TERRAIN CLASSIFICATION WITH AN OMNI-DIRECTIONAL CAMERA USING CONVOLUTIONAL NEURAL NETWORKS

Yuto Suebe[1], Kenji Nagaoka[1], Kazuya Yoshida[1]

[1]*Tohoku University, 6-6-01 Aoba, Aramaki, Aoba-ku, Sendai, Miyagi, 980-8579, Japan,*
E-mail: {suebe, nagaoka, yoshida}@astro.mech.tohoku.ac.jp

ABSTRACT

In the case of autonomous lunar and planetary exploration, terrain classification is necessary. In recent years, research has been focused on terrain classification using convolutional neural networks (CNNs). However, these studies did not consider the method of updating the terrain classifier when new data is obtained. In this paper, two updating methods for CNNs are applied and evaluated. One fine-tunes the terrain classifier with all the data obtained until the training and the other trains it with only the new data using elastic weight consolidation [10]. Both the methods enabled the terrain classifier to classify the terrain of a new environment while retaining the ability to recognize the terrain types of the previous environment.

1 INTRODUCTION

In the case of lunar and planetary missions, because of communication delays between the on-site rovers and a ground station on Earth, teleoperation is inefficient. Therefore, lunar and planetary rovers are required to have an autonomous exploration ability. In order to perform exploration, firstly, rovers are required to move to an explored point. Therefore, rovers are required to have autonomous mobility to accomplish exploration autonomously. Autonomous mobility can be realized in three steps. The first step involves understanding the environment in which robots recognize the location of obstacles, shape of the ground, properties of the ground, or where the robots are located. Then, a path is generated based on the information obtained from the first step. Finally, the robots move along the generated path. Therefore, environmental understanding is an essential element of autonomous locomotion. Terrain classification is an important technique for environmental understanding as the surface of celestial bodies is covered with sand or rocks, and the mobility and optimal control performances depend on what surface the rovers are moving.

Thus far, several studies have been conducted on terrain classification[1][2][3][4][5][6]. A support vector machine with color feature or vibration feature has been used in [1][5] and classifies a Mars-analogous terrain into three types of terrain in [1]. In [2], a vision-based terrain classifier using convolutional neural networks (CNNs) was employed for pixelwise prediction by using images as inputs. This classifier successfully classified Mars images from a Mars rover into six classes based on the size of the rocks in the images. In [6], lidar data had been processed using a 3D CNN for finding the landing site of helicopters. A CNN has also been employed as a terrain classifier for pixelwise prediction by combining point cloud data with an RGB image to improve the performance of terrain classification in [3]. However, these studies assume that all the necessary data is available when the terrain classifier is trained. In the case of an actual mission, it is difficult to obtain all the data before landing at the site and obtaining new environmental images as rovers travel over the surface of the celestial bodies. Therefore, a method of updating the trained terrain classifier should be considered. In this paper, we present two methods of updating the terrain classifier and evaluate them using image data.

2 METHODOLOGY OF TERRAIN CLASSIFICATION

2.1 Structure of Terrain Classifier

We constructed a vision-based terrain classifier using CNNs. The CNN automatically extracts features from the inputs and uses them to represent the outputs while other types of machine learning requires handcrafted features. As it is difficult to determine significant features before encountering the site, the application of conventional machine learning that requires handcrafted features is difficult. In contrast, as CNNs do not require handcrafted features, the terrain classifier can adapt to unknown environments. As the structure of the CNN, we employed the pyramid scene parsing network (PSPNet) [8], which is used for semantic segmentation. The PSPNet comprises a CNN that extracts features from the input, a pyramid pooling module with four different sizes of pooling layers in parallel, and convolution layers.

