Towards Open Scene Understanding For Construction Analysis

Emily Steiner
Stanford University
easteine@stanford.edu

Iro Armeni
Stanford University
iarmeni@stanford.edu

Abstract

Construction tracking currently lacks strong quantitative methods to understand the progress and efficiency of a construction site over time. This work aims to improve construction monitoring and provide feedback mechanisms by evaluating a 3D open-scene understanding method for the unique challenges of the construction environment. With the recent surge in robust 2D vision foundational models, this method uses foundational models SAM and GroundingDINO to provide context from multi-view renderings of the scene. Semantic segmentation and labels are predicted for user-defined queries in 2D and are then projected to 3D to create a unified semantically segmented mesh model. Quantitative evaluation on an evaluation dataset show promising zero-shot capabilities, however, segmentation includes errors due to ambiguity between assets and dataset limitations. Qualitative evaluation of the 3D construction dataset shows some successful segmentation, but significant improvements are required to properly capture the long-tail concepts of interest in the construction industry.

1. Introduction

Real-world scenes, particularly in the built environment, may undergo significant structural or usage changes when observed over long periods. Analyzing and tracking structural or usage changes can provide valuable feedback to the historically wasteful construction and design industries. Currently, construction managers rely on anecdotal and visual estimates to track construction progress. Beyond the building plans, there is no consistent captured documentation of the progress of the building, structural changes, location of installed assets, or construction errors causing rework. In addition, in constructed buildings, usage changes such as rearranging room assets can provide valuable feedback to architects and designers to optimize layouts and designs for future buildings. Data cannot be captured continuously, and scene measurements may be spaced weeks or months apart. In a construction site environment, the scene may have extreme structural changes. This project explores the initial stages of creating a spatiotemporal scene understanding model to provide queryable resources for the architecture, engineering, and construction industries.

Given the timeline for this course project, the focus will be limited to a simplified version of the overall project as the first step toward a unified spatiotemporal model. This project hopes to develop a simplified version for constructing a unified 3D semantic segmentation given a scene given an arbitrary set of open vocabulary assets. For the scope of the class, the initial focus lies on one-shot open vocabulary segmentation on long-tailed concepts of interest. This represents the first steps toward a unified 4D scene model to provide consistent instances.

Open vocabulary semantic segmentation will facilitate the construction tracking task described. By querying a 3D semantic model for a question of interest, volume estimates can provide quantified metrics that represent the Ideally, a construction manager would be able to query progress and usage using statements such as "Where is every *light bulb* in the building?", "What percent of the insulation has been installed?" or "Identify the rework which occurred". Although evaluation for topic-agnostic queries is essential for open vocabulary methods, this paper will specifically focus on the ability to consider construction-specific queries to consider the construction industry application.

The proposed method leverages existing foundational 2D vision language models to provide scene semantic understanding when lifted to the 3D scene. It inputs a texture 3D mesh and predicts semantic labels per mesh triangle for arbitrary user queries. This allows the semantic segmentation of arbitrary assets of interest, which differs from training only on assets available in standard 3D segmentation datasets. In addition to enabling several construction tracking tasks, open-scene understanding may facilitate and improve the annotation for other various topic-specific datasets.

2. Related Work

2.1. Zero-Shot 2D Scene Understanding

Unlike closed-set scene understanding, which leverages ground truth annotation, zero-shot vocabulary scene understanding has recently been enabled due to robust foundational vision models. Works like CLIP have shown that large-scale pre-training on image-text pairs can result in powerful models capable of zero-shot transfer to various downstream tasks, including scene understanding [18]. While CLIP provides a per-image feature embedding, LSeg is a language-driven model that provides text embeddings per image pixel and produces segmentation predictions that align with text queries [13]. OpenSeg is a similar model which outperforms LSeg due to its scalability [7]. Working with large-scale indoor scenes requires multi-view fusion from thousands of images; these per-image or per-pixel embedding methods are not feasible with the computing available for this project.

Segment Anything (SAM) is a foundational model that can produce open-world segmentations of 2D images with solid performance due to the immense scale of training data used [12]. Grounding DINO, a text queryable object detector that pairs objects with input text queries, also has strong open-world performance [14]. From GroundingSAM, which marries these approaches [20], this project leverages both SAM and GroundingDINO foundational models for semantic segmentation. These methods will be discussed in more detail in Section 3.

2.2. 3D Scene Understanding

ScanNet200 is a standard benchmarking dataset for 3D segmentation and, when proposed, increased the categories for 3D segmentation from 30 to 200 [21]. Current leading models of the ScanNet200, PTv3 advances performance by recognizing model performance can be improved more significantly with scale rather than model design [31]. Supervised methods such as Contrastive Lift leverage neural field representations to lift 2D instance segmentation to 3D, encouraging consistency [3].

