HIGH SCHOOL EDITION
Journal of Student Research
Volume 10 Issue 3 (2021)

Classification Using 3D Point Cloud and 2D Image on Abstract Objects

Mark Yang[1] and Guillermo Goldsztein[#]

[1]The Quarry Lane School, Dublin, CA, USA
[#]Advisor

ABSTRACT

While classification using machine learning is exceptionally successful with 2D images, it is more challenging to classify 3D objects. However, 3D objects classification is critical because of its application in autonomous vehicles and robotics. This paper compared neural networks with similar structures using 3D point clouds and 2D images on the same objects. We also generated objects with abstract design and input them into the neural networks we created. We find clear disadvantages with classifying abstract objects compared to ordinary objects for both neural networks. We believe having contextual information will help to address this problem. We also observed that the neural network based on images performs worse than that based on point clouds. However, image based classification takes less time to train compared to point cloud based classification.

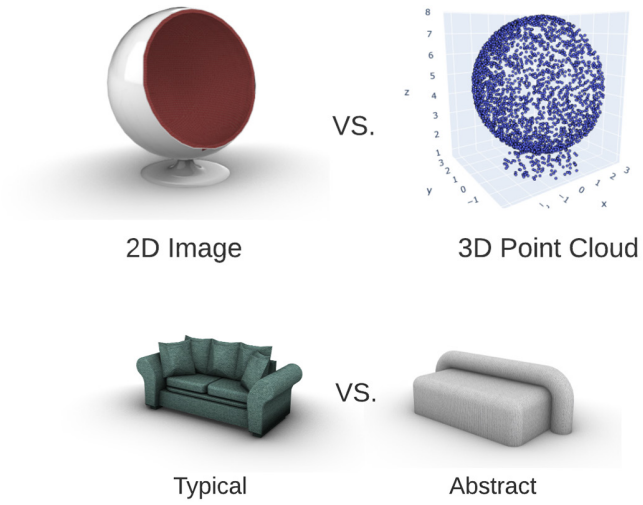

Figure 1. 2D Image representation and 3D Point Cloud representation of the same Eero Aarno chair. Renders of a typical sofa and an abstract designed sofa.

Introduction

Recent development in LiDAR, a type of sensor that emits light to detect objects and measure distances for autonomous vehicles and robots (Wandinger, 2005), led to much effort researching 3D geometric data. One of the research areas is classification based on 3D data. Classification is crucial for achieving autonomy in cars and robots because autonomous vehicles and robots have to distinguish objects to understand what they can or cannot do. The alternative

to accomplishing this task is using multi-view cameras. Though both methods have been in research for many years, and researchers even developed ways to combine these methods (Zhang et al., 2014), few papers have compared the difference between classification using 3D geometric data from a LiDAR and 2D image data from a camera. Furthermore, most of the research on accuracy drop for classification is on the effect of losing data (Xiao et al., 2015) or environment (Filgueira et al., 2017) but not much on the effect of abstract designs, where the object's shape does not primarily relate to its intended function or purpose.

This paper aims to answer both of these questions: classification based on image vs. point cloud, and classification of abstract objects from training on typical objects (Figure 1). First, we find the performance difference between classification of neural networks based on 3D and 2D data on ModelNet10 (Wu et al., 2015). Comparing the performance of these neural networks is extremely important because high accuracy is required for real-world applications. Besides the performance, we also record the time it took for the neural network to learn from the training dataset to look at their scalability. Finally, we create CAD models based on modern designs that are in the categories of ModelNet10 dataset to analyze how well these neural networks perform on more abstract designed objects.

We hypothesize that classification based on 3D data will better distinguish objects since it records spatial information better than 2D images. This is because 3D data includes higher-dimensional geometric data than flat images that will help the neural network extract better features. We also hypothesize that classification based on 3D data will also be slower at training than the image-based classification because of the complexity of the data. 3D data is more complicated to format in a well-defined way than 2D data represented as a two-dimensional matrix.

Our work begins with generating point clouds and images from ModelNet10. Thus, we can ensure that the amount of input into our neural network and the number of data going into the training process are the same. Consequently, it allows us to control the data for a fair comparison between neural networks based on point clouds and images. Then we trained our neural networks on the point clouds and images. We recorded the time for each of them to run through their training process. Then we tested their accuracy on the testing data in ModelNet10. We also created 3D models ourselves, but with the object in each category slightly harder to recognize. Finally, we tested the neural networks on these 3D models and compared their performance.

