DLI Edge AI and Robotics Teaching Kit

Module 2 Lab:   
**Hello AI World**

This lab assigns sections from the Hello AI World portions of the Jetson Inference project on Github. The content and Instructions for the entire Hello AI World can be found in the readme documents at: <https://github.com/dusty-nv/jetson-inference>.

This lab is meant to run on any of the Jetson Developer kits. The github project contains all the instructions to run the lab. This document contains additional information to help faculty and students work through connecting remotely and getting images and video to display on their remote host machines.

For the System Setup, we will be using a Docker Container and follow the instructions under ‘Running the Docker Container.’

Download and **Launch** the container by running the following commands **separately**:

sudo git clone --recursive https://github.com/dusty-nv/jetson-inference

cd jetson-inference

docker/run.sh

The run.sh script contains the commands to download and run the appropriate docker container that matches the software version of your Jetson Nano. The first time you execute the command ‘run.sh’, you will be prompted to download neural net models.

You can accept the default models for now:

Graphical user interface

Description automatically generated

In the future, if you want to download additional models, you can run the download model tool again by typing the following commands:

Text

Description automatically generated

It will take a while for everything to download. Once the containers are installed, you will be at the root inside the container:



If you need to open up another shell inside the container, find out the container image name (**sudo docker container ls** is the command to list the containers) and use the container name in the following docker exec command. For example, if the container name is romantic\_banach, the command is:

**docker exec -it romantic\_banach /bin/bash**

Then you can navigate to the jetson\_inference directory

cd /jetson-inference

# Viewing Images:

For Mac, it is recommended that you install [XQuartz](https://www.xquartz.org/) on your host Mac machine which will enable the display the X11 windows. Also, each time that you ssh into the Jetson Nano, you should use the -Y parameter (E.G. **ssh -Y user@nanoIPaddress**).

For Windows, I recommend [MobaXterm](https://mobaxterm.mobatek.net/) on your host machine when using the command line to connect to the Nano. It simplifies the setup process as it is preconfigured to enable X11 and provides other advantages such as viewing the remote file directories

Both Windows and Mac users can use the eog command in the terminal to open an image. For instance, if you are same directory as the image and you need to open an image called lena.jpg then the command is:

**eog lena.jpg**

# Using RTP for Video

Streaming video and camera feeds remotely requires a bit more setup for both Mac and Window host machines. One options is to use RTP to remotely view video streamed from our camera or a video file from the Jetson.

## **First install gstreamer**

## Install on Windows.

Install gstreamer (download from <https://gstreamer.freedesktop.org/download/> )

## Install on Mac.

<https://gstreamer.freedesktop.org/download/#macos> (Download Both; runtime installer and developer installer)

— Open with Installer:

continue through the installation.

Note:

Installation on Windows or Mac will not create a launchable desktop icon but will create the libraries that you will load at a command prompt when ready to view the video output from your Jetson Nano.

## Launch the python program within the Container on Jetons using RTP

Now inside the container on your jetson, test out your camera. First transmit the images from your camera over rtp. You must use your own Windows or Mac host IP address here. This is an important step. Your Jetson Nano must know what IP address to stream the video to. Not using the right IP address will result in the video running, but you will not see it from your host. Mine is 10.200.79.119 as shown below.

Execute either of these commands to generate camera or video feeds remotely on your Jetson Nano but you much specify the IP of your host machine to direct the RTP output to your host machine.

### Transmit camera feed – call video-viewer to test(substitute the IP with your Windows or Mac host IP) :

**video-viewer /dev/video0 rtp://10.200.79.119:1234**

### Transmit video file – call video-viewer to test (edit to use your Windows or Mac host IP for remote-ip and name of video file on jetson):

**video-viewer --output-codec=h265 my\_video.mp4 rtp://<remote-ip>:1234**

### Call detectnet python program using your camera(substitute the IP with your host IP)

**detectnet /dev/video0 rtp://10.200.79.119:1234**

## Now launch Gstreamer Windows to view the output:

**Command Line** - use this command on your Windows machine from the command line (port should match port specified when launching the python program on the nano) :

