0.1 Test dataset exploration

After showing the model's capability of correctly separate classes' features we utilized Grad-CAM to visualize some patches inside the quarry dataset. The aim of this section is to show how the model looks at meaningful features in each input to make the prediction even when it fails. For instance, imagine we feed to the model a not traversable patch with an obstacle and the network label is as traversable. Clearly, the output is wrong but the model's error may be caused by two different situations. The first one, the model could just have ignored the obstacle and looked away, meaning that it was not even able to understand there was an obstacle in the first place. Second, the network could have correctly look at the obstacle but thought that obstacle is traversable, showing the correct ability to find and use important features in the map. In the following sections, we showed that, even when the predictions are wrong, our model always look at the most important features of each input to determine its traversability.

We divided the dataset patches into four classes based on the model's performance: worst, best, false positive and false negative. Then, we took twenty inputs from those sets and applied a technique called Grad-CAM to visualize which part of the ground the model is looking. Those regions are highlighted in 3D render to better visualize which spot of the inputs caused the prediction. Before starting the exploration, let us introduce the method that allowed us to identify regions of interest on the patches.

0.1.1 Grad-CAM

Before starting the exploration, let us introduce the method used to identify region of interest on the patches. Gradient-weighted Class Activation Mapping (Grad-CAM) **?** is a technique to produce v̈isual explanationsf̈or convolutional neural networks. It highlights the regions of the input image that contribute the most to the predictions. For example, given a classifier able to recognize cats and cat image, Grad-CAM will point outs the cat. In detail, the output with respect to a target

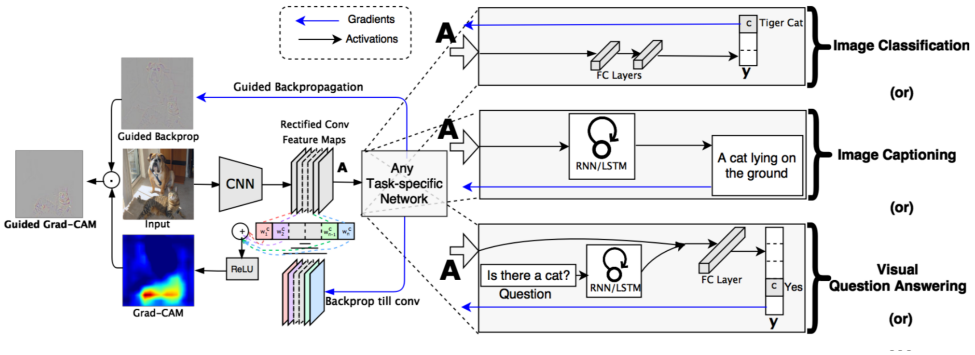

Figure 1. Grad-CAM procedure on an input image. Image from the original paper **?**.

class is backpropagated while storing the gradient and the last convolution layer's output. Then, a global average is applied to the saved gradient keeping the channel dimension in order to get a 1-d tensor, this will represent the importance of each channel in the target convolutional layer. After, each element of the convolutional layer outputs is multiplied with the averaged gradients to create

the grad cam. This whole procedure is fast and it is architecture independent. The procedure is summarized by figure **??**

0.1.2 Traversable patches

Those samples are the patches correctly predicted as traversable. We plotted twenty different inputs sampled uniformly according to the robot advancement to include as many interesting situations as possible. We recognized two main clusters of images: flat, figures 2a, 2b, 2f, 2g, 2h, 2i, 2j, 2k, 2o, 2q, 2s, and slopes, figures 2c, 2d, 2e, 2l, 2m, 2n, 2p, 2t. The models is mostly interested in the left part of the patches with uneven ground on the left region, 2a, 2b, 2c, 2d, 2e, 2g, 2j, 2l, 2m, 2n, 2o, 2p, 2r. This is due to the fact that an obstacle in that region can block the rear legs and prevent the robot to advance. In other patches where most probably a untraversable features may also be present ahead, the model discriminates different parts of the map. For instance, figure 2d shows an interest region on the slope near the end and figure 2l on two stop of the first faily big step ahead of the robot. Similarly, also figures 2n, 2q, 2s.