We hope our work can help people understand the pros and cons of using different data for classification aiding autonomy. We also believe that through observing the behavior of the neural networks when classifying abstract objects, we can explore the limitations other than data augmentation in autonomous systems based on computer vision.

Method

Data

We used the ModelNet10 (Wu et al., 2015) dataset to train both of our neural networks. ModelNet10 contains a total of 4899 3D CAD models that are separated into ten categories, ranging from bathtub, table, monitor, chair, etc. Each one of the 3D CAD models is manually aligned and scaled to its actual size. ModelNet10 also divided the entire set into 3991 models for training and 908 models for testing. Our neural networks are trained entirely on the training set of ModelNet10.

Table 1. Number of the data used in classification based on images and point clouds.

	Amount of Training Data	Amount of Testing Data	Amount of Self-generated Data	Shape of the Input Data[1]
Image Based Classification	3991	908	20	(32, 32, 3)

Point Cloud Based Classification	3991	908	20	(1024, 3)

There are many existing ways of processing 3D geometric data for classification and segmentation, ranging from using 3D meshes (MeshNet (Feng et al., 2018)), 3D voxels (ORION (Sedaghat et al., 2017)), and 3D point clouds (PointNet (Qi et al., 2017)) to represent the 3D shapes. However, the point cloud representation is the most widely used for classifying and segmenting 3D data from all three of these options. Point clouds are high-quality 3D representations of the world that are generated by LiDAR sensors and depth cameras. Each point in the point cloud has its own (x,y,z) coordinate that view together resembles the 3D shape of the object (Rusu and Cousins, 2011). We convert the 3D models in the dataset into point clouds that contain 1024 points with trimesh library (Dawson-Haggerty et al., 2019). Next, we normalized the point clouds, so each point in the point cloud has three coordinates (x, y, z) between -1 and 1. To classify 2D data, we first rendered the 3D models and took one RGB image from the same perspective. We also scaled the renders down a fixed amount to contain the full models from different classes. Then the images are down scaled to 32x32 resolution to match the size of 3D data as recorded in Table 1.

The self-generated testing data we used in the paper are created with CAD software based on modern furniture designs, including the classic Eero Aarno chair. We used Rhino 6 to make the models and then convert them into the same format as the ModelNet10 dataset (.off) using MeshLab. The process of creating point clouds is the same as the training set. And for the 2D images, we rotate the rendered 3D model and take the RGB image from the same perspective as the images from the training set.

Both the 3D and 2D data have as little noise as possible since the purpose of this research is to find the difference between processing 3D data and 2D data from the same source. We also kept the number of the training data and testing data the same. We also made sure that the size of each sample in the training and testing set was consistent across the board.

Neural Network Model

Between all the existing methods of classifying 3D point clouds, we built our neural network based on one of the early architectures used to classify directly on point clouds: PointNet Vanilla. PointNet Vanilla was developed to solve the problem of the structure of point clouds. Since the point cloud data is orderless, meaning that roll permutation won't change what the point cloud represents, PointNet Vanilla must be prone to permutation invariant to classify directly on the point cloud (Qi et al., 2017). Therefore, PointNet Vanilla proposed a structure that contains a 1D max pooling layer in the middle of the network to extract features from all the points. We modify this structure to build our neural network, using four 1D convolutional layers to extend the dimension of each point from 3 to 1024 in the beginning. Then we use max pooling for each dimension of all the points, just like PointNet Vanilla. Finally, we end the neural network with fully connected layers with the softmax activation function for the last layer to predict between 10 classes as shown in Figure 2.