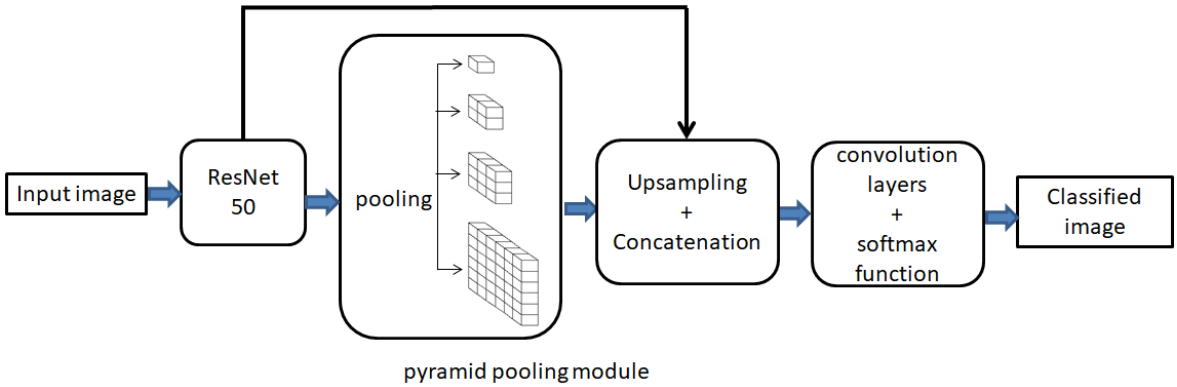


Figure 1: Structure of terrain classifier

In our terrain classifier, as a feature extractor in the PSPNet, a 50-layer residual network (ResNet) [9] is used. The softmax function is applied to the outputs from the last layer in order to predict the probability of the object class of each pixel. The overall structure is shown in Figure 1.

2.2 Updating Methods for the Terrain Classifier

We compared two updating methods for the terrain classier. First, fine-tuning the terrain classifier with all the data obtained until the training step. In this method, each time new data is obtained the terrain classifier is trained with all the data while using the optimal parameters of last training as the initial parameters.

Secondly, updating the CNN based on the importance of each parameter in the CNN by using elastic weight consolidation (EWC) [10]. The EWC measures the importance of the parameters in the CNN by computing the diagonal components of the Fisher information matrix. When training the terrain classifier, parameters that are not important for the previous task are preferentially adjusted by adding a penalty term to the loss function of the previous task, which measures the difference between the desired outputs and actual outputs, as follows

$$L(x;\theta) = L_{new}(x;\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta^*_{prev,i})^2 \quad (1)$$

where x is the input data, θ represents all the parameters in the CNN, $L(x;\theta)$ is the new loss function, $L_{new}(x;\theta)$ is the loss function of the new task, λ represents how important the previous task is, F_i is the i th element of the diagonal of the Fisher information matrix, θ_i is the i th parameter, and $\theta^*_{prev,i}$ is the i th optimal parameter that achieved

Figure 2: Testbed

the best performance in the previous task. In this research, the Fisher information matrix is calculated by averaging the gradient of the loss function of the previous task using 100 samples of the previous task with an optimal parameter as follows.

$$F_i = \frac{1}{100} \sum_n^{100} \left(\frac{\partial L_{prev}(x;\theta)}{\partial \theta_i}\bigg|_{x=x_n} \right)^2 \quad (2)$$

where $L_{prev}(x;\theta)$ is the loss function of the previous task and x_n is the input data of the n th sample of the previous task.

3 EXPERIMENT AND EVALUATION

3.1 Dataset

In order to obtain the image data for constructing a dataset, were used a rover testbed (Figure 2) developed in our lab [7], which has an omni-directional camera on top of it that provides

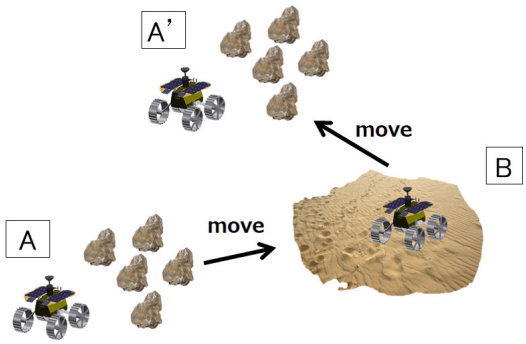
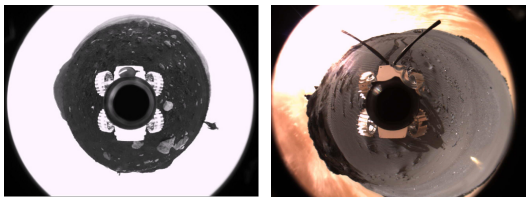


