2017 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
AUTONOMOUS GROUND SYSTEMS (AGS) TECHNICAL SESSION
AUGUST 8-10, 2017 - NOVI, MICHIGAN

GESTURE RECOGNITION FOR ROBOTIC CONTROL USING DEEP LEARNING

Chris Kawatsu
Frank Koss
Andy Gillies
Aaron Zhao
Jacob Crossman
Ben Purman
Soar Technology Inc.
Ann Arbor, MI

Dave Stone
Dawn Dahn
Marine Corps
Warfighting Laboratory[1]

ABSTRACT

Can convolutional neural networks (CNNs) recognize gestures from a camera for robotic control? We examine this question using a small set of vehicle control gestures (move forward, grab control, no gesture, release control, stop, turn left, and turn right). Deep learning methods typically require large amounts of training data. For image recognition, the ImageNet data set is a widely used data set that consists of millions of labeled images. We do not expect to be able to collect a similar volume of training data for vehicle control gestures. Our method applies transfer learning to initialize the weights of the convolutional layers of the CNN to values obtained through training on the ImageNet data set. The fully connected layers of our network are then trained on a smaller set of gesture data that we collected and labeled. Our data set consists of about 50,000 images recorded at ten frames per second, collected and labeled in less than 15 man-hours. Images contain multiple people in a variety of indoor and outdoor settings. Approximately 4,000 images are held out for testing and contain a person not present in any of the training images. After training, greater than 99% of the images in the test set are correctly recognized. Additionally, we use the system to control a small unmanned ground vehicle. We also investigate using a Long Short-Term Memory (LSTM) layer for recognizing gestures that require analyzing sequences of images. On this more difficult set of gestures, we achieve a recognition rate of approximately 80% using a smaller data set of approximately 26,000 images.

[1] This Project Agreement Holder (PAH) effort was sponsored by the U.S. Government under Other Transaction number W15QKN-08-9-0001 between the Robotics Technology Consortium, Inc, and the Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S; Government.

INTRODUCTION

Can we apply deep learning to recognize vehicle control gestures from a standard camera with high enough accuracy to control an unmanned vehicle? Our goal is to recognize standard gestures defined in Field Manual (FM) 21-60 [1] to allow warfighters to control an unmanned vehicle in the same manner as a vehicle driven by a human.

SoarTech has previously investigated intuitive human-robot interfaces that leverage natural modes of interaction such as speech, gesture, and sketch to enable two-way dialogue between operators and robots. Our Smart Interaction Device (SID) [2] [3] has applied a speech and sketch interface on a tablet to control a variety of unmanned ground vehicles. The present paper focuses on adding gestures as an additional modality to SID.

Gesture recognition varies considerably across two dimensions: the type of gestures, for example American sign language, and the type of sensor(s) used to recognize the gesture, for example an accelerometer. In the present paper, we limit ourselves to full body gestures used for vehicle control specified in FM 21-60. By using these gestures, warfighters should not need any additional training to use gestures to control an unmanned ground vehicle. Our gesture set consists of the following gestures: Attention, As You Were, Turn Right, Turn Left, Slow Down, Increase Speed, Halt, Move Forward, Move In Reverse. Examples of these gestures taken from FM 21-60 are shown in Figure 1 through Figure 4. In addition to these gestures, we also add two additional categories: No User and No Gesture. No User indicates that there is no person in the image. No Gesture indicates that the operator is standing with arms down and not performing a gesture.

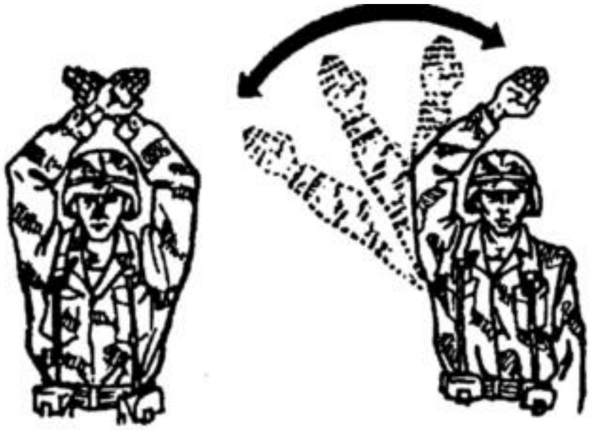


Figure 1:As You Were (left) and Attention (right).

