# Chapter 1

# Interpretability

In this section, we interpret MicroResNet7x7-SE's predictions using different techniques to test its ability to properly estimate traversability by classifying ground patches. We highlight its strength, robustness to understand its limitations. We evaluated the quality of our traversability estimator with different methodology. First, we showed that the model has correctly learned grounds' features and was able to separable terrains based on them. Second, we utilized the test dataset to visualize the most traversable, the lest traversable and the misclassified patches. Utilizing a special method, we determined that the model always looked at the correct features in the ground even if when it fails. Lastly, we crafted several patches with different unique features, such as walls, bumps, etc, to test the robustness of the model by comparing its predictions to the real data gathered from the simulator.

## 1.1 Features separability

In general, convolutional neural networks learn to encode images by applying filters of increasing size at each layer. Usually, the first layers learn basic features, such as edges, while the final one encodes complex shapes. The outputs in the final convolution layer are usually referred to as *features space*, consequently, a feature vector is just the output of the last layer for a given image. Those last features are combined and mapped to the correct classes by one or more fully connected layers. Lee et al. **?** have visualized the features learned by the first and last layers. This is visualize in figure 1.1 where there are the low level features (down) and the high level features (up) for four different classes, faces, cars, elephants and chairs, learned by a convolution neural network. So, a correctly trained network should be able to separate the inputs features based on the predicted classes. Intuitively, given two classes $\mathcal{A}$ and $\mathcal{B}$, for example, *chairs* and *cars*, the high-level features for each class should not be the same, otherwise, the model may misclassify the input due to the overlap of different classes' features. For instance, if the network believes that big wheels are features of both chairs and cars then chairs may be wrongly classified as cars. Similarly, two patches have a small and big wall in front of the robot should not be mapped in the same position in the features. Because, from a traversability point of view, have different characteristic, one has a traversable wall, the other not. However, those patches are close to each other in the features space, the model could foolishly believe they are similar and belong to the same class. One technique to discover the degree of separability is to directly visualize the features vectors for each class. In our case, MicroResNet7x7-SE, maps inputs to a 128 dimensional feature space. Since, we cannot

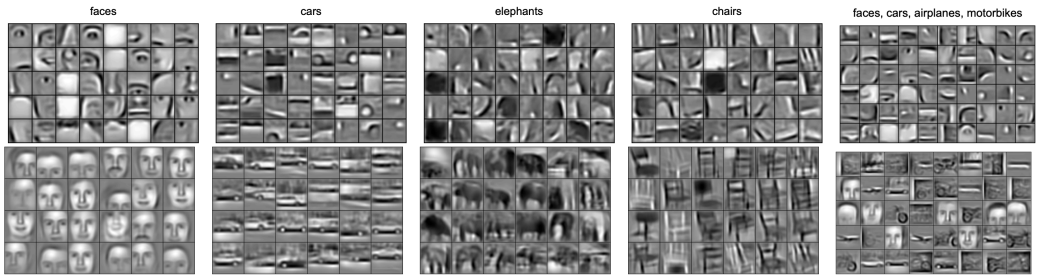



Figure 1.1. Figure from Lee et al. **?** paper where they show for fours different classes the low-level features (up) and the high-level features (down) learned by a convolution neural network.

