Neural Point Cloud Highlighting of Affordance Region

Lukas Loiodice
POLITO - Grenoble INP
s334726@studenti.polito.it

Louise Tiger
POLITO - ESILV
s335276@studenti.polito.it

Antoine Wencel
POLITO - Centrale Nantes
s321430@studenti.polito.it

1. Introduction

This paper aims to present our exploration, implementation and propositions of extension of the 3D highlighter developed by students from the University of Chicago. The goal of the project was to build a label-free pipeline for affordance generation on point cloud in a zero-shot fashion, leveraging pre-trained language vision models.

The project was realized in four parts. In the first part we studied the paper ”3D Highlighter: Localizing Regions on 3D Shapes via Text Descriptions” [1] and the literature associated with it. Secondly, we implemented Neural Highlighting Model and the CLIP [4] loss in the given skeleton for the code base, then we evaluated these implementations and explore the hyperparameters by visualizing the results with diverse configuration. During the third step we adapted the pipeline to work on point cloud data on the AffordanceNet dataset [2]. Finally, we defined a testing pipeline to assess how good our model is at finding the affordance regions and explored diverse possible extensions.

Our code is publicly available at https://github.com/Daviruss1969/Affordance_Highlighting_Project_2024.

2. Related Work

The main paper studied for this project [1] present a 3D highlighter. Using as input a mesh (3D representation of an object, animal or character) and a text. The output will be the mesh coloured in the region of interest described in the text. The particularity of this algorithm is his ability to provide “out-of-domain” localization. For example, by localising the region for hat on a lamp, it is called hallucinated highlighting.

The pipeline presented in this paper is composed of a multi-layer perceptron (MLP) used as a Neural Highlighter. The MLP will predict the probability for each vertex of the mesh to be highlighted. The mesh will then be coloured depending on this probability. At the beginning each vertex is a assigned a random colour (half highlight H and half grey G). The mesh is said half-highlighted. Then it is weighted with the probability given by the neural highlighter and evaluate towards 1 (H) or 0 (G). The gradient of the colouring function is continuous generating a smooth change. At each iteration, the model is guided by the joint vision language embedding space of CLIP. A fixed number of points of view and image augmentation is selected and compared to the 2D images predictions made by CLIP and use as target value. A CLIP-loss is then generated. The goal of the model is then to minimize this loss.

The pipeline presented is said to be global semantic understanding, making a real difference between accessories with same geometric characteristics such as a belt and a neckless. It shows consistency regardless of the selected primary viewpoints. It is not restricted to specific categories and able to produce hallucinated highlighting. The highlights are specified, we can obtained different part of the same object depending on the input text. Neural field optimization allowed to transfer localization result to different meshes. Coloration can be transfer from a mesh of an object to another mesh of the same object.

This highlighting technic can be helpful in many applications as explained in the paper. It is possible to add an object with different shape, colour and texture in selected regions highlighted without affecting the rest of the object. When doing composition, some text models struggle to select the distinct part of the object or body to select. Resulting in an overall mix of points of different object without the parts being really identified and so without answering the initial composition demand. This method allows clear frontier to cut before combining the different parts. Finally, the model is not restricted to hallucinated highlighting and can localize semantically specified geometric regions.

When detailing the components of the pipeline, the researchers emphasize the significance of several key aspects such as the combination of the optimization neural field, the probability weighted blending and the image augmentations is necessary to obtain meaningful results. None of these three parts of this method can be suppressed or simplified without degrading the predictions. They also insist on the prompt formulation which need to be clearly understood by CLIP and must correspond to the geometry of the object.

