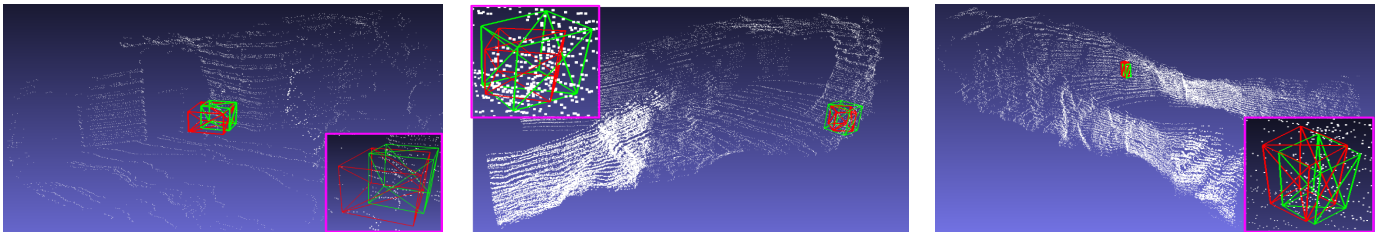
Fig. 2 shows the adversarial discriminator taking input from the fourth level of feature encoding (called "Level4"). In this experiment, for both the point and voxel-based architectures, we examine the effect of taking the feature vector input from different levels of the encoding pipeline.

The results are shown in Table V. For the point-based architecture, the results are fairly intuitive: as the hierarchy of features are built up, the encoded feature vector is more useful to the adversarial discriminator. For the voxel-based architecture it is interesting to note that the performance goes down by using the most-encoded feature vector, *Level4*. This could be due to the more aggressive pooling that occurs with the PartA2 feature extraction pipeline.

V. CONCLUSION

In this work we developed a 3D adversarial training architecture for point cloud object detection in robotic environments where real data is extremely limited. To our knowledge this is the first 3D adversarial approach targeting 3D robotic scenes. Our approach does not compress information into 2D and is lightweight enough to scale to 3D robotics scenes. Additionally, we developed an adaptive sampling module to maintain important 3D information throughout the discriminator. These characteristics offered large performance benefits for unstructured point-based representations.

There are a number of interesting future research directions. Given the performance improvement from using adaptive sampling for point-based architectures, it would be interesting to adapt the module to work on voxelized representations.



(a) Survivor detection (b) Backpack detection (c) Fire extinguisher detection

Fig. 7: Qualitative results for our voxel-based adversarial architecture. Green boxes are ground truth, red boxes are predicted bounding boxes. All clouds were captured in the Edgar Experimental Mine, CO as part of the DARPA SubT STIX Integration Exercise. Additional results with 3D manipulation of detections can be found at: https://youtu.be/oXk88gmsx8g

Another interesting research direction is to improve our small object detection ability which could be achieved by using point-voxel fusion methods [28]. This could improve our performance on the fire extinguisher and enable the detection of other small objects of interest indicative of nearby survivors, such as a helmet.

ACKNOWLEDGMENTS

We thank Sebastian Scherer, Vasu Agrawal, and other members of SubT Team Explorer for help deploying the robots and collecting data. We thank Ratnesh Madaan, Sai Vemprala, and Jim Piavis from Microsoft who helped setup the AirSim simulator and build the underground environment. We also thank Graeme Best for the helpful discussions and the anonymous reviewers for their helpful suggestions.

REFERENCES

[1] W. Wang, W. Dong, Y. Su, D. Wu, and Z. Du, "Development of search-and-rescue robots for underground coal mine applications," *Journal of Field Robotics*, vol. 31, no. 3, pp. 386–407, 2014.

[2] H. Porav, T. Bruls, and P. Newman, "I can see clearly now: Image restoration via de-raining," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7087–7093.

[3] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep Hough voting for 3D object detection in point clouds," in *Proc. IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9277–9286.

[4] R. Barth, J. IJsselmuiden, J. Hemming, and E. J. van Henten, "Optimising realism of synthetic agricultural images using cycle generative adversarial networks," in *Proc. IEEE Conference on Intelligent Robots and Systems Workshop on Agricultural Robotics*, 2017, pp. 18–22.

[5] "DARPA subterranean challenge [online]," www.subtchallenge.com.

[6] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[7] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7167–7176.

[8] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 4376–4382.

[9] C. Qin, H. You, L. Wang, C.-C. J. Kuo, and Y. Fu, "PointDAN: A multi-scale 3D domain adaption network for point cloud representation," in *Proc. Conference on Neural Information Processing Systems*, 2019.

[10] S. Shi, Z. Wang, X. Wang, and H. Li, "Part-A2 net: 3D part-aware and aggregation neural network for object detection from point cloud," *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2019.

[11] C. Baur, S. Albarqouni, and N. Navab, "Generating highly realistic images of skin lesions with GANs," in *OR 2.0 Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis*, 2018, pp. 260–267.

[12] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *Proc. International Conference on Machine Learning*, 2018, pp. 40–49.

[13] D. W. Shu, S. W. Park, and J. Kwon, "3D point cloud generative adversarial network based on tree structured graph convolutions," in *Proc. IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3859–3868.

[14] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-YOLO: An Euler-region-proposal for real-time 3D object detection on point clouds," in *European Conference on Computer Vision*, 2018, pp. 197–209.

[15] Y. Wang, X. Chen, Y. You, L. E. Li, B. Hariharan, M. Campbell, K. Q. Weinberger, and W.-L. Chao, "Train in Germany, test in the USA: Making 3D object detectors generalize," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 713–11 723.

[16] J. Yang, S. Shi, Z. Wang, H. Li, and X. Qi, "St3D: Self-training for unsupervised domain adaptation on 3D object detection," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[17] E. Nezhadarya, E. Taghavi, R. Razani, B. Liu, and J. Luo, "Adaptive hierarchical down-sampling for point cloud classification," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[18] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "PointASNL: Robust point clouds processing using nonlocal neural networks with adaptive sampling," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[19] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*. Springer, 2018, pp. 621–635.

[20] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," in *Proc. Conference on Neural Information Processing Systems*, 2017, pp. 5769–5779.

[21] K. Mo, H. Wang, X. Yan, and L. Guibas, "PT2PC: Learning to generate 3D point cloud shapes from part tree conditions," in *European Conference on Computer Vision*, 2020, pp. 683–701.

[22] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5828–5839.

[23] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[24] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. International Conference on Machine Learning*, 2017, pp. 214–223.

[25] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9, 2014.

[26] P. Kumar Rath, A. Ramirez-Serrano, and D. Kumar Pratihar, "Real-time moving object detection and removal from 3D pointcloud data for humanoid navigation in dense gps-denied environments," *Engineering Reports*, vol. 2, no. 12, p. e12275, 2020.

[27] D. Z. Wang and I. Posner, "Voting for voting in online point cloud object detection." in *Robotics: Science and Systems*, vol. 1, no. 3. Rome, Italy, 2015, pp. 10–15 607.

[28] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 529–10 538.