In some situations, the model identify a possible untraversable spot only forward being sure the robot is no immeaditely stopped. There are two obvious cases, figures 2h , 2i and 2s. The first one is a totally flat surface, so the model looks as far as possible to check if there are obstacles. We have a similar situation in the last two patches, a surface with a some noise and a big bump respectively, where the network verifies those spots. So, rightly, the network analysis the first region of the patch that may contain an untraversable obstacle.

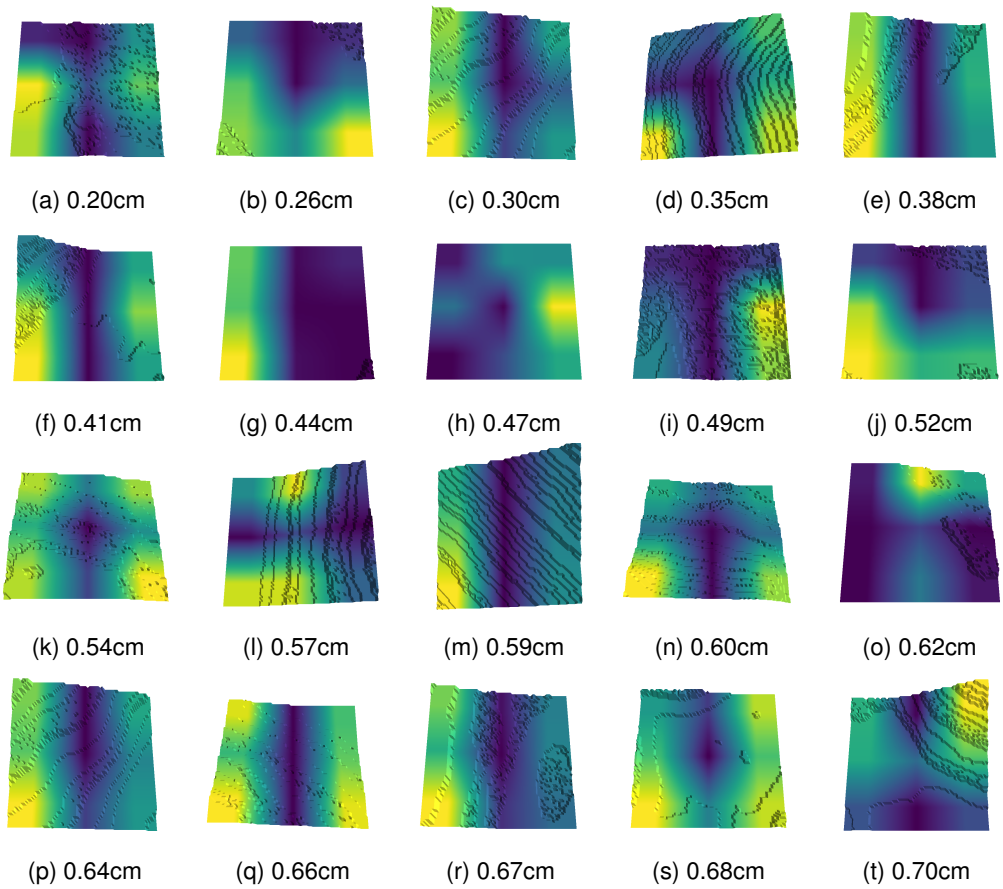

Figure 2. Traversable patches sampled from the test dataset correctly predicted by the model. We compute dthe Grad-CAM and applied as texture in the 3D render. The yellow highlights the ground region that contributed the most the model's prediction.

0.1.3 Non traversable patches

Not traversable patches correctly classified by the model. In this case, the region highlighted by the Grad-CAM more varied. Like in the previous section, in some patches, 3b, 3f, 3l, 3n, 3q, 3s, the non traversable features is directly under the robot. While, on other grounds with a atraversable left part but a big obstacle ahead, figures 3c, 3e, 3i, 3o, 3r, the model identified the huge obstacle as the main reason for their untraversability. There mixed cases, figures 3b, 3d, 3j, 3m. Other grounds are mostly uneven, 3d, 3k and 3l and the cam highlighted the biggest bumps. The last patch, 3h, is very interesting. There is a trail with a obstacle parallel to it. Perfectly, the network identified one part obstacle as responsabile for the prediction. Probably, those spot is the points where the robot hitted the obstacle while going forward.