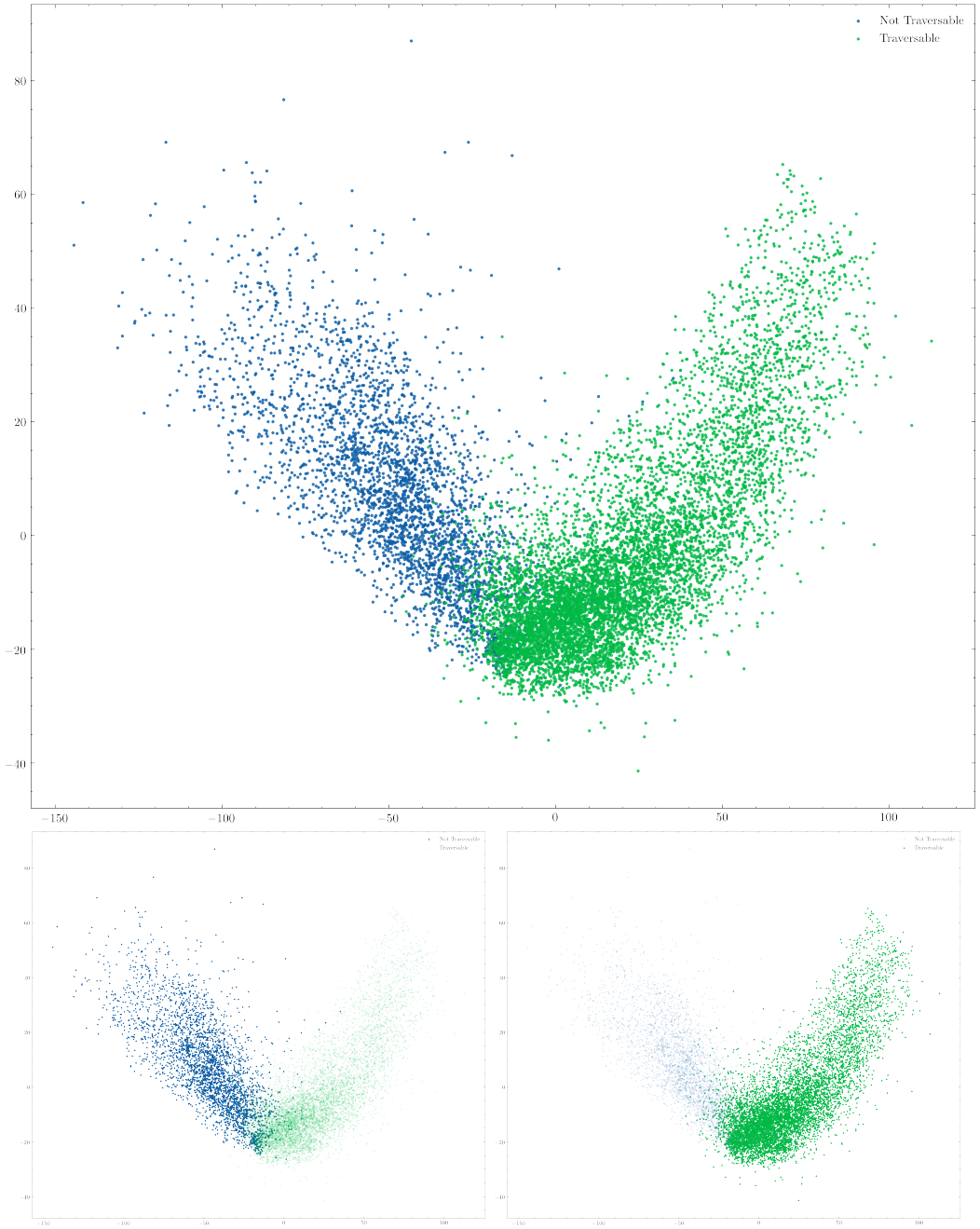
directly visualize such highg dimension space, we reduced the feature vectors to a two-dimension space by applying Principle Analysis Component (PCA) **?**. We investigat the features space of the model both in the train and test set.

### 1.1.1 Features space of the train set

Figure **??** shows the features space for 11K images sampled from the train set labeled with their classes, *traversable* and *not traversable*. We can clearly recognize two main clusters based on the labels' color, one on the left and one of the right. Those points are easily separable, even by human eyes, meaning that the model was able to learn meaning features from the dataset and use to make accurate predictions. To be totally sure the center of each class' point cloud is not overlapping we plotted the density of each cluster. Clearly, there is some distance between the centers. Furthermore, we can directly plot the patch corresponding to each feature vector to identify clusters of inputs based on their similarities. Intuitively, if similar inputs are close to each other in the features space then the model also learned to effectively encode terrains features. We decid to not show all images on the same plot to avoid overcrowding the image. Instead, we cluster the points using K-Means with $k = 50$ clusters and then we took the patch that corresponded to the center point in each cluster. In this way, even by showing only a few inputs, we include all the meaningful ground types. Figures 1.4 visualizes the results. Definitely, patches with similar features are close to each other yielding a quality features encoding. On the left-top side, we can distinguish highly untraversable patches with walls/bumps in front of the robot. Going down, we encounter patches with smaller obstacles. On the plateau, there are traversable patches with small obstacles such as light bumps. Importantly, those patches are the closest ones to the not traversable ones, so they were the hardest to separate, thus, to classify. Going up on the right side, we see some grounds with small steps. Finally, on the top, we find all the downhill patches, the simplest ones to traverse.