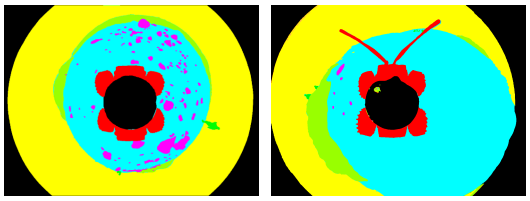
Figure 3: Mission scenario overview



(a) Environment A (b) Environment B

Figure 4: Sample images of two environments

a 360 degree field of view image. The omni-directional camera is useful for path-planning because it provides the direction in which the rover should move once the terrain classification is completed. In this study, we assume the scenario wherein the rover starts its exploration at environment A and then moves to environment B and then to environment A', which is similar to environment A, as shown in Figure 3. At environments A and B, the rover obtains the image data and the terrain classifier is trained with this data. We prepared datasets for the environments A and B. Example images of the two environments are shown in Figures 4(a) and 4(b). Environment A has numerous rocks on the ground (dataset A) and environment B has a few small rocks (dataset B). The datasets consist of image data and the corresponding label data that has been labeled by a human. An example labeled image is shown in Figure 5(a) and 5(b). The label data comprises seven classes: sky, rover, ground, rock, person, hill, and null. The number of training data is 40 for environment A and 8 for environment B. These data are augmented by rotating the image data by 6 degrees from 0 to 360 degrees because the images obtained using an omni-directional camera has rotational symmetry. The number of testing data is 7 and 4, respectively.



(a) Environment A (b) Environment B

Figure 5: Sample label images of two environments

3.2 Training Condition

In this paper, we conducted the following four trainings.

- Training 1: Train terrain classifier with dataset A. The number of iterations is 12000. The part of ResNet is initialized using the parameters obtained from the training on a task of image classification and the other parameters are randomly initialized.
- Training 2: The terrain classifier trained in training 1 by fine-tuning with dataset B is updated. The number of iterations is 4800.
- Training 3: The terrain classifier trained in training 1 by fine-tuning with datasets A and B is updated. The number of iterations is 4800.
- Training 4: The terrain classifier trained in training 1 using EWC with dataset B is updated. The number of iterations is 2400. λ was determined from the preliminary training and set as 10000000 for the EWC.

As a loss function for training 1, 2, and 3, we used cross entropy, which is defined as follows.

$$L(x;\theta) = \frac{1}{WH}\sum_{i=1}^{W}\sum_{j=1}^{H}\sum_{k=1}^{C} d_{ijk}\ln(y_{ijk}(x,\theta)) \quad (3)$$

where W is the height of the input image, H is the width of the input image, c is the number of the class, d_{ijk} is the target output that takes a value of 1 if the pixel (i, j) belongs to the class k, and $y_{ijk}(x,\theta)$ is the output of CNN. The loss function for training 4 is equation (1) wherein L_{new} is substituted by equation (3). In order to minimize this loss function, we used the Adam optimizer[11] with the hyperparameters shown in Table 1 through all training.

3.3 Result

For the purpose of evaluation, the below equation is used.

$$accuracy = \frac{NP_c}{NP} \times 100\ [\%] \tag{4}$$

where NP is the number of all the pixels, and NP_c is the number of the pixels predicted correctly. The per-class accuracy is also evaluated. The pixel accuracy is shown in Tables 2 and 3 and per-class accuracy is shown in Tables 4 and 5. In addition, examples of the segmented images are shown in Figures 6 and 7, where the input images are Figures 4(a) and 4(b) and the corresponding ground truth images are Figures 5(a) and 5(b).