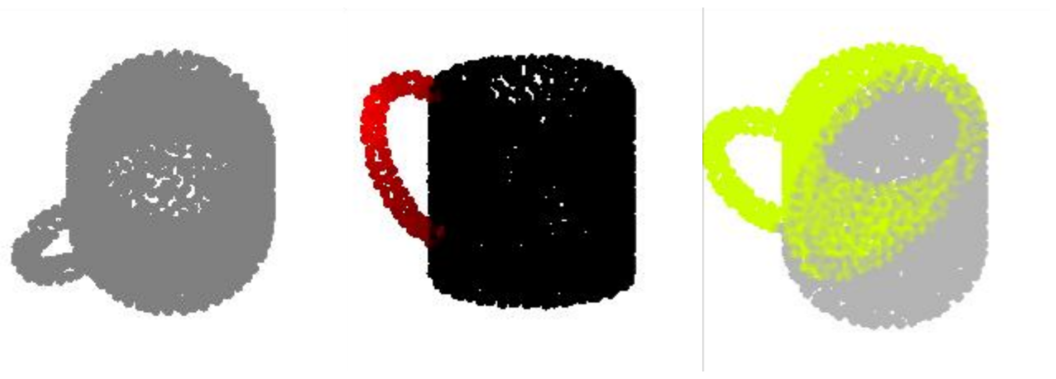

Figure 1. PointCLoud of Mug with highlighted affordance of the grabbing part in the AffordanceNet dataset [2] compare to ours.

In 2021, researchers from OpenAI, San Francisco published a paper introducing a state-of-the-art computer vision systems named CLIP [4]. It is trained by matching captions with images over a large dataset of 400M of pair and then using raw text about images as supervision. CLIP uses two encoders: one for images based on a visual model like ResNet or Vision Transformer and one for text based on a transformer. Both encoders map images and texts into a shared embedding space where related pairs are close together. This enables zero-shot transfer, allowing the model to perform tasks without task-specific training. The performance was benchmark over 30 dataset showing competitive results compare to over models. In addition to that, the model is available already pre-trained and weighted on GitHub which make it easy to implement. In 2022 another paper [5] from Open IA proposed a two-stage model using CLIP to generate an image with an input text. In the first stage, CLIP is used to generate image embedding corresponding to the given text. And in the second stage, a decoder named unCLIP generates an image conditioned on the images embedding, preserving its semantics and style while variating non important details. This model creates high-quality images matching the prompt.

Affordance, describing possible interactions between objects and their environment, is crucial for enabling robots to operate in complex settings. Unlike prior datasets, 3D AffordanceNet [2] provides fine-grained annotations of affordance probabilities for over 22,000 shapes across 18 affordance types and 23 object categories, derived from the PartNet dataset. This resource supports tasks such as full-shape, partial-view, and rotation-invariant affordance estimation. Additionally, the paper evaluates state-of-the-art 3D learning models on these tasks and proposes a semi-supervised method to leverage unlabeled data. By integrating geometric and functional object properties, 3D AffordanceNet advances affordance understanding in dynamic and realistic scenarios.

The Neural Highlighter proposed the researchers is a Neural Field “parameterize physical properties of scenes or objects across space and time.” [8]. These methods show great success in 3D images fields. A typical neural fields algorithm is described as succession of five steps. First the coordinates are sampled from an image for example. Then this coordinates are given as input to a Neural Network which generate field quantities. These field quantities are projected in the desired reconstruction domain. For 3D problems, this could involve 3D shapes, point clouds, or other forms of spatial data. Successively a forward map is applied to transfer the information to a sensor domain, in our project it would be the coloured mesh. Finally a reconstruction loss is computed, for us it will be the CLIP loss.

3. Method

3.1. Neural Highlighting Model and Clip Loss

To Recreate the pipeline presented in the ”3D Highlighter: Localizing Regions on 3D Shapes via Text Descriptions” paper, we were given a skeleton of this pipeline with some missing functions. We needed to implement the Neural Network, to import the clip model and to create the clip loss function. For the Neural Network we based our structure on the one presented on the researchers GitHub. Our Network start with an input layer followed by an activation function which was imposed as ReLU and a normalization layer using LayerNorm. Then we have the hidden layers grouped by 3 a linear, an activation and a normalization layers, the number of groups depend on the depth of the network. Finally, we have the output layer and the activation layer using SoftMax. All these layers are then compiled in the ModuleList of the network.

As seen in the literature, a meaningful loss function is essential for a network field to improve his performance between each iteration. Once our mesh is coloured, a number of sample views is selected (in our project 5 images). This set of 2D images are input to our clip loss function with our prompt text, our clip model, the number of image augmentation wanted and the augmentation transformation function. For each augmented image, the CLIP model will project the input text and the image into a common vector space which will allow us to compute the cosine similarity function. If there is no image augmentation we use the original image directly by using the function clip model allowing us to resize the image at the right size for CLIP. All cosin similarity results are subtracted to the loss. Meaning the less the vectors corresponding to the text and to the image are similar the higher the loss will be. Consequently the goal will be to minimize the lost.