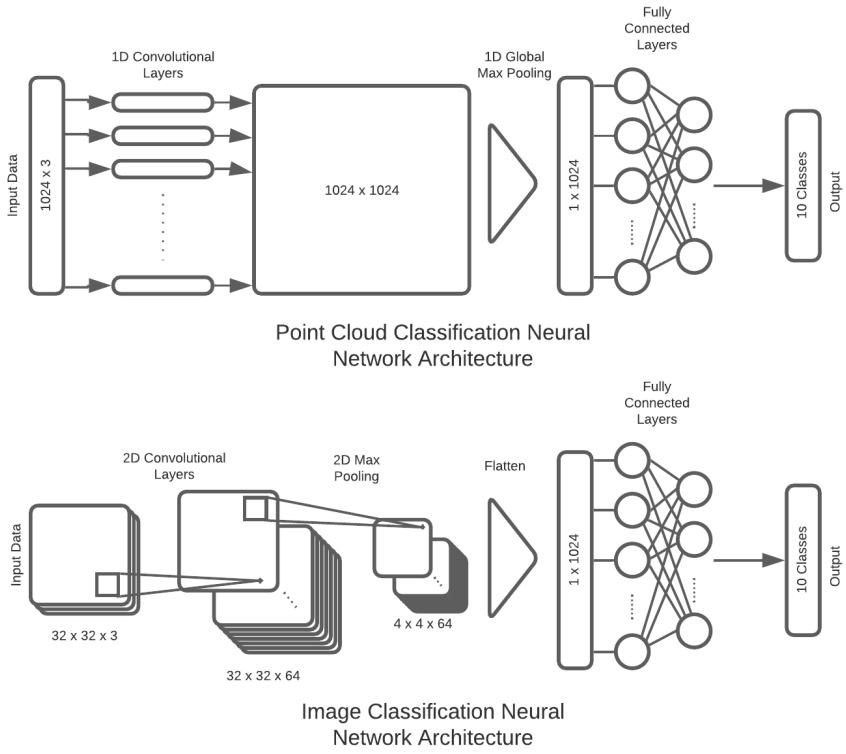

Figure 2. Architectures of Neural Network based on Point Cloud and Image. Note that they have similar hyper-parameters and structures.

We used 2D convolutional neural network to classify the 2D images. The building blocks of a convolutional neural network are convolutional layers. 2D convolutional layers designed for classifying images extract features with filters. These filters are two-dimensional matrices that show features after applying them to the original pictures. After the convolutional layers, the filters are flattened and connected to fully connected layers to learn these features and output. We designed our 2D convolutional neural network with the same number of convolutional layers as the 3D neural network. The convolutional layers connect to a 2D max pooling layer, followed by the same fully connected layers at the end of the network.

We created neural networks very similar to each other to control the effect of hyper-parameters on performance (Figure 2). The only difference between the networks is that they used the different convolutional layers and used them for various purposes. For example, the 3D classification neural network used the 1D convolutional layers to extend the dimension of the points for max pooling, and the 2D convolutional neural network used 2D convolutional layers to extract features from the images.

Results and Analysis

We observed that both point cloud based and image based neural networks performed well on the testing data included in the ModelNet10 dataset: 79.07% accuracy for classification based on point clouds and 81.5% accuracy for classification based on images. However, when the neural networks classified the generated data based on more abstracted designed furniture, the performance decreases by 35-45% accuracy in Figure 3. Furthermore, the performance of the 2D neural network is worse than the performance of the 3D neural network with a 10% gap on the new data, showing

that classification based on point cloud extracts more features that help classify these abstract design furniture compared to classification based on images.

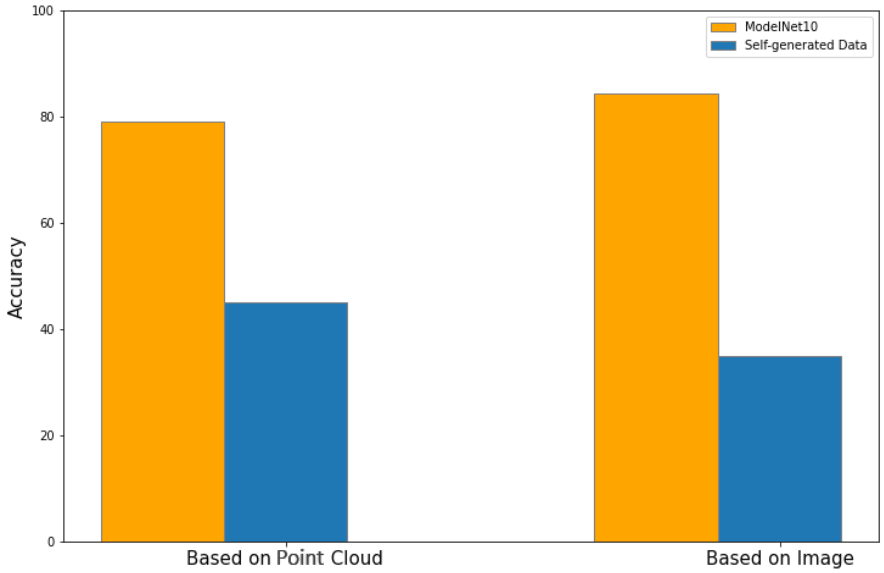

Figure 3. Comparison the accuracy between classification based on point clouds and images for ModelNet10 and self-generated datasets.