3.3.1 Training 1

From Figure 6 and Table 2, the terrain types in environment A are successfully classified with an accuracy of 97.6% in training 1. In contrast, from Figure 7 and Table 3, it is observed that the terrain classifier is not applicable to environment B and the accuracy is 76.5 %. This is because only dataset A is used for training the terrain classifier and the information of environment B is not included.

3.3.2 Training 2

Although from Table 2, it is observed that the accuracy does not decrease much on fine-tuning the terrain classifier with dataset B, from Figure 6, it is observed that the terrain classifier lost the ability to recognize rocks on the ground. This can be also observed by comparing Tables 4 and 5. This is because the knowledge regarding environment A is lost while fine-tuning the terrain classifier with dataset B.

3.3.3 Training 3

From Figures 6 and 7 and Tables 2 and 3, we can see that the terrain classifier updated by fine-tuning with all the data can classify terrain types of environment B with keeping the capability of terrain classification of environment A. This is because the information of both environments A and B is taken into account while training.

Table 1: Hyperparameters of Adam optimizer

learning rate	0.00001
momentum term $\beta 1$	0.9
momentum term $\beta 2$	0.999

3.3.4 Training 4

From Figure 7, it can be observed that the terrain classifier updated using EWC can classify the terrain of environment B. From Figure 6, although the terrain classifier lost the ability to recognize small rocks, which can be classified by using the terrain classifier trained through fine-tuning with all the data, it can still classify large rocks. As described in section 2.2, the use of EWC results in a restriction on the parameters when the terrain classifier is trained. Therefore, it is important to determine how much the restriction influences the training of the new data. According to Table 2, the accuracy of the result of training 4 is 94.4 % and reduces accuracy by only 1.0 % from the result of training 2. This result indicates that the restriction does not result in a profound difference.

3.3.5 Comparison between Fine-tuning with All the Data and EWC

From Tables 2 and 3 fine-tuning with all the data achieves better performance than updating by EWC. However, in fine-tuning with all the data, the amount of data and computational cost increases as rover moves. In contrast, Updating by EWC does not require to keep all the data but only new data. This keeps computational cost constant. So if the rover does not move so much and does not get a lot of data, fine-tuning method is applicable and better, however, if the rover travels long distance and obtains a lot of data, updating by EWC is better.

4 APPLICATION OF TERRAIN CLASSIFICATION

In this section, we present an application of the terrain classification, which is presented in section 3. For wheeled rovers, determining whether

Table 2: Pixel accuracy result for environment A

Training number	Percentage of correct pixels [%]
1	97.6
2	94.3
3	97.9
4	96.8

Table 3: Pixel accuracy result for environment B

Training number	Percentage of correct pixels [%]
1	76.5
2	95.4
3	94.6
4	94.4

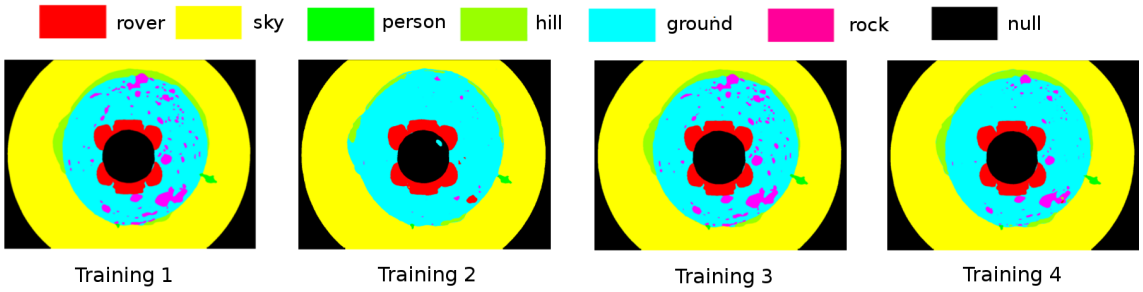


Figure 6: Segmented result for environment A (Input image:Figure 4(a))