3.2. Point cloud adaptation

The goal of this section is to adapt the 3D Highlighter pipeline, which was originally designed for meshes, to work with point clouds. To achieve this, we decided to update the rendering component, replacing the original differentiable mesh renderer with a differentiable point cloud renderer.

First, we implemented a component to load a point cloud either from a tensor, a PLY file, or by sampling points from a mesh using Open3D (https://www.open3d.org/). Since the rendering will be done with pytorch3D (https://pytorch3d.org/), a conversion between the two libraries is required. Once the point cloud is created, it is normalized by centering and resizing it uniformly.

Next, we implemented the rendering part, starting with a rasterizer that defines the radius of the rendered points, as well as the dimensions and number of points per pixel. Additionally, we included a renderer that specifies the background.

For each view, we define a specific camera to capture multiple angles, helping CLIP [4] guide the model toward the desired result. To compute the settings for each camera, we adjust the elevation and azimuth angles by specifying their mean and standard deviation.

Finally, the pipeline follows the original approach. For each rendered view, we apply various augmentations and compute the cosine similarity between the CLIP text embedding and the CLIP augmented images embedding.

3.3. Hyperparameters optimization

This section outlines the chosen pipeline for selecting the appropriate hyperparameters and prompts based on a specific task.

To achieve this, with the help of the AffordanceNet dataset [2], we focus on a specific object and a specifIc affordance. We select the Mug object with the grasp affordance.

Initially, we conduct some experiments based only on visual results, without considering any metrics. This allows us to establish a starting point for further optimization.

To select the best hyperparameters and prompts, we do not follow the traditional train, validation, and test set pipeline. Instead, we begin by defining a small subset of the dataset as our validation set, typically consisting of four examples in our case, and adjust the hyperparameters to achieve the best mean intersection over union (mIoU).

Once we obtain convincing results for the validation set, we apply the model to a test set, which is another subset of the dataset, to evaluate its generalization to previously unseen point clouds.

The final goal is to use our model as a zero-shot fashion, in the sense that we can obtain results without checking the ground truth. In our case, the selected shapes for the validation set are described in Fig. 7. The threshold selected for the mean intersection-over-union is 0.4.

3.4. Choice of Extensions

Our pipeline for implementing extensions is as follows: we first apply them to meshes to evaluate their effectiveness, leveraging the extensive testing presented in the 3D Highlighter paper, and fine-tune any potential hyperparameters. Once this is achieved, we proceed to implement a similar solution for the point cloud pipeline.

During our training we observed that more complex mesh with a high number of vertices may struggle to highlight objects located on distinct part of the body like shoes for instance. As this association is efficient for the dog (1480 vertices) it is not for the horse (21867). As it was establish in the main paper highlight localization from one mesh to the other describing the same object is doable and does not affect the quality of the results. Consequently we had the idea to create a simplified mesh for the horse, allowing the model to have a better understanding of the all structure and then transfer the results to the initial mesh.
To implement this idea, we converted our mesh into a trimesh object and utilized the simplify quadric decimation function from trimesh to reduce the number of vertices. This function effectively preserves the overall shape and integrity of the model by prioritizing the reduction of finer details using quadratic error. The function offers the possibility to controle the diminution of the mesh complexity.

In order to study the influence of the training augmentations on the results obtained, we added some Horizontal flips, random small rotations and Colour jitter in the transformation pipeline (that already contained some random perspectives and resized crop). Furthermore, we implemented a function that replaces the white background of the image with a random colour (the background obtained is either uniform or stripped) Fig. 2.

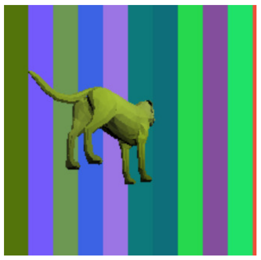

Figure 2. Background generator used on the dog

To apply the extensions on the point cloud, we first focus on the augmentation component, we attempted to implement strides for the background but were unable to achieve a functional solution. As a result, we only applied the basic augmentations. For the reduction component, we simply down sampled the points using open3D (https://www.open3d.org/) before passing them to the model.

