Qualcomm Developer Project DriverWarn-demo2

Project Submission

|  |  |  |
| --- | --- | --- |
| **Attribution:** |  | |
| **Email address** | [zhangzz6687@thundersoft.com](mailto:zhangzz6687@thundersoft.com)  [zhanglei0706@thundersoft.com](mailto:sunzhen@thundersoft.com)  Yuandk0305@thundersoft.com | |
| **Project Title**\* | DriverWarn | |
| **Images**  *Upload up to 5 images of your project*  *Please submit/send the original JPEG/PNG files for all images included in the document* | **Qualcomm® QCS610 SoC.png**    [Alt tag: “DriverWarn-dem2 using The Qualcomm® QCS610 SOC Open Kit ”]  **usb.png**   |  | | --- | | **typc** |   [Alt tag: “using the USB line to develop on Qualcomm® QCS610 SOC development board” ]  **charger.jpg**  charger  [Alt tag: “using round-hole charger to power Qualcomm® QCS610 SOC development board”]  **dpline.jpg**  dpline  [Alt tag: “using the universal DP line to connect LED displayer to Qualcomm® QCS610 SOC development board”]  **LED-Displayer.png**  LED-Displayer  [Alt tag: “using a LED Displayer to display the objectTracker-dem2 interface from Qualcomm® QCS610 SOC development board”] | |
| **Description**\*  *High level description of the project* ***(75 words or less)*** | This project relies on the QCS610 development kit, using the AI computing power and image processing capabilities of the development kit to collect images in real time, perform AI reasoning after preprocessing, and output the reasoning results. I use gstreamer and tflite (support NNAPI) to complete the above functions. The front camera collects facial images to detect the driver's fatigue state, and the rear camera is used to display road conditions outside the vehicle. | |
| **Objective**   * *What inspired you to create this project?* * *What is your desired outcome?* | The current deployment of deep learning is extremely hot, so I think it is necessary to deploy deep learning models in embedded devices. In life, fatigue driving causes a lot of traffic accidents, so I thought of this small project of driving warning. | |
| **Materials Required / Parts List / Tools** | Part Name | Link to purchase |
| Qualcomm® QCS610 SOC Open Kit | https://www.thundercomm.com/app\_zh/product/1593776185472315 |
| USB line | https://item.jd.com/40759941966.html |
| Charger |  |
| LCD Displayer |  |
| Charger |  |
|  |  |
|  |  |
| **Source Code / Source Examples / Application Executable**  *Link to open source / shareable code repository* | Description | Link |
| [Source Code](https://github.com/ThunderSoft-XA/C610-objectTracker-demo2) | https://github.com/ThunderSoft-XA/C610-DriverWarn-demo2 |
|  |  |
|  |  |
|  |  |
| **Additional Resources**  *List related links or resources such as websites, videos, presentations, or other materials* | Resource Title | Link or File Name (and provide file) |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |  |
| --- | --- | --- |
| **Build / Assembly Instructions** | Sample outline:   1. Overall design framework and Test environment construction method.     Green box:   1. get image data by gstreamer from Dual Camera   Blue box:   1. Process the data from gst, complete the image correction, and put it in the data pool 2. Manage thread pools and allocate data to threads that need data   Red box:   1. Start the AI runtime by Android NNAPI.Then,load models and get AI result   Black box:   1. Process AI results.And display image processing results by opencv render 2. Software Build Instructions    1. Prepare a PC (Ubuntu 18.04) with Yocto Project;    2. Install adb ;    3. Configure the compilation environment according to the release note document    4. Configure the DP display environment    5. Write BB file(<driver\_warn\_1.0.bb>),Executing the “bitbake DriverWarn” command generates executable file 3. Start DriverWarn-dem2    1. Copy “DriverWran” executable file to project root directory, then “adb push ” whole project file to Qualcomm® QCS610 SOC,ex./data/ dir.    2. Execute adb root && adb shell. Next, enter the DriverWran-dem2 project root directory, execute ./gst\_test. | |
|  | Sample outline:   1. How does it work?   The running process of demo is relatively clear. After the program runs, the environment initialization of GStreamer and tflite runtime will be carried out immediately. Then, the local video stream pipeline will be built and the tflite AI model will be loaded; After receiving the signal from the pipeline, the captured video frame is transformed into picture frame format, and the object is tracked and detected by neural network, and the object coordinate value and category is returned.  int main (int argc, char \*\* argv)  {  MulitGstCamera::GstEnvInit();  GMainLoop \*main\_loop = g\_main\_loop\_new(NULL,false);  IniConf ini\_conf;  memset(&ini\_conf, 0, sizeof(IniConf));  sprintf(ini\_conf.ini\_node, "conf\_info");  get\_ini\_info((char \*)DEFAULT\_GST\_CONFIG,&ini\_conf);  rgb\_object\_frame = std::make\_shared<BufManager<cv::Mat> > ();  for(int i = 0; i < ini\_conf.conf\_count; i++) {  std::cout << "gstreamer config info " << i << ",as follow:" <<std::endl;  gst\_camera\_vec.push\_back(new GstCamera(DEFAULT\_GST\_CONFIG,i));  }  datapool::DataPool <unsigned char> \*data\_pool = new DataPool<unsigned char>(1024,FALSE);  for(auto gst\_camera : gst\_camera\_vec) {  gst\_camera->Init();  // std::thread gst\_thread(&GstCamera::RunGst,gst\_camera);  std::thread gst\_thread([=]{  DEBUG\_FUNC();  gst\_camera->RunGst(data\_pool);  });  gst\_thread.join();  }  memset(&ini\_conf, 0, sizeof(IniConf));  sprintf(ini\_conf.ini\_node, "conf\_info");  get\_ini\_info((char \*)DEFAULT\_AI\_CONFIG,&ini\_conf);  for(int i = 0; i < ini\_conf.conf\_count; i++) {  std::cout << "AI runtime config info " << i << ",as follow:" <<std::endl;  ai\_inference\_vec.push\_back(new AiInference(DEFAULT\_AI\_CONFIG,i));  }  for(auto ai\_inference : ai\_inference\_vec) {  std::cout << "loading tflite model ......" <<std::endl;  ai\_inference->loadTfliteModel();  }  std::thread handleThread(getDataFromSample);  handleThread.detach();  std::this\_thread::sleep\_for(std::chrono::seconds(1));  std::thread inferenceThread\_landmark(ai2nfrencelandmark);  inferenceThread\_landmark.detach();  std::thread inferenceThread\_sign(ai2nfrencesign);  inferenceThread\_sign.detach();  std::this\_thread::sleep\_for(std::chrono::seconds(2));  std::thread showThread(result\_show);  showThread.join();g\_main\_loop\_run(main\_loop);  MulitGstCamera::GstEnvDeinit();  g\_main\_loop\_unref(main\_loop);  } | |
| **Usage Instructions** | The Demo running results are as follows：  configure file parse result:    traffic\_sign\_detect.tflite AI model result:    face\_landmark.tflite AI model result:  (There are a total of 468 points with coordinates [x, y, z] )    DP show result:  show_157 | |
| **Contributor(s) Info**  *Feel free to include headshots!* | Name | Title  Company |
|  |  |
|  |  |
|  |  |