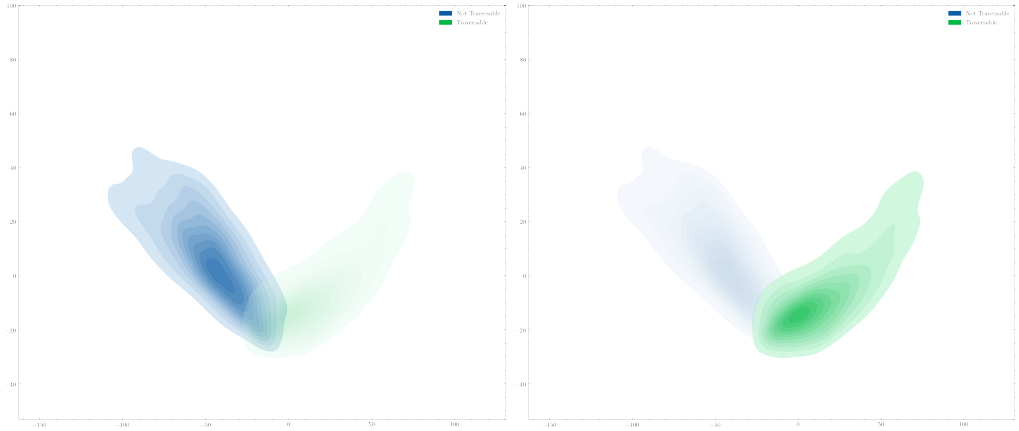
### 1.1.2 Features space of the test set

We can apply the same procedure on the test set. Since it is a real world quarry, this dataset is harder than the train set and present challenging situations for the robot. Figure 1.5 displays the features space after reducing its dimension to two using PCA. Interesting, the traversable patches in figure **??** are very near to each other, while the others span a very big surface. This suggests that there are many not traversable terrains with different features. The traversable points are clustered near the




(a) Not Traversable (b) Traversable

Figure 1.2. Principal Component Analysis on the features space computed using the outputs from the last convolutional layers on the train dataset. The two point clouds are perfectly separable.



(a) Not Traversable (b) Traversable

Figure 1.3. Density plot for train set features space. The more opaque the color the close to the cluster center. The centers of the cluster are not overlapping yielding a good separability and correct learning.

center, this implies that most of them share similar features. We plotted the density for each class to better understand where the most points are mapped. The two centers are really close to each other, making those samples harder to separate and some not traversable points are mixed up with the traversable ones. This explains the elevated number of false negative that lower down the AUC score on this dataset. As we did before, we can also visualize the patches by plotting them using their features coordinates. Figure **??** shows the patches directly into the features space. On the top left, from the not traversable cloud, we can see patches with a high level of bumps. Going down we find surfaces with huge walls in front of the robot while going close to the center we start to see all the traversable patches. Those samples have not too steep slopes. If we move to the density center, green double shown in figure 1.6b, we encounter lots of flat patches with little obstacles. Going up on the right branch we find downhill and on the top there are falls.

To summarized, we showed how the model correctly learned to separate the inputs based on their traversability, to encode meaningful grounds' information and to map close to each other patches with similar characteristic in the features spaces.

In the following section, we will take a deep look at the test set to find which patches confuse the most the model. Probably, those samples will be located between the two clusters' center where the difference between classes' features is minimum.