4. Experiments

4.1. Hyperparameters exploration

This part of the paper will describe the impact of the hyperparameters on the highlighting results. To make the comparison we decided to highlight the Neckless of the horse using the prompt: ”a grey horse with green Neckless.”. We started by using the same parameters as the ones used in the main paper. And compute the results when increasing and decreasing each parameters.

After training the model with the paper parameters, we tried to use a shallow network, but the more iterations, the further from the solution we wanted to obtain, as we can see on the image obtained Fig. 3. So, while taking less computational time, this method is not efficient. On the contrary, by having a deeper network, the prediction appears to focus the region of interest on the horse’s chest rather than its neck.

Then we focused on the number of augmentations – without any, the model does not converge to a meaningful solution as the highlighted part on the Fig. 3. This confirm the paper observation claiming that image augmentation is necessary for the model to work. By adding more of those, we also obtained results close to what we were aiming for. We got our best results by combining some hyperparameters changes while doing various tests (i.e. 5 augmentations and a network depth of 2): with only 500 iterations, we obtained a result really close to the reality as the image shows. The only drawback of these settings is the extended computational time caused by the high number of image augmentation.

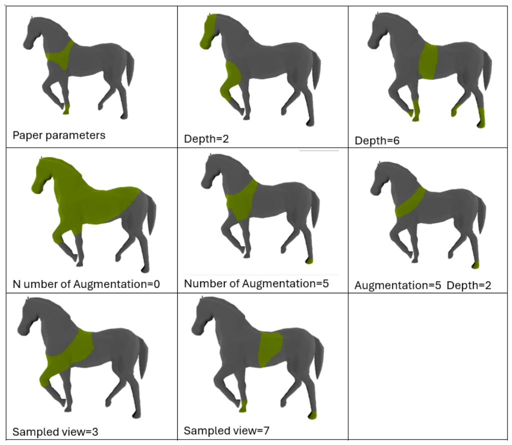

Figure 3. Test of hyperparameters on ”a grey horse with green Neckless.” Starting from the parameters of the paper. depth=4 ; number of augmentation = 1 ; number of sampled view = 5 and learning rate = 0.0001

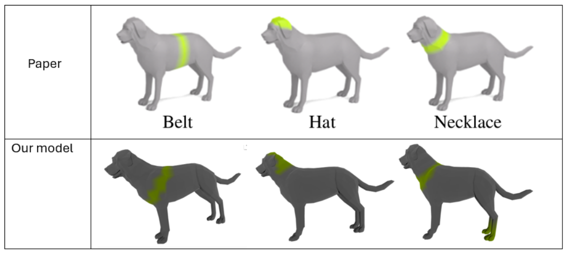

Figure 4. Dog results in the paper VS using our model

The Learning rate remained unchanged during our experiments. Indeed, setting a higher learning rate resulted in a total extinction of the non-highlighted parts in less than 100 iterations, and with a lower one, it took too many iterations to have any meaningful results.

4.2. Comparison between paper model and our model

To compare our model to the one presented on paper we conducted a series of experiments on diverse objects to see if we had the same observations. We started by comparing the association of the dog with diverse object using the same parameters as the researchers. Our results diverge slightly from the paper while still being coherent to the task Fig. 4. Then we tried to evaluate the impact of the text formulation on the results. We observed that adding a capital letter to the object help the model to highlight the right region. Plus, compare to the paper where adding a colour had a negative impact on their model, it is for us necessary to add the colour green in our text Fig. 5. In the paper it is mentioned that the model can make a different between objects and

Prompt	Views	Augmentations	Depth	Point radius	Augs	mIoU
A 3D point cloud rendering of a gray mug with a highlighted area intended for gripping.	5	1	4	0.03	False	0.09
A 3D point cloud representation of a gray mug with the precise area intented for gripping shown in green.	5	1	4	0.03	False	0.15
A 3D point cloud representation of a gray mug with the precise area intented for gripping shown in green.	5	2	5	0.02	False	0.11
A 3D point cloud representation of a gray mug with the precise area intented for gripping shown in green.	5	2	5	0.03	False	0.18
A 3D point cloud representation of a gray mug with the precise area intented for gripping shown in green.	5	2	5	0.03	True	0.16

Table 1. Subset of the results containing pertinent ones on the selected shapes on the validation set. ”Augs” refers to the application of the extentions augmentations.

parts which have the same geometry. Meaning the model should make a clear difference between a neckless and a belt Fig. 6. Globally our model behave in the same manner as the one presented while being less precise. we still observe the same strength and vulnerability.

