Intel® Vision Accelerator Design with Intel® Movidius™ VPUs Workload Distribution

User Guide

September 2019

Document Number: 613514-0.9

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting: http://www.intel.com/design/literature.htm

No computer system can be absolutely secure.

Contents

1.0 Introduction ... **6**
- 1.1 Terminology ... 6
- 1.2 Prerequisites ... 6
- 1.3 Reference Documents ... 7

2.0 Configure the Host System ... **9**
- 2.1 Set the Environment Variables ... 9
- 2.2 Update the HDDL Configuration File (`hddl_service.config`) ... 10
- 2.3 Start the hddldaemon ... 10

3.0 Run the Same Neural Network on all Vision Processing Units (VPUs) ... **11**
- 3.1 Preparation ... 11
- 3.2 Build and Run the Image Classification Sample Application ... 11

4.0 Run Different Neural Network on Different Sets of VPUs ... **14**
- 4.1 Preparation ... 14
- 4.2 Run the Security Barrier Camera Sample Application ... 15

5.0 Run Combinations of Neural Networks across Multiple Accelerator Cards ... **18**
- 5.1 Preparation ... 18
  - 5.1.1 Hardware ... 18
  - 5.1.2 Software ... 19
- 5.2 Run the Security Barrier Camera Sample Application ... 21

Figures

Figure 1. Location of IR and Labels Files ... 11
Figure 2. Label and Confidence for Image Classification Sample Application Top 10 Categories ... 12
Figure 4. Change Tag Scheduler ... 14
Figure 5. Location of Three Pre-Trained IRs ... 15
Figure 6. Security Barrier Camera Sample Results ... 16
Figure 7. VPU Workload Distribution ... 17
Figure 8. DIP Switch Location on Intel® Vision Accelerator Design with Intel® Movidius™ VPU Add-in Cards ... 19
Figure 9. Autoboot Configuration File change ... 20
Figure 10. Tag Scheduler Change in Service Configuration File ... 20
Figure 11. Location of Three Pre-Trained IRs ... 21
Figure 12. Security Barrier Camera Sample Results ... 22
Figure 7. VPU Workload Distribution ... 23

Tables

Table 1. Terminology 6
Table 2. Reference Documents 7

Revision History

Date	Revision	Description
June 2019	0.5	Initial release.

§

1.0 Introduction

This document guides you through enabling workload distribution tasks on the Intel® Vision Accelerator Design with Intel® Movidius™ VPU. This VPU product accelerates inference of computer vision applications.

In this document, you learn how to:

- Configure the Host System to run the applications in this document
- Run the Same Neural Network on all Vision Processing Units (VPUs)
- Run Different Neural Network on Different Sets of VPUs
- Run Combinations of Neural Networks across Multiple Accelerating Cards

1.1 Terminology

Table 1. Terminology

Term	Description
VPU	Vision Processing Unit

1.2 Prerequisites

- Linux host system
- Familiarity with editing Linux configuration files
- Two Intel® Vision Accelerator Design with Intel® Movidius™ VPU add-in cards installed in the host system. When multiple cards are used, make sure the DIP switches are set differently from each other.
  - Run the Same Neural Network on all Vision Processing Units (VPUs)
  - Run Different Neural Network on Different Sets of VPUs
  - Run Combinations of Neural Networks across Multiple Accelerating Cards
- Installed Intel® Distribution of OpenVINO™ toolkit

1.3 Reference Documents

Table 2. Reference Documents

Document	Document No./Location

2.0 Configure the Host System

Each example in this document requires you to:

- Set environment variables
- Edit `hddl_service.config` settings
- Start the hddldaemon

This section provides instructions for one-time configuration settings. **Make these changes before you use the examples in this document**.

The examples in this guide require you to make other configuration changes. Therefore, you may need to return to this section later.

2.1 Set the Environment Variables

You must update several environment variables before you can compile and run Intel® Distribution of OpenVINO™ toolkit applications. The OpenVINO toolkit provides a script to make setting the environment variables easy. See Using the Temporary Environment Variable Script to run this script.

As an alternative, use Setting Permanent Environment Variables to permanently set the environment variables.

Using the Temporary Environment Variable Script

Run the following script to temporarily set your OpenVINO environment variables.

```
source /opt/intel/openvino/bin/setupvars.sh
```

When using this script, the environment variables are removed when you close the shell. If you run the examples in this document across sessions, return to this step to reset your environment variables when you open the shell.

Setting Permanent Environment Variables

Use these steps to permanently set the environment variables. If you permanently set the environment variables, you don't need to return to these steps to set your environment variables again later.

1. Open the `.bashrc` file in <user_directory>:

```
vi <user_directory>/.bashrc
```

2. Add this line to the end of the file:

```
source /opt/intel/openvino/bin/setupvars.sh
```

3. Save and close the file: press the **Esc** key and type `:wq`

4. Test your change, open a new terminal. You see
   `[setupvars.sh] OpenVINO environment initialized`

2.2 Update the HDDL Configuration File (`hddl_service.config`)

The HDDL plugin invokes an hddldaemon process as a backend service to manage VPUs and dispatch inference tasks to VPUs, as directed by the scheduler settings.

The `hddl_service.config` file provides configuration settings to control hddldaemon behaviors, like log level, VPUs assignment to schedulers, timeout, log formatter etc.

Each example in this guide uses both common and unique `hddl_service.config` file settings. In this section, you change the common settings.

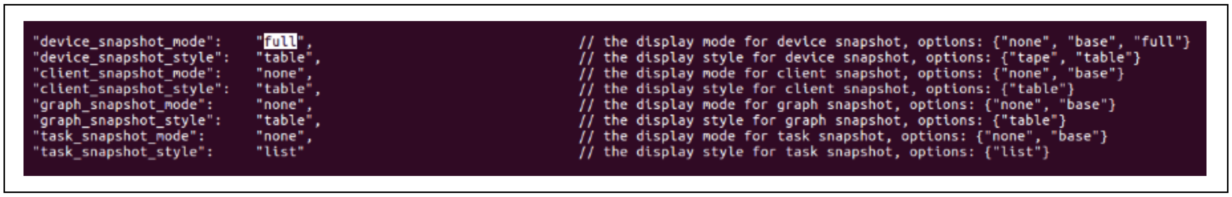
See the descriptions in the `hddl_service.config` file for more information about the settings.