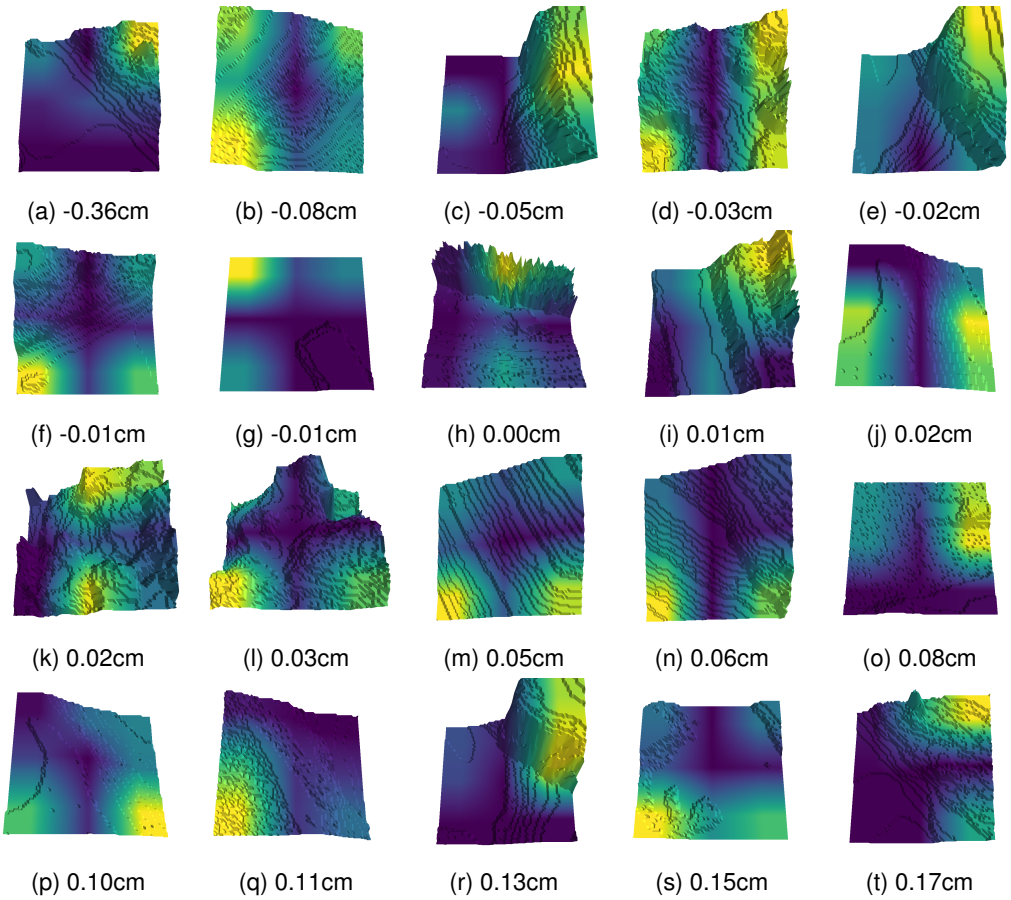

Figure 3. Not traversable patches sampled from the test dataset correctly predicted by the model. We compute dthe Grad-CAM and applied as texture in the 3D render. The yellow highlights the ground region that contributed the most the model's prediction.

