SATR: Zero-Shot Semantic Segmentation of 3D Shapes
Ahmed Abdelrehem, Ivan Skorokhodov, Maks Ovsjanikov, Peter Wonka

To cite this version:
Ahmed Abdelrehem, Ivan Skorokhodov, Maks Ovsjanikov, Peter Wonka. SATR: Zero-Shot Semantic Segmentation of 3D Shapes. ICCV 2023 - International Conference on Computer Vision, Oct 2023, Paris, France. hal-04352372

HAL Id: hal-04352372
https://hal.science/hal-04352372
Submitted on 19 Dec 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
SATR: Zero-Shot Semantic Segmentation of 3D Shapes

Ahmed Abdelreheem\textsuperscript{1} Ivan Skorokhodov\textsuperscript{1} Maks Ovsjanikov\textsuperscript{2} Peter Wonka\textsuperscript{1}

\textsuperscript{1}KAUST \textsuperscript{2}LIX, Ecole Polytechnique
{asamirh.95,iskorokhodov,pwonka}@gmail.com, {maks}@lix.polytechnique.fr

The \{arm, head, leg, torso\} of a man.

The \{leg, top, support\} of a table.

The \{case, handle\} of a bag.

Figure 1: SATR performs zero-shot 3D shape segmentation via text descriptions by using a zero-shot 2D object detector. It infers 3D segmentation from multi-view 2D bounding box predictions by exploiting the topological properties of the underlying surface. SATR is able to segment the mesh from both single and multiple queries and provides accurate predictions even for fine-grained categories.

Abstract

We explore the task of zero-shot semantic segmentation of 3D shapes by using large-scale off-the-shelf 2D image recognition models. Surprisingly, we find that modern zero-shot 2D object detectors are better suited for this task than contemporary text/image similarity predictors or even zero-shot 2D segmentation networks. Our key finding is that it is possible to extract accurate 3D segmentation maps from multi-view bounding box predictions by using the topological properties of the underlying surface. For this, we develop the Segmentation Assignment with Topological Reweighting (SATR) algorithm and evaluate it on ShapeNetPart and our proposed FAUST benchmarks. SATR achieves state-of-the-art performance and outperforms a baseline algorithm by 1.3% and 4% average mIoU on the FAUST coarse and fine-grained benchmarks, respectively, and by 5.2% average mIoU on the ShapeNetPart benchmark. Our source code and data will be publicly released. Project webpage: https://samir55.github.io/SATR/.

1. Introduction

Recent developments in vision-language learning gave rise to many 2D image recognition models with extreme zero-shot generalization capabilities (e.g., [76, 70, 55, 54, 102]). The key driving force of their high zero-shot performance was their scale [11]: both in terms of the sheer amount of data [80, 49] and parameters [5, 97] and in terms of developing the architectures with better scalability [85, 31, 59]. However, extending this success to the 3D domain is hindered by the limited amount of available 3D data [2, 3], and also the higher computational cost of the corresponding architectural components [19]. For example,
the largest openly available 2D segmentation dataset [52] contains two orders of magnitude more instance annotations than the largest 3D segmentation one [79]. This forces us to explore other ways of performing zero-shot recognition in 3D, and in our work, we explore the usage of off-the-shelf 2D models for zero-shot 3D shape segmentation.

Zero-shot 3D shape segmentation is a recently emerged research area [22] with applications in text-based editing [75, 4], stylization [65], and interactive visualization. Given a 3D mesh, the user provides one or several text descriptions of their regions of interest, and the task is to categorize each face on the mesh into one of the given descriptions (or “background” class if it does not suit any). To the best of our knowledge, the only previous work which explores this task is 3D Highlighter (3DH) [22]. The method uses an optimization-based search algorithm guided by CLIP [76] to select the necessary faces for a given text prompt. While showing strong zero-shot performance, 3DH has two drawbacks: 1) it struggles in fine-grained segmentation, and 2) it is very sensitive to initialization (see Figure 2). Moreover, due to its per-query optimization, the segmentation process is slow, taking up to ≈5-10 minutes on a recent GPU for a single semantic part.