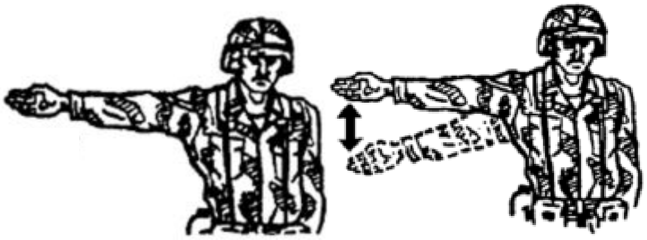


Figure 2: Turn Right (left) and Slow Down (right).

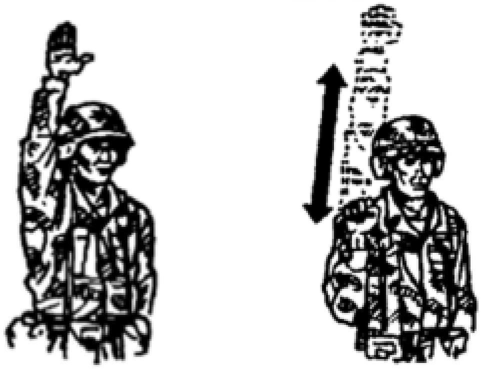


Figure 3:Halt (left) and Increase Speed (right).

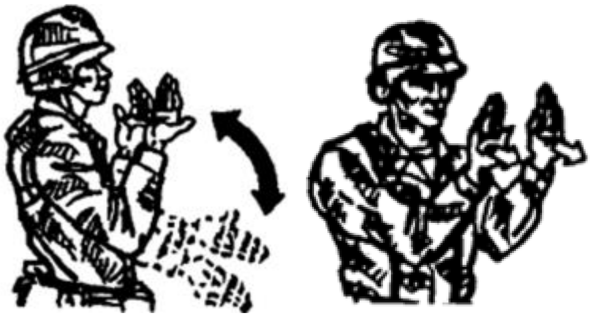


Figure 4: Move Forward (left) and Move In Reverse (right).

There are a wide variety of sensors which can be used to recognize gestures. For arm gestures such as those in our vehicle control gesture set, the Microsoft Kinect provides very high-fidelity data for the position of each joint in the arm. We have previously used the Kinect on similar types of vehicle control gestures, and were able to achieve close to a 100% recognition rate. Unfortunately, the Kinect will not work in outdoor environments because it relies on an IR laser which is overwhelmed by sunlight. For outdoor operation, typically four types of sensors are used: accelerometers, Lidar, stereo cameras, or monocular cameras. Accelerometers and Lidar are both active sensors, while cameras are passive; therefore, assuming other considerations such as recognition rates are equal, cameras are the preferred solution. Lidar sensors that have high point density are prohibitively expensive. Stereo cameras are also expensive compared to monocular cameras and require significant processing power to perform stereo matching. For these reasons, we decided to use a monocular camera for our sensor.

In recent years, deep learning approaches based on Convolutional Neural Networks (CNNs) have achieved state of the art results in a variety of image processing tasks such as object recognition and segmentation. Advances in this area have largely been driven by increases in processing power and the availability of large collections of labeled images to use during training. On the processing side, Graphics Processing Units (GPUs) increased enough in computation power and memory size to support running gradient descent on multiple layer neural networks with hundreds of millions of parameters. On the data side, competitions such as ImageNet [4] have released public datasets containing millions of labeled images which are necessary for training large neural networks. In order to take advantage of these improvements in image processing, we decided to use deep learning as the core of our gesture recognition system.

