Wall - Z the Robot v1.0

Ryan Hunt and Nhan Tran

February 2018

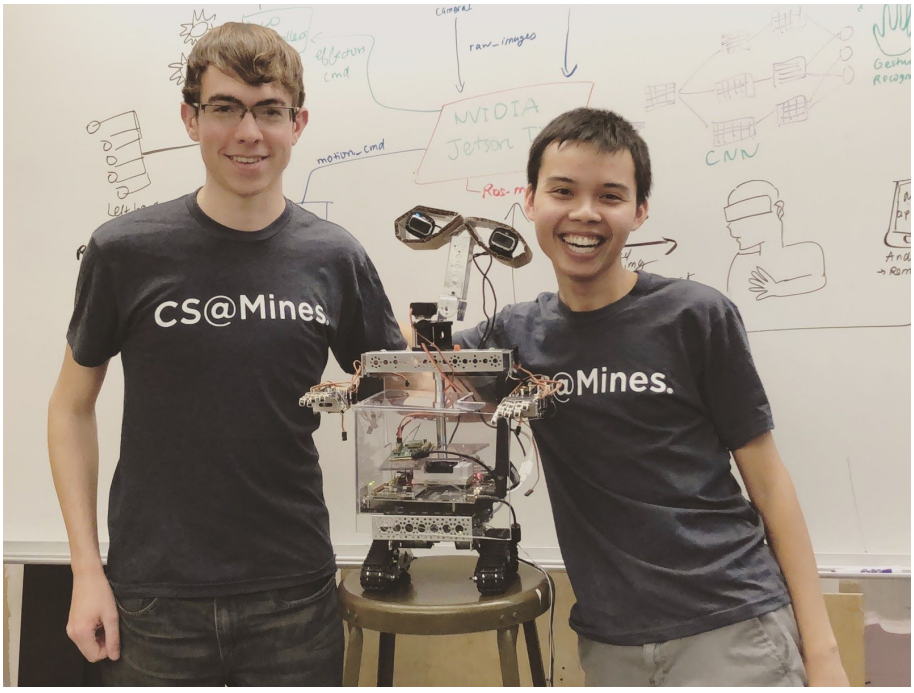

Table of Contents

1. Abstract

With dramatic improvements in embedded computational capacity arising in just the past couple of years, techniques such as neural network inference and high bandwidth video processing are now easily deployable to consumers on a wide scale. However, not all members of society have reaped the benefits of such technology. Those who are deaf struggle to communicate verbally in real time over traditional telecommunication links such as cellular networks and video conference calls, although these struggles could be resolved with improved embedded computational capacity. Here, we present a prototype of a remotely controlled security robot which doubles as a real-time communication link by processing and responding to American Sign Language (ASL) gestures via deep learning and gesture classification.

2. Project Description

The robot presented here (nicknamed Wall-Z for its resemblance to Pixar's Wall-E) is fully functional prototype of a robot capable of bridging real communication challenges faced by those who are deaf. It also serves as an effective security and task-oriented robot that would be capable of remotely monitoring and securing a home or business. The Wall-Z robot features built-in neural-network-based ASL letter recognition, VR-based remote visualization of the target environment, and controllable motor and servo movements for precise interaction with the target environment. It truly represents a viable application of advanced embedded computational power to the improvement of everyday lives of regular people.

3. System Architecture

In the interest of simplifying the overall system architecture and easing the development process, the Jetson TX2 onboard the Wall-Z robot was chosen as the lone system master. The Jetson was responsible for managing gesture recognition, motor movement, and VR/Communication infrastructure. ROS (Robot Operating System) was utilized to simplify the process of integrating components due to its logistical and organizational prowess.

Gesture recognition was the primary responsibility of the Jetson TX2 onboard the Wall-Z robot. Custom TensorFlow models trained remotely were imported to the Jetson TX2 and loaded onto the GPU for inference. Subsequently, frames captured from one of the two cameras located on the Wall-Z robot were fed to the model, and the inference results were published to the robot observers.

Motor movement was another high priority task for the Jetson TX2 onboard the Wall-Z robot. Drive motor control signals originating from an Android remote control application were fed to an Arduino board serving as a DC motor controller via serial commands over Jetson's USB interface, while servo commands collected from accelerometer and magnetometer sensors on the observer's Android device were fed to a dedicated servo controller on the robot, also via serial commands on the USB bus. This effectively allows a remote observer to virtually explore an environment, by issuing drive commands and altering the orientation of the cameras onboard the robot which mirror the orientation of the controlling VR headset.