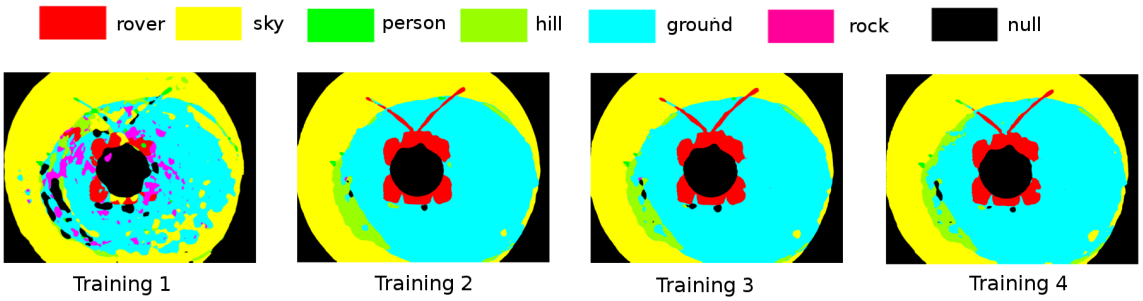


Figure 7: Segmented result for environment B (Input image:Figure 4(b))

the surface is smooth or abundant with obstacles is important for obtaining efficient and safe movement. In order to determine the amount of rocks, we divide the image obtained from the omni-directional camera radially by 30 degrees from 0 to 360 degrees as the omni-directional camera has a 360-degree field of view. We then calculate how rocky the surface is using the following equation.

$$ratio = \frac{NP_{Rock}}{NP_{Rock} + NP_{Ground}} \tag{5}$$

where NP_{Rock} is the number of pixels predicted as rocks, and NP_{Ground} is the number of pixels predicted as part of the ground.

We applied this method to the segmented image. The result is shown in Figures 8 and 9. From Figure 8, we can say that the forward area of the rover is abundant with rocks and the rover should avoid that area intuitively. The resulting image shows that the forward area is rocky terrain and this result corresponds to the human intuition. Hence, the rover can determine the direction to which the rover should move by using resulting image to move safely. From Figure 9, we can say same thing. Although the accuracy of the classification of the rock class is relatively low as compared to the other classes because the small rocks or gravels cannot be recognized, the terrain classifier can recognize large rocks that the rover should avoid. Hence, this method will be useful for path planning of the rover missions.

5 CONCLUSION

In this paper, we presented two methods for updating the terrain classifier constructed using CNNs. One method comprises fine-tuning with all the data obtained until the training step. The other method comprises updating using EWC with only the new data. We evaluated these two updating methods using image data obtained from an omni-directional camera. The results showed that the fine-tuning method provides a better performance than that obtained while updating using EWC. In terms of the computational cost, although the fine-tuning method increases the computational cost, updating using EWC does not increase the computational cost because the method requires only the new data. In addition, we presented an application of terrain classification to determine which direction the rover should move based on the omni-directional camera image.

References

[1] K. Otsu, M. Ono, T. J. Fuchs, I. Baldwin, and T. Kubota, "Autonomous terrain classification with co- and self-training approach," IEEE Robotics and Automation Letters, Vol. 1, no. 2, pp. 814-819, 2016.

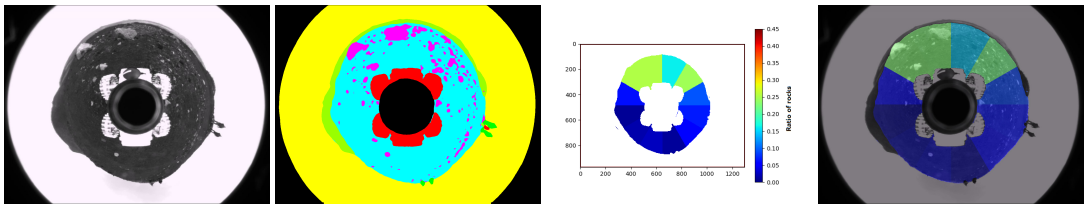
[2] B. Rothrock, J. Papon, R. Kennedy, M. Ono, and M. Heverly, "SPOC: Deep learning-