METHODOLOGY

Our gesture recognition system is subject to several constraints not considered in most deep learning research. First, there is no labeled dataset containing images of our vehicle control gestures. We must create and label training datasets ourselves. Second, we would like to run our gesture recognition system using on board computation power from a small unmanned ground vehicle, an iRobot PackBot. This means that a large GPU such as the Nvidia Titan X is out of the question. Our target is to run on the Nvidia Jetson which has 4-8GB of RAM and about one tenth the computational power of a Titan X. Our approach is designed to work around these constraints in two different ways. First, we initialize the weights of our CNN to values obtained by training on the ImageNet classification task. We then perform fine tuning of the upper layers of our network using our much smaller vehicle control gesture dataset. Second, we limit ourselves to CNN architectures which we expect to fit in memory of a Jetson and run in real time. Canziani et al. [5] provide an excellent comparison of popular CNN architectures shown in Figure 5. Initially we identified AlexNet [6] as the most promising network to run in real time on a Jetson TX1. With the release of the Jetson TX2 we have also evaluated using ResNet-50 [7] which provides a good tradeoff between performance and required operations.

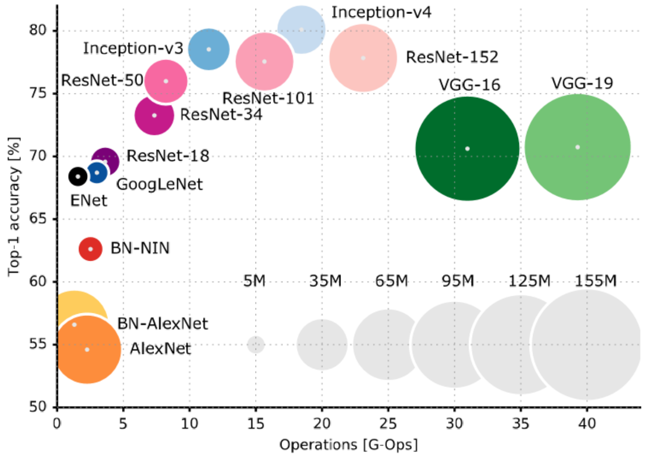


Figure 5: Comparison of popular CNN architectures. The vertical axis shows top 1 accuracy on ImageNet classification. The horizontal axis shows the number of operations needed to classify an image. Circle size is proportional to the number of parameters in the network.

Person Segmentation

Using deep architectures such as Faster R-CNN [8] it is possible to localize objects within a larger image. Due to our computational constraints of running in real time on a Jetson, it is not feasible to use this type of architecture without using a very small CNN. For this reason, we use a correlation filter tracker [9] which requires a user to initialize the starting position of the tracked person. The tracker segments the upper body of the person from the larger image. This segmented image is then rescaled to the size expected by the CNN used for gesture recognition.

AlexNet Architecture

Initially we tried to classify gestures using only information available in a single image. For this reason, we created a static gesture set by removing gestures involving motion: Slow Down, Speed Up, and Move In Reverse. Our architecture uses the same convolutional and max pooling layers as AlexNet, but significantly reduces the size of the two fully connected layers of the network. The architecture of our network is shown in Figure 6. Weights for the convolutional layers are initialized to values obtained through training on ImageNet. Fully connected and logistic regression layers are initialized to values normally distributed with mean zero and variance 0.1. Training uses minibatch stochastic gradient descent with a learning rate of 0.01. We found that it is only necessary to update weights in the fully connected layers; weights for convolutional layers remain fixed during training. Dropout is also used during training to randomly remove 50% of the connections between fully connected layers.

