Chapter 1

Results

In the section, we describe and evaluate the model's results. We first show the model selection process. Then we quantitatively evaluate the classifier and regressor performance on different numerical scores. Later, we compare the two approaches and select the classifier. Finally, we qualitatively evaluate the best model by predicting traversability in real-world terrains.

1.1 Experiment setup

We run all the experiment on a Ubuntu 18.10 work station equipped with a Ryzen 2700x, a powerful CPU with 8 cores and 16 threads, and a NVIDIA 1080 GPU with 8GB of dedicated RAM.

1.2 Dataset

To perform classification, we selected a threshold of 0.2m, according to the process explained in section **??**, on a time window, Δt, of two seconds to label the patches, meaning that a patch with an advancement less than 20 centimeters are labeled as *no traversable* and vice-versa. For the regression thanks, we did not label the patch and directly regress on the raw advancement.

Initially, to train the models we first used Standard Gradient Descent with momentum set to 0.95, weight decay to $1e-4$ and an initial learning rate of $1e-3$ as was originally proposed in He et al. **?**. However, we later switched Leslie Smith's 1cycle policy **?** that allowed us to train the network faster and with higher accuracy. The average training time is $\approx$ 10 minutes. We minimized the Binary Cross Entropy (BCE) for the classifier and the Mean Square Error (MSE) for the regressor.

1.2.1 Experimental validation

We selected as *validation* set ten percent of the training data. Since we stored each run of Krock as a *.csv* file, validation and train set do not overlap. We also assessed the model's performance on the *arck rocks* map, a 10×10m terrain with a big rocky bump in the middle. The test set is composed entirely by the quarry map **??**, a real-world terrain. Table **??** describes in detail the configuration used in to generate all datasets.

add arck rocks

1.2.2 Metrics

Classification: To evaluate the model's classification performance we used two metrics: *accuracy* and *AUC-ROC Curve*. Accuracy scores the number of correct predictions made by the network while the AUC-ROC Curve represents degree or measure of separability, informally it tells how much model is capable of distinguishing between classes. For each experiment, we selected the model with the higher AUC-ROC Curve during training to be evaluated on the test set.

Regression: We used the Mean Square Error to evaluate the model's performance.

1.3 Quantitative results

1.3.1 Model selection

We compared two different MicroRenset **??**, MicroRenset-3x3-SE and MicroRenset-7x7-SE with the initial channel dimension of 16 and the Vanilla CNN **??** proposed in the original work **?**. Ideally, the second MicroResnet model should be able to extrapolate more features from the patch thanks to the big kernel size. We evaluated those models using a time window of two seconds, a threshold of 20cm and we applied data augmentation as we described in section **??**. We run five experiments per architecture and we selected the best performing network. Table 1.1 shows the results for each architecture.

		Vanilla	MicroResnetSE	
			3×3 stride 1	7×7 stride 2
AUC	Top	0.892	0.888	**0.896**
	Mean	**0.890**	0.883	0.888
Params		974,351	313,642	314,282

Table 1.1. Model comparison on the test set.

Based on this data, we selected MicroResNet-7x7 with squeeze and excitation. The model has a first convolution's kernel size of 7×7 with a stride of 2. This network has one-third of the parameter of the original model proposed by Chavez-Garcia et all **?** making faster to train and lighter to deploy.

As proof of work, we also train the best network architecture, MicroResnetSE with a first convolution's kernel size of 7×7 and stride$= 2$ **??**, with and without the squeeze and excitation operator. Table 1.2 confirms the squeeze operator boosted performance while adding just a bunch of parameters.

1.4 Final results

1.4.1 Classification

Table 1.3 table shows in deep the score of the best network for each dataset. We obtained 95% accuracy and 0.96 AUC on the validation set and 88% accuracy and 0.89 AUC on the test set.

	MicroResnet7 $\times$ 7	MicroResnet7 $\times$ 7-SE	Improvement
Top	0.875	**0.896**	$+0.021$
Mean	0.867	**0.888**	$+0.021$

Table 1.2. AUC top value and mean value for MicroResnet with a fist convolution of 7×7 and stride $= 2$ with and without the SE module. The SE module boosts the performance by 0.021 while keeping the number of parameters almost identical.

