Xilinx AI SDK Programming Guide

UG1355(v1.0) April 04,2019

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
04/04/2019 Version 1.0	
General updates	Initial Xilinx release

Contents

Chapter 1

Overview

1.1 The Xilinx AI SDK

The Xilinx AI SDK is a set of high level libraries built on top of DNNDK (Deep Neural Network Development Kit) and DPU (Deep-learning Processor Unit). By encapsulates a large number of efficient and high-quality neural networks in the use of DNNDK, the Xilinx AI SDK provides a simple and easy-to-use unified interfaces, which make it easy for users without deep learning knowledge and FPGA knowledge to use deep learning neural networks. the Xilinx AI SDK allows users to focus more on the development of the business layer rather than the underlying hardware.

1.1.1 The Xilinx AI SDK Block Diagram

The XILINX AI SDK block diagram is shown in the following figure.

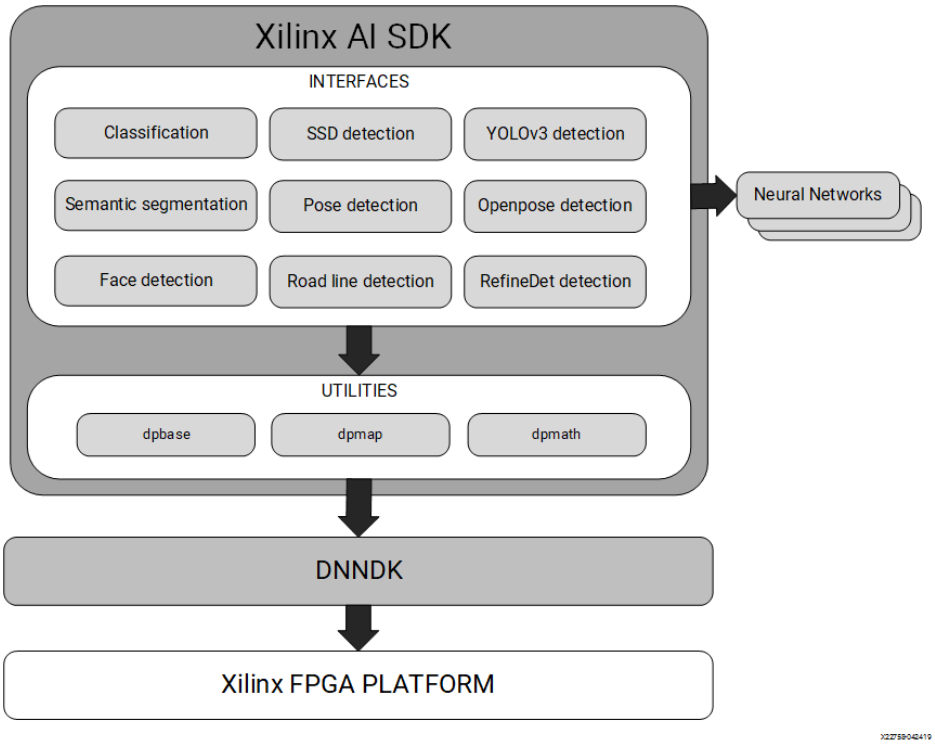

Figure 1.1: the Xilinx AI SDK Block Diagram

The user calls the Xilinx AI SDK through the interface to send the image(cv::Mat) to the neural network. the Xilinx

AI SDK will call DNNDK to pruning, quantization,compilation, optimization and so on. Finally, the neural network will runing on the DPU to get the neural network return results.

1.1.2 The Xilinx AI SDK Features

The Xilinx AI SDK Features are listed here.

- Full stack
- Embedded
- Optimized
- Unified interface
- Practical application

1.2 How to use

Development language using C++.

First , the users needs to prepare the development board and crosscompilation environment. For detailed environment construction, please refer to the Xilinx AI SDK User Guide .

During the development of the user , you need to pay attention to the header files, library files and caffe model library files. These files in the development environment must match the version provided in the SDK.