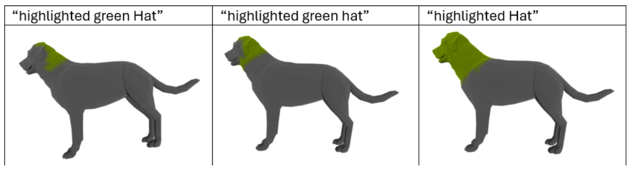

Figure 5. Dog results using diverse texts to express the object

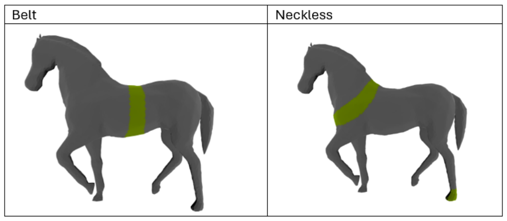

Figure 6. Horse results for a Neckless and for a Belt

4.3. Point cloud experiments

In this section, we examine and discuss the performance of our 3D point cloud highlighter on the AffordanceNet dataset [2], following the methodology described above. We selected four objects as our training set. Fig. 7 shows the selected shapes with their corresponding affordances highlighted. To improve generalization on the test set, we intentionally avoided selecting too similar mugs.

To achieve optimal results, we first focus on refining our prompting strategies, evaluating only the visual outcomes while maintaining a consistent set of hyperparameters from the previous section. We experimented with a variety of prompts during training to assess their impact on the model, though many were not effective across the entire validation set. Tab. 1, present the two prompts that yielded the best visual results. These prompts follow a similar structure to the initial 3D highlighter prompt, starting with the context (’A 3D point cloud...’), followed by the object description (’of a grey...’), and ending with the specific area we wanted to highlight. Short and overly general prompts typically resulted in poor outcomes, often producing a full gray output. For this reason, we opted for more detailed prompts, which wasn’t problematic as we primarily focused on a single object and affordance.

We also highlight relevant results for the sets of hyperparameters in Tab. 1 emphasizing the importance of them on the result. Indeed, by tweaking only one of them, we can have completely different results on the validation set. In Fig. 8, we illustrate the impact of the point cloud radius, which plays a significant role in our pipeline. Specifically, we hypothesize that reducing the radius makes the point

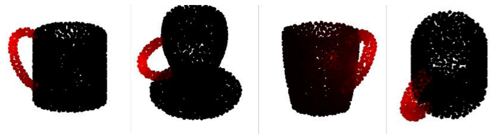

Figure 7. Selected shapes for the validation set. The corresponding affordance targeted (grasp) is shown in red

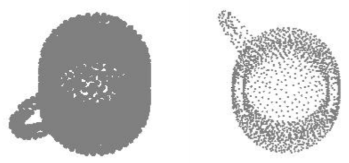

Figure 8. Differences between a rendered point cloud with a point cloud radius of 0.03 (left) and 0.01 (right)

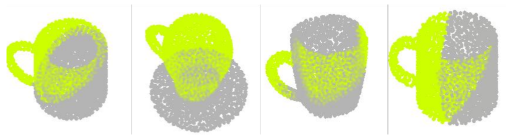
Figure 9. Visual results on the selected shapes for the validation set. The corresponding affordance target (grasp) is shown in green

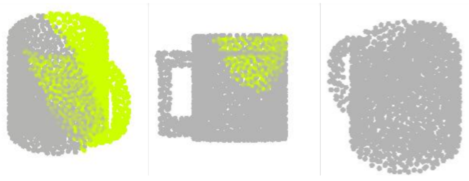
Figure 10. Visual results on a subset of the test set. The corresponding affordance target (grasp) is shown in green. The left image represents a correctly localized region, the center one shows a missed region, and the right shows no highlighted regions.

cloud sparser, potentially leading to misinterpretations by CLIP [4].