Although this approach works well for tasks evaluated within the scope of the training data, a sufficiently diverse 3D dataset does not exist to emulate the robustness of current 2D foundational vision models. As a result, a considerable focus of open-vocabulary 3D scene representation research has been leveraging 2D foundational vision models. CLIP-FO3D [35] uses transfer learning from CLIP by extracting image features in 2D and projecting them to 3D voxels. A 3D CNN scene understanding model is trained from the projected CLIP features, which act as a pseudo-ground truth; however, it is limited by an upper bound of the CLIP to the projection network. OpenScene also uses CLIP and Lseg to extract feature vectors from multiple views of the 3D scene [17]. This method uses these views to train a 3D network to produce 3D labels, then later ensemble the predictions from the 2D views and 3D model based on the cosine similarity of the feature embeddings. One drawback is that these approaches result in heat maps when queried rather than instance segmentations.

OpenMask3D instead approaches [25] 3D scene understanding with instance segmentation. It relies on mask3D, a 3D mask proposal network, the transformer-based mask module trained on ScanNet200 [23]. For each proposed mask, multiple views on specific instances aggregate CLIP features. OpenMask3D shows that the limiting factor in their model is the 3D masking method. SAM3D proposes using SAM to propose 2D masks, which are then projected to 3D and heuristically merged [33]; however, the merging process is the primary cause of failure cases. Instead, our method uses GroundingDINO to ground object masks to the queries of interest, discarding unwanted masks before 3D projection. Segment3D has improved 3D masking on arbitrary classes by leveraging 2D segmentation foundational models [10].

The proposed algorithm will build off of work from OpenMask3D [25]. However, rather than CLIP to semantically embedded information, we explore the efficacy of using GroundedSAM to[20] to create annotated masks of 2D renders of the 3D scene. Another difference between existing and proposed methods is the choice of 3D representation. Semantic segmenting often occurs on 3D point cloud data, or mesh information is voxelized for computational efficiency. To directly produce semantic labels for the 3D input mesh, instead, this method predicts labels per mesh triangle.

2.3. Construction Applications

Computer vision application to the architecture, engineering, and construction (AEC) industries has been an ongoing area of research, especially as a monitoring tool[32]. Golparvar et al. utilized 3D point clouds captured from construction sites, and 4D Building Information Models (BIMs) were used to train a material classification model for construction progress tracking. However, the method relies on manual assignment of correspondence and now outdated methods. Within deep learning, most methods reviewed by Xu et al. follow CNN-based approaches [32]. In addition to the application of construction progress tracking, as discussed by Armeni et al., computer vision has a strong potential in other areas of the AEC industry, such as predicting re-use material patterns and opportunities [1]. This project instead examines the application of a 3D scene understanding method, which leverages 2D foundational vision language models on long-tail distributed concepts important to the architecture, engineering, and construction (AEC) industries.

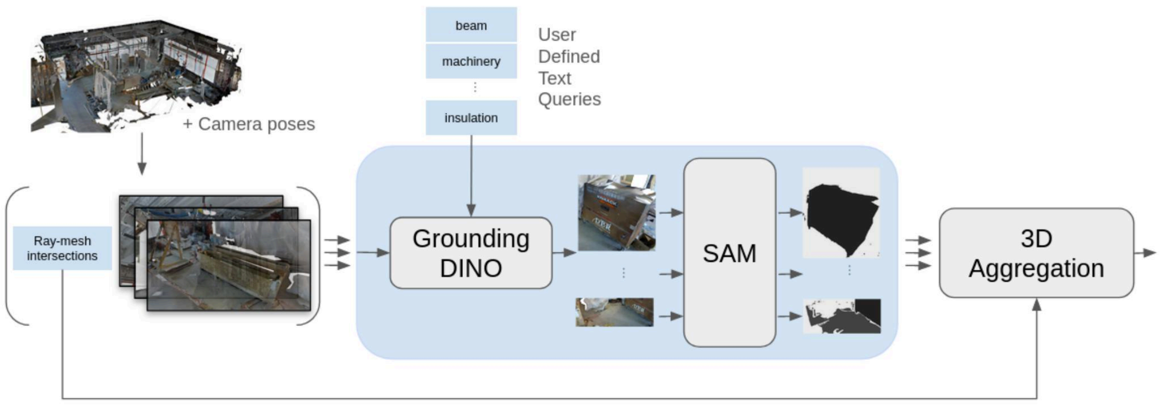


Figure 1. **Technical Implementation**: The method can be divided into three stages: 1. Mutli-view 3D scene, 2. 2D segmentation, and 3. Accumulation in 3D

3. Implementation

The overall process is outlined in Figure 1. It can be split into three steps: 3D data processing, 2D segmentation, and 3D aggregation.