Dataset			MicroResNet		Size(m)	Resolution(cm/px)
Type	Name	Samples	ACC	AUC		
Synthetic	Training	429312	-	-	10×10	2
	Validation	44032	95.2 %	0.961	10×10	2
	Arc Rocks	37273	85.5 %	0.888	10×10	2
Real evaluation	Quarry	36224	88.2 %	0.896	32×32	2

Table 1.3. Classification results on different datasets.

1.4.2 Regression

Table 1.4 table shows in deep the score of the best network for each dataset. The regressor is more

Dataset			MicroResNet	Size	Resolution(cm/px)
Type	Name	Samples	MSE		
Synthetic	Training	-	-	10×10	2
	Validation	44032	0.006	10×10	2
	Arc Rocks	37273	0.020	10×10	2
Real evaluation	Quarry	36224	0.022	32×32	2

Table 1.4. Classification results on different datasets.

flexible. We can use the same model in different situations by just see if the predicted advancement is greater or lower than a threshold. However, we always need to fix a threshold to classify a patch and once the value is fixed, the classifier drastically outperforms the regressor. The classifier is able to generalize better thanks to data augmentation that cannot be applied when predicting directly the advancement since it will change its value. Intuitively, if we transform an untraversable surface by applying the data argumentation techniques illustrated in section **??** we do not change its label. While it will change the actual advancement. Table 1.5 shows that the accuracy of a classifier is greater than a regressor using the same model, MicroResNet7x7-SE, and a fixed threshold of twenty centimeters. For this reason, all the next evaluation have been done using the best performing

classifier and not the regressor.

	Regression	Classification
Top	72.8%	**88.2%**
Mean	73.6%	**87.8%**

Table 1.5. Regression and classification accuracy for the same model, MicroResNet7x7-SE, once a threshold of 20cm is fixed. The classifier outperforms the regressor by 15%.

1.5 Qualitative results

We qualitative evaluate the classifier predictions by plotting the traversability probability on different maps in 3D. We used a sliding window to extract the patches from the tested maps to collect the model's predictions. Then, we created a texture based on the traversable probability, we have colored it using a colormap and finally applied on the 3D model of each map. We evaluated four rotations, from bottom to top, top to bottom, left to right and right to left since those are the most human understandable. By using those angles, we can optimize the cropping process since to avoid rotate each patch based on the Krock's head position. Instead, we can just rotate the entire map beforehand.

1.5.1 Quarry

The first map we evaluate is Quarry, this map is 32×32m long and has a maximum height of 10m. We can use some of the terrain interesting features, such as are three bis slopes and the rocky ground on the top, to evaluate the model. For instance, w expect the trail on the slopes to be traversable at almost any rotation, especially from left to right and vice-versa. The top part should be hard to traverse in almost any case. Figure 1.1 shows the traversability probability directly on the map.

some where place the colormap bar

Correctly, the lower part of the map, composed by flat regions, was labeled with high confidence as traversable in all rotations. On the other hand, the traversability of the slopes and the bumps on the top region depends on the robot orientation.