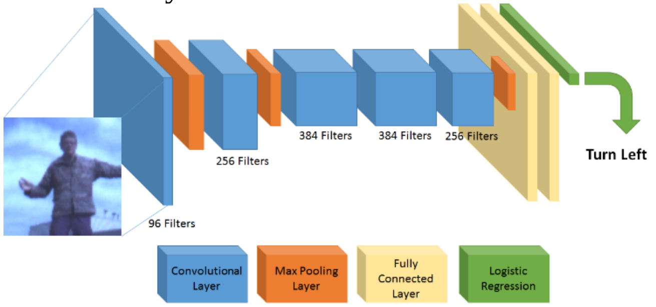


Figure 6: The architecture of our AlexNet based CNN.

ResNet Architecture

With the release of the Jetson TX2, we have been able to explore the use of more computationally expensive CNNs. We found that ResNet-50 will run at close to 10 frames per second on the TX2. This architecture provides a good tradeoff between computation time and accuracy on ImageNet (shown in Figure 5).

In addition to using a deeper CNN architecture, we would also like to account for motion in our gestures. Three pairs of gestures are distinguished largely by motion versus no motion. For example in Figure 2, Turn Right is almost exactly the same as Slow Down, except the latter involves motion. An example of these gestures recorded through our camera is shown in Figure 7.

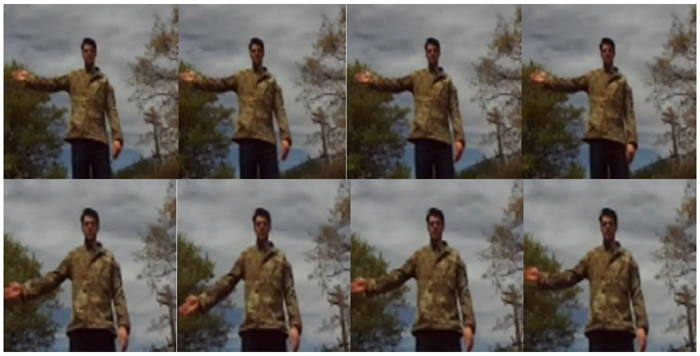


Figure 7: Example of a gesture pair distinguished by motion. Turn Right (top) has the right arm extended and not moving. Slow Down (bottom) has the right arm extended but moving up and down.

We are currently experimenting with different methods of accounting for motion. We have had limited success with two different approaches. Our first approach takes the difference between the previous and current image and stores this information in one of the color channels. This makes motion very obvious in cases where the background is static. The second approach adds a Long Short Term Memory (LSTM) [10] layer between the CNN and softmax layer. The LSTM will accumulate state as the network runs. This state could be used to determine whether or not the person is moving from frame to frame. The gesture recognition architecture with the addition of a LSTM is shown in Figure 8.

Weights for ResNet-50 are initialized using values obtained through training on ImageNet. Unlike in the AlexNet architecture, we fine tune the weights of the last 10 convolutional layers while keeping the remaining convolutional layers fixed. An average pooling layer is added to the end of ResNet to reduce the output dimension to 2048. When using an LSTM, 32 hidden states are used and dropout of 0.5.