In our work, we explore modern zero-shot 2D object detectors [55] and segmentors [54, 61] for 3D shape segmentation. Intuitively, 2D segmentation networks are a natural choice for this task: one can predict the segmentations for different views, and then directly propagate the predicted pixel classes onto the corresponding mesh faces. Moreover and surprisingly, we found that it is possible to achieve substantially higher performance using a zero-shot 2D object detector [55]. To do this, we develop Segmentation Assignment with Topological Reweighting (SATR): a method that estimates a 3D segmentation map from multi-view 2D bounding box predictions by using the topological properties of the underlying 3D surface.

For a given mesh and a text prompt, our method first uses GLIP [55] to estimate the bounding boxes from different camera views. However, relying exclusively on the bounding boxes provides only coarse guidance for 3D segmentation and is prone to “leaking” unrelated mesh faces into the target segment. This motivates us to develop two techniques to infer and refine the proper segmentation. The first one, gaussian geodesic reweighting, performs robust reweighting of the faces based on their geodesic distances to the potential segment center. The second one, visibility smoothing, uses a graph kernel, which adjusts the inferred weights based on the visibility of its neighbors. When combined together, these techniques allow for achieving state-of-the-art results on zero-shot 3D shape segmentation, especially for fine-grained queries.

To the best of our knowledge, there are currently no quantitative benchmarks proposed for 3D mesh segmenta-

![Figure 2: 3DHighlighter [23] is very sensitive to initialization. We observe that 3DHighlighter produces quite different results when using different seeds for the same prompt on different 3D shapes.](image)

2. Related Work

Zero-shot 2D detection and segmentation. Zero-shot 2D object detection is a fairly established research area [77, 7]. Early works relied on pre-trained word embeddings [67, 74] to generalize to unseen categories (e.g., [77, 7, 25, 94]). With the development of powerful text encoders [26] and vision-language multi-modal networks [76], the focus shifted towards coupling their representation spaces with the representation spaces of object detectors (e.g., [38, 82]). The latest methods combine joint text, caption, and/or self-supervision to achieve ex-
treme generalization capabilities to unseen categories (e.g., [55, 99, 87, 102, 88, 78, 64, 35, 30]).

Zero-shot 2D segmentation is a more challenging problem [102] but had a similar evolutionary path to zero-shot detection. Early works (e.g., [12, 39, 6, 100, 18, 89, 71, 98]) focused on adapting open-vocabulary word embeddings to align the semantic representations with the image recognition model. Later ones (e.g., [27, 63, 54, 61, 29, 36, 62, 93, 92]) developed the mechanisms to adapt rich semantic spaces of large-scale multi-modal neural networks (e.g., [76, 70, 43]) for segmentation tasks. Some recent works show that one can achieve competitive segmentation performance using only text or self-supervision [90, 14, 13, 73, 15]. The current state-of-the-art zero-shot 2D semantic segmentation models are based on rich multi-modal supervision (e.g., [91, 99, 102]). A limiting factor of 2D segmentation is the absence of large-scale segmentation datasets due to the high annotation cost [96, 56, 51, 49], which hinders generalization. However, we observed that these models struggle to produce fine-grained segmentation even in 2D, especially for fine-grained categories (see Figure 3) and develop an algorithm that constructs accurate segmentation predictions from estimated bounding boxes.

**Zero-shot 3D segmentation.** Zero-shot 3D segmentation is a new research topic, and the main focus of the community was targeted towards point cloud segmentation [66, 17, 28, 57, 48]. With the rise of Neural Radiance Fields (NeRFs) [68, 60], there were several methods developed to model semantic fields (e.g., [101, 86, 32, 50, 33, 84, 83, 40, 1]) by reconstructing ground-truth or estimated semantic annotations from multi-view renderings. By distilling zero-shot 2D segmentation networks (e.g., [54, 14]) into a NeRF, these methods can perform 3D segmentation of a volumetric scene (e.g., [47, 81, 37]) and can generalize to an open-set vocabulary. By fusing representations from additional feature extractors of non-textual modalities, ConceptFusion [42] can also support zero-shot visual and audio queries. In our case, we are interested in shape segmentation and show that employing a 2D object detector yields state-of-the-art results.