3.1. 3D Data Module

Preprocessing is performed for both datasets. First, mesh information is transformed with respect to a base scan as defined by the metadata provided. Once normalized, Trimesh produces 2D camera renderings of the 3D mesh at various poses [6] and consistent camera intrinsic parameters with a field of view of 90°. These poses are determined from the original capture locations of each dataset.

In addition to capturing a rendering at each camera location, rays are cast. Ray casting uses the same camera's intrinsic and extrinsic properties to determine the source of the pixel in question. Equation 1 represents the relationship between the point in 3D and the image pixel location given the camera matrix K, the rotation and translation matrix controlling extrinsic parameters R and T, respectively. The 3D point can be solved with a scale ambiguity, represented by α. As a result, the equation is represented by a line in 3D space or a 'ray.' This ray is traced until an intersection with the 3D mesh occurs. Then, this intersection is used to associate every pixel in the rendered image with the corresponding mesh triangle from which it is sampled. The implementation for this project utilizes Trimesh and Intel Embree, a high-performance ray tracing library to accelerate calculations [29]. This provides the 3D sources from each image and allows 2D to 3D correspondences later in the pipeline.

$$\alpha \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1}$$

3.2. 2D Segmentation Module

In the second stage, we employ transfer learning. 2D foundation vision models are leveraged to enable open vocabulary queries inspired by the GroundedSAM demo [20]. First, GroundingDINO uses the input query in pair with every rendered image. The queries used on the 3RScan dataset are all classification categories for training and evaluation, but in practice, any open-set vocabulary can be used.

Grounding DINO is a union between DINO, the transformer-based 2D object detector, and grounding pre-training, which pairs text and object detection. The text queries and image features are encoded and augmented to lie in a unified representation space. This allows Grounded-DINO to recognize relevant features from both modalities and assess their similarity. The model outputs are a set of objects detected by defining bounding boxes and similarity scores with each of the text queries. For this implementation, GroundingDINO's object recognition model assigns only the maximum text label for each object recognized if similarity exceeds a certain threshold.

In the next stage, bounding box proposals associated with text labels are input into Segment Anything (SAM) to produce segmentation masks to isolate the object from the background. The SAM architecture leverages a transformer backbone to extract image embedding and a smaller mask decoder network to generate masks, both of which were trained on over 1 billion segmentation tasks. An example of a single rendering after being queried for 'column' is shown in Figure 2.

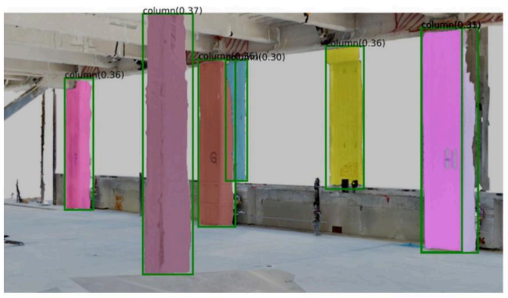


Figure 2. Example Output from GroundedSAM

Alternative methods using CLIP and OpenSeg were also considered. Both are transformer-based methods that produce image feature vectors by using an image encoder trained on a large amount of paired text and image data. The prior creates one feature vector for each image while the latter produces per-pixel features by visual grouping [**?**]. With this implementation, text queries are encoded, and these can be projected and ensembled at the 3D level to enable open-vocabulary queries after 3D aggregation. To prevent feature collapse rather than voting on query similarity or simply averaging features from different views, the cosine similarity of feature vectors for the same mesh triangle from different views could be evaluated. A clustering method could group inliers and outliers based on whether the views agreed on the semantics of the triangle. Inliers could then be averaged. The initial decision to use GroundedSAM rather than CLIP features is due to the preliminary investigation into the performance of foundational models on construction objects.

3.3. 3D Aggregation Module

Since ray-mesh intersections are pre-processed in the first step of the pipeline. Aligning the resulting semantic masks in 2D to their corresponding 3D mesh triangle is simple. Each mesh triangle has a weighted average accumulation of 'votes' for each query input to aggregate the information from multiple views. The weighting is determined by the confidence of the 2D segmentation models, both the confidence in the image-text object detection and the mask. The resulting output is a labeled 3D mesh of the entire scene based on the user-defined text queries.

Other design options for the 3D aggregation model were also considered. Inspired by OpenMask3D [25] rather than voting per mesh triangle, instead vote per mask produced by a separate 3D mask proposal network such as Mask3D [23] or Segment3D [10]. However, a drawback to this method is the mask proposal networks are often trained on datasets like ScanNet or S3DIS [2], which are not representative of the NSS dataset. In addition, challenges such as holes and artifacts that are present in the mesh present challenges for these models.