So why are the neural networks failing to classify the furniture that we generated? We looked into the feature maps of the 2D convolutional neural network that we created for processing 2D images. Figure 4 shows that the first convolutional layer is already failing to extract correct features to classify the Eero Aarnio ball chair as a chair. The ball-shaped body does not resemble the appearance of a chair; however, the feature of the Eero Aarnio ball chair is closely related to the feature extracted from a monitor image. The chair's foot is similar to a monitor stand. The 2D convolutional neural network gave us a prediction of 64.87% as a monitor. On the 3D end, we observed the same thing happening. The neural network processing 3D point cloud failed to distinguish this chair and gave it 76.57% as a monitor, an even higher percentage on the failed prediction.

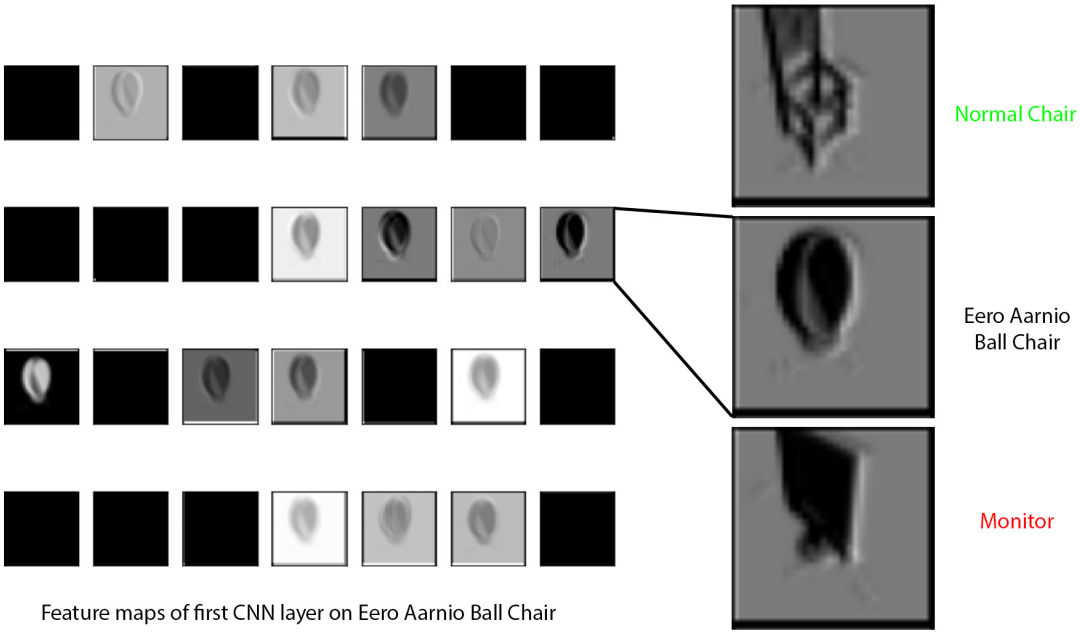

Figure 4. Feature comparison for image-based classification on Eero Aarnio chair. Note that the correct label is colored in green and incorrect label in red.

We observed the training process of each neural network on the same data set. The accuracy trend on the testing set of ModelNet10 is very similar between using 3D data and 2D data. However, these two neural networks converge differently. For example, CNN can come to the high prediction on the pictures with a low learning rate $\alpha = 0.00001$ in 20 epochs. On the other hand, the neural network based on the point cloud requires a higher learning rate $\alpha = 0.001$ to achieve the same performance with the same number of epochs. This results in the accuracy of the testing set going up and down from overshooting when making gradient descent. In addition, it shows that neural networks based on 3D point clouds are more challenging to converge than 2D images.