PartSLIP [57] is concurrent work that performs zero/few-shot part segmentation of point clouds and, similarly to us, also relies on the GLIP [55] model. It clusters a point cloud, predicts bounding boxes via GLIP for multiple views, and assigns a score to each cluster depending on the number of its visible points inside each bounding box. Their method is designed for point clouds while ours is designed for meshes.

To the best of our knowledge, the only prior work which explores zero-shot 3D mesh segmentation is 3D Highlighter (3DH) [22]. It solves the task by optimizing a probability field of a point to match a given text prompt encoded with CLIP [76]. While showing strong generalization capabilities, their approach struggles to provide fine-grained predictions and is very sensitive to initialization (see Figure 2).

### 3. Method

As input, we assume a 3D shape, represented as a polygon mesh \( \mathcal{F} = \{ f_n \}_{n=1}^{N} \) of \( d \)-sided polygon faces \( f_n \in \mathbb{R}^{d \times 3} \), and \( K \) semantic text descriptions \( \{ t_k \}_{k=1}^{K} \), provided by the user. The prompts \( t_k \) are single nouns or compound noun phrases consisting of multiple words. Then, the task of 3D shape segmentation is to extract \( K \) non-intersecting subsets \( \{ \mathcal{F}_k \mid \mathcal{F}_k \subseteq \mathcal{F} \}_{k=1}^{K} \) in such a way that the \( k \)-th subset \( \mathcal{F}_k \) is a part of the mesh surface which semantically corresponds to the \( k \)-th text prompt \( t_k \).

In our work, we explore modern powerful 2D vision-language models to solve this task. The simplest way to incorporate them would be employing a zero-shot 2D segmentation network, like CLIPSeg [61] or LSeg [54], which can directly color the mesh given its rendered views. But as we show in experiments, this leads to suboptimal results since modern zero-shot 2D segmentors struggle to produce fine-grained annotations and high-quality segments, while 2D detectors could be adapted for shape segmentation with surprisingly high precision.

In our work, we consider untextured meshes, but it is straightforward to apply the method to textured ones.

In this way, our method relies on a zero-shot 2D object detector \( D(x, t) = \{ (b_\ell, p_\ell) \}_{\ell=1}^{L} \), which takes as input an RGB image \( x \in \mathbb{R}^{H \times W \times 3} \) of size \( H \times W \) and a text prompt \( t \), and outputs \( L \geq 0 \) bounding boxes \( b_\ell = (x_\ell, y_\ell, h_\ell, w_\ell) \in \mathbb{R}^4 \) with their respective probability...
In our notation, each bounding box $b_m$ is defined by its left lower corner coordinate $(x_l, y_l)$, width $w_l$, and height $h_l$. We choose GLIP [55] as the detector model due to its state-of-the-art generalization capabilities.

In this section, we first describe a topology-agnostic baseline framework (denoted as SATR-F) that can leverage a 2D object detector to segment a meshed 3D surface. Then, we explain why it is insufficient to infer accurate segmentation predictions (which is also confirmed by the experiments in Tables 1 to 3) due to the coarse selection nature of the bounding boxes. After that, we develop our Segmentation Assignment with Topological Reweighting (denoted as SATR (F+R) or SATR in short) algorithm, which alleviates this shortcoming by using the topological properties of the underlying surface.

### 3.1. Topology-Agnostic Mesh Segmentation

Our topology-agnostic baseline method works the following way. We render the mesh $F$ from $M$ random views (we use $M = 10$ in all the experiments, if not stated otherwise) to obtain RGB images $x_m$. To create the $M$ views, we generate random camera positions where the elevation and azimuth angles are sampled using the normal distribution ($\mu = 0.7, \sigma = 4$). We use this view generation to be directly comparable to 3DHighlighter [22]. After that, for each view $x_m$ and for each text prompt $t_k$, we use the detector model to predict the bounding boxes with their respective confidence scores:

$$\left\{ \left( b^m_{\ell, k}, p^m_{\ell, k} \right) \right\}_{\ell = 1}^{L_{m, k}} = \text{D}(b_m, t_k). \quad (1)$$