4. Analysis

4.1. Data

The datasets used for this project are 4D mesh scan data for indoor spaces. The 3D scans have a temporal aspect; some change occurs between scenes. Although the temporal information was not explored for the scope of this course, these datasets were explicitly chosen as opposed to other more common datasets like ScanNetV2 with future project development in mind.

4.1.1 Nothing Stands Still (NSS) Dataset

The project's overall goals align with the data available in the Nothing Stands Still Dataset [24]. This dataset captures six buildings in construction or renovation with 2-6 time stages spreads weeks or months apart. Overall, there are 27 individual scans of large building areas. A 3D textured mesh reconstruction is available for each building stage, providing RGB-D information. However, original images that reconstruct the mesh scene and ground truth annotations are unavailable. Additionally, the dataset does not have ground truth annotation. Although NSS will be included in the qualitative evaluation and will drive the design of the method, a second dataset is included to ensure quantitative evaluation.

For NSS, only the x and y coordinates are provided for camera poses. To get a sufficient view of the scene in the 3D Data model of the pipeline, at each camera point, views vary in the pitch and yaw of the camera orientation.

There are several challenges with this dataset. Rendered images have some mesh artifacts or holes, do not fully resemble natural images, and are sometimes challenging for a human to discern. An example rendering is presented in 9. Another significant challenge of this dataset is the size. To get sufficient coverage of the elements within the scene, around 6500 camera views were defined. However, rendering an NSS data mesh that exceeds 1 million vertices was extremely slow. To combat this, smaller mesh sections were cropped and individually processed. Additionally, the texture is removed from the mesh when determining ray mesh intersections to aid with runtime. Finally, for the use of foundational models, the contents and assets of interest in this dataset may also present a challenge. Due to the specific nature of construction tracking, many concepts or objects may not be featured in commonly used computer vision datasets. For example, common structural elements such as 'column' are not included in ImageNet, while 'beam' is included in a class that also contains 'balance beam' [22]. Without high-quality data, fine-tuning existing models for construction tracking can also be challenging.

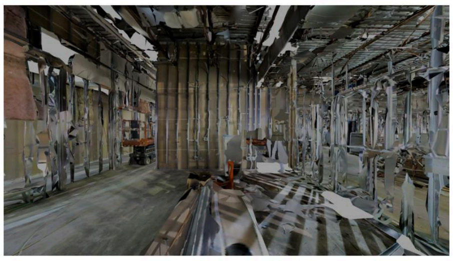


Figure 3. NSS rendering Example

4.1.2 3RScan Dataset

The method will be quantitatively evaluated with the 3RScan dataset [30]. This dataset contains 3D mesh reconstructions from naturally changing indoor environments. It is limited concerning the long-term scope of this project as it captures only small-scale room changes, such as furniture rearrangement, without any structural change. There are 1482 scans of 478 scenes. Each mesh triangle is annotated following several different common semantic categories, including the eigensplit, ScanNet, and NYU40. For this project, the specific 3Rscan semantic segmentation categories, including additions to the NYU40 list, were used with the extension to temporal instance segmentation in mind. Scans are tagged in test (46), validation (47), and training categories (385). For the purpose of the quantitative metrics, input queries are the set of ground truth labels.

3RScan metadata includes the specific camera extrinsic matrices used for the original scene capture. These positions are vertical images that are very close to the boundaries of the scene and, thus, do not give much context to the entire scene. Therefore, camera positions are adapted to 'zoom out' and rotate to the horizontal position.

4.2. Metrics

Using 3RScan, the quantitative evaluation will use the standard 3D instance segmentation metric, average precision (AP). AP is a weighted mean of precision at each threshold where the difference in recall at the current versus the previous threshold determines the weighting. The equation for average precision is found in 2, P_n and R_n indicate nth threshold precision and recall, respectively.

$$AP = \sum_n (R_n - R_{n-1}) P_n \tag{2}$$

To evaluate this method's semantic labels, 3RScan provides a unified ground truth for AP scores. Note that although this method is not an instance segmentation method, the output of this model classifies each triangle index as a label based on the input query. A common segmentation metric Jaccard (IOU) score compares the overlap of predicted and true segmentation maps. Since output does not create a segmentation map with a spatial definition, this metric cannot be used to compare it to baselines. The expected qualitative results from the NSS dataset are limited to visual inspection of resulting 3D scenes and masks across the available time stages.