––– Continued on next page –––

Filters and Tags for QDN projects page

|  |  |  |
| --- | --- | --- |
| **Platform/Hardware** | CSR 101x/102x Bluetooth  DragonBoard 410c  mangOH Red/Yellow  √ Qualcomm QCS610 SoC | MDM920x LTE for IoT  QCA-402x WiFi/BLE/Zigbee  Qualcomm Robotics RBx Dev Kit |
| **Software Tools** | 3D Audio Plugin for Unity  Adreno GPU SDK  Hexagon DSP SDK | Neural Processing SDK for AI  　Snapdragon Profiler |
| **Operating System** | Android  √ Linux  ThreadX RTOS | Ubuntu Core  Windows 10 IoT Core |
| **Cloud Services/Platform** | Sierra Wireless AirVantage  Gizwits Cloud Platform  AT&T M2X  IBM Bluemix | IBM Watson IoT  Microsoft Azure IoT  Amazon AWS IoT |
| **Skill Level Required** | Advanced  Beginner  √ Intermediate |  |
| **Areas of Focus** | 3D Printing & Modeling  Alexa Voice Service  √ Artificial Intelligence  Bluetooth  √ Computer Vision  Digital Signage  Education  √ Embedded  Gaming | Healthcare  √ IoT  √ Robotics  √ Security  Sensors  √ Smart Cities  Smart Home  Toys |

*By submitting your content (“Submission”), you are granting Qualcomm a royalty-free, perpetual, non-exclusive, unrestricted, worldwide license to: (a) post, use, copy, sublicense, adapt, transmit, publicly perform or display any such Submission, (b) use, reproduce, modify, adapt, publish, translate, create derivative works from, distribute, perform, play, host, communicate, make available and publish your Submission without restriction and (c) sublicense to third parties the unrestricted right to exercise any of the foregoing rights granted with respect to the Submission. The foregoing grants shall include the right to exploit any ideas, concepts, intellectual property, or proprietary rights in such Submission, including but not limited to rights under copyright, trademark, servicemark or patent laws under any relevant jurisdiction without Qualcomm owing any monies to you whatsoever. You represent and warrant that you own all right, title and interest in and to the Submission, or you have been granted sufficient rights in and to the Submission allowing the foregoing use of such Submission.*