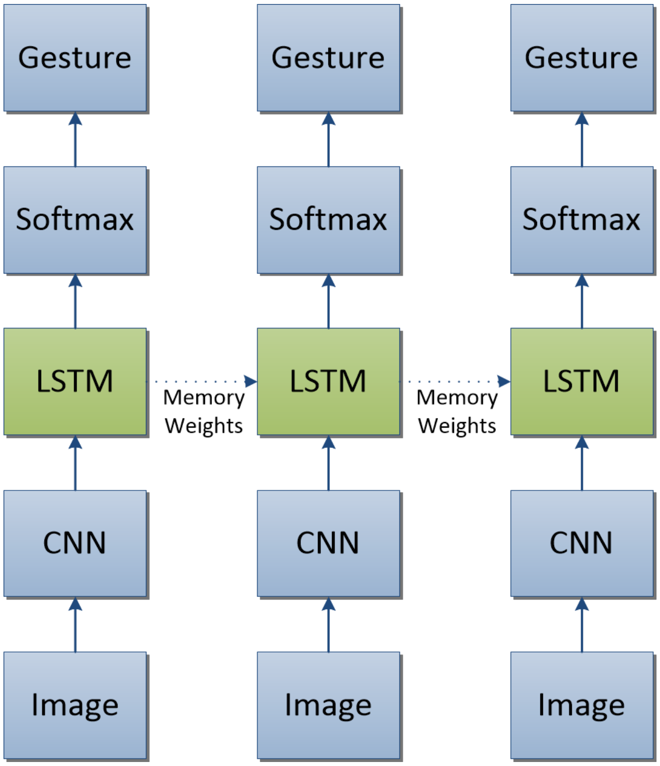


Figure 8: Gesture recognition architecture with the addition of a LSTM between the CNN and the softmax layer.

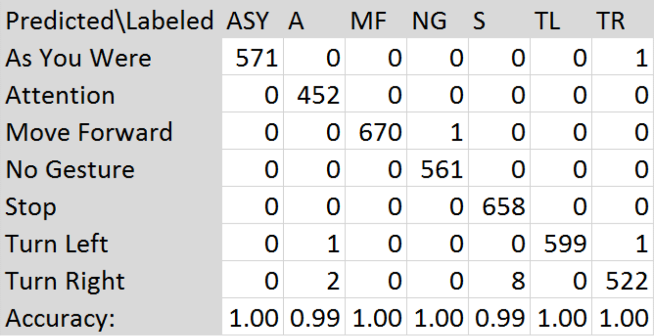
RESULTS

AlexNet Architecture

The AlexNet architecture was trained using the reduced static gesture set. The network is trained on about 45,000 images. We tested the performance of the architecture using an indoor data set consisting of about 4,000 images. Gestures were recognized by selecting the highest confidence output from the softmax layer. Using this criterion, the CNN correctly classified 99.79% of the test images. The confusion matrix is show in Table 1.

We deployed this network on a Jetson TX1 mounted on an iRobot PackBot and were able to use gestures to control the motion of the robot. While testing on the robot, we made several discoveries that were not apparent from the confusion matrix. First, we found that the network had overfit to very specific lighting conditions. To solve this issue, we introduced a random, artificial adjustment to each image during the training process. Second, we found that the network was very particular about the orientation of the person's arms for the Move Forward Gesture. We have since modified our data collection procedure to try to introduce more variation in how the gestures are performed. We do not demonstrate how to perform the gesture to people prior to collecting data; instead, we show them the image from Field Manual 21-60 describing the gesture. This introduces significantly more variation into the data compared to demonstrating the gesture.

Table 1: Confusion matrix for AlexNet architecture on indoor test set. Rows show the gesture predicted by the network, column show how the image was labeled in the data set.

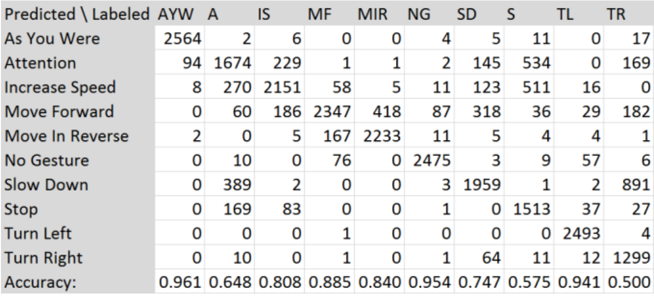


Predicted\Labeled	ASY	A	MF	NG	S	TL	TR
As You Were	571	0	0	0	0	0	1
Attention	0	452	0	0	0	0	0
Move Forward	0	0	670	1	0	0	0
No Gesture	0	0	0	561	0	0	0
Stop	0	0	0	0	658	0	0
Turn Left	0	1	0	0	0	599	1
Turn Right	0	2	0	0	8	0	522
Accuracy:	1.00	0.99	1.00	1.00	0.99	1.00	1.00

ResNet Architecture

The ResNet architecture was trained on the full gesture set which includes three pairs of gestures which are distinguished primarily by motion or no motion. The network was trained on approximately 26,000 images. During training, we feed the network minibatches consisting of 32 sequences of 10 images, with each sequence demonstrating a randomized gesture type. We perform gradient descent for each image in the sequence and reset the LSTM state between minibatches. We performed cross validation by holding out all images associated with each person in the data set and training on the remaining images. The average accuracy of these models was 78.69%.