VR and communication infrastructure represented the last main responsibility for the Jetson TX2. The other camera located on the Wall-Z robot published split-frames directly to the robot observer, who could virtually visualize the robot's environment through a VR headset. Eventually, speech-to-text software running on the observer's Android device will transmit text to the robot so that remote conversation may occur between the observer and any person visible by the Wall-Z robot.

The entire system architecture is illustrated in **Figure 1**.

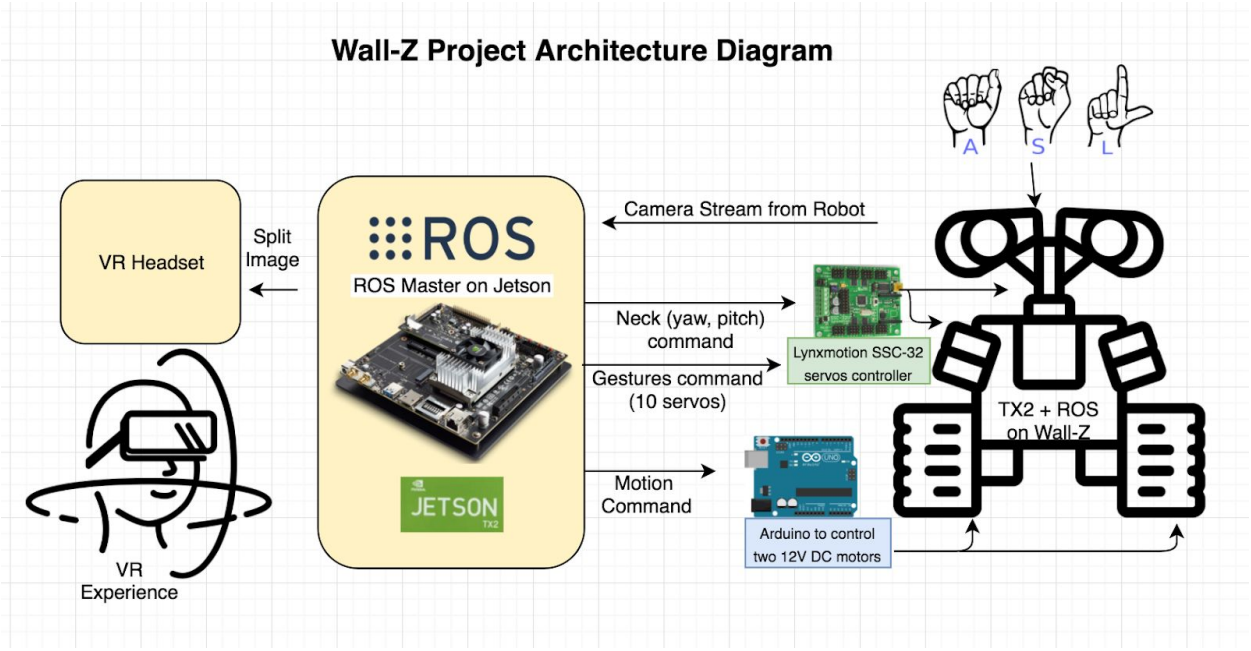

Figure 1. The user interacts directly with a VR headset (e.g., Google Cardboard) which streams processed images from the robot's camera. These devices send data to and receive data from the Wall Z robot using an instance of the ROS architecture whose Master node is run on the NVIDIA Jetson TX2. The TX2 also does inferencing of ASL gestures.