Fig. 9 presents the results obtained using our optimal set of hyperparameters and prompts. As shown, the method successfully highlights the corresponding area in all cases. However, the highlighted area is consistently too large, resulting in a low intersection-over-union score. For the test set, the results are significantly worse, with a mean intersection-over-union of approximately 0.03 across all the tested objects. In many cases, there are either no parts highlighted, fully highlighted or appears in completely incorrect areas shown in Fig. 10.

We hypothesize that these results are influenced by several factors. First, the limited time available to optimize the models due to the deadline likely contributed to the outcome. There is certainly room for improving the mean intersection over union for both the validation and test sets by further fine-tuning the prompts and hyperparameters. Additionally, the area designed for grasping a cup defined by AffordanceNet may not be sufficiently precise in the semantic for our pipeline. As shown in Fig. 11, there are multiple ways to grasp a cup, and this is further supported by the AffordanceNet dataset, which includes a similar affordance, 'wrap-grasp,' as illustrated in Fig. 12. This could lead CLIP to direct the model in multiple directions when specifying the keyword 'precise area'.

In Tab. 2, we compare our results with the state-of-the-art models, DGCNN [7] and PointNet++ [3], to provide a sense of scale within this dataset. It is important to note that the mean intersection over union for our pipeline is computed exclusively for the mugs we tested, whereas for the other two models, it is calculated across the entire dataset. Furthermore, our approach operates in a zero-shot fashion, whereas the other models are trained on the entire training set.

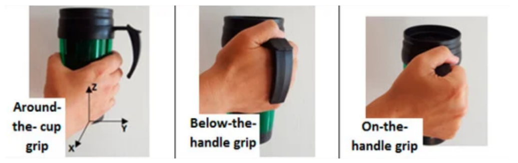

Figure 11. Picture that shows different ways to hold a cup, found in the paper "Does Cup-Grip Type Affect Tremor among People with Essential Tremor?" [6]

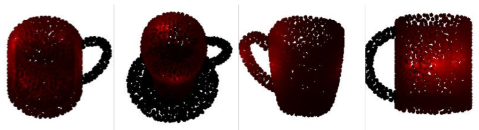
Figure 12. Selected shapes for the validation set. The affordance "wrap-grasp" is shown in red

At first, these results may seem strange: while our mean IoU val is similar to the others, our mean IoU test is nearly 6 times smaller that the ones obtained in the original paper. This can be explained by the fact that our validation set is made of only 4 elements chosen for their better results, thus the generalization is less efficient.

4.4. Extensions experiments

We evaluated the performance of our model with diverse mesh representing the horse. We used our simplifying mesh function to obtain less complex mesh of the horse Fig. 13. We observed that decreasing the number of vertices allow the model to perform a better detection of certain area like all four feet for the shoes. It also take less time to generate using the same hyperparameters (three minutes for 500 iterations with the original mesh agains two minutes for the 909 vertices mesh). In the other hand it will anormally struggle to converge with another object such as the belt. Plus, this model is really sensible and decreasing the number of vertices may obtains bad results for every object. If the results were the right highlighted region but less precise we could have start the program with simplified mesh before tranfering to the original set to obtain more precise results. But, for some object the result is clearly to far away from expected highlighting the leg instead of the head for instance which force us to abandon this idea. In addition to that the background extension didn't change a lot the results of the first tests.

4.5. Application of extensions on Point Cloud

We initially attempted to apply the augmentation extension but were unable to implement the striding background for the point cloud within the available time. Therefore, we applied only the augmentations. As shown in Fig. 14, the

Model	Mean IoU val	Mean IoU test
DGCNN	17.5	17.8
PointNet++	19.2	19.3
Ours	18	3

Table 2. Comparaison between our model and the state-of-the art DGCNN [7] and PointNet++ [3]. The results for DGCNN and PointNet++ was found in the AffordanceNet [2] paper. The "Mean IoU val" refer to the results on the validation set. The "Mean IoU test" refer to the results on the test set. Important to notice that our validation set is only made of 4 Mugs, and our test set around 20 Mugs.