Refer to the figure below for dependencies between libraries:

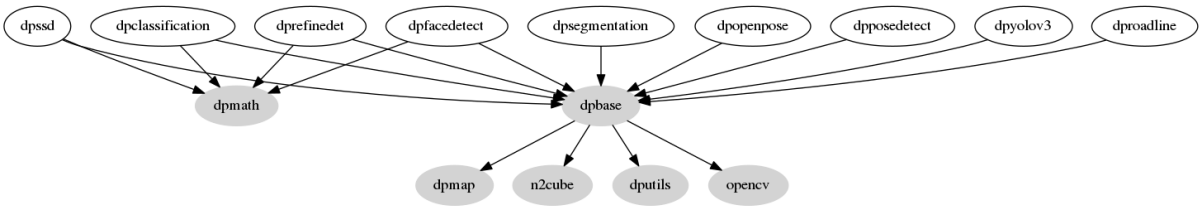

Figure 1.2: the Xilinx AI SDK dependencies between libraries

The grey module is the underlying dependency and does not open the interface for this part.

This libraies can running the FPAG plateform, support ZCU102 , ZCU104 ,Ultra96 and so on.

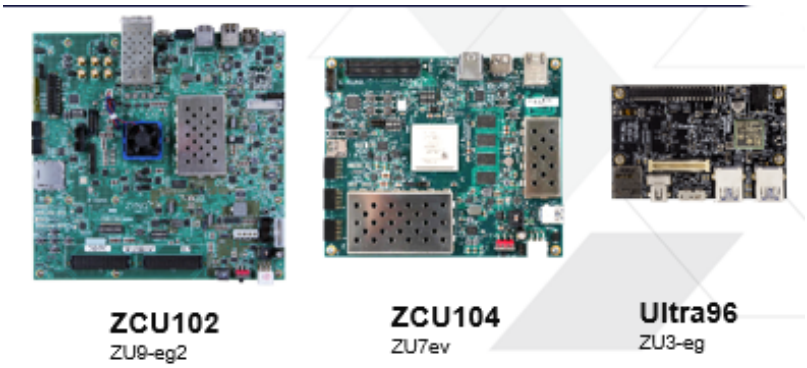

Figure 1.3: Support FPGA Platform

1.2.1 The basic Process

The basic process:

- select a image (cv::Mat).
- call the create method privided by the corresponding library to get class instance. If need_mean_scale_process set as `false` , the model will not minus its mean and scale. please use it only int the pre-minus means and scale.
- call getInputWidth() and getInputHeight() to get the network need cols and rows of the input image.
- resize image to inputWidth x inputHeight
- call run() to get result of the network.

1.2.2 Table of Libraries

Network Name	Header file/c++ classes/Library files/modle files	Description
dpfacedetect	xilinx/facedetect/detect.hpp xilinx::facedetect::Detect libdpfacedetect.so∗ libdpumodeltiling_v6_320.so libdpumodeltiling_v6_640.so	face detection. including: DENSE_BOX_320x320 DENSE_BOX_640x360
dpssd	xilinx/ssd/ssd.hpp xilinx::ssd::SSD libdpssd.so∗ libdpumodelssd_vehicle_v3_480x360.so libdpumodelssd_traffic_480x360.so libdpumodelssd_pedestrian_640x360.so libdpumodelssd_mobilenet_480x360.so libdpumodelssd_mobilenet_v2_480x360.so libdpumodelssd_voc_300x300.so	object detection , including : ADAS_VEHICLE_V3_480x360 TRAFFIC_480x360 ADAS_PEDESTRIAN_640x360 MOBILENET_480x360 MOBILENET_V2_480x360 VOC_300x300_TF
dpclassification	xilinx/classification/classification.hpp xilinx::classification::Classification libdpclassification.so∗ libdpumodelresnet_50.so libdpumodelinception_v1.so libdpumodelinception_v2.so libdpumodelinception_v3.so libdpumodelmobilenet_v2.so libdpumodelresnet_50_tf.so libdpumodelinception_v1_tf.so libdpumodelmobilenet_v2_tf.so	image classification for ImageNet, including: RESNET_50 INCEPTION_V1 INCEPTION_V2 INCEPTION_V3 MOBILENET_V2 RESNET_50_TF INCEPTION_V1_TF MOBILENET_V2_TF
dpyolov3	xilinx/yolov3/yolov3.hpp xilinx::yolov3::YOLOv3 libdpyolov3.so∗ libdpumodelyolov3_adas_512x256.so libdpumodelyolov3_adas_512x288.so libdpumodelyolov3_voc_416.so libdpumodelyolov3_voc_416x416_tf.so	for object detection, including: ADAS_512x256 ADAS_512x288 VOC_416x416 VOC_416x416_TF