4. Convolutional Neural Network Architecture

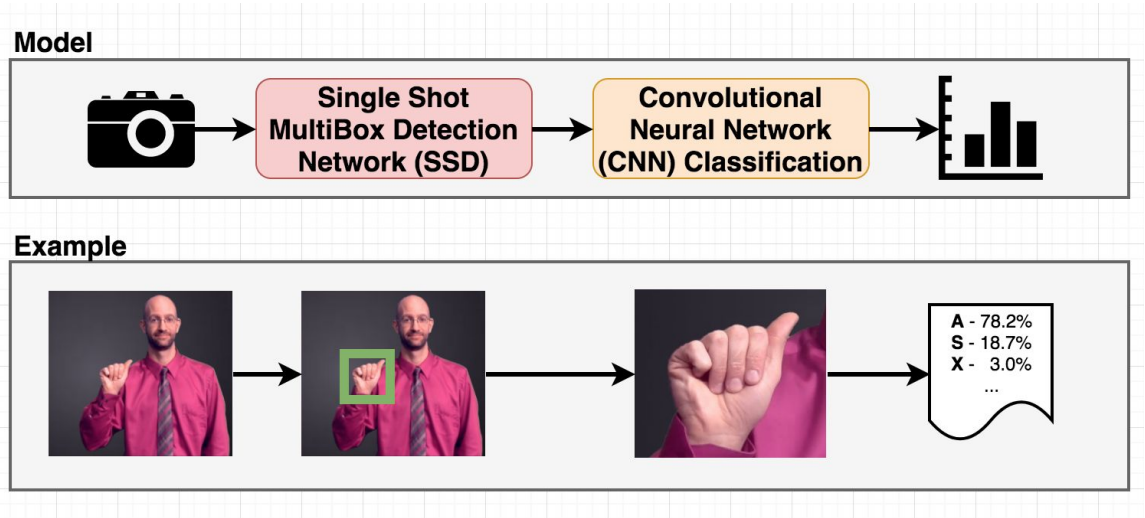

Figure 2. (Above) The deep learning model used to perform detection and classification of ASL gestures. (Below) An example pipeline showing the processing of the ASL gesture for the letter 'a'.

In order to perform high quality, robust, and rapid gesture detection, neural network inference based on a model trained via deep learning was the technology chosen for this task. Properly implemented neural networks boast rapid response times, robustness against environmental variations, relatively inexpensive hardware requirements, and infinite configurability.

Based on previous research in the field of computer vision, it was determined early on that it would be infeasible to effectively train a single stage neural network capable of detecting ASL gestures. Instead, we pursued a dual stage neural network composed of a Regional Convolutional Neural Network (R-CNN) pipelined with a Convolutional Neural Network (CNN). The R-CNN stage identifies and isolates regions of the image containing hands, while the CNN stage classifies the hand gesture itself.

Our R-CNN is based on an popular object localization architecture known as Single Shot Multi-Box Detection Network (SSD). The network provides a list of proposed bounding boxes and their associated probabilities, but at a much quicker pace than a traditional brute force search of an image for an object because all the proposals are generated in just a single pass of the image.

The bounding boxes with a probability above a given threshold are passed through the pipeline to the next neural network stage: the CNN classification stage. Our CNN stage crops the image based on the bounding box produced by the previous stage and passes the fragment into the neural network that has been trained on ASL gestures. The output of this stage is a distribution of probabilities across the various ASL gestures that are known by the network. The gesture with the highest probability is forwarded to the observer for processing.

This architecture is outlined in **Figure 2** and an example is given to illustrate the processing that occurs for a sample ASL gesture.

5. Discussion

5.1 Challenges

The first challenge was the accuracy of the gesture recognition, especially with the second phase when we tried to classify the gestures. Besides reading related research paper, we reached out to several graduate students and engineers in the industry to ask their perspectives on several approaches. The two frequently mentioned networks were “You Only Look Once” (YOLO) and “Single Shot Detector” (SSD). During the experimental stage, our application got around 1 frame per second when testing on our laptop, but an average of 14 frames per second when running on the TX2! Then, we decided to go with SSD as it yielded better performance.

The second challenge were limited time and resources. We had only 3.5 weeks to define the project scope, learn from prior work and the related topics, order materials, and meet up ~2 hours every day to do collaborate on the project. Moreover, the servos on the robot’s arms were

damaged; we had been waiting for new replacement to arrive. However, we gained super useful knowledge after weeks of teaching ourselves and reading related research papers.

5.2. What we learned

Besides gaining exposures to the new technologies, the powerful TX2, and the useful Robot Operating System, we learned about the importance of architecture when dealing with problem related to neural networks and big set of data. Our test on the laptop which maximized GPU knowledge ran slowly. A quick change to the GPU fixed our problem and gave us 10x performance!

In addition to the software of Wall-Z, we enjoyed learning more about the hardware side of robotics while working on this project. We spent a significant amount of time in the university's machine shop trying to create parts in order to assemble to a function Wall-E inspired robot.

5.3 Future Improvements

In the future, we hope to finalize hardware and software that would allow complete remote conversations to be held between the remote observer and a person in view of the Wall-Z robot. This would require implementing software to successfully plan and control robotic arm gestures on the Wall-Z robot and implementing speech-to-text software on the observer's device to encode verbal communication into a textual format.

In conclusion, thank you for reading. Participating in the NVIDIA Jetson Developer Challenge gave us the opportunity to learn more about deep learning and its application in robotics. If you might have suggestion, feedback, or question, please reach out to us at either nhantrantnt@gmail.com or ryhunt@mymail.mines.edu.