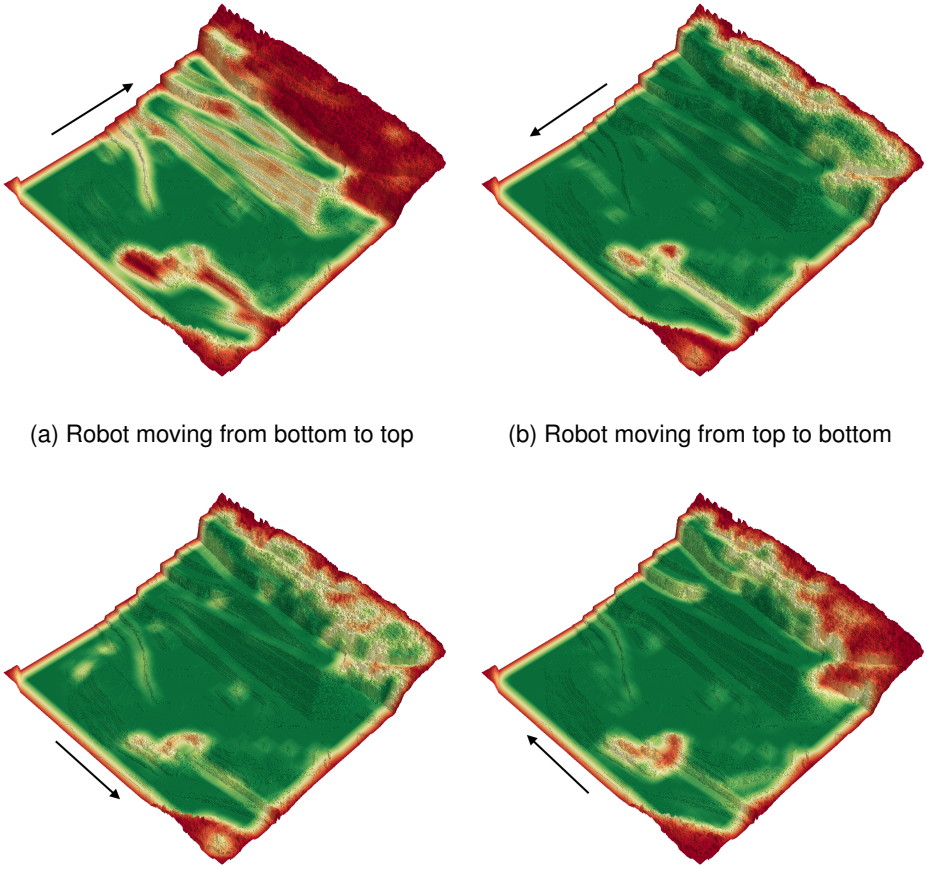
1.5.2 Bars

Bars is a map composed of different heights wall, thus we expected similar probabilities with different orientation. Figure **??** shows the traversability probabilities on the terrain.

Definitely, this was a hard map for the robot due to the high number of not traversable walls. Interesting, we can identify a tunnel near the bottom center of the maps that shows how the model correctly label those patches depending on the orientation. Figure 1.3 highlights this detail.

1.5.3 Small village

We applied the same procedure to also a 10×10m small village map. Figure 1.4 describes its traversability for four different rotations.