dpsegmentation	xilinx/segmentation/segmentation.hpp xilinx::segmentation::Segmentation libdpsegmentation.so∗ libdpumodelfpn_deconv.so	for segmentation
dprefinedet	xilinx/refinedet/refinedet.hpp xilinx::refinedet::RefineDet libdprefinedet.so∗ libdpumodelrefinedet_480x360.so libdpumodelrefinedet_480x360_10G.so libdpumodelrefinedet_480x360_5G.so libdpumodelrefinedet_640x480.so	body detection, including : REFINEDET_480x360 REFINEDET_480x360_10G REFINEDET_480x360_5G REFINEDET_640x480
dproadline	xilinx/roadline/roadline.hpp xilinx::roadline::RoadLine libdproadline.so∗ libdpumodelroadline.so	roadline detection
dpposedetect	xilinx/posedetect/posedetect.hpp xilinx::posedetect::PoseDetect libdpposedetect.so∗ libdpumodelpose2.so	14-pt gesture detection
dpopenpose	xilinx/openpose/openpose.hpp xilinx::openpose::OpenPose libdpopenpose.so∗ libdpumodelopenpose_368x368.so	14-pt gesture detection

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 MultiTaskResult2 Struct Reference

Struct of the result returned by the MultiTask network, when you need to visualize.

`#include <xilinx/multitask/multitask.hpp>`

3.1.1 Detailed Description

Struct of the result returned by the MultiTask network, when you need to visualize.

3.2 xilinx::classification::Classification Class Reference

Base class for detecting objects in the input image (cv::Mat).

`#include <xilinx/classification/classification.hpp>`

Public Member Functions

- **Classification** (const Classification &)=delete
- virtual ClassificationResult run (const cv::Mat &image)=0

  Function of get running result of the Classification neuron network.
- virtual int getInputWidth () const =0

  Function to get InputWidth of the classification network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the classification network (input image rows).

Static Public Member Functions

- static std::unique_ptr
  < Classification > create (Type type, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Classification.
- static std::unique_ptr
  < Classification > create_ex (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Classification.
- static const char * lookup (int index)

  Get the classification corresponding by index.

3.2.1 Detailed Description

Base class for detecting objects in the input image (cv::Mat).

Input is a image (cv::Mat).

Output is index and score of objects in the input image.

- sample code:

```
auto image = cv::imread("test.jpg");
auto network = xilinx::classification::Classification::create
      (
               xilinx::classification::RESNET_50,
               true);
auto result = network->run(image);
for (const auto &r : result.scores) {
   auto score = r.score;
   auto index = network->lookup(r.index);
}
```

3.2.2 Member Function Documentation

3.2.2.1 static std::unique_ptr<Classification> xilinx::classification::Classification::create (Type *type*, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class Classification.

Parameters

type	Enum Type
@param	need_mean_scale_process Normalize with mean/scale or not, default value is true.

Returns

An instance of Classification class.

3.2.2.2 static std::unique_ptr<Classification> xilinx::classification::Classification::create_ex (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class Classification.

Parameters

type	string
@param	need_mean_scale_process Normalize with mean/scale or not, default value is true.

Returns

An instance of Classification class.

3.2.2.3 static const char* xilinx::classification::Classification::lookup (int *index*) [static]

Get the classification corresponding by index.

Parameters

index,the	network result

Returns

classification

3.2.2.4 virtual ClassificationResult xilinx::classification::Classification::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the Classification neuron network.

Parameters

img	Input data of input image (cv::Mat).

Returns

ClassificationResult.

3.2.2.5 virtual int xilinx::classification::Classification::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the classification network (input image cols).

Returns

InputWidth of the classification network

3.2.2.6 virtual int xilinx::classification::Classification::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the classification network (input image rows).

Returns

InputHeight of the classification network.

3.3 xilinx::classification::ClassificationResult Struct Reference

Struct of the result with the classification network.

`#include <xilinx/classification/classification.hpp>`

Classes

- struct Score

Public Attributes

- int width
  width of a input image
- int height
  height of a input image
- std::vector< Score > scores
  all objects, a vector of Score.

3.3.1 Detailed Description

Struct of the result with the classification network.

3.4 xilinx::classification::ClassificationResult::Score Struct Reference

`#include <xilinx/classification/classification.hpp>`

Public Attributes

- int index
  Result's index in ImageNet.
- float score
  Confidence of this category.

3.4.1 Detailed Description

Struct of a classification

3.5 xilinx::facedetect::FaceDetect Class Reference

Base class for detecting the position of faces in the input image (cv::Mat).