Then, we use them to construct the initial face weights matrix $W'_m \in [0, 1]^{N \times K}$ for the $m$-th view. Let $F_m \subset F$ denote a subset of visible (in the $m$-th view) faces with at least one vertex inside bounding box $b^m_{\ell, k}$, whose projection falls inside $b^m_{\ell, k}$. Then

$$W_m[n, k] = \sum_{\ell=1}^{L_{m, k}} W^\ell_m[n, k, \ell] \quad (2)$$

$$W^\ell_m[n, k, l] = \begin{cases} p^m_{\ell, k} & \text{if } f_n \in F^m_{\ell, k} \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

In this way, the score of each face $f_n$ for the $m$-th view is simply set to the confidence $p^m_{\ell, k}$ of the corresponding bounding box(es) it fell into.

The face visibility is determined via the classical Z-culling algorithm [34]. In this way, if there are no bounding boxes predicted for $x_m$ for prompt $t_k$, then $W_m[n, k]$ equals the zero matrix. The latter happens when the region of interest is not visible from a given view or when the detector $D$ makes a prediction error.

Next, we take into account the area sizes of each face projection. If a face occupies a small area inside the bounding box, then it contributed little to the bounding box prediction. The area $s^m_{n, \ell}$ of the face $f_n$ in the view $x_m$ and bounding box $\ell$ is computed as the number of pixels which it occupies. We use the computed areas to re-weight the initial weights matrix $W'_m$ and obtain the area-adjusted weights matrix $W_m \in \mathbb{R}^{N \times K}$ for the $m$-th view:

$$W_m[n, k] = \sum_{\ell=1}^{L_{m, k}} W^\ell_m[n, k, \ell] \times s^m_{n, \ell} \quad (4)$$

To compute the final weights matrix, we first aggregate the predictions from each view $x_m$ by summing the scores.
of the un-normalized matrix \( \tilde{W} \in \mathbb{R}^{N \times K} \):

\[
\tilde{W}[n, k] = \sum_m W_m[n, k],
\]

and then normalize it by dividing each column by its maximum value to obtain our final weights matrix \( W \in \mathbb{R}^{N \times K} \):

\[
W[n, k] = \tilde{W}[n, k] / \max_k \tilde{W}[n, k].
\]

The above procedure constitutes our baseline method of adapting bounding box predictions for 3D shape segmentation. As illustrated in Figure 8, its disadvantage is “segmentation leaking”: some semantically unrelated parts of the surface get incorrectly attributed to a given text prompt \( t \), because they often fall into predicted bounding boxes from multiple views. To alleviate this issue, we develop a more careful score assignment algorithm that uses the topological properties of the surface, thus allowing us to obtain accurate 3D shape segmentation predictions from a 2D detector.

### 3.2. Gaussian Geodesic Reweighting

Bounding box estimates give only coarse estimates about the semantic region being queried, and we found that using the surface topology allows localizing it with substantially better precision. For this, we construct a method that utilizes geodesic distances between mesh faces, i.e., path lengths from one face to another along the surface, instead of their direct distances in Euclidian space.

Consider the following example of segmenting a human palm. When the hand is in a resting state, the palm lies close to the waistline in Euclidean space (as illustrated in Figure 5). Then, a simple topology-agnostic method would lead to the waistline leaking into the segmentation prediction. At the same time, one can observe that the predicted bounding boxes are always centered around the palm, and the waistline is far away from it in terms of the geodesic distance. In this way, discarding such outlying polygons yields a precise segmentation of the required region. And this is the main intuition of our developed algorithm.

As a first step, for each predicted bounding box \( b_{\ell}^{m,k} \), we estimate its central face, which we call the capital face \( g_{\ell}^{m,k} \in F \). It is computed by taking the (area-weighted) average of all the vertices from all the faces inside \( F_{\ell}^{m,k} \), projecting this average point onto \( F \) and taking the face on which the projection lies. After that, we compute a vector of geodesic distances \( d_{\ell}^{m,k} \) from the capital face \( g_{\ell}^{m,k} \) to every other face \( f \in F_{\ell}^{m,k} \):

\[
d_{\ell}^{m,k}[n] = \begin{cases} 
gdist(g_{\ell}^{m,k}, f_n) & \text{if } f_n \in F_{\ell}^{m,k} \\ 0 & \text{otherwise} \end{cases}
\]

where \( gdist(\cdot, \cdot) \) denotes the geodesic length between two faces computed on the mesh \( F \) using the Heat method [21].