Time-wise, the image-based neural network wins with the total training time on the same amount of data being 3 times faster than the neural network trained on the point cloud using the same hardware. The time it took to preprocess the data for training also favors the 2D algorithm, averaging half of time compared to 3D. However, it took us around 10 min to generate photos from the 3D model format. To create point clouds from the same format, it took around 1 min.

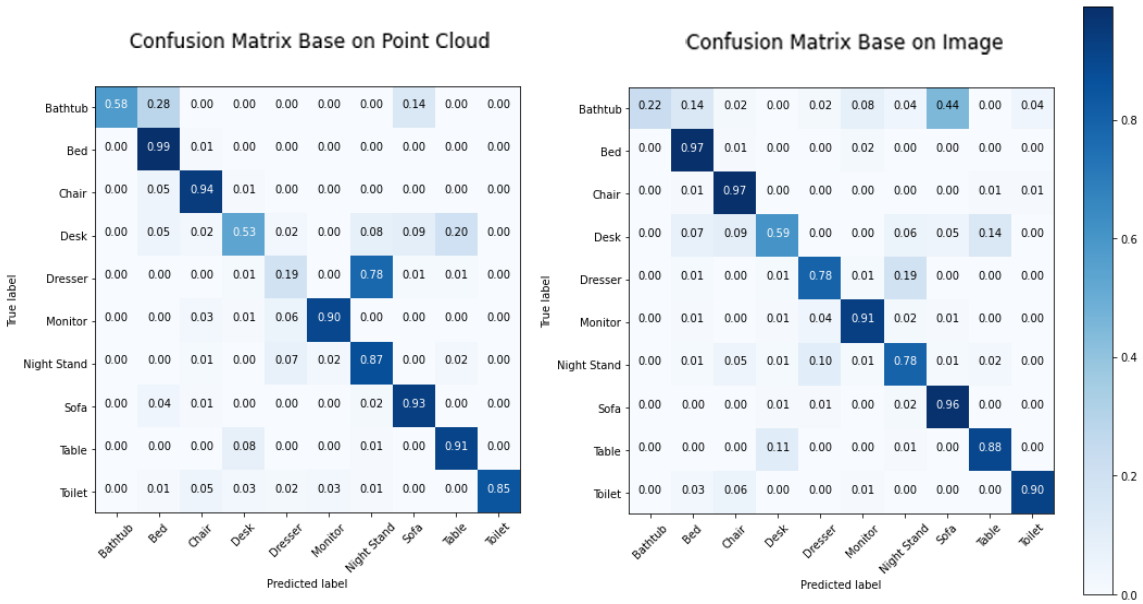

Figure 5. Confusion Matrices of classification based on point clouds and images.

For the performance of each different class in the testing set of ModelNet10, we found an interesting correlation between the results of 3D classification on point clouds and 2D classification on images in Figure 5. In addition, we observed that both classification algorithms have similar false positives and classes with the highest or lowest accuracy. This trend tells us that both neural networks are picking up similar features that contribute to their predictions. However, the classification based on the point cloud performs poorly in some classes compared to images. For example, the neural network based on the point cloud predicts most dressers as nightstands, which is due to us normalizing the point cloud and losing the scale of its original model. This does not happen to images because the image generated from the model still conserves its original size. In addition, normalizing the images scales down the RGB value instead of the coordinates for point clouds. We also observed that classification based on images is significantly worse at distinguishing between a bathtub and a sofa. The cause of this phenomenon is the fact that images do not contain any 3D information. Bathtubs are curved down on the top so they can hold water. Images won't be able to record all the curves in bathtubs that are blocked. Therefore, it is hard for the neural network to extract features corresponding to that curve with just an image from one direction.

Discussion

We observed a significant performance drop when our algorithm performs on abstract objects. We believed that other than research around the performance of these algorithms with losing data, we should put more effort into researching the influence of different designs. Computer vision used in autonomous vehicles and robots needs to classify not only living objects but also artificial objects. Animals are generic enough for neural networks to classify them with minimal errors. However, artificial things, especially furniture, are a lot trickier than animals because they are influenced by human design. Design is not easy to predict, making it hard to extract features from them similar to an ordinary object. Nevertheless, we still believe that there are ways of classifying them using neural networks. Using videos instead of static data like images and point clouds, deep neural networks can analyze human interaction with the object to understand these abstract objects' functions and categories. However, further research is still needed in this area.