Table 2: Confusion matrix for cross validation on ResNet architecture. Rows show the gesture predicted by the network, column show how the image was labeled in the data set.



Predicted \ Labeled	AYW	A	IS	MF	MIR	NG	SD	S	TL	TR
As You Were	2564	2	6	0	0	4	5	11	0	17
Attention	94	1674	229	1	1	2	145	534	0	169
Increase Speed	8	270	2151	58	5	11	123	511	16	0
Move Forward	0	60	186	2347	418	87	318	36	29	182
Move In Reverse	2	0	5	167	2233	11	5	4	4	1
No Gesture	0	10	0	76	0	2475	3	9	57	6
Slow Down	0	389	2	0	0	3	1959	1	2	891
Stop	0	169	83	0	0	1	0	1513	37	27
Turn Left	0	0	0	1	0	0	0	0	2493	4
Turn Right	0	10	0	1	0	1	64	11	12	1299
Accuracy:	0.961	0.648	0.808	0.885	0.840	0.954	0.747	0.575	0.941	0.500

Unfortunately, this accuracy is not sufficient for controlling the PackBot. We estimate that greater than 95% accuracy is required in order to control the PackBot without requiring an excessive number of gestures to repair incorrectly interpreted commands. Looking at the confusion matrix, the hardest to recognize gestures were Turn Right and Stop. Turn Right was primarily confused with Slow Down, while Stop was confused with Increase Speed and Attention. In both cases the confused gesture is the static gesture incorrectly classified as a similar looking dynamic gesture. Our hypothesis is that the network is primarily using the person's pose in a single image to classify the gesture, rather than using the LSTM to account for the entire sequence of images.

CONCLUSION

We were able to train a CNN to recognize static vehicle control gestures at a rate high enough to use for vehicle control in real world situations. Our network is small enough to run in real time on a Jetson TX1, allowing us to perform all processing onboard the iRobot PackBot. When our gesture set is expanded to include dynamic gestures, which appear similar to some of the static gestures, recognition rates decrease. We are continuing to investigate architectures to handle motion in images from frame to frame in order to increase recognition rates for the full set of vehicle control gestures.

REFERENCES

[1] U.S. Department of the Army, "Field Manual 21-60," 1987.

[2] G. Taylor, B. Purman, P. Schermerhorn, G. Garcia-Sampedro, M. Lanting, M. Quist and C. Kawatsu, "Natural interaction for unmanned systems," in *SPIE Defense and Security: Unmanned Systems Technology XVII*, 2015.

[3] G. Taylor, M. Quist, M. Lanting, C. Dunham and P. Muench, "Multi-modal interaction for robotic mules," in *SPIE Defense and Security: Unmanned Systems Technology XIX*, 2017.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma and Z. Huang, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, pp. 1-42, 2014.

[5] A. Canziani, A. Paszke and E. Culurciello, "An Analysis of Deep Neural Network Models for Practical Applications," *arXiv preprint,* vol. arXiv:1605.07678, 2016.

[6] A. Krizhevsky, I. Sutskever and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, pp. 1097-1105, 2012.

[7] 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.

[8] S. Ren, R. Girshick and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, pp. 91-99, 2015.

[9] M. Danelljan, G. Häger, F. Khan and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference*, Nottingham, 2014.

[10] G. Yarin and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," *Advances in Neural Information Processing Systems*, pp. 1019-1027, 2016.