It feels natural to use those geodesic distances directly to reweight the original weight matrix \( W \). However, this leads to sub-optimal results for two reasons: 1) there are natural errors in selecting the capital face, which would bias reweighting towards incorrect regions; and 2) as we show in Figure 5, it is difficult to tune the decay rate for such reweighting. Instead, we propose Gaussian reweighting and demonstrate that it is more robust and accurate in practice. It works the following way.

First, we fit a Gaussian distribution over the distances and compute the corresponding probability density values for each face given its geodesic distance from the capital face:

\[
r_{\ell}^{m,k} \triangleq \{N(d; \mu_{\ell}^{m,k}, \sigma_{\ell}^{m,k})^2 | d \in d_{\ell}^{m,k} \},
\]

where \( \mu_{\ell}^{m,k} \), \( \sigma_{\ell}^{m,k} \) denote the mean and standard deviation of the distances \( d_{\ell}^{m,k} \). This formulation nudges the weights away from the capital and works like adaptive regularization. If there are inaccuracies in the capital face selection, then it will have less influence on the segmentation quality.

We aggregate the weights from multiple views into a single vector of scores \( r_{\ell}^{m,k} \in \mathbb{R}^N \) and reweigh the original weight matrix \( W_m \) for the \( m \)-th view to obtain the weight matrix \( W_m^g \) with Gaussian geodesic reweighting:

\[
W_m^g[n, k] = \sum_{\ell=1}^{L_m} W_m[n, k, \ell] \times s_{m,\ell}^{n} \times r_{\ell}^{m,k}.
\]

After that, we compute the final weight matrix \( W^g \in \mathbb{R}^{N \times K} \) for each face \( f_n \in F \) in a similar manner to \( W \) by taking the summing over \( W_m^g \) from different views. We find that not normalizing the weight matrix whenever the Gaussian Geodesic Reweighting is used yields better performance than applying normalization.

This procedure takes into account the topological structure of the mesh surface, which allows for obtaining more accurate segmentation performance, as can be seen from Tables 1 and 2 and Figure 7. However, it has one drawback: it pushes the weights away from the capital face, which might decrease the performance. To compensate for this, we develop the technique of visibility smoothing, which we describe next.

### 3.3. Visibility Smoothing

The score matrix \( W^g \) with Gaussian geodesic reweighting might allocate too little weight on the central faces around the capital face \( g_{\ell}^{m,k} \), which happens for regions with a large average distance between the faces. To compensate for this, we propose visibility smoothing. It works independently in the following way.

For each visible face \( f \in F_{\ell}^{m,k} \), we compute its local neighborhood, where the neighbors are determined via
Figure 5: Gaussian Geodesic Weights can help reduce the scores for triangles inside a detected bounding box that do not belong to the segment of the corresponding text prompt.

mesh connectivity: a face $g$ is a neighbor of face $f$ if they share at least one vertex. For this, we use a $q$-rank neighborhood $N_q(f)$ (we use $q = 5$ in all the experiments unless stated otherwise), which is constructed the following way. For face $f_n \in \mathcal{F}$, we take the face $g \in \mathcal{F}$ if there exists a path on a graph between $f$ and $g$ of at most $q$ other vertices.

After that, we compute the neighborhood visibility scores vector $v_{\ell}^{m,k}[n] \in [0,1]^N$ for each face $f_n \in \mathcal{F}$ by computing the ratio between visible faces in its neighborhood and the overall neighborhood size:

$$v_{\ell}^{m,k}[n] = |N_q(f_n) \cap \mathcal{F}^{m,k}_{\ell}| / |N_q(f_n)|.$$  

(10)

Similarly to geodesic weighting, we aggregate the neighborhood visibility scores $v_{\ell}^{m,k}[n]$ across the bounding boxes into $v^{m,k}[n] \in [0,1]^N$ via element-wise vector summation:

$$v^{m,k}[n] = \sum_{\ell=1}^{L} v_{\ell}^{m,k}[n].$$  

(11)

This gives us our final per-view score matrix $\mathcal{W}^*_{m}[n] \in \mathbb{R}^{N \times K}$:

$$\mathcal{W}^*_{m}[n,k] = \sum_{\ell=1}^{L_{m,k}} \mathcal{W}_{m}[n,k,\ell] \times s_{m,\ell} \times r_{\ell}^{m,k} \times v^{m,k}[n].$$  

(12)

Again, we aggregate our multi-view scores $\mathcal{W}^*_{m}[n]$ into the final weights matrix $\mathcal{W}^*[n]$ by taking the maximum across the views.