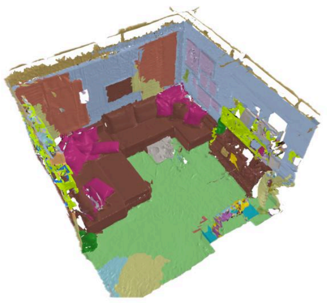


Figure 4. 3RScan Segmented 3D mesh using given classes as the queries

4.3. 3RScan Results

Several method parameters were explored using the 3RScan Dataset. These include text and box thresholds for the 2D segmentation models. In addition, camera parameters were explored, which control the trade-off between the amount of image context and the performance. Images rendered too close to the mesh will not provide meaningful context for the model to determine similarity to the queries; for example, the difference between a wall and the floor may be minimal if the camera only sees a surface. Images too far from the mesh may be too busy and challenging to parse and discern. These were chosen after validating on 3RScan using a grid search. Batches were chosen as 10 scenes due to the high computing time required for processing. Other structural changes were explored in an ablation study discussed in Section 4.3.

The final design has a mean AP score (mAP) of 0.4482. A qualitative segmentation example is presented in Figure 4. A legend defining the color-to-label associations can be found in Figure 12 in the Appendix.

Currently, other semantic segmentation benchmarks are not reported for the 3RScan dataset, and since existing methods do not produce triangle index labels, they cannot be directly compared. Although quantitatively, this is well below the performance of current state-of-the-art methods on benchmarks like ScanNet200, this is expected because the open vocabulary approach increases generalization, which improves flexibility but reduces performance on closed-set tasks.

The resulting confusion matrix from the 3RScan test set in Figure 5 shows common failure cases. The most common semantic segmentation error was towels labeled as curtains,

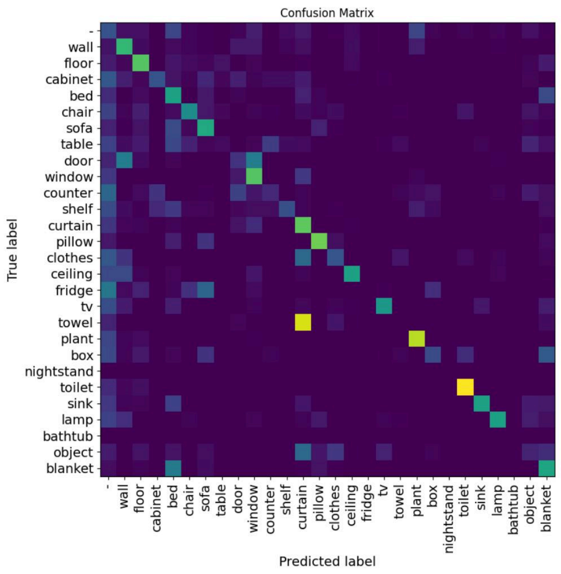


Figure 5. Confusion Matrix Evaluated on 3RScan Test Set

doors as windows or walls, and blankets as beds. These failures are intuitive. The challenges arise where labels are semantically similar, as in the case of curtains and blankets, visually similar, as in windows and doors, or hierarchical, as in blankets on beds. The categories with the highest number of true positives were toilets, plants, curtains, pillows, walls, and floors. These classes tend to be either visually distinct or very common, thus well represented in foundational models.

To evaluate the qualitative results for 3RScan, the previous example scene will be analyzed in more detail. A textured close-up of a corner of the room is presented in Figure 6, along with the ground truth segmentation Figure 7 and our method's segmentation results in Figure 8.

This example demonstrates both successful areas and failure cases seen with this method. First, the model performs very well with pictures, the couch and pillows, and most walls. However, the black cabinet is misclassified as a couch. Visually referencing the textured example scene, holes in the mesh in these areas are present, and the black color makes the cabinet hard to discern. Another notable failure is classifying the wall behind the TV as a curtain. In the ground truth, the wallpaper print accent is labeled as a uniform wall; it is not completely clear visually what the material wall element is. In addition, this method successfully segments a plant located on the black cabinet unit, which is not labeled the ground truth segmentation. These failures highlight limitations in the 3RScan data; meshes have some artifacts, some ambiguity in the features exists, and there are some errors in annotations.

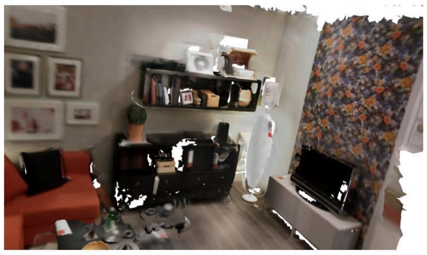


Figure 6. Textured Example Scene

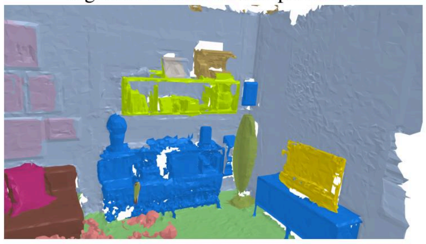


Figure 7. Ground Truth Segmentation

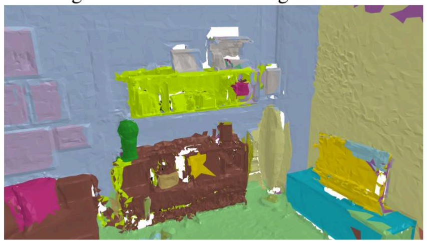


Figure 8. Our Method Segmentation

Table 1 shows the results of the ablation study, which demonstrates the importance of considering each model's predicted confidence. Labels are simply aggregated by majority voting from the multi-view renderings for the baseline. Weighting each vote by Grounding DINO and SAM's confidence are independently evaluated. Finally, the best-performing model weights each vote by the product of each model's confidence.