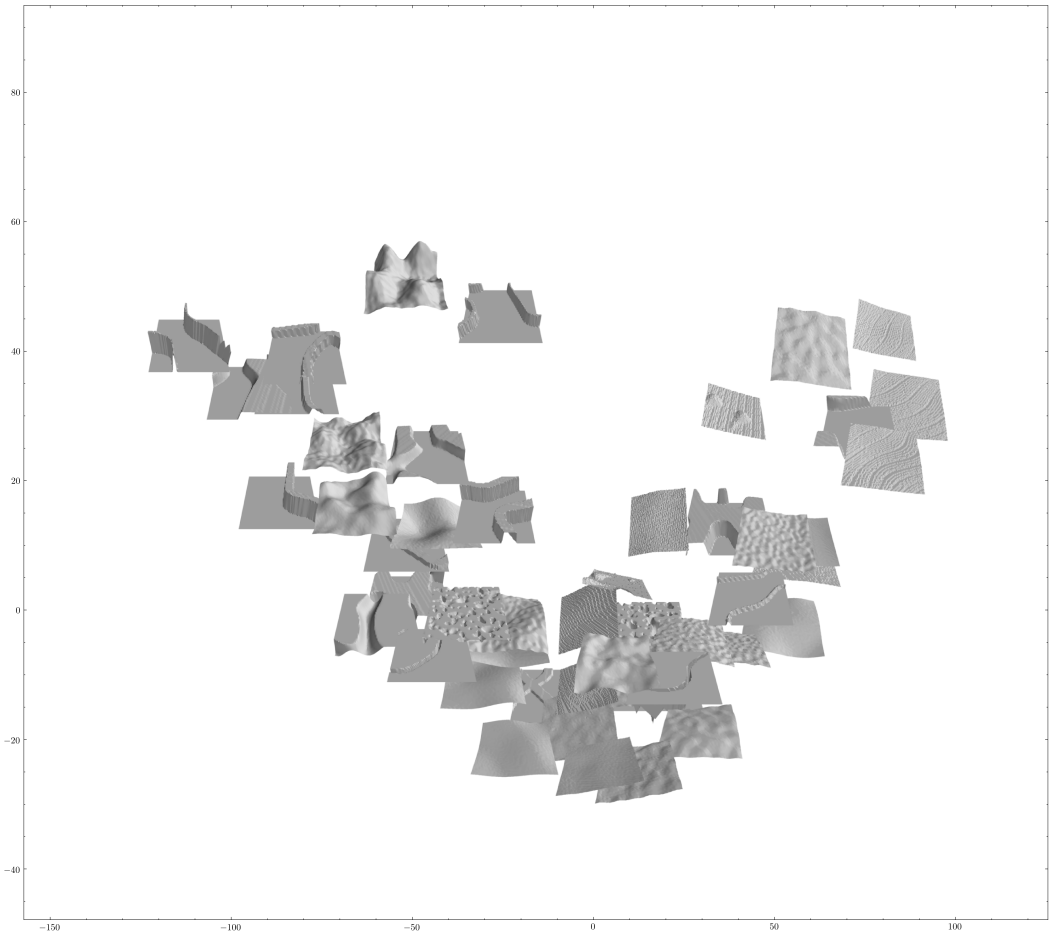



Figure 1.4. Patches plotted using the coordinate of the features vector obtained from the last convolutional layer's output and then reduced using PCA to a two-dimensional vector. Similar grounds are close to each other. From the top left in counterclockwise order, there are not traversable patches with walls, steps, and big bumps. On the plateau, we can found traversable patches with light bumps. Going up we encounter the downhills.