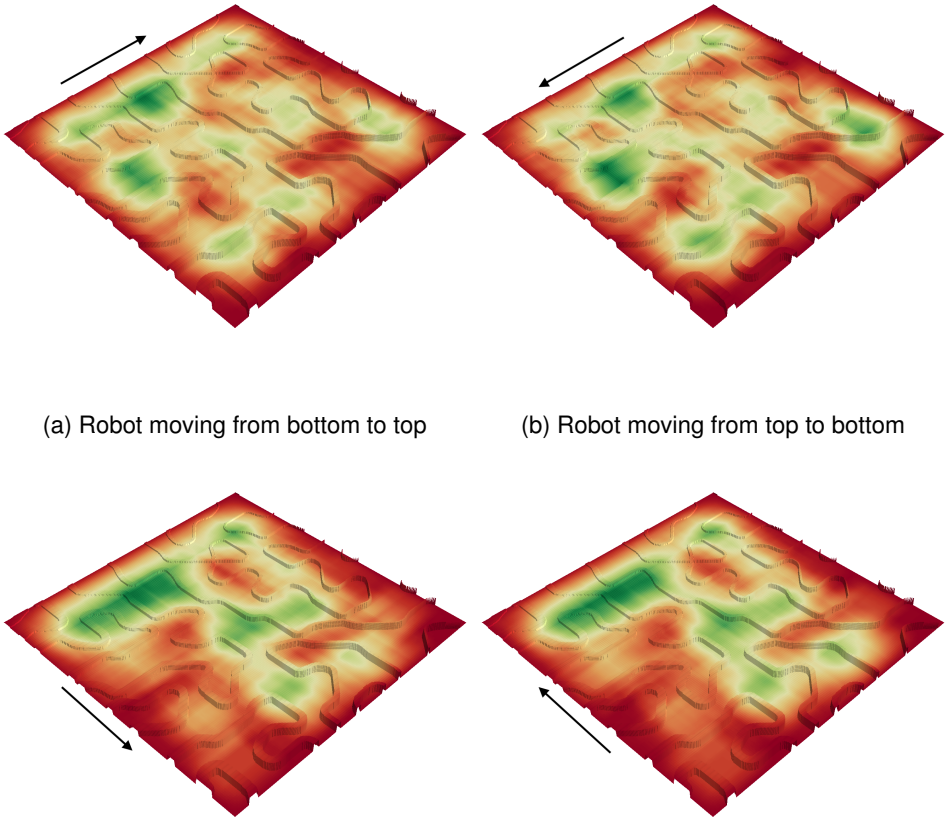
(a) Robot moving from bottom to top

(b) Robot moving from top to bottom

(c) Robot moving from left to right

(d) Robot moving from right to left

Figure 1.1. Traversability probability on the Quarry map, 32×32m, for different Krock's rotation. The values are obtained by sliding a window on the map to create the patches to predict their traversability.



(a) Robot moving from bottom to top

(b) Robot moving from top to bottom

(c) Robot moving from left to right

(d) Robot moving from right to left

Figure 1.2. Traversability probability on the bars map, a 10×10m , for different Krock's rotation. The values are obtained by sliding a window on the map to create the patches to predict their traversability.

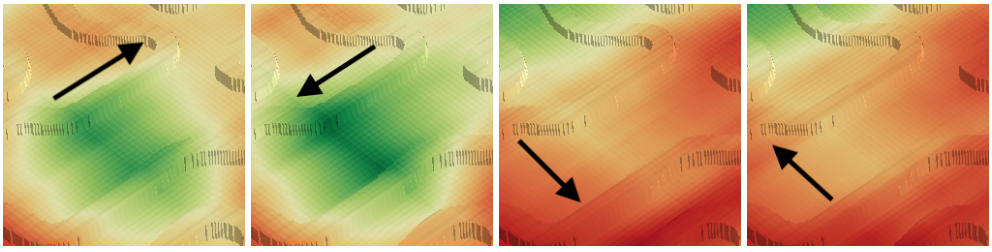
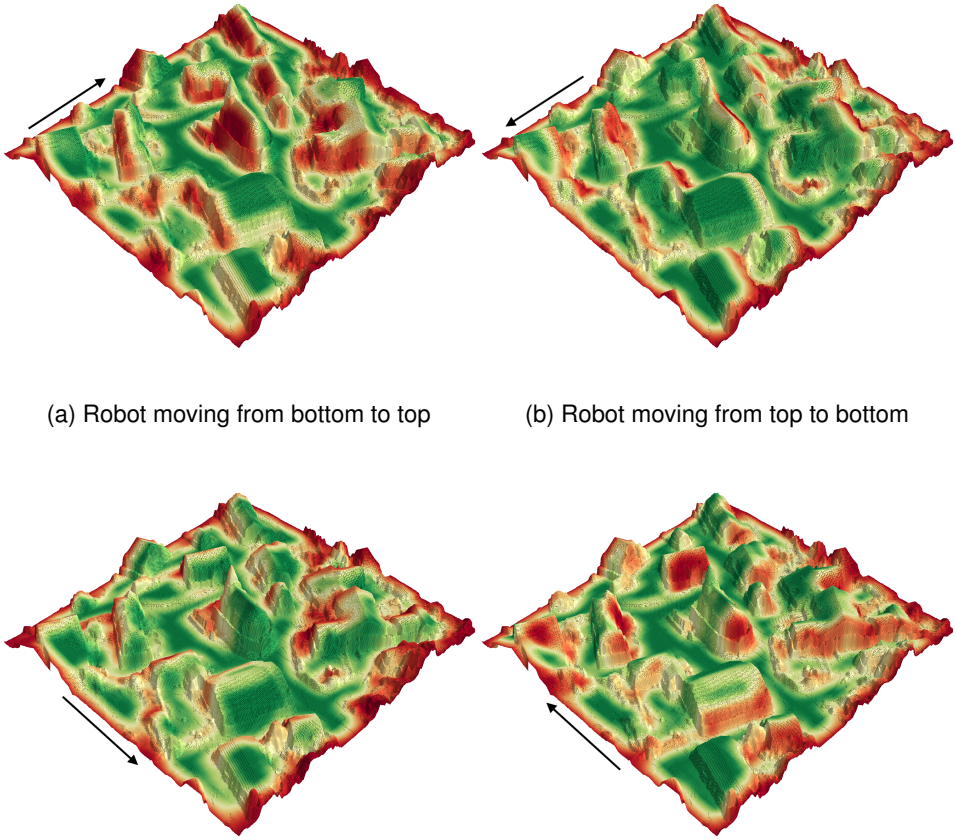


Figure 1.3. Detail of a region in the bars map where there are two walls forming a tunnel. Correctly, when the robot is travel following the trail the region is label as traversable.