We call the above technique visibility smoothing since it smoothes face weights according to their neighborhood visibility and can be seen as a simple convolutional kernel over the graph. It allows for repairing the weights in the central part of the visible surface region without damaging the rest of the faces. This consequently leads to a noticeable improvement in the scores, as we report in Tables 1 and 5.

The pseudo-code of our algorithm is provided in Algorithm 1 in Appx B, together with additional implementation details. Also, note that the source code of our method will be publicly released.

4. Experiments

Datasets and splits. We evaluate the zero-shot performance on our proposed FAUST [9] benchmark and on the ShapeNetPart [95] dataset. For the FAUST dataset, we use all of the 100 scans available. We manually collected the coarse and fine-grained annotations in the following way. First, we manually annotate the registered mesh of one human scan as shown in Figure 6 using vertex paint in Blender [8]. Since we have access to the FAUST mesh correspondences, we are able to transfer our annotations to all other scans. We then re-mesh each human scan independently to contain around 20K triangular faces. To generate annotations for these re-meshed scans, we assign the label of each vertex to the label of the nearest vertex before remeshing. For the ShapeNetPart dataset [95], it contains 16 object categories and 50 annotated parts for shapes from the ShapeNet [16] dataset. We use all of the 2460 labeled shapes of the ShapeNetPart dataset provided by [44], where the point labels are transferred to mesh polygon labels via a...
<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mIoU</th>
<th>arm</th>
<th>belly</th>
<th>button</th>
<th>chin</th>
<th>ear</th>
<th>elbow</th>
<th>eye</th>
<th>foot</th>
<th>fore-head</th>
<th>hand</th>
<th>head</th>
<th>knee</th>
<th>leg</th>
<th>mouth</th>
<th>neck</th>
<th>nose</th>
<th>shoulder</th>
<th>torso</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DH [23]</td>
<td>CLIP [76]</td>
<td>3.89</td>
<td>18.39</td>
<td>1.99</td>
<td>0.46</td>
<td>0.72</td>
<td>0.08</td>
<td>0.0</td>
<td>20.81</td>
<td>0.70</td>
<td>0.02</td>
<td>3.49</td>
<td>6.17</td>
<td>3.91</td>
<td>0.05</td>
<td>1.94</td>
<td>0.07</td>
<td>0.04</td>
<td>7.28</td>
<td></td>
</tr>
<tr>
<td>SATR-F</td>
<td>CLIPSeg</td>
<td>10.88</td>
<td>11.51</td>
<td>0.10</td>
<td>0.30</td>
<td>0.0</td>
<td>0.03</td>
<td>0.0</td>
<td>0.0</td>
<td>25.80</td>
<td>39.99</td>
<td>0.07</td>
<td>50.52</td>
<td>0.0</td>
<td>0.05</td>
<td>0.0</td>
<td>5.11</td>
<td>48.24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SATR</td>
<td>GLIP</td>
<td>46.01</td>
<td>50.51</td>
<td>29.41</td>
<td>27.74</td>
<td>47.45</td>
<td>26.80</td>
<td>18.90</td>
<td>81.99</td>
<td>38.11</td>
<td>81.45</td>
<td>51.11</td>
<td>33.34</td>
<td>65.22</td>
<td>27.29</td>
<td>41.95</td>
<td>57.60</td>
<td>38.94</td>
<td>64.35</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Performance of SATR on the fine-grained semantic segmentation on FAUST dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mIoU</th>
<th>arm</th>
<th>head</th>
<th>leg</th>
<th>torso</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSeg [54]</td>
<td>6.50</td>
<td>26.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>SATR-F</td>
<td>CLIPSeg [61]</td>
<td>60.34</td>
<td>46.55</td>
<td>58.01</td>
<td>76.22</td>
<td>59.80</td>
</tr>
<tr>
<td>GLIP [55]</td>
<td>81.16</td>
<td>81.56</td>
<td>82.46</td>
<td>82.01</td>
<td>88.17</td>
<td>86.54</td>
</tr>
<tr>
<td>SATR</td>
<td>GLIP [55]</td>
<td>82.46</td>
<td>85.92</td>
<td>90.56</td>
<td>85.75</td>
<td>67.60</td>
</tr>
</tbody>
</table>