1. Open the configuration file:

```
${HDDL_INSTALL_DIR}/config/hddl_service.config
```

2. Edit the file for the device snapshot display mode and style:

```
"device_snapshot_mode":    "full",
"device_snapshot_style":   "table",
```



```
"device_snapshot_mode":    "full",                                      // the display mode for device snapshot, options: {"none", "base", "full"}
"device_snapshot_style":   "table",                                     // the display style for device snapshot, options: {"tape", "table"}
"client_snapshot_mode":    "none",                                      // the display mode for client snapshot, options: {"none", "base"}
"client_snapshot_style":   "table",                                     // the display style for client snapshot, options: {"table"}
"graph_snapshot_mode":     "none",                                      // the display mode for graph snapshot, options: {"none", "base"}
"graph_snapshot_style":    "table",                                     // the display style for graph snapshot, options: {"table"}
"task_snapshot_mode":      "none",                                      // the display mode for task snapshot, options: {"none", "base"}
"task_snapshot_style":     "list"                                       // the display style for task snapshot, options: {"list"}
```

2.3 Start the hddldaemon

Whenever you change `hddl_service.config`, you must start or restart the hddldaemon.

In a separate terminal, enter the following command to run the hddldaemon:

```
${HDDL_INSTALL_DIR}/bin/hddldaemon
```

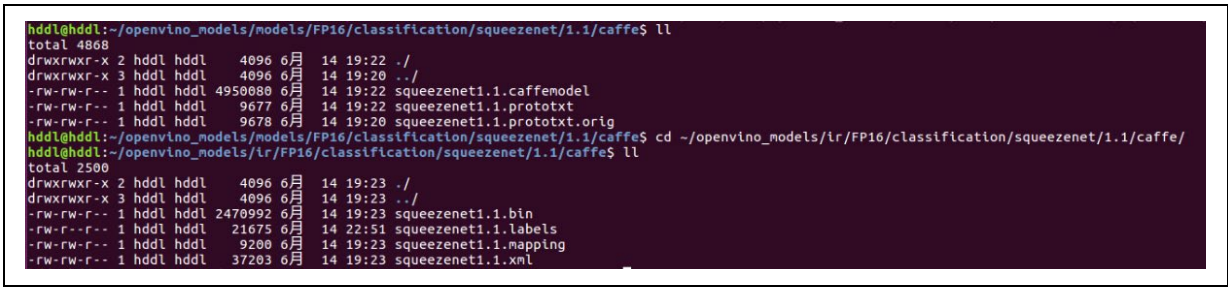
The HDDLPlugin starts the hddldaemon automatically if there is no hddldaemon running on the system.

3.0 Run the Same Neural Network on all Vision Processing Units (VPUs)

Purpose: Verifies the workload distribution of the same neural network running on all VPUs.

OpenVINO™ toolkit Sample Application: Image Classification

Figure 1. Location of IR and Labels Files



```
hddl@hddl:~/openvino_models/models/FP16/classification/squeezenet/1.1/caffe$ ll
total 4868
drwxrwxr-x 2 hddl hddl    4096 6月  14 19:22 ./
drwxrwxr-x 3 hddl hddl    4096 6月  14 19:20 ../
-rw-rw-r-- 1 hddl hddl 4950080 6月  14 19:22 squeezenet1.1.caffemodel
-rw-rw-r-- 1 hddl hddl    9677 6月  14 19:22 squeezenet1.1.prototxt
-rw-rw-r-- 1 hddl hddl    9678 6月  14 19:20 squeezenet1.1.prototxt.orig
hddl@hddl:~/openvino_models/models/FP16/classification/squeezenet/1.1/caffe$ cd ~/openvino_models/ir/FP16/classification/squeezenet/1.1/caffe/
hddl@hddl:~/openvino_models/ir/FP16/classification/squeezenet/1.1/caffe$ ll
total 2500
drwxrwxr-x 2 hddl hddl    4096 6月  14 19:23 ./
drwxrwxr-x 3 hddl hddl    4096 6月  14 19:23 ../
-rw-rw-r-- 1 hddl hddl 2470992 6月  14 19:23 squeezenet1.1.bin
-rw-r--r-- 1 hddl hddl   21675 6月  14 22:51 squeezenet1.1.labels
-rw-rw-r-- 1 hddl hddl    9200 6月  14 19:23 squeezenet1.1.mapping
-rw-rw-r-- 1 hddl hddl   37203 6月  14 19:23 squeezenet1.1.xml
```

3.1 Preparation

1. If you use temporary OpenVINO® toolkit environment settings and just opened a shell, Set the Environment Variables.
2. Start the hddldaemon if it isn't running.

You are ready to Build and Run the Image Classification Sample Application.

3.2 Build and Run the Image Classification Sample Application

The Image Classification script:

- Downloads a SqueezeNet* model.
- Uses the Model Optimizer to convert the model to the `.bin` and `.xml` Intermediate Representation (IR) files and a labels file. The Inference Engine requires this model conversion to use the IR as input. This achieves optimum performance on Intel hardware.

To verify the workload distribution, use these steps to build and run the Image Classification sample application script that was provided with the OpenVINO™ toolkit. The sample uses `car.png` in `/opt/intel/openvino/deployment_tools/demo`

1. Go to the Inference Engine demo directory:

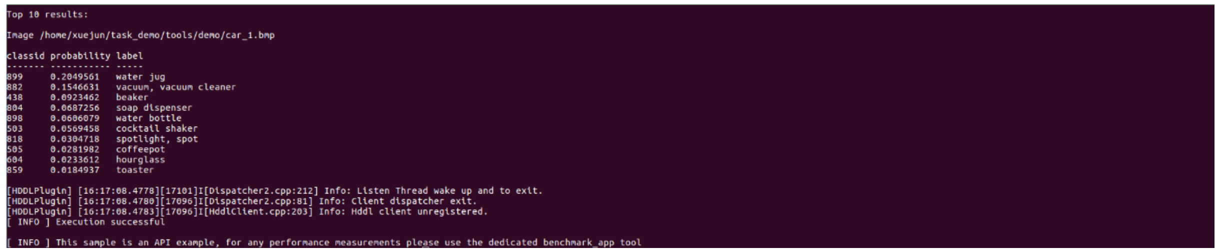
```
cd /opt/intel/openvino/deployment_tools/demo
```