(a) Robot moving from bottom to top

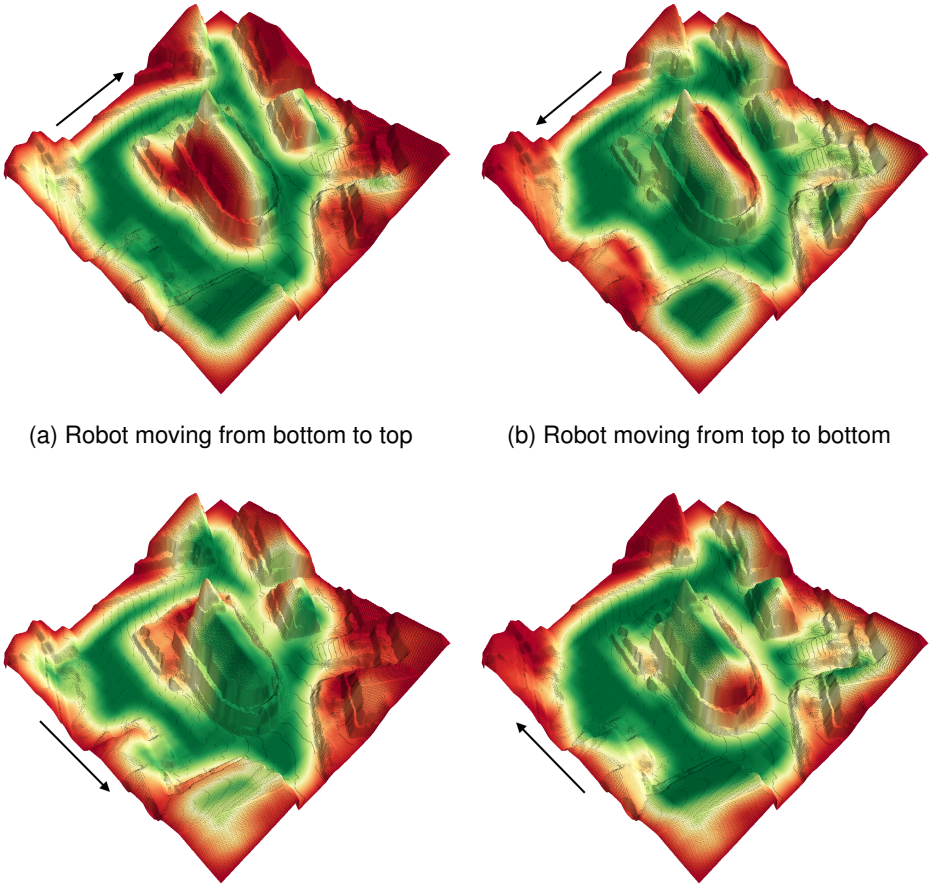
(b) Robot moving from top to bottom

(c) Robot moving from left to right

(d) Robot moving from right to left

Figure 1.4. Traversability probability on the map of a small village for different Krock's rotation. The surface covers 30×30m and has a maximum height of 10m. The values are obtained by sliding a window on the map to create the patches to predict their traversability.

All the streets were label as traversable, while the church's roof traversability depend on the orientation. For example, most steep roofs are not traversable when walking uphill. On the other hand, if krock walks side by side they can be traversed. Figure 1.5 shows this behavior on the church's roof.



(a) Robot moving from bottom to top

(b) Robot moving from top to bottom

(c) Robot moving from left to right

(d) Robot moving from right to left

Figure 1.5. Detail of the church in the small village map for different Krock's rotation. All the fours images are correctly labeled accordingly to the robot orientation. In the first two images only downhill part of the roof is traversable. While in the last two the robot is able to travel till the end.