Table 2: Performance of SATR on the coarse-grained semantic segmentation on FAUST dataset.

The meshes in ShapeNetPart have triangular faces that are very large, covering a large portion of the mesh surface. For this reason, during model inference, we provide re-meshed ShapeNetPart shapes as input, where each mesh contains at most 30K triangular faces. We use the Isotropic Explicit Remeshing algorithm [41] provided by MeshLab [20] to do the re-meshing. During the evaluation, we transferred the predicted face labels back to the original ShapeNetPart shapes. Figure 1 includes examples from the TextANIMAR [53], Objaverse [24], and TOSCA [10] datasets.

Metrics. We use the semantic segmentation mIoU as described in [69]. We first calculate the mIoU for each part category across all the test shapes and then compute for each object category the average of the part mIoU.

4.2. Implementation Details

We use a single Nvidia V100 GPU for each experiment. We use the Nvidia Kaolin library [34] written in PyTorch [72] for rendering in all of our experiments. To ensure fairness, we use the same views in all of our GLIP-based model experiments. As a pre-processing step, we center the input mesh around the origin and normalize it inside a unit sphere. For rendering, we use a resolution of 1024 × 1024 and a black background color.

4.3. Zero-Shot Semantic Segmentation

4.3.1 FAUST Benchmark

We compare our method with 3DHighlighter [22] and CLIPSeg [61]. To obtain semantic segmentation results from 3DHighlighter, we run the optimization separately to get a highlighted mesh for each of the input semantic regions. If a face were highlighted for different semantic regions, its predicted label would be the semantic class with the highest CLIP similarity score. To obtain semantic segmentation results from CLIPSeg, we generate a segmentation mask for each rendered view for each semantic label, and we aggregate the segmentation scores for each face and assign the most predicted label for each face.

In Table 2, we report the overall average mIoU and the mIoU for each semantic part on our proposed FAUST benchmark. SATR significantly outperforms 3DHighlighter on the coarse-grained parts. As shown in Figure 7, our method SATR outperforms 3DHighlighter and CLIPSeg on all of the four semantic parts (leg, arm, head, and torso) by an overall average mIoU of 82.46%. In addition, we show that our proposed components(SATR) help improve the results upon our GLIP-based baseline (SATR-Baseline). In Figure 7, we show the qualitative results of our method, and we compare it to 3DHighlighter. In addition, in Table 1, our method SATR overall outperforms 3DHighlighter with a margin of 42.12% in the fine-grained semantic segmentation. In Figure 7, we show the results of SATR on fine-grained segmentation compared to the ground truth.

4.3.2 ShapeNetPart Dataset

In Table 3, SATR consistently outperforms 3DHighlighter in every shape category by a large margin. The results suggest that our proposed method SATR is applicable not only to human shapes but can also perform well in a wide range of categories. In Figure 7, we compare SATR and 3DHighlighter. The main challenge is that 3DHighlighter only works with the right random initialization, which doesn’t happen too often.
4.4. Ablation Studies

Effectiveness of the proposed components. We ablate the effectiveness of our proposed components for FAUST coarse and fine-grained benchmarks in Tables 5 and 7. Using both Gaussian geodesic re-weighting and visibility smoothing gave the best performance in both the coarse and fine-grained FAUST benchmarks. In addition, each component is effective on its own and performs better than SATR-Baseline. The results suggest that both components work in a complementary fashion.