Approach	**mAP**
Baseline (majority voting)	0.3542
Weighted by SAM confidence	0.3867
Weighted by GroundingDINO	0.4027
Full Method (weighted by both)	**0.4482**

Table 1. Ablation Experiments

4.4. NSS Results

Although the results on the 3RScan dataset are promising and provide a quantitative way to refine the approach, performance on the NSS dataset is more important to the application purpose of this project. The method was tested on subareas of the NSS dataset and qualitatively evaluated.

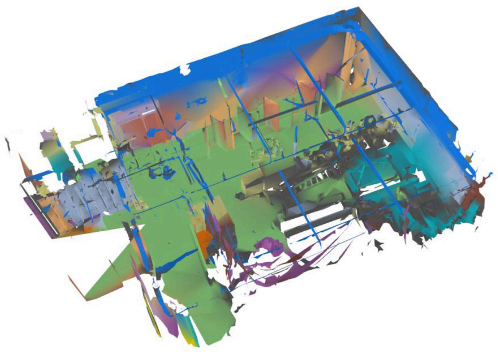


Figure 9. Results Example of a Cropped NSS building

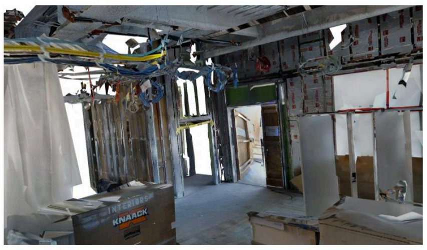


Figure 10. Textured Example Scene

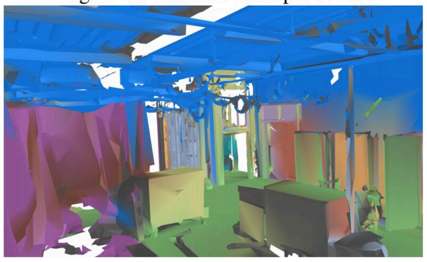


Figure 11. Our Method Segmentation

Figure 9 shows the qualitative results on an example section of NSS Building 6 Stage 2.

Figures 10 and 11 show a close-up example of the textured scene and the resulting labeled scene, respectively. For a detailed legend of the colors associated with the queries used for this evaluation, see Figure 12 in the Appendix. Note that the color blurring between triangle labels is an artifact of the visualization library visible only on the NSS data since triangle density is lower. Each mesh triangle is assigned only one label.

The model accurately labels the scene's columns, floors, plastic sheets, ceilings, carts, and doors. However, the labels do not capture more specific details, such as pipes, insulation panels, and air ducts. These concepts are more challenging than the foundational vision models. Although the model provides some semantic context to the scene, the qualitative results are unsuccessful.

The rendering speed limitation on the NSS data is the major limiting point for the number of views. Increasing the number of views would increase the number and variety of votes, if possible. One possible reason for the failure of more specific assets is the renderings capturing too wide of a scene. The original scan positions chosen were near center points in individual rooms of the buildings. Images are very busy and contain a lot of construction topics. Overall, the model is more likely to be confident the asset is a more general feature, such as a wall, rather than insulation, which is a child feature of the wall. Limiting the context in each image may improve the recognition of more specific assets. However, determining adequate camera position in construction mesh data is particularly challenging since scenes are large and very open; thus, isolating elements is not always possible. One possible change to address this problem would be to use the top 5 object labels for the voting scheme rather than only assigning the top-scoring label.

5. Conclusions

Overall, this project represented an initial investigation of the application of foundational vision models for construction tracking. The performance on the construction dataset, Nothing Stands Still, was initially limited. Although performance metrics on indoor scenes in 3RScan show promise, there is a large gap in the transfer to long-tailed concepts in construction data.

Data is the primary limiting factor for the project. First is the lack of labeled semantically segmented construction datasets. Fine-tuning SAM on relevant 2D datasets could improve the segmentation results. Due to holes and artifacts in the NSS dataset, renderings do not always resemble natural images. Data cleaning methods should be explored, such as 3D mesh super-resolution to attend to clean up artifacts or image in-painting for holes in the mesh renderings. Finally, augmenting the annotations in 3RScan or selecting specifically relevant scenes to increase the context overlap between the NSS and 3RScan results would allow for more specific development.