`#include <xilinx/facedetect/facedetect.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0
  Function to get InputWidth of the facedetect network (input image cols).
- virtual int getInputHeight () const =0
  Function to get InputHeigth of the facedetect network (input image rows).
- virtual float getThreshold () const =0
  Function to get detect threshold.
- virtual void setThreshold (float threshold)=0
  Function of set detect threshold.
- virtual FaceDetectResult run (const cv::Mat &img)=0
  Function of get running result of the facedetect network.

Static Public Member Functions

- static std::unique_ptr
  < FaceDetect > create (Type type, bool need_preprocess=true)
  Factory function to get a instance of derived classes of class FaceDetect.
- static std::unique_ptr
  < FaceDetect > create_ex (const std::string &model_name, bool need_preprocess=true)
  Factory function to get instance of derived classes of class FaceDetect.

Protected Member Functions

- **FaceDetect** (const FaceDetect &)=delete
- FaceDetect & **operator=** (const FaceDetect &)=delete

3.5.1 Detailed Description

Base class for detecting the position of faces in the input image (cv::Mat).

Input is a image (cv::Mat).

Output is a vector of position and score infomation for faces in the input image.

sample code:

```
auto image = cv::imread("test_faces.jpg");
auto network = xilinx::facedetect::FaceDetect::create(
               xilinx::facedetect::DENSE_BOX_640x360,
               true);
auto result = network->run(image);
for (const auto &r : result) {
   auto score = r.score;
   auto x = r.x * image.cols;
   auto y = r.y * image.rows;
   auto width = r.width * image.cols;
   auto height = rz.height * image.rows;
}
```

Display of the facedetect model results:

Figure 3.1: facedetect result image

3.5.2 Member Function Documentation

3.5.2.1 static std::unique_ptr<FaceDetect> xilinx::facedetect::FaceDetect::create (Type *type*, bool *need_preprocess* = true) [inline], [static]

Factory function to get a instance of derived classes of class FaceDetect.

Support 2 types of input size:

1. width = 640 and height = 360
2. width = 320 and height = 320

Parameters

type	DENSE_BOX_320 or DENSE_BOX_640
need_mean_-scale_process	Normalize with mean/scale or not, default value is true.

Returns

An instance of FaceDetect class.

3.5.2.2 static std::unique_ptr<FaceDetect> xilinx::facedetect::FaceDetect::create_ex (const std::string & *model_name*, bool *need_preprocess* = `true`) [static]

Factory function to get instance of derived classes of class FaceDetect.

Note

for internal use

Parameters

modle_name	Model name
need_mean_-scale_process	Normalize with mean/scale or not, default value is true.

Returns

An instance of FaceDetect class.

3.5.2.3 virtual int xilinx::facedetect::FaceDetect::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the facedetect network (input image cols).

Returns

InputWidth of the facedetect network

3.5.2.4 virtual int xilinx::facedetect::FaceDetect::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the facedetect network (input image rows).

Returns

InputHeight of the facedetect network.

3.5.2.5 virtual float xilinx::facedetect::FaceDetect::getThreshold ( ) const [pure virtual]

Function to get detect threshold.

Returns

detect threshold , the value range from 0 to 1.

3.5.2.6 virtual void xilinx::facedetect::FaceDetect::setThreshold (float *threshold*) [pure virtual]

Function of set detect threshold.

Note

The results will filter by detect threshold (score > threshold).

Parameters

threshold,the	value range from 0 to 1.

3.5.2.7 virtual FaceDetectResult xilinx::facedetect::FaceDetect::run (const cv::Mat & *img*) [pure virtual]

Function of get running result of the facedetect network.

Parameters

img	Input Data , input image (cv::Mat) need to be resized to InputWidth and InputHeight required by the network.

Returns

the results of the face detect network , a collection of FaceDetectResult filter by score >= det_threshold

3.6 xilinx::facedetect::FaceDetectResult Struct Reference

Struct of the result with the facedetect network.

`#include <xilinx/facedetect/facedetect.hpp>`

Classes

- struct BoundingBox
  Struct of a face coordinate and confidence.

Public Attributes

- int width
  width of a input image
- int height
  height of a input image
- std::vector< BoundingBox > rects
  all faces, a vector of BoundingBox

3.6.1 Detailed Description

Struct of the result with the facedetect network.

3.7 xilinx::facedetect::FaceDetectResult::BoundingBox Struct Reference

Struct of a face coordinate and confidence.

`#include <xilinx/facedetect/facedetect.hpp>`

Public Attributes

- float x
- float y
- float width
- float height
- float score

  face confidence, the value range from 0 to 1.