Different reweighting methods. We compare using different re-weighting methods as shown in Table 6. As dis-
Table 4: mIoU comparison between SATR and SATR-F with recent SAM-based [46] backbones.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone(s)</th>
<th>Coarse-Grained mIoU</th>
<th>Fine-Grained mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>SATR-F</td>
<td>GLIP [55]</td>
<td>81.16</td>
<td>41.96</td>
</tr>
<tr>
<td>SATR</td>
<td>GLIP-SAM [45]</td>
<td>78.59</td>
<td>27.52</td>
</tr>
<tr>
<td>DINO-SAM</td>
<td>[14]</td>
<td>80.42</td>
<td>21.90</td>
</tr>
<tr>
<td>SATR</td>
<td>GLIP [55]</td>
<td><strong>82.46</strong></td>
<td><strong>46.01</strong></td>
</tr>
</tbody>
</table>

Table 5: Ablation to show the effectiveness of our proposed components for the coarse-grained FAUST benchmark.

<table>
<thead>
<tr>
<th>Gaussian Geodesic Reweighting</th>
<th>Visibility Smoothing</th>
<th>mIoU</th>
<th>arm</th>
<th>head</th>
<th>leg</th>
<th>torso</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>81.16</td>
<td>82.01</td>
<td>88.17</td>
<td>86.54</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td></td>
<td>81.69</td>
<td>82.68</td>
<td>88.61</td>
<td>86.85</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td></td>
<td>82.39</td>
<td>85.73</td>
<td>90.61</td>
<td><strong>85.81</strong></td>
</tr>
<tr>
<td></td>
<td>✓ ✓</td>
<td></td>
<td>82.46</td>
<td>85.92</td>
<td>90.56</td>
<td>85.75</td>
</tr>
</tbody>
</table>

Table 6: Ablation on using different re-weighting methods. Our proposed Gaussian Geodesic Re-weighting method outperforms other normalization methods. This shows its effectiveness in the fine-grained and more difficult semantic segmentation task.

<table>
<thead>
<tr>
<th>Re-weighting Method</th>
<th>Coarse-Grained mIoU</th>
<th>Fine-Grained mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Geodesic</td>
<td>82.41</td>
<td>44.57</td>
</tr>
<tr>
<td>Softmax Geodesic</td>
<td>81.69</td>
<td>43.34</td>
</tr>
<tr>
<td>Gaussian Geodesic (ours)</td>
<td><strong>82.46</strong></td>
<td><strong>46.01</strong></td>
</tr>
</tbody>
</table>

5. Conclusion

In our work, we explored the application of modern zero-shot 2D vision-language models for zero-shot semantic segmentation of 3D shapes. We showed that modern 2D object detectors are better suited for this task than text-image similarity or segmentation models, and developed a topology-aware algorithm to extract 3D segmentation mapping from 2D multi-view bounding box predictions. We proposed the first benchmarks for this area. We compared to previous and selected concurrent work qualitatively and quantitatively and observed a large improvement when using our method. In future work, we would like to combine different types of language models and investigate in what way the names of the segments themselves can be proposed by a language model. We discuss the limitations of our approach in Appx A.

References


[23] Dale Decatur, Itai Lang, and Rana Hanocka. 3d highlighter: Localizing regions on 3d shapes via text descriptions. arXiv, 2022. 2, 7, 8, 16


[30] Xiaoyi Dong, Jianmin Bao, Yinglin Zheng, Ting Zhang, Dongdong Chen, Hao Yang, Ming Zeng, Weiming Zhang,


[43] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In International Conference on Machine Learning, pages 4904–4916. PMLR, 2021. 3


[53] Trung-Ngiah Le, Tam V Nguyen, Minh-Quan Le, Trong-Thuan Nguyen, Viet-Tham Huynh, Trong-Le Do, Khanh-Duy Le, Mai-Khiem Tran, Hnat Hoang-Xuan, Thang-Long


Hanoona Rasheed, Muhammad Maaz, Muhammad Uzair Khattak, Salman Khan, and Fahad Shahbaz Khan. Bridging the gap between object and image-level representations for open-vocabulary detection. In 36th Conference on Neural Information Processing Systems (NIPS), 2022. 3


Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kontschieder. Panoptic lifting for 3D scene understanding with neural fields, 2022. 3


Li Yi, Vladimir G. Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3D shape collections. SIGGRAPH Asia, 2016. 2, 6