2. Run the Image Classification verification script:

```
./demo_squeezenet_download_convert_run.sh -d HDDL
```

When the verification script completes, you will have the label and confidence for the top-10 categories:

Figure 2. Label and Confidence for Image Classification Sample Application Top 10 Categories



```
Top 10 results:

Image /home/xuejun/task_demo/tools/demo/car_1.bmp

classid probability label
------- ----------- -----
899     0.2049561   water jug
882     0.1546631   vacuum, vacuum cleaner
438     0.0923462   beaker
804     0.0687256   soap dispenser
898     0.0606079   water bottle
503     0.0569458   cocktail shaker
818     0.0304718   spotlight, spot
505     0.0281982   coffeepot
604     0.0233612   hourglass
859     0.0184937   toaster

[HDDLPlugin] [16:17:08.4778][17101]I[Dispatcher2.cpp:212] Info: Listen Thread wake up and to exit.
[HDDLPlugin] [16:17:08.4780][17096]I[Dispatcher2.cpp:81] Info: Client dispatcher exit.
[HDDLPlugin] [16:17:08.4783][17096]I[HddlClient.cpp:203] Info: Hddl client unregistered.
[ INFO ] Execution successful

[ INFO ] This sample is an API example, for any performance measurements please use the dedicated benchmark_app tool
```

Based on the downloaded model (`squeezenet1.1.xml` and `squeezenet1.1.bin`), `benchmark_app` measures the performance of the model and show the workload distribution.

1. Go to Inference Engine Samples Build directory:

```
cd ${HOME}/inference_engine_samples_build
```

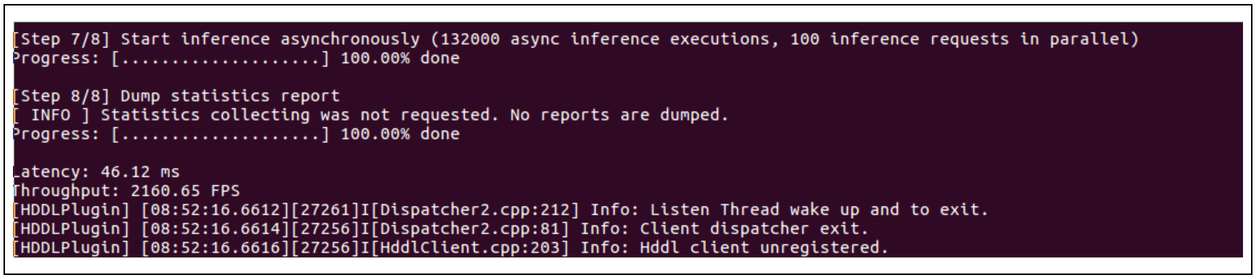
2. Build the `benchmark_app` demo:

```
make -j 8 benchmark_app
```

3. Run the `benchmark_app` demo:

```
./intel64/Release/benchmark_app
-i /opt/intel/openvino/deployment_tools/demo/car_1.bmp
-m
~/openvino_models/ir/FP16/classification/squeezenet/1.1/caffe/
squeezenet1.1.xml -d HDDL -nireq 100 -niter 132000
```

4. The output is:



```
[Step 7/8] Start inference asynchronously (132000 async inference executions, 100 inference requests in parallel)
Progress: [....................] 100.00% done

[Step 8/8] Dump statistics report
[ INFO ] Statistics collecting was not requested. No reports are dumped.
Progress: [....................] 100.00% done

Latency: 46.12 ms
Throughput: 2160.65 FPS
[HDDLPlugin] [08:52:16.6612][27261]I[Dispatcher2.cpp:212] Info: Listen Thread wake up and to exit.
[HDDLPlugin] [08:52:16.6614][27256]I[Dispatcher2.cpp:81] Info: Client dispatcher exit.
[HDDLPlugin] [08:52:16.6616][27256]I[HddlClient.cpp:203] Info: Hddl client unregistered.
```

4.0 Run Different Neural Network on Different Sets of VPUs

Purpose: Verify the workload distribution of different neural network running on different sets of VPUs.

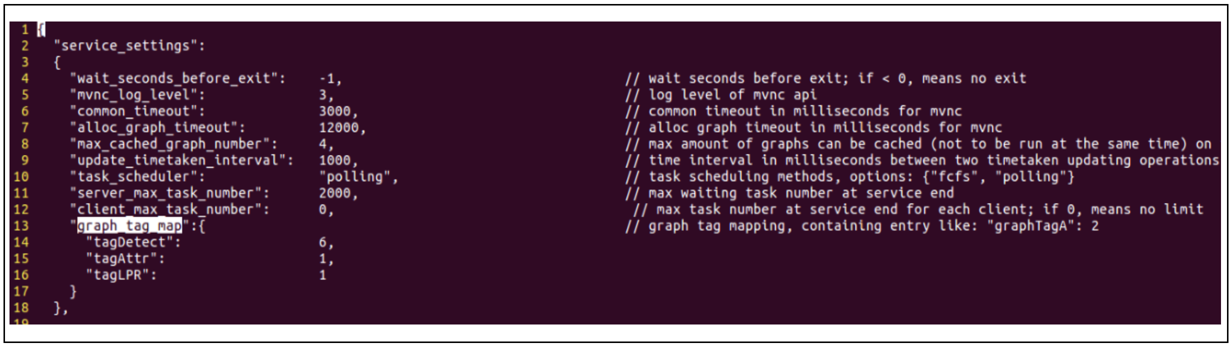
OpenVINO™ toolkit Sample Application: Security Barrier Camera Demo

4.1 Preparation

1. If you use temporary OpenVINO® toolkit environment settings and just opened a shell, Set the Environment Variables.
2. Edit `${HDDL_INSTALL_DIR}/config/hddl_service.config` to update the tag scheduler:

```
"scheduler_config":
{
  // Tag Scheduler
  "graph_tag_map":{
    "tagDetect": 6,
    "tagAttr": 1,
    "tagLPR": 1
  }
}
```