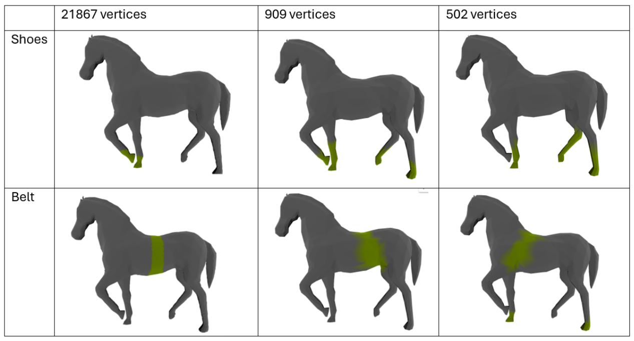

Figure 13. Results for the Horse depending on the mesh complexity

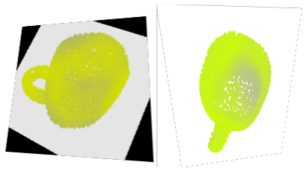

Figure 14. Comparison between the same point cloud with extension augmentations (left) and without (right). It is important to note that normalization based on the CLIP [4] mean and standard deviation was removed in these augmentations to prevent color alterations.

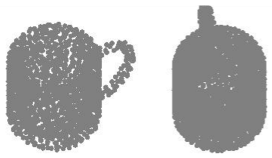

Figure 15. Comparison between the same point cloud with down sampling (left) and without (right).

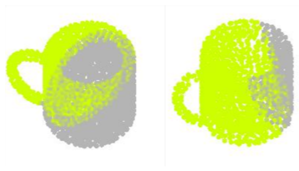

Figure 16. Visual comparaison on the same mug according to the grasp affordance with down sampling (right) and without (left).

augmented images produced by the extension differ from the original ones. However, as indicated in Tab. 1, even on the validation set, the results did not improve. With more time, it might be possible to enhance the results by fine-tuning the prompts or hyperparameters according to the augmentations.

Next, we experimented with the down sampling extension. For this approach, we relied solely on visual results, as the mean intersection over union may differ between the objects and the ground truth in the AffordanceNet dataset due to reduction. Fig. 15 illustrates the difference between an original mesh and its down sampled version, showing that even with a 50% reduction in points, the mesh remains recognizable. Fig. 16 also compares the results for a training set element with and without down sampling. As observed, the results are worse with down sampling. However, with more time and by fine-tuning the hyperparameters especially the point radius in the renderer, it is likely possible to achieve at least comparable results while reducing computational costs thanks to down sampling.

5. Conclusions

We present an adaptation of the 3D Highlighter paper [1] to operate on point clouds, specifically on the AffordanceNet dataset, without any pre-training and in a zero-shot fashion, leveraging the pre-trained language-vision model CLIP [4]. In our pipeline, we fine-tune hyperparameters and prompts on a specific subset of the AffordanceNet dataset [2], referred to as the validation set, focusing on the Mug object with the Grasp affordance. We then evaluate the generalization of our pipeline on another subset of the dataset, termed the test set. Finally, we explore potential extensions to the existing pipeline, testing improvements by incorporating custom augmentations and experimenting with down sampling the object.

As a result, without any extensions, we achieved average performance on the validation set after extensive hyperparameter and prompt fine-tuning, but poor results on the test set. We hypothesize that the primary reasons for these outcomes are the limited time available for thorough fine-tuning and the semantic ambiguity of the grasp area for a mug. There are multiple ways to grasp a cup, leading to a semantic conflict as the model attempts to accommodate diverse interpretations. The implementation of the extensions did not significantly improve the model's ability to identify the corresponding region. However, the down sampling extension, while yielding slightly worse results, did reduce the computational cost.

References

[1] Dale Decatur, Itai Lang, and Rana Hanocka. 3d highlighter: Localizing regions on 3d shapes via text descriptions, 2022. 1, 7

[2] Shengheng Deng, Xun Xu, Chaozheng Wu, Ke Chen, and Kui Jia. 3d affordancenet: A benchmark for visual object affordance understanding, 2021. 1, 2, 3, 5, 7

[3] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017. 6, 7

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 1, 2, 3, 6, 7

[5] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022. 2

[6] Navit Roth and Sara Rosenblum. Does cup-grip type affect tremor among people with essential tremor? *Sensors*, 21(23), 2021. 6

[7] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds, 2019. 6, 7

[8] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond, 2022. 2