Compute significantly impacted the model's performance on the NSS dataset due to the processing limitations of renders. Since renders were expensive, the number of views was limited to test a reasonable number of rooms. This, however, heavily impacted performance as mesh triangles may only be in view of a few images, and the voting will not have an adequate number of votes to benefit from the confidence weighting. Major future development is required for the NSS dataset to improve computational processing time, enabling a more thorough accumulation of 2D views. Specifically transitioning to open3D [36], a 3D, more efficient data handler.

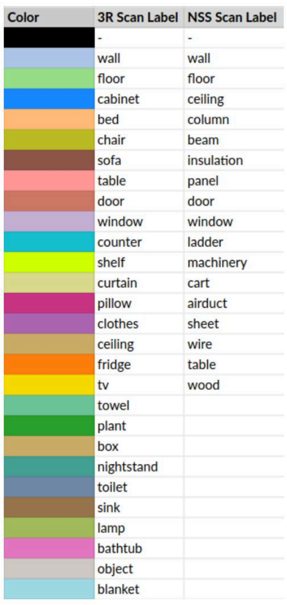
If computational resources had not been a problem, additional investigation of using feature vectors, which require much more memory, should have been explored. Applying foundational models such as OpenSeg to produce feature vectors per mesh triangle would have allowed user queries to interact with a unified 3D model rather than before the aggregation of views.

Future goals of this project include creating instance-based tracking. This will help understand how the construction scene changes over time and enable temporally based queries. A potential area of development would be to explore the adaptation of AST-GRU's spatial and temporal transformers to output instance labels will be adapted to output instance labels rather than the classification of entire sequences [34] to explore consistent temporal segmentation, semantic embedding, and instance labeling enforcement. The goal will be to jointly optimize by incorporating both the 3D geometry and temporal information.

Appendix

Supplemental Figures

Figure 12 shows the legend for the query asset to visualization pairs for each dataset.



Color	3R Scan Label	NSS Scan Label
⬛	-	-
🟦	wall	wall
🟩	floor	floor
🟦	cabinet	ceiling
🟧	bed	column
🟨	chair	beam
🟫	sofa	insulation
🟥	table	panel
🟫	door	door
🟪	window	window
🟦	counter	ladder
🟨	shelf	machinery
🟨	curtain	cart
🟥	pillow	airduct
🟪	clothes	sheet
🟫	ceiling	wire
🟧	fridge	table
🟨	tv	wood
🟩	towel	
🟩	plant	
🟫	box	
🟩	nightstand	
🟦	toilet	
🟫	sink	
🟩	lamp	
🟪	bathtub	
⬜	object	
🟦	blanket	

Figure 12. Color-Label Legend

Contributions & Acknowledgements

This project is jointly used for the CS231N project. CS231A focuses on working with 3D data, mainly ray-tracing, multi-view imaging, and exploring different methods for aggregating 2D semantic segmentation back to the original 3D model.

E.S. managed project design and implementation decisions, designed and wrote all code for data processing, finetuning training, and evaluation, decided on a smaller scope for the course, and wrote the paper. Prof I.A. provided the conceptual guidance, suggested datasets, and advised. The Gradient Spaces lab provided GPU resources. Cesar Portocarrero Rodriguez is a collaborator in the context of the larger scope of this project. However, their contributions did not overlap with this class's project.

Code

The Grounding-SAM project [20] which combines Grounding Dino [14] and SAM [12] The GroundingSAM repository is used for installation, and the sub-packages and some interfacing API are used. In addition, following Python libraries are used in the code: Trimesh, [6], PyTorch [15], NumPy [8], Pillow [26], OpenCV [4], MatPlotLib [11], SciPy [28], scikit-learn [16], HDF5 for Python [5], pyglet [9], and plyfile [19]. As well as Python 3.10 and supporting libraries io, os, sys, argparse, collections, zipfile, Pathlib, json, and itertools [27].

References

[1] I. Armeni, D. Raghu, and C. De Wolf. Artificial Intelligence for Predicting Reuse Patterns. In C. De Wolf, S. Çetin, and N. M. P. Bocken, editors, *A Circular Built Environment in the Digital Age*, pages 57–78. Springer International Publishing, Cham, 2024. Series Title: Circular Economy and Sustainability.

[2] I. Armeni, S. Sax, A. R. Zamir, and S. Savarese. Joint 2D-3D-Semantic Data for Indoor Scene Understanding, Apr. 2017. arXiv:1702.01105 [cs].

[3] Y. Bhalgat, I. Laina, J. F. Henriques, A. Zisserman, and A. Vedaldi. Contrastive Lift: 3D Object Instance Segmentation by Slow-Fast Contrastive Fusion, Dec. 2023. arXiv:2306.04633 [cs].