Figure 3. Change Tag Scheduler



```
{
  "service_settings":
  {
    "wait_seconds_before_exit":    -1,                                       // wait seconds before exit; if < 0, means no exit
    "mvnc_log_level":              3,                                        // log level of mvnc api
    "common_timeout":              3000,                                     // common timeout in milliseconds for mvnc
    "alloc_graph_timeout":         12000,                                    // alloc graph timeout in milliseconds for mvnc
    "max_cached_graph_number":     4,                                        // max amount of graphs can be cached (not to be run at the same time) on
    "update_timetaken_interval":   1000,                                     // time interval in milliseconds between two timetaken updating operations
    "task_scheduler":              "polling",                                // task scheduling methods, options: {"fcfs", "polling"}
    "server_max_task_number":      2000,                                     // max waiting task number at service end
    "client_max_task_number":      0,                                         // max task number at service end for each client; if 0, means no limit
    "graph_tag_map":{                                                        // graph tag mapping, containing entry like: "graphTagA": 2
      "tagDetect":                 6,
      "tagAttr":                   1,
      "tagLPR":                    1
    }
  },
```

3. Restart the system.
4. Start the hddldaemon.

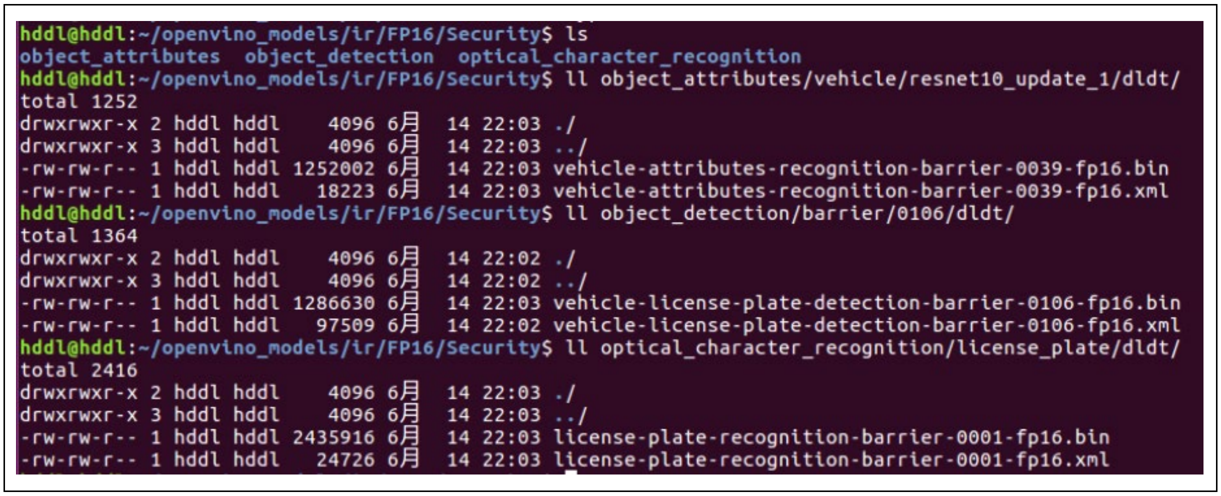
You are ready to Run the Inference Pipeline Verification Sample Application.

4.2 Run the Security Barrier Camera Sample Application

The Security Barrier Camera Sample Application script:

- Downloads three pre-trained models Intermediate Representations.
- Builds and runs the Security Barrier Camera Demo application.
- Uses vehicle recognition in which vehicle attributes build on each other to narrow in on a specific attribute:
  - With the vehicle license plate detection model, an object is identified as a vehicle. This identification provides input to the vehicle attributes recognition model.
  - The vehicle attributes recognition model identifies specific vehicle attributes, including the license plate. The license plate attributes provide input to the license plate recognition model
  - The license plate recognition model recognizes specific characters in the license plate.

Figure 4. Location of Three Pre-Trained IRs



```
hddl@hddl:~/openvino_models/ir/FP16/Security$ ls
object_attributes  object_detection  optical_character_recognition
hddl@hddl:~/openvino_models/ir/FP16/Security$ ll object_attributes/vehicle/resnet10_update_1/dldt/
total 1252
drwxrwxr-x 2 hddl hddl    4096 6月  14 22:03 ./
drwxrwxr-x 3 hddl hddl    4096 6月  14 22:03 ../
-rw-rw-r-- 1 hddl hddl 1252002 6月  14 22:03 vehicle-attributes-recognition-barrier-0039-fp16.bin
-rw-rw-r-- 1 hddl hddl   18223 6月  14 22:03 vehicle-attributes-recognition-barrier-0039-fp16.xml
hddl@hddl:~/openvino_models/ir/FP16/Security$ ll object_detection/barrier/0106/dldt/
total 1364
drwxrwxr-x 2 hddl hddl    4096 6月  14 22:02 ./
drwxrwxr-x 3 hddl hddl    4096 6月  14 22:02 ../
-rw-rw-r-- 1 hddl hddl 1286630 6月  14 22:03 vehicle-license-plate-detection-barrier-0106-fp16.bin
-rw-rw-r-- 1 hddl hddl   97509 6月  14 22:02 vehicle-license-plate-detection-barrier-0106-fp16.xml
hddl@hddl:~/openvino_models/ir/FP16/Security$ ll optical_character_recognition/license_plate/dldt/
total 2416
drwxrwxr-x 2 hddl hddl    4096 6月  14 22:03 ./
drwxrwxr-x 3 hddl hddl    4096 6月  14 22:03 ../
-rw-rw-r-- 1 hddl hddl 2435916 6月  14 22:03 license-plate-recognition-barrier-0001-fp16.bin
-rw-rw-r-- 1 hddl hddl   24726 6月  14 22:03 license-plate-recognition-barrier-0001-fp16.xml
```

To verify the workload distribution, use these steps to build and run the Security Barrier Camera script that was provided with the OpenVINO™ toolkit. The sample uses the `car.png` image in `/opt/intel/openvino/deployment_tools/demo`.