0.1.4 False negative patches

Not traversable inputs predicted as traversable by the model. Some of the grounds are similar to ones from the last section. Figure 3b is almost identical to 4d. In this case, the model must have misclassified the first part of the map thinking it could traverse till the two bumps in the end. If so, the robot should have been able to move for more than twenty centimeters. There are two grounds with a trail, 4n, 4t. In both cases, the model looked at the initial position of the robot, left, and the final part of the patch to understand if the robot fits on the trail. Moreover, there are different slopes in which Grad-CAM highlighted the region under the robot, 4c, 4d, 4e, 4g, 4h, 4s. Those regions may be hard to estimate when the sizes of the obstacles are closes to the edge cases. The model may overestimate when the size of the obstacle is just a few centimeters more than the real traversable one. Figure 4k has a small step on the right region, correctly highlighted by Grad-CAM. Other patches have obstacles close to the end, 4j, 4o, 4p, 4q, 4r. ONe patch clearly showed the model confusion, figure 4m. Even if the obstacle was ahead, the model discriminated and an empty spot on the top left. in general, most of the region of interested on those patches are located on the left,

close to the position of the robot's leg. Correctly, even if the prediction is wrong, the model looks at the first region of the surface, located near the legs, that can effect traversability. This shown the correct behavior even when misclassifying the inputs, meaning that the network is always looking in the correct spot.

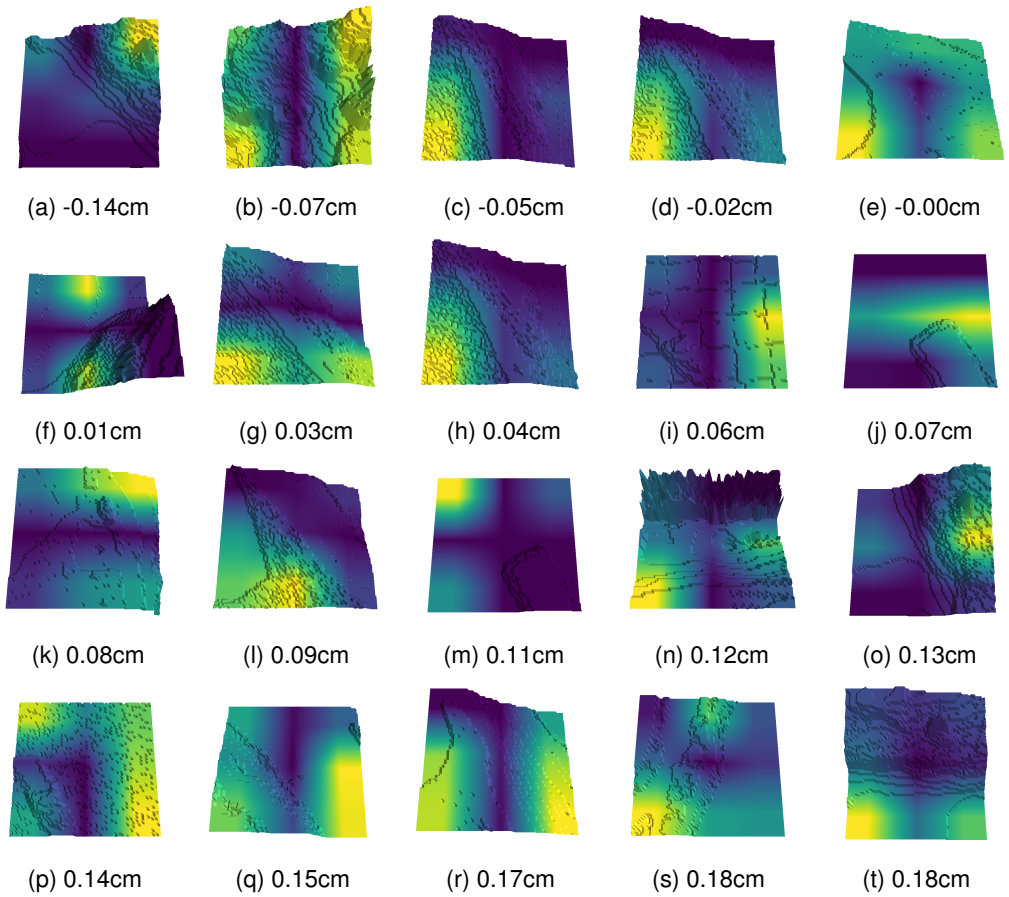

Figure 4. Not traversable patches predicted as traversable sampled from the test dataset. We compute dthe Grad-CAM and applied as texture in the 3D render. The yellow highlights the ground region that contributed the most the model's prediction.