3.7.1 Detailed Description

Struct of a face coordinate and confidence.

3.7.2 Member Data Documentation

3.7.2.1 float xilinx::facedetect::FaceDetectResult::BoundingBox::x

x-coordinate , x is normalized relative to the input image cols ,the value range from 0 to 1.

3.7.2.2 float xilinx::facedetect::FaceDetectResult::BoundingBox::y

y-coordinate , y is normalized relative to the input image rows ,the value range from 0 to 1.

3.7.2.3 float xilinx::facedetect::FaceDetectResult::BoundingBox::width

face width , width is normalized relative to the input image cols , the value range from 0 to 1.

3.7.2.4 float xilinx::facedetect::FaceDetectResult::BoundingBox::height

face height , heigth is normalized relative to the input image rows , the value range from 0 to 1.

3.8 xilinx::multitask::MultiTask Class Reference

Base class for ADAS MuiltTask from a image (cv::Mat).

`#include <xilinx/multitask/multitask.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the multitask network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHight of the multitask network (input image rows).
- virtual MultiTaskResult run_8UC1 (const cv::Mat &image)=0

  Function of get running result from the MultiTask network.
- virtual MultiTaskResult run_8UC3 (const cv::Mat &image)=0

  Function of get running result from the MultiTask network.

Static Public Member Functions

- static std::unique_ptr< MultiTask > create (bool need_preprocess=true)

  Factory function to get a instance of derived classes of class MuliTask.
- static std::unique_ptr< MultiTask > **create_ex** (const std::string &model_name, bool need_preprocess=true)

Protected Member Functions

- **MultiTask** (const MultiTask &)=delete

3.8.1 Detailed Description

Base class for ADAS MuiltTask from a image (cv::Mat).

Input a image (cv::Mat).

Output is struct MultiTaskResult include segmentation results, detection detection results and vehicle towards;

sample code:

```
auto det = xilinx::multitask::MultiTask::create();
auto image = cv::imread("sample_multitask.jpg");
auto result = det->run_8UC3(image);
cv::imwrite("res.jpg",result.segmentation);
```

Display of the multitask model results:

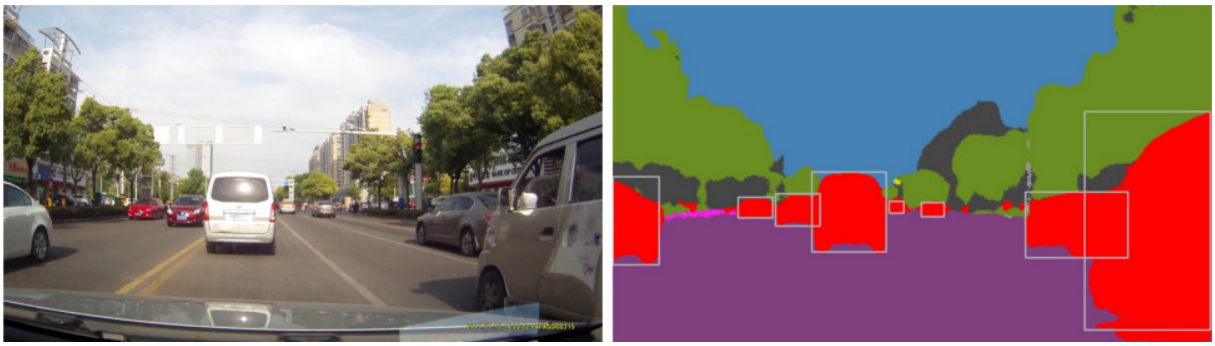

Figure 3.2: multitask visualization result image

3.8.2 Member Function Documentation

3.8.2.1 static std::unique_ptr<MultiTask> xilinx::multitask::MultiTask::create (bool *need_preprocess* = `true`) [inline], [static]

Factory function to get a instance of derived classes of class MuliTask.

Parameters

need_preprocess	normalize with mean/scale or not, default value is true.

Returns

An instance of MultiTask class.

3.8.2.2 virtual int xilinx::multitask::MultiTask::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the multitask network (input image cols).

Returns

InputWidth of the multitask network.

3.8.2.3 virtual int xilinx::multitask::MultiTask::getInputHeight ( ) const [pure virtual]

Function to get InputHight of the multitask network (input image rows).

Returns

InputHeight of the multitask network.

3.8.2.4 virtual MultiTaskResult xilinx::multitask::MultiTask::run_8UC1 (const cv::Mat & *image*) [pure virtual]

Function of get running result from the MultiTask network.