[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[5] A. Collette. *Python and HDF5*. O'Reilly, 2013.

[6] Dawson-Haggerty et al. trimesh.

[7] G. Ghiasi, X. Gu, Y. Cui, and T.-Y. Lin. Scaling Open-Vocabulary Image Segmentation with Image-Level Labels, July 2022. arXiv:2112.12143 [cs].

[8] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020.

[9] Holkner et al. pyglet.

[10] R. Huang, S. Peng, A. Takmaz, F. Tombari, M. Pollefeys, S. Song, G. Huang, and F. Engelmann. Segment3D: Learn-

ing Fine-Grained Class-Agnostic 3D Segmentation without Manual Labels, Dec. 2023. arXiv:2312.17232 [cs].

[11] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[12] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment Anything, Apr. 2023. arXiv:2304.02643 [cs].

[13] B. Li, V. Koltun, K. Q. Weinberger, R. Ranftl, and S. Belongie. LANGUAGE-DRIVEN SEMANTIC SEGMENTATION. 2022.

[14] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang. Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection, Mar. 2023. arXiv:2303.05499 [cs].

[15] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] S. Peng, K. Genova, C. M. Jiang, A. Tagliasacchi, M. Pollefeys, and T. Funkhouser. OpenScene: 3D Scene Understanding with Open Vocabularies, Apr. 2023. arXiv:2211.15654 [cs].

[18] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning Transferable Visual Models From Natural Language Supervision, Feb. 2021. arXiv:2103.00020 [cs].

[19] Ranjan et al. plyfile.

[20] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, Z. Zeng, H. Zhang, F. Li, J. Yang, H. Li, Q. Jiang, and L. Zhang. Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks, Jan. 2024. arXiv:2401.14159 [cs].

[21] D. Rozenberszki, O. Litany, and A. Dai. Language-Grounded Indoor 3D Semantic Segmentation in the Wild. In S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, editors, *Computer Vision – ECCV 2022*, pages 125–141, Cham, 2022. Springer Nature Switzerland.

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, Jan. 2015. arXiv:1409.0575 [cs].

[23] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe. Mask3D: Mask Transformer for 3D Semantic Instance Segmentation, Apr. 2023. arXiv:2210.03105 [cs].

[24] T. Sun, Y. Hao, S. Huang, S. Savarese, K. Schindler, M. Pollefeys, and I. Armeni. Nothing Stands Still: A Spatiotemporal Benchmark on 3D Point Cloud Registration Under Large Geometric and Temporal Change, Nov. 2023. arXiv:2311.09346 [cs].

[25] A. Takmaz, E. Fedele, R. W. Sumner, M. Pollefeys, F. Tombari, and F. Engelmann. OpenMask3D: Open-Vocabulary 3D Instance Segmentation, Oct. 2023. arXiv:2306.13631 [cs].

[26] P. Umesh. Image processing in python. *CSI Communications*, 23, 2012.

[27] G. Van Rossum. *The Python Library Reference, release 3.10.2*. Python Software Foundation, 2024.

[28] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[29] I. Wald, S. Woop, C. Benthin, G. S. Johnson, and M. Ernst. Embree: a kernel framework for efficient cpu ray tracing. *ACM Transactions on Graphics (TOG)*, 33(4):1–8, 2014.

[30] J. Wald, A. Avetisyan, N. Navab, F. Tombari, and M. Niessner. RIO: 3D Object Instance Re-Localization in Changing Indoor Environments. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7657–7666, Seoul, Korea (South), Oct. 2019. IEEE.

[31] X. Wu, L. Jiang, P.-S. Wang, Z. Liu, X. Liu, Y. Qiao, W. Ouyang, T. He, and H. Zhao. Point Transformer V3: Simpler, Faster, Stronger, Mar. 2024. arXiv:2312.10035 [cs].

[32] S. Xu, J. Wang, W. Shou, T. Ngo, A.-M. Sadick, and X. Wang. Computer Vision Techniques in Construction: A Critical Review. *Archives of Computational Methods in Engineering*, 28(5):3383–3397, Aug. 2021.

[33] Y. Yang, X. Wu, T. He, H. Zhao, and X. Liu. SAM3D: Segment Anything in 3D Scenes, June 2023. arXiv:2306.03908 [cs].

[34] J. Yin, J. Shen, X. Gao, D. Crandall, and R. Yang. Graph Neural Network and Spatiotemporal Transformer Attention for 3D Video Object Detection from Point Clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):9822–9835, Aug. 2023. arXiv:2207.12659 [cs].

[35] J. Zhang, R. Dong, and K. Ma. CLIP-FO3D: Learning Free Open-world 3D Scene Representations from 2D Dense CLIP. In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 2040–2051, Paris, France, Oct. 2023. IEEE.

[36] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.