1. Go to the Inference Engine demo directory:

   ```
   cd /opt/intel/openvino/deployment_tools/demo
   ```

2. Run the Security Barrier Camera Verification script:

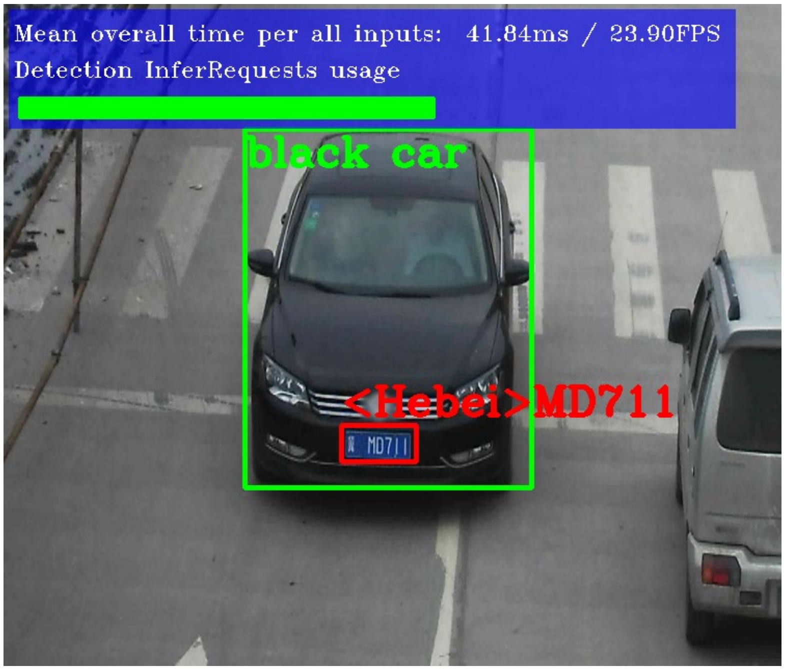
   ```
   ./demo_security_barrier_camera.sh -d HDDL -sample-options -tag
   ```

The three models are used as follows:

- To identify a vehicle, the vehicle license plate detection model runs on six VPUs with the tag, "tagDetect".
- To identify the license plate, the vehicle attributes recognition model runs on one VPU with the tag, "tagAttr".
- To identify characters in the license plate, the license plate recognition model runs on one VPU with the tag, "tagLPR".

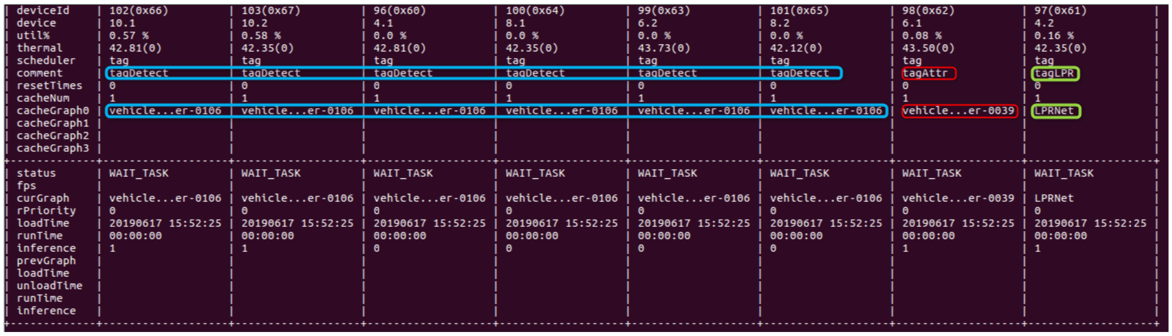
When the verification script completes, you see an image that displays the resulting frame with detections rendered as bounding boxes, and text:

Figure 5. Security Barrier Camera Sample Results


Mean overall time per all inputs: 41.84ms / 23.90FPS
Detection InferRequests usage
black car
<Hebei>MD711
冀 MD711

A screen also displays workload distribution across the VPUs.

Figure 6. VPU Workload Distribution



```
| deviceId    | 102(0x66)         | 103(0x67)         | 96(0x60)          | 100(0x64)         | 99(0x63)          | 101(0x65)         | 98(0x62)          | 97(0x61)          |
| device      | 10.1              | 10.2              | 4.1               | 8.1               | 6.2               | 8.2               | 6.1               | 4.2               |
| util%       | 0.57 %            | 0.58 %            | 0.0 %             | 0.0 %             | 0.0 %             | 0.0 %             | 0.08 %            | 0.16 %            |
| thermal     | 42.81(0)          | 42.35(0)          | 42.81(0)          | 42.35(0)          | 43.73(0)          | 42.12(0)          | 43.50(0)          | 42.35(0)          |
| scheduler   | tag               | tag               | tag               | tag               | tag               | tag               | tag               | tag               |
| comment     | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagAttr           | tagLPR            |
| resetTimes  | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 |
| cacheNum    | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 |
| cacheGraph0 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0039 | LPRNet            |
| cacheGraph1 |                   |                   |                   |                   |                   |                   |                   |                   |
| cacheGraph2 |                   |                   |                   |                   |                   |                   |                   |                   |
| cacheGraph3 |                   |                   |                   |                   |                   |                   |                   |                   |
+-------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+
| status      | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         |
| fps         |                   |                   |                   |                   |                   |                   |                   |                   |
| curGraph    | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0039 | LPRNet            |
| rPriority   | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 |
| loadTime    | 20190617 15:52:25 | 20190617 15:52:25 | 20190617 15:52:25 | 20190617 15:52:25 | 20190617 15:52:25 | 20190617 15:52:25 | 20190617 15:52:25 | 20190617 15:52:25 |
| runTime     | 00:00:00          | 00:00:00          | 00:00:00          | 00:00:00          | 00:00:00          | 00:00:00          | 00:00:00          | 00:00:00          |
| inference   | 1                 | 1                 | 0                 | 0                 | 0                 | 0                 | 1                 | 1                 |
| prevGraph   |                   |                   |                   |                   |                   |                   |                   |                   |
| loadTime    |                   |                   |                   |                   |                   |                   |                   |                   |
| unloadTime  |                   |                   |                   |                   |                   |                   |                   |                   |
| runTime     |                   |                   |                   |                   |                   |                   |                   |                   |
| inference   |                   |                   |                   |                   |                   |                   |                   |                   |
+-------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+
```

5.0 Run Combinations of Neural Networks across Multiple Accelerator Cards

Purpose: Verify the workload distribution of different neural network across multiple VPUs.