0.1.5 False positive patches

False positive patches are the most interesting inputs. They are traversable patches classified as not. By looking at the features space, **??**, we can noticed how the patches are mostly located close to the non traversable features. This explained why most of them are very close the surfaces presented before in figure 3. Those samples included different types of terrains, some with obstacles ahead 5b, 5c, 5d, 5g, 5h, 5k, 5m, 5n, 5q, 5s, 5t, slopes 5i, 5j and mixed grounds, 5a, 5e, 5l, 5o, 5p, 5r. In each case, the model identify meaningful features that can effect traversability. In the slopes, 5i and **??**, the first part of the surface is highlighted. Similar to the traversable patches, the model tried to evaluated if the rear legs was able to move. This is also true for 5f. The only patch that consufed

the model is the first one, 5r, where it wrongly discriminated the flat region and not the obstacle. In all the others inputs, the network identifed importat region of the map that realistically could have caused the predicted not traversability.

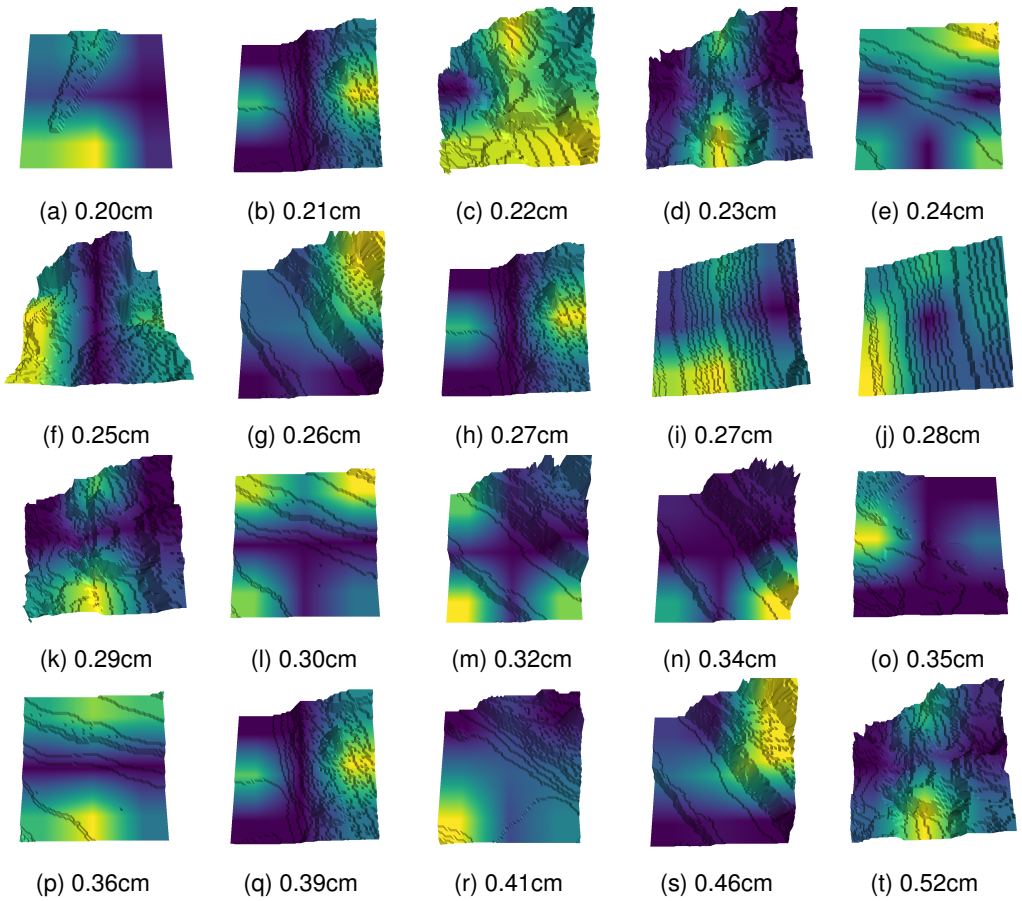

Figure 5. Traversable patches predicted as not traversable sampled from the test dataset. We compute dthe Grad-CAM and applied as texture in the 3D render. The yellow highlights the ground region that contributed the most the model's prediction.