Note

The type is CV_8UC1 of the MultiTaskResult.segmentation.

Parameters

image	Input image

Returns

The struct of MultiTaskResult

3.8.2.5 virtual MultiTaskResult xilinx::multitask::MultiTask::run_8UC3 (const cv::Mat & *image*) [pure virtual]

Function of get running result from the MultiTask network.

Note

The type is CV_8UC3 of the MultiTaskResult.segmentation.

Parameters

image	Input image;

Returns

The struct of MultiTaskResult

3.9 xilinx::multitask::MultiTask8UC1 Class Reference

Base class for ADAS MuiltTask8UC1 from a image (cv::Mat).

`#include <xilinx/multitask/multitask.hpp>`

Public Member Functions

- virtual int getInputWidth () const

  Function to get InputWidth of the multitask network (input image cols).
- virtual int getInputHeight () const

  Function to get InputHight of the multitask network (input image rows).
- virtual MultiTaskResult run (const cv::Mat &image)

  Function of get running result from the MultiTask network.

Static Public Member Functions

- static std::unique_ptr
  < MultiTask8UC1 > create (bool need_preprocess=true)

  Factory function to get a instance of derived classes of class MuliTask8UC1.
- static std::unique_ptr
  < MultiTask8UC1 > **create_ex** (const std::string &model_name, bool need_preprocess=true)

Protected Member Functions

- **MultiTask8UC1** (std::unique_ptr< MultiTask > multitask)
- **MultiTask8UC1** (const MultiTask8UC1 &)=delete

3.9.1 Detailed Description

Base class for ADAS MuiltTask8UC1 from a image (cv::Mat).

Input a image (cv::Mat). Output is struct MultiTaskResult include segmentation results, detection results and vehicle towards; The result cv::Mat type is CV_8UC1

sample code:

```
auto det = xilinx::multitask::MultiTask8UC1::create();
auto image = cv::imread("sample_multitask.jpg");
auto result = det->run(image);
cv::imwrite("res.jpg",result.segmentation);
```

3.9.2 Member Function Documentation

3.9.2.1 static std::unique_ptr<MultiTask8UC1> xilinx::multitask::MultiTask8UC1::create (bool *need_preprocess* = `true`) [inline], [static]

Factory function to get a instance of derived classes of class MuliTask8UC1.

Parameters

need_preprocess	normalize with mean/scale or not, default value is true.

Returns

An instance of MultiTask8UC1 class.

3.9.2.2 virtual int xilinx::multitask::MultiTask8UC1::getInputWidth ( ) const [inline], [virtual]

Function to get InputWidth of the multitask network (input image cols).

Returns

InputWidth of the multitask network.

3.9.2.3 virtual int xilinx::multitask::MultiTask8UC1::getInputHeight ( ) const [inline], [virtual]

Function to get InputHight of the multitask network (input image rows).

Returns

InputHeight of the multitask network.

3.9.2.4 virtual MultiTaskResult xilinx::multitask::MultiTask8UC1::run (const cv::Mat & *image*) [inline], [virtual]

Function of get running result from the MultiTask network.

Note

The type is CV_8UC1 of the MultiTaskResult.segmentation.

Parameters

image	Input image

Returns

The struct of MultiTaskResult

3.10 xilinx::multitask::MultiTask8UC3 Class Reference

Base class for ADAS MuiltTask8UC3 from a image (cv::Mat).

`#include <xilinx/multitask/multitask.hpp>`

Public Member Functions

- virtual int getInputWidth () const
  Function to get InputWidth of the multitask network (input image cols).
- virtual int getInputHeight () const
  Function to get InputHight of the multitask network (input image rows).
- virtual MultiTaskResult run (const cv::Mat &image)
  Function of get running result from the MultiTask network.

Static Public Member Functions

- static std::unique_ptr
  < MultiTask8UC3 > create (bool need_preprocess=true)
  Factory function to get a instance of derived classes of class MuliTask8UC3.
- static std::unique_ptr
  < MultiTask8UC3 > **create_ex** (const std::string &model_name, bool need_preprocess=true)

Protected Member Functions

- **MultiTask8UC3** (std::unique_ptr< MultiTask > multitask)
- **MultiTask8UC3** (const MultiTask8UC3 &)=delete

3.10.1 Detailed Description

Base class for ADAS MuiltTask8UC3 from a image (cv::Mat).