OpenVINO™ toolkit Sample Application: Security Barrier Camera Demo

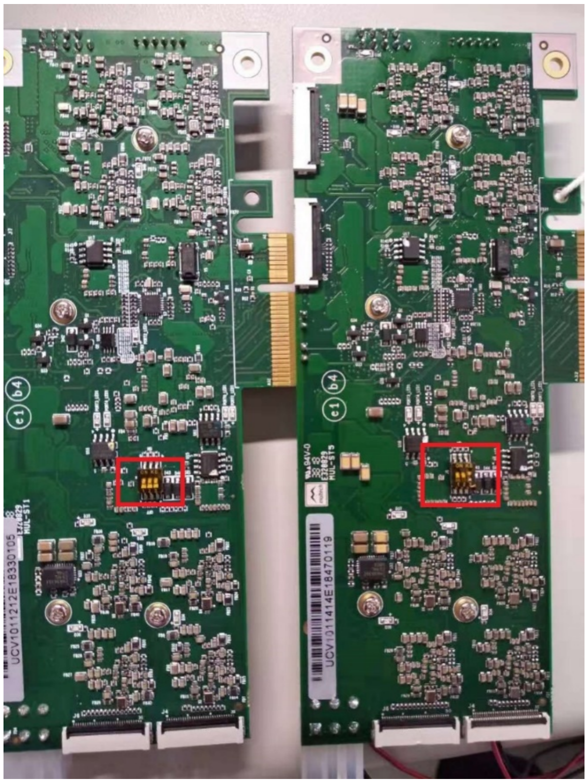
5.1 Preparation

5.1.1 Hardware

Note: These steps assume you are using two Intel® Vision Accelerator Design with Intel® Movidius™ VPU add-in cards.

1. Power down and unplug the system.
2. Remove the system cover and the add-in cards.
3. Set the DIP switches on the cards to differ from each other. See the following illustration for the hardware location.

Figure 7. DIP Switch Location on Intel® Vision Accelerator Design with Intel® Movidius™ VPU Add-in Cards



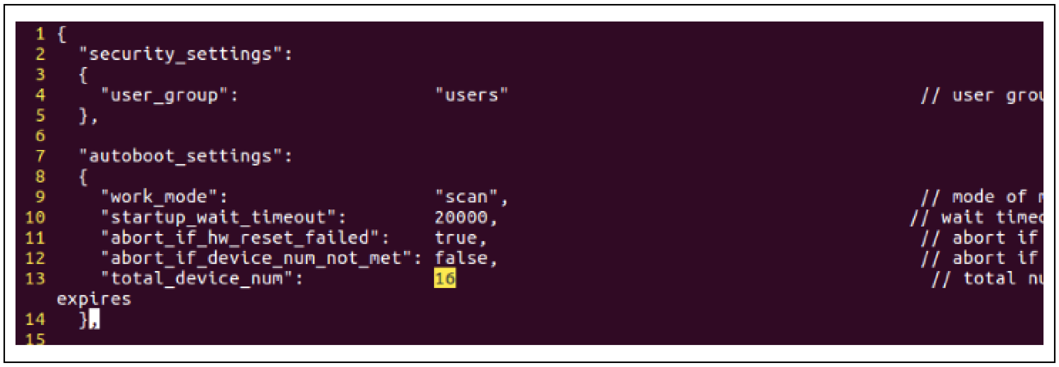
4. Install the add-in cards and the system cover.
5. Plug in and power on the system.

5.1.2 Software

1. If you use temporary OpenVINO® toolkit environment settings, Set the Environment Variables.
2. Edit `${HDDL_INSTALL_DIR}/config/hddl_autoboot.config` to change the total device number:

```
"total_device_num":    "16",
```

Figure 8. Autoboot Configuration File change



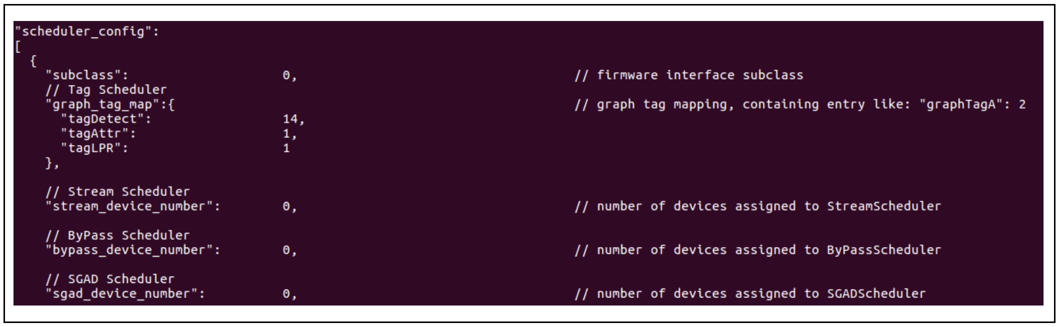
```
{
  "security_settings":
  {
    "user_group":                "users"                                    // user grou
  },

  "autoboot_settings":
  {
    "work_mode":                 "scan",                                    // mode of r
    "startup_wait_timeout":      20000,                                    // wait timeo
    "abort_if_hw_reset_failed":  true,                                      // abort if
    "abort_if_device_num_not_met": false,                                   // abort if
    "total_device_num":          16                                          // total nu
expires
  },
```

3. Edit `${HDDL_INSTALL_DIR}/config/hddl_service.config` to change the tag scheduler:

```
"scheduler_config":
{
  // Tag Scheduler
  "graph_tag_map":{
    "tagDetect": 14,
    "tagAttr": 1,
    "tagLPR": 1
  }
}
```

Figure 9. Tag Scheduler Change in Service Configuration File



```
"scheduler_config":
[
  {
    "subclass":                 0,                                  // firmware interface subclass
    // Tag Scheduler
    "graph_tag_map":{                                               // graph tag mapping, containing entry like: "graphTagA": 2
      "tagDetect":              14,
      "tagAttr":                1,
      "tagLPR":                 1
    },

    // Stream Scheduler
    "stream_device_number":     0,                                  // number of devices assigned to StreamScheduler

    // ByPass Scheduler
    "bypass_device_number":     0,                                  // number of devices assigned to ByPassScheduler

    // SGAD Scheduler
    "sgad_device_number":       0,                                  // number of devices assigned to SGADScheduler
```