Table 4: Per-class result for environment A

Training	Null	Sky	Ground	Rover	Rock	Hill	Person
Training 1	98.9%	99.4%	96.4%	96.5%	77.9%	87.6%	85.0%
Training 2	97.6%	98.4%	98.5%	97.8%	5.0%	45.7%	59.7%
Training 3	99.1%	99.5%	96.8%	97.1%	75.3%	91.3%	87.9%
Training 4	98.1%	99.0%	98.5%	97.7%	42.1%	83.4%	81.0%

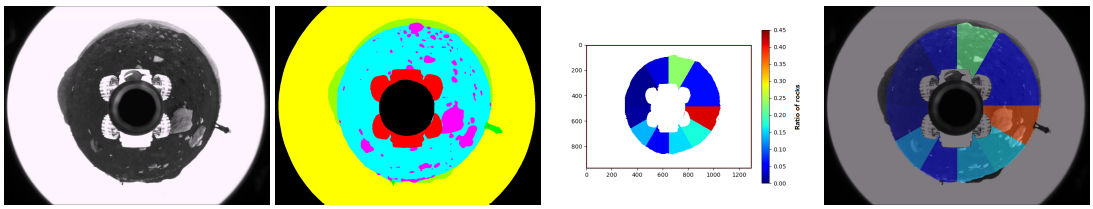
Table 5: Per-class result for environment B

Training	Null	Sky	Ground	Rover	Rock	Hill	Person
Training 1	99.7%	89.9%	68.1%	45.9%	48.9%	10.3%	42.5%
Training 2	97.4%	97.8%	97.1%	93.7%	4.4%	66.0%	63.5%
Training 3	97.4%	97.9%	97.0%	92.7%	12.1%	51.4%	62.3%
Training 4	98.4%	97.0%	96.7%	87.8%	10.6%	57.1%	52.6%



(a) Input image (b) Output image (c) Resulting image of terrain classification (d) Resulting image overlayed on input image

Figure 8: Terrain classification result 1



(a) Input image (b) Output image (c) Resulting image of terrain classification (d) Resulting image overlayed on input image

Figure 9: Terrain classification result 2

based terrain classification for Mars rover missions," AIAA SPACE, p. 5539, 2016.

[3] D. K. Kim, D.Maturana, M. Uenoyama, S. Scherer. "Season-invariant semantic segmentation with a deep multimodal network," Proceedings of the Field and Service Robotics, pp. 255-270, Springer, Cham, 2018.

[4] J. F. Lalonde, N. Vandapel, D. F. Huber, and M. Hebert, "Natural terrain classification using threedimensional ladar data for ground robot mobility," Journal of field robotics Vol. 23, No. 10, pp. 839-861, 2006.

[5] C. Weiss, H. Frohlich, A. Zell, "Vibration-based terrain classification using support vector machines," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4429-4434, 2006.

[6] D. Maturana, S. Scherer, "3d convolutional neural networks for landing zone detection from lidar," Proceedings of the IEEE International Conference on Robotics and Automation, pp. 3471-3478, 2015.

[7] K. Yoshida, N. Britton, and J. Walker, "Development and Field Testing of Moonraker, a Four-Wheel Rover in Minimal Design," Proceedings of the 12th International Symposium on Artificial Intelligence, Robotics and Automation in Space, 2013.

[8] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," ,Proceedings of the IEEE Conference on Computer

Vision and Pattern Recognition, pp. 2881-2890, 2017.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770-778, 2016.

[10] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," Proceedings of the National Academy of Sciences, Vol. 114, No. 13, pp. 3521-3526, National Academy of Sciences, 2017.

[11] P. D. Kingma and J. Ba, "Adam: A method for stochastic optimization," Proceedings of the International Conference on Learning Representation, 2015.