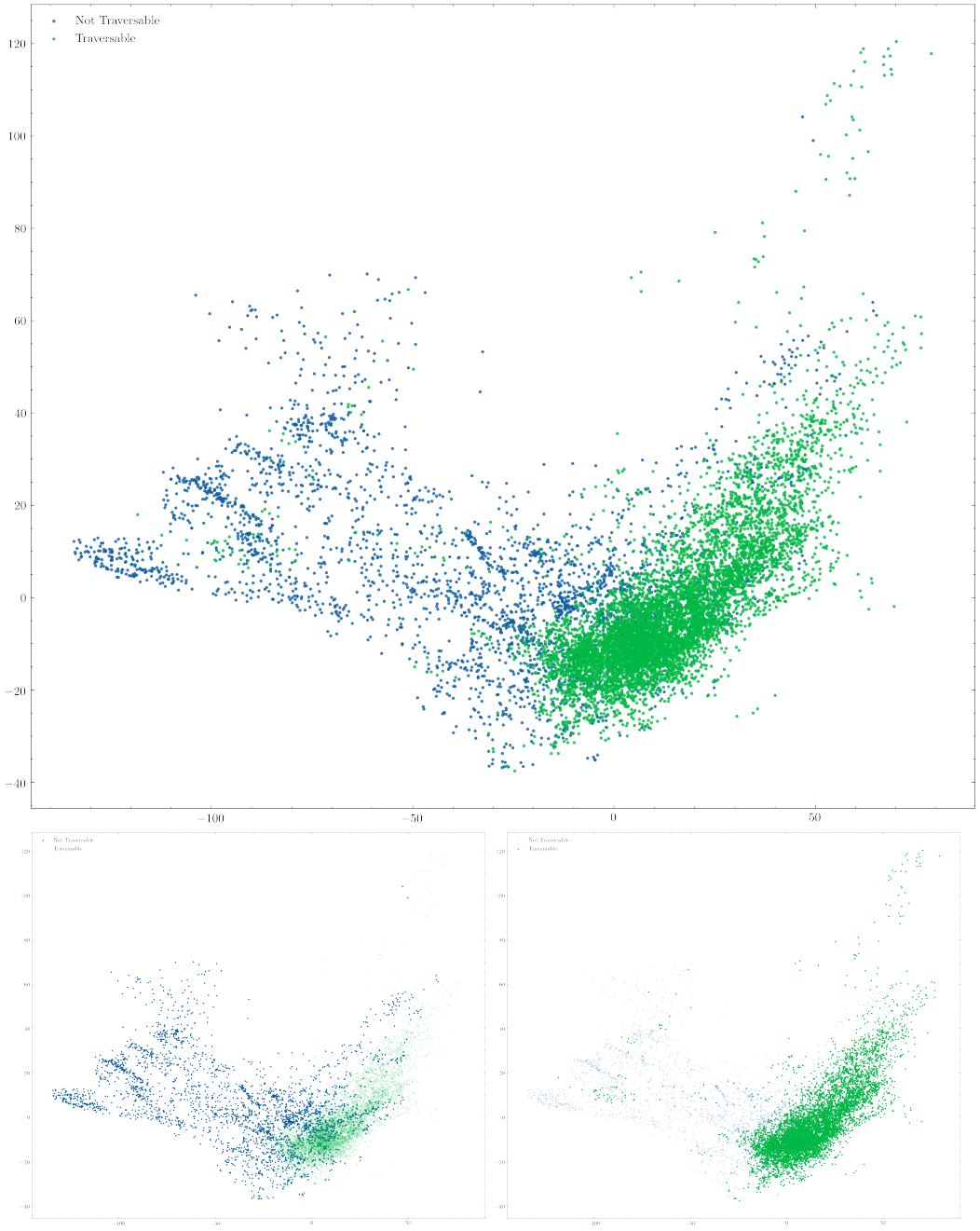
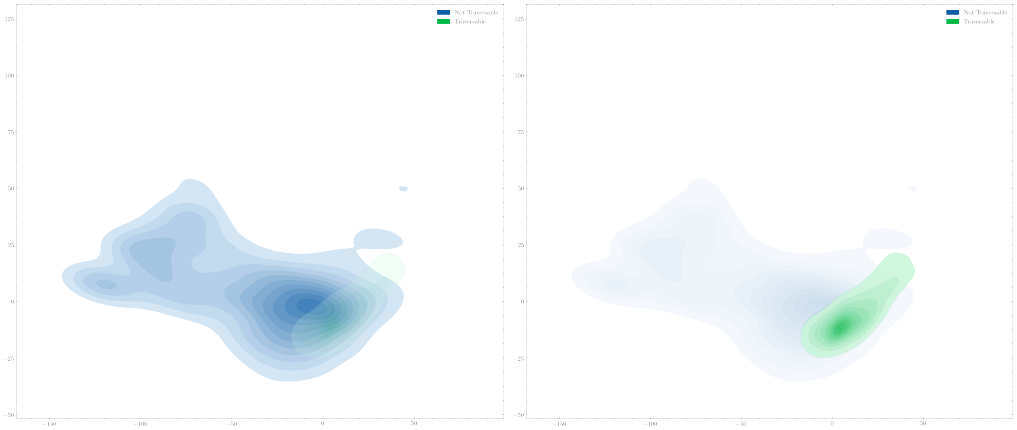



Figure 1.5. Principal Component Analysis on the features space computed using the outputs from the last convolutional layers on the test dataset. We can distinguish two main clusters. However, some points are mixed up between classes.