4. Restart the system.
5. Start the hddldaemon.

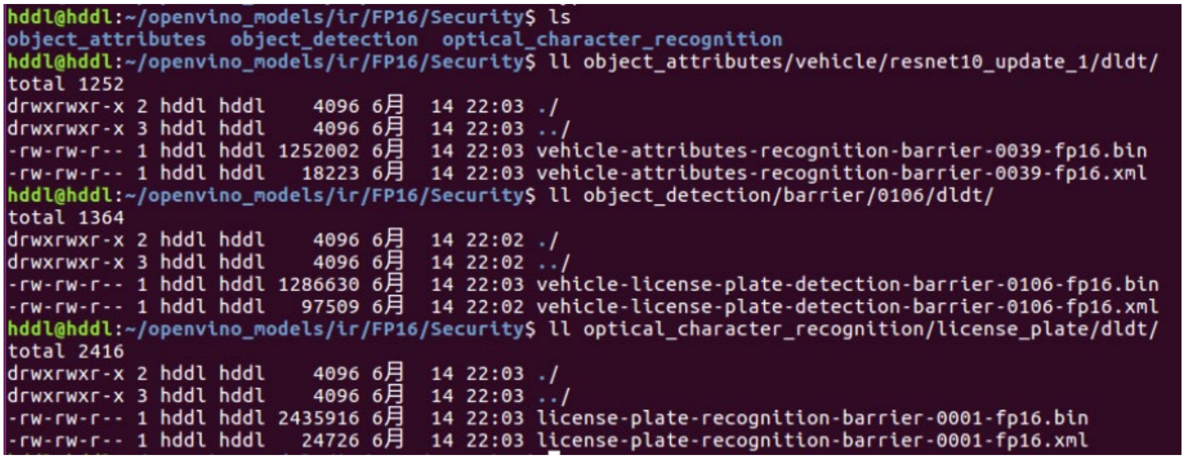
You are ready to Run the Security Barrier Camera Sample Application.

5.2 Run the Security Barrier Camera Sample Application

The Security Barrier Camera Sample Application script:

- Downloads three pre-trained models Intermediate Representations.
- Builds and runs the Security Barrier Camera Demo application.
- Uses vehicle recognition in which vehicle attributes build on each other to narrow in on a specific attribute:
  - With the vehicle license plate detection model, an object is identified as a vehicle. This identification provides input to the second model.
  - The vehicle attributes recognition model identifies specific vehicle attributes, including the license plate. The license plate attributes provide input to the third model
  - The license plate recognition model recognizes specific characters in the license plate.

Figure 10. Location of Three Pre-Trained IRs



```
hddl@hddl:~/openvino_models/ir/FP16/Security$ ls
object_attributes  object_detection  optical_character_recognition
hddl@hddl:~/openvino_models/ir/FP16/Security$ ll object_attributes/vehicle/resnet10_update_1/dldt/
total 1252
drwxrwxr-x 2 hddl hddl    4096 6月  14 22:03 ./
drwxrwxr-x 3 hddl hddl    4096 6月  14 22:03 ../
-rw-rw-r-- 1 hddl hddl 1252002 6月  14 22:03 vehicle-attributes-recognition-barrier-0039-fp16.bin
-rw-rw-r-- 1 hddl hddl   18223 6月  14 22:03 vehicle-attributes-recognition-barrier-0039-fp16.xml
hddl@hddl:~/openvino_models/ir/FP16/Security$ ll object_detection/barrier/0106/dldt/
total 1364
drwxrwxr-x 2 hddl hddl    4096 6月  14 22:02 ./
drwxrwxr-x 3 hddl hddl    4096 6月  14 22:02 ../
-rw-rw-r-- 1 hddl hddl 1286630 6月  14 22:03 vehicle-license-plate-detection-barrier-0106-fp16.bin
-rw-rw-r-- 1 hddl hddl   97509 6月  14 22:02 vehicle-license-plate-detection-barrier-0106-fp16.xml
hddl@hddl:~/openvino_models/ir/FP16/Security$ ll optical_character_recognition/license_plate/dldt/
total 2416
drwxrwxr-x 2 hddl hddl    4096 6月  14 22:03 ./
drwxrwxr-x 3 hddl hddl    4096 6月  14 22:03 ../
-rw-rw-r-- 1 hddl hddl 2435916 6月  14 22:03 license-plate-recognition-barrier-0001-fp16.bin
-rw-rw-r-- 1 hddl hddl   24726 6月  14 22:03 license-plate-recognition-barrier-0001-fp16.xml
```

To verify the workload distribution, use these steps to build and run the Security Barrier Camera script that was provided with the OpenVINO™ toolkit. The sample uses the `car.png` image in `/opt/intel/openvino/deployment_tools/demo`.

1. Go to the Inference Engine demo directory:

```
cd /opt/intel/openvino/deployment_tools/demo
```

2. Run the Security Barrier Camera Verification script:

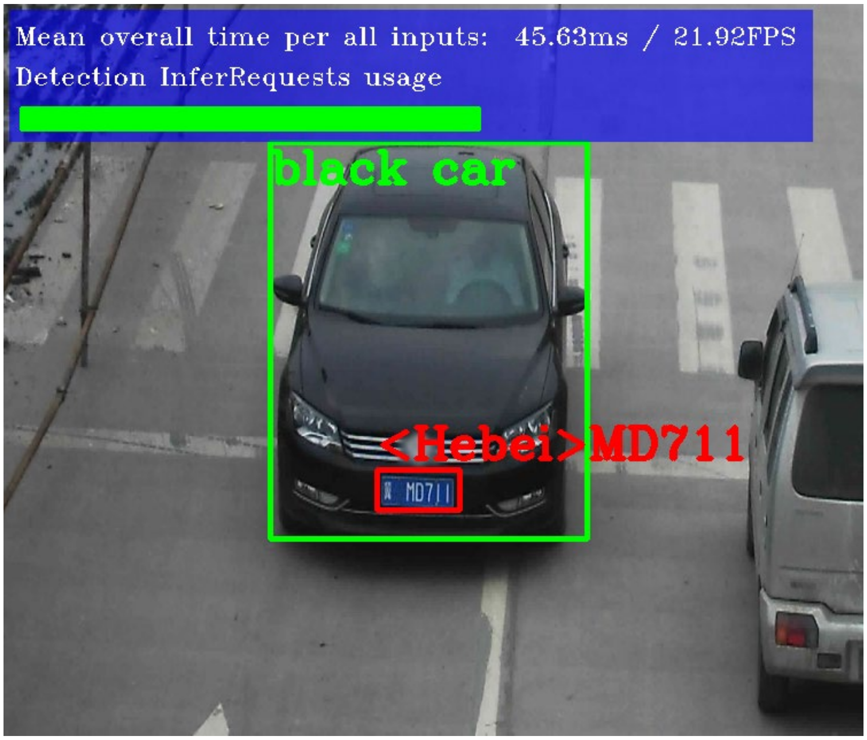
```
./ demo_security_barrier_camera.sh -d HDDL -sample-options -
tag
```

The three models are used as follows:

- To identify a vehicle, the vehicle license plate detection model runs on 14 VPUs with tag "tagDetect".
- To identify the license plate, the vehicle attributes recognition model runs on one VPU with tag "tagAttr".
- To identify characters in the license plate, the license plate recognition model runs on one VPU with tag "tagLPR".