From the designer's perspective, we believe that the extensive application of artificial intelligence using computer vision will significantly influence designers' creativity. This is because designers have to consider not only human interaction with the object but also computer interactions. Considering that computers currently are not as knowledgeable as humans on how to interact with manufactured objects, it will be challenging to design something in a world with humans and computers. Therefore, we suspect that computer vision will limit the potential of abstract designs.

Conclusion

In this project, we created our neural networks classifying two different data types, 3D point clouds and 2D images. We then trained the neural networks on the same 3D models from ModelNet10. For 3D point clouds, we used trimesh to convert the 3D models into point coordinates and feed them into a neural network based on PointNet Vanilla. And for 2D images, we rendered the models and took pictures from the same perspective and then input them into a 2D convolutional neural network. After training both neural networks, we tested their performance on the ModelNet10 testing set and the 3D models we generated. We generated these 3D models based on abstract designs from the categories of ModelNet10 and converted them into the same format as other testing data. To get the result, we compared the performance of the classification neural networks based on point cloud and images on ModelNet10 and the self-generated data.

We found that the classification based on point cloud and images are pretty similar performance-wise on the ModelNet10. However, when we input self-generated data, the point cloud based neural network has significantly higher accuracy than the image based neural network. Classification based on images also takes less time to train and is easier to converge than classification based on point clouds.

We conclude that though both point cloud and image based classification algorithms have similar performance, it seems more beneficial to combine these techniques for better performance and scalability. Furthermore, classifying abstract objects is very inefficient with both using point clouds and images. We suspect that we can minimize the performance lost on those objects with the help of deep neural networks and contextual analysis. However, further research is needed.

Code

Github Repository: github.com/M4rkX-Y/Point-Cloud-vs.-Image-Classification

Acknowledgments

I would like to express my sincere appreciation to Dr. Guillermo Goldsztein and TA Davida Kollmar, who gave me the golden opportunity to write this paper on machine learning. Secondly, I would also like to thank my family and friends, Sam Liu and James Li, for peer-reviewing the paper. Thanks again to all who supported me.

References

Dawson-Haggerty et al. (n.d.). trimesh. Version 3.2.0. url: https://trimsh.org/

Feng, Yutong, Yifan Feng, Haoxuan You, Xibin Zhao, and Yue Gao (2018). MeshNet: Mesh Neural Network for 3D Shape Representation. arXiv: 1811.11424 [cs.CV]

Filgueira, A., H. González-Jorge, S. Lagüela, L. Díaz-Vilariño, and P. Arias (2017). "Quantifying the influence of rain in LiDAR performance". In: Measurement 95, pp. 143–148.

Qi, Charles R., Hao Su, Kaichun Mo, and Leonidas J. Guibas (2017). Point-Net: Deep Learning on Point Sets for 3D Classification and Segmentation. arXiv: 1612.00593 [cs.CV]

Rusu, Radu Bogdan and Steve Cousins (2011). "3D is here: Point Cloud Library (PCL)". In: 2011 IEEE International Conference on Robotics and Automation, pp. 1–4. doi: 10.1109/ICRA.2011.5980567

Sedaghat, N., Zolfaghari, M., Amiri, E., & Brox, T. (2016). Orientation-boosted voxel nets for 3d object recognition. arXiv preprint arXiv: 1604.03351[cs.CV]

Wandinger, Ulla (2005). Introduction to Lidar. Ed. by Claus Weitkamp. New York, NY: Springer New York, pp. 1–18. isbn: 978-0-387-25101-1. doi: 10.1007/0-387-25101-4_1. url: https://doi.org/10.1007/0-387-25101-4_1

Wu, Zhirong et al. (2015). 3D ShapeNets: A Deep Representation for Volu-metric Shapes. arXiv: 1406.5670 [cs.CV]

Xiao, Tong, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang (2015). "Learning From Massive Noisy Labeled Data for Image Classification". In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Zhang, Feihu, Daniel Clarke, and Alois Knoll (2014). "Vehicle detection based on LiDAR and camera fusion". In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 1620–1625. doi: 10.1109/ITSC.2014.6957925