Navigate to the directory first:

**cd C:\gstreamer\1.0\msvc\_x86\_64\bin**

Then execute this script to launch on your host machine:

**gst-launch-1.0 -v udpsrc port=1234 caps = "application/x-rtp, media=(string)video, clock-rate=(int)90000, encoding-name=(string)H264, payload=(int)96" ! rtph264depay ! decodebin ! videoconvert ! autovideosink**

Alternatively, in Powershell, the format is slightly different in Windows:

Navigate to the directory first:

**cd C:\gstreamer\1.0\msvc\_x86\_64\bin**

Then execute this script to launch on your host machine:

**.\gst-launch-1.0 -v udpsrc port=1234 caps = "application/x-rtp, media=(string)video, clock-rate=(int)90000, encoding-name=(string)H264, payload=(int)96" ! rtph264depay ! decodebin ! videoconvert ! autovideosink**

## Now launch Gstreamer Mac to view the output:

Locate Gstreamer Folder (usually in this path shown) and navigate to that directory in a command prompt:

**cd /Library/Frameworks/GStreamer.framework/Commands**

Launch Gstreamer by running the following (port should match port specified when launching the python program on the nano):

**gst-launch-1.0 -v udpsrc port=1234 \**

**caps = "application/x-rtp, media=(string)video, clock-rate=(int)90000, encoding-name=(string)H264, payload=(int)96" ! \**

**rtph264depay ! decodebin ! videoconvert ! autovideosink**

You will need to run the Gstreamer command each time you load a python script that uses RTP.

# Hello AI World Image Classification:

This lab makes use of the [Hello AI World Inferencing Projec](https://github.com/dusty-nv/jetson-inference)t developed by Nvidia. You should have already followed the instructions to download the container.

For this lab, you are to follow the i[nstructions on writing a python program](https://github.com/dusty-nv/jetson-inference/blob/master/docs/imagenet-example-python-2.md) to detect images.

Please read through the instructions and then follow to create a directory and python file outside of the container. Then you will test it this file inside the container.

A video tutorial from Nvidia can be viewed [here](https://www.youtube.com/watch?v=QatH8iF0Efk&list=PL5B692fm6--uQRRDTPsJDp4o0xbzkoyf8&index=11). For the streaming camera portion, you will need to use RTP and G-Streamer to view the output. Refer back to the [instructions](https://umbc.box.com/s/n4p8ktq4cf2l9ddvd2ez78nhihkym06j) on how to enable RTP using G-Streamer.

# Hello AI World Image Segmentation:

This makes use of the Jetson\_inference Hello AI World that we previously used. Refer back to this [instruction](https://umbc.box.com/s/n4p8ktq4cf2l9ddvd2ez78nhihkym06j) to run the container.

Follow the tutorial in the readme file of the Jetson\_inference github site for the [Semantic Segmentation with SegNet](https://github.com/dusty-nv/jetson-inference/blob/master/docs/segnet-console-2.md) and execute the sample python code provided along with the test images. Try a few images of your own.

Here is a helpful [video from Nvidia](https://github.com/dusty-nv/jetson-inference/blob/master/docs/segnet-console-2.md) for this tutorial.

# Hello AI World Pose Estimation

This makes use of the Jetson\_inference Hello AI World that we previously used. Refer back to this [instruction](https://umbc.box.com/s/n4p8ktq4cf2l9ddvd2ez78nhihkym06j) to run the container.

Follow the tutorial in the readme file of the Jetson\_inference github site for the [Pose Estimation with PoseNet](https://github.com/dusty-nv/jetson-inference/blob/master/docs/posenet.md) and execute the sample python code provided along with the test images and videos. You will need to use Gstreamer for the video when connecting the Jetson remotely. Alternatively, you can directly connect to your Jetson with a monitor, keyboard, and mouse.

**Deliverables** – demonstrate to your instructor that you are able to run through each of the exercises and create a print screen of each final section.