When the verification script completes, you see an image that displays the resulting frame with detections rendered as bounding boxes, and text:

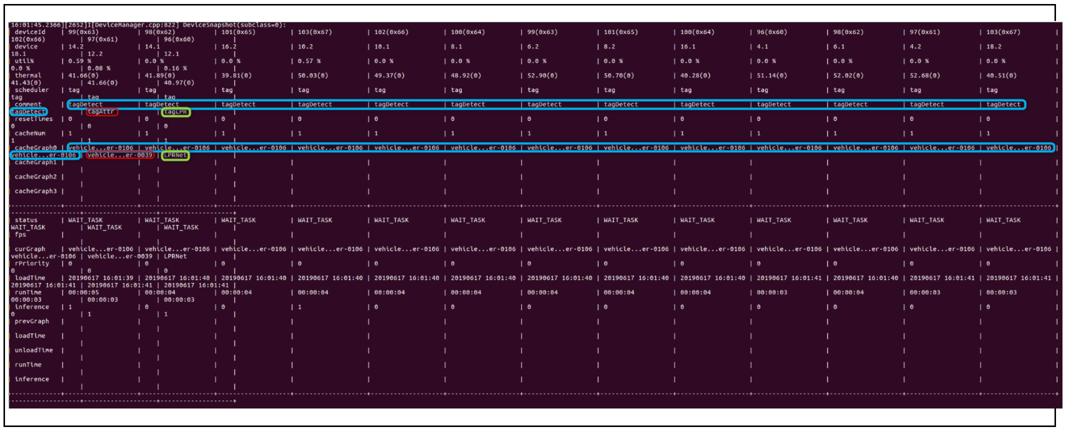
Figure 11. Security Barrier Camera Sample Results



```
Mean overall time per all inputs:  45.63ms / 21.92FPS
Detection InferRequests usage
```

A screen also displays VPU workload distribution.

Figure 12. VPU Workload Distribution



```
16:01:45.2366][2652]I[DeviceManager.cpp:822] DeviceSnapshot(subclass=0):
 deviceId    | 99(0x63)          | 98(0x62)          | 101(0x65)         | 103(0x67)         | 102(0x66)         | 100(0x64)         | 99(0x63)          | 101(0x65)         | 100(0x64)         | 96(0x60)          | 98(0x62)          | 97(0x61)          | 103(0x67)         |
102(0x66)         | 97(0x61)          | 96(0x60)          |
 device      | 14.2              | 14.1              | 16.2              | 10.2              | 10.1              | 8.1               | 6.2               | 8.2               | 16.1              | 4.1               | 6.1               | 4.2               | 18.2              |
18.1              | 12.2              | 12.1              |
 util%       | 0.59 %            | 0.0 %             | 0.0 %             | 0.57 %            | 0.0 %             | 0.0 %             | 0.0 %             | 0.0 %             | 0.0 %             | 0.0 %             | 0.0 %             | 0.0 %             | 0.0 %             |
0.0 %             | 0.08 %            | 0.16 %            |
 thermal     | 41.66(0)          | 41.89(0)          | 39.81(0)          | 50.03(0)          | 49.37(0)          | 48.92(0)          | 52.90(0)          | 50.70(0)          | 40.28(0)          | 51.14(0)          | 52.02(0)          | 52.68(0)          | 40.51(0)          |
41.43(0)          | 41.66(0)          | 40.97(0)          |
 scheduler   | tag               | tag               | tag               | tag               | tag               | tag               | tag               | tag               | tag               | tag               | tag               | tag               | tag               |
tag               | tag               | tag               |
 comment     | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         | tagDetect         |
tagDetect         | tagAttr           | tagLPR            |
 resetTimes  | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 |
0                 | 0                 | 0                 |
 cacheNum    | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 | 1                 |
1                 | 1                 | 1                 |
 cacheGraph0 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 |
vehicle...er-0106 | vehicle...er-0039 | LPRNet            |
 cacheGraph1 |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
 cacheGraph2 |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
 cacheGraph3 |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
-------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+
------------------+-------------------+-------------------+
 status      | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         | WAIT_TASK         |
WAIT_TASK         | WAIT_TASK         | WAIT_TASK         |
 fps         |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
 curGraph    | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 | vehicle...er-0106 |
vehicle...er-0106 | vehicle...er-0039 | LPRNet            |
 rPriority   | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 |
0                 | 0                 | 0                 |
 loadTime    | 20190617 16:01:39 | 20190617 16:01:40 | 20190617 16:01:40 | 20190617 16:01:40 | 20190617 16:01:40 | 20190617 16:01:40 | 20190617 16:01:40 | 20190617 16:01:40 | 20190617 16:01:40 | 20190617 16:01:41 | 20190617 16:01:40 | 20190617 16:01:41 | 20190617 16:01:41 |
20190617 16:01:41 | 20190617 16:01:41 | 20190617 16:01:41 |
 runTime     | 00:00:05          | 00:00:04          | 00:00:04          | 00:00:04          | 00:00:04          | 00:00:04          | 00:00:04          | 00:00:04          | 00:00:04          | 00:00:03          | 00:00:04          | 00:00:03          | 00:00:03          |
00:00:03          | 00:00:03          | 00:00:03          |
 inference   | 1                 | 0                 | 0                 | 1                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 | 0                 |
0                 | 1                 | 1                 |
 prevGraph   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
 loadTime    |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
 unloadTime  |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
 runTime     |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
 inference   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
                  |                   |                   |
-------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+
------------------+-------------------+-------------------+
```