(a) Not Traversable (b) Traversable

Figure 1.6. ensity plot for test set features space. The more opaque the color the close to the cluster center. The centers of the clusters are close to each other yielding less separability/

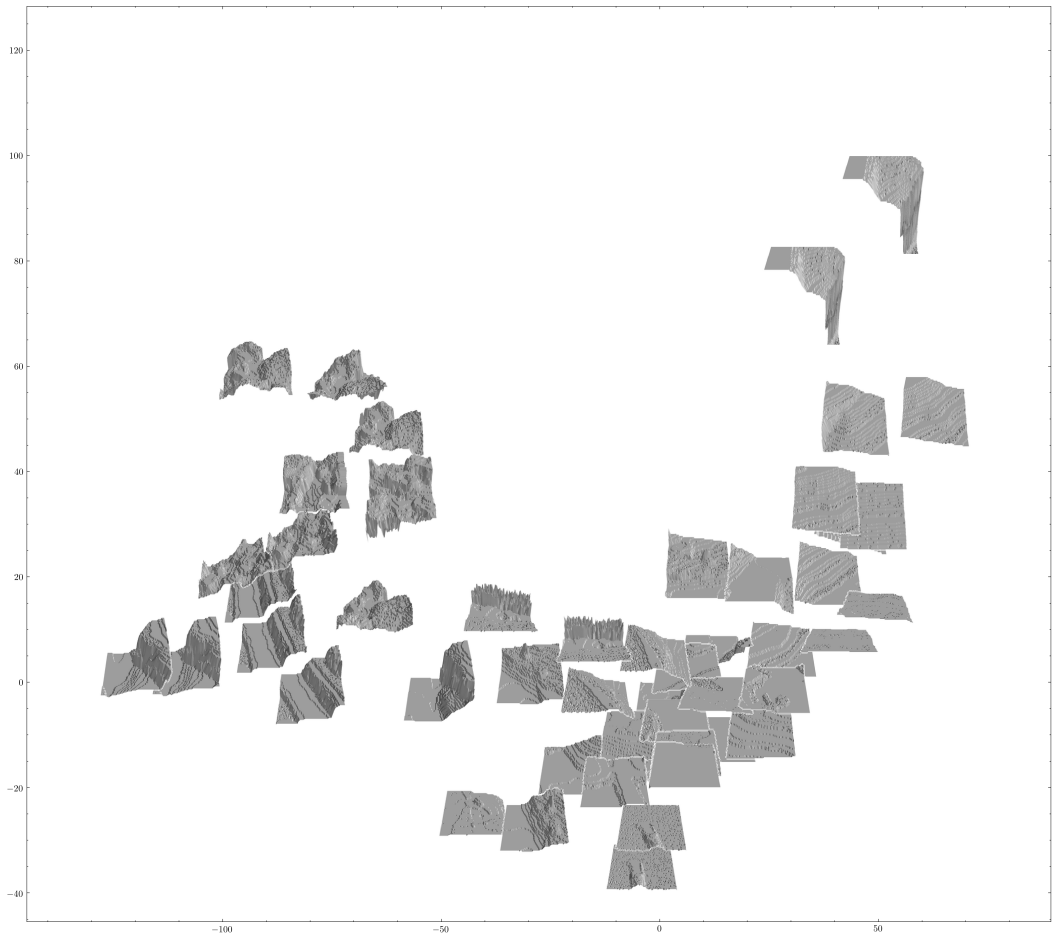



Figure 1.7. Patches that correspond to coordinates in the features space of the last convolutional layers on the test dataset. Similar grounds are close to each other. From the top left in counterclockwise order, we found not traversable patches with clearly not traversable features such us big bumps, huge obstacle towards the end. On the center, there are hardest surfaces to separate composed by slopes and small obstacles. Going up we found downhills and huge cliffs, highly traversable patches.