Input a image (cv::Mat). Output is struct MultiTaskResult include segmentation results, detection results and vehicle towards; The result cv::Mat type is CV_8UC3

sample code:

```
auto det = xilinx::multitask::MultiTask8UC3::create();
auto image = cv::imread("sample_multitask.jpg");
auto result = det->run(image);
cv::imwrite("res.jpg",result.segmentation);
```

3.10.2 Member Function Documentation

3.10.2.1 static std::unique_ptr<MultiTask8UC3> xilinx::multitask::MultiTask8UC3::create (bool *need_preprocess* = true) [inline], [static]

Factory function to get a instance of derived classes of class MuliTask8UC3.

Parameters

need_preprocess	normalize with mean/scale or not, default value is true.

Returns

An instance of MultiTask8UC3 class.

3.10.2.2 virtual int xilinx::multitask::MultiTask8UC3::getInputWidth ( ) const [inline], [virtual]

Function to get InputWidth of the multitask network (input image cols).

Returns

InputWidth of the multitask network.

3.10.2.3 virtual int xilinx::multitask::MultiTask8UC3::getInputHeight ( ) const [inline], [virtual]

Function to get InputHight of the multitask network (input image rows).

Returns

InputHeight of the multitask network.

3.10.2.4 virtual MultiTaskResult xilinx::multitask::MultiTask8UC3::run (const cv::Mat & *image*) [inline], [virtual]

Function of get running result from the MultiTask network.

Note

The type is CV_8UC3 of the MultiTaskResult.segmentation.

Parameters

image	Input image

Returns

The struct of MultiTaskResult

3.11 xilinx::multitask::MultiTaskResult Struct Reference

Public Attributes

- int **width**
- int **height**
- std::vector< VehicleResult > **vehicle**
- cv::Mat **segmentation**

3.12 xilinx::multitask::VehicleResult Struct Reference

A struct to define detection result of MultiTask.

`#include <xilinx/multitask/multitask.hpp>`

Public Attributes

- int **label**
- float **score**
- float **x**
- float **y**
- float **width**
- float **height**
- float **angle**

3.12.1 Detailed Description

A struct to define detection result of MultiTask.

3.13 xilinx::openpose::OpenPose Class Reference

Base class for detecting pose from a input image (cv::Mat).

`#include <xilinx/openpose/openpose.hpp>`

Public Member Functions

- **OpenPose** (const OpenPose &)=delete
- virtual OpenPoseResult run (const cv::Mat &image)=0

  Function of get running result of the openpose neuron network.

- virtual int getInputWidth () const =0

  Function to get InputWidth of the openpose network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the openpose network (input image rows).

Static Public Member Functions

- static std::unique_ptr< OpenPose > create (bool need_preprocess=true)

  Factory function to get a instance of derived classes of class OpenPose.
- static std::unique_ptr< OpenPose > create_ex (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class OpenPose.

3.13.1 Detailed Description

Base class for detecting pose from a input image (cv::Mat).

Input a image (cv::Mat).

Output is OpenPoseResult.

sample code:

```
auto image = cv::imread(argv[1]);
if (image.empty()) {
  std::cerr << "cannot load " << argv[1] << std::endl;
  abort();
}
auto det = xilinx::openpose::OpenPose::create();
int width = det->getInputWidth();
int height = det->getInputHeight();
cv::Mat res_img;
cv::resize(image, res_img, cv::Size(width, height));
auto results = det->run(res_img);
for(auto &r : results.poses){
  cv::Point2f a = r.point_a;
  cv::Point2f b = r.point_b;
  a.x = a.x * image.cols;
  a.y = a.y * image.rows;
  b.x = b.x * image.cols;
  b.y = b.y * image.rows;
  cv::circle(image, a, 5, cv::Scalar(0, 255, 0), -1);
  cv::circle(image, b, 5, cv::Scalar(0, 255, 0), -1);
  cv::line(image, a, b, cv::Scalar(255, 0, 0), 3, 4);
}
cv::imwrite("sample_openpose_result.jpg", image);
```

Display of the openpose model results:

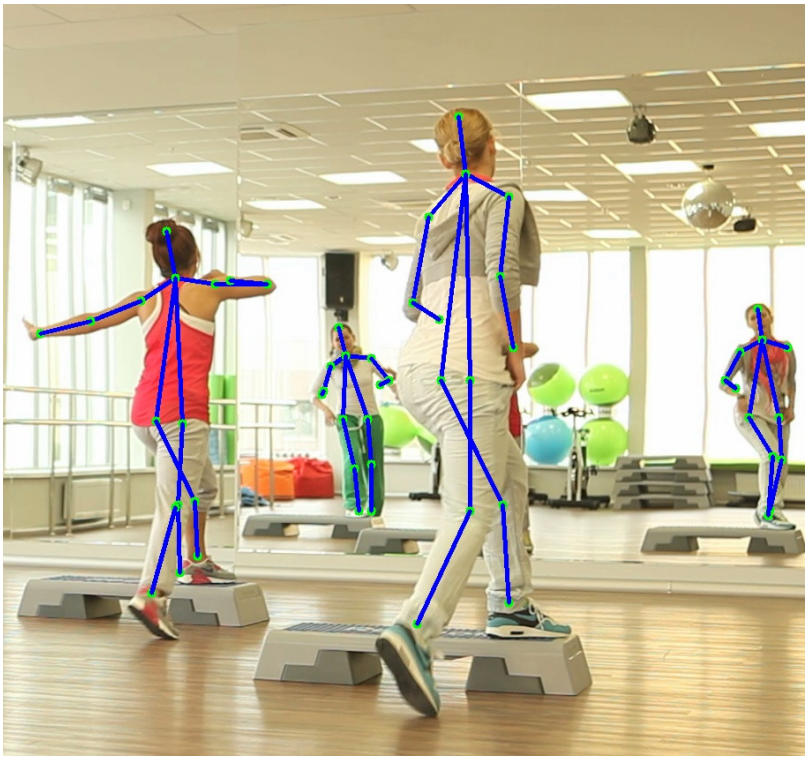

Figure 3.3: openpose image

3.13.2 Member Function Documentation

3.13.2.1 static std::unique_ptr<OpenPose> xilinx::openpose::OpenPose::create (bool *need_preprocess* = `true`) [inline], [static]

Factory function to get a instance of derived classes of class OpenPose.

Parameters

need_mean_-scale_process	Normalize with mean/scale or not, default value is true.

Returns

An instance of OpenPose class.

3.13.2.2 static std::unique_ptr<OpenPose> xilinx::openpose::OpenPose::create_ex (const std::string & *model_name*, bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class OpenPose.

Parameters

model_name	openpose_368x368
need_mean_-scale_process	Normalize with mean/scale or not, default value is true.

Returns

An instance of OpenPose class.

3.13.2.3 virtual OpenPoseResult xilinx::openpose::OpenPose::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the openpose neuron network.

Parameters

img	Input data of input image (cv::Mat).

Returns

OpenPoseResult.

3.13.2.4 virtual int xilinx::openpose::OpenPose::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the openpose network (input image cols).

Returns

InputWidth of the openpose network

3.13.2.5 virtual int xilinx::openpose::OpenPose::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the openpose network (input image rows).

Returns

InputHeight of the openpose network.

3.14 xilinx::openpose::OpenPoseResult Struct Reference

Struct of the result returned by the OpenPoseResult network.

`#include <xilinx/openpose/openpose.hpp>`

Classes

- struct Line

  Struct of the line.

Public Attributes

- int width

  width of input image.
- int height

  height of input image.
- std::vector< Line > poses

  a vector of Line.

3.14.1 Detailed Description

Struct of the result returned by the OpenPoseResult network.

3.15 xilinx::openpose::OpenPoseResult::Line Struct Reference

Struct of the line.

`#include <xilinx/openpose/openpose.hpp>`

Public Attributes

- cv::Point2f point_a
  point A.
- cv::Point2f point_b
  point B.

3.15.1 Detailed Description

Struct of the line.

3.16 xilinx::posedetect::PoseDetect Class Reference

Base class for detecting a pose from a input image (cv::Mat).

`#include <xilinx/posedetect/posedetect.hpp>`

Public Member Functions

- **PoseDetect** (const PoseDetect &)=delete
- virtual int getInputWidth () const =0
  Function to get InputWidth of the PoseDetect network (input image cols).
- virtual int getInputHeight () const =0
  Function to get InputHeigth of the PoseDetect network (input image rows).
- virtual PoseDetectResult run (const cv::Mat &image)=0
  Function of get running result of the posedetect neuron network.

Static Public Member Functions

- static std::unique_ptr
  < PoseDetect > create (bool need_preprocess=true)
  Factory function to get a instance of derived classes of class PoseDetect.

3.16.1 Detailed Description

Base class for detecting a pose from a input image (cv::Mat).

Note

support detect a signle pose.

Input a image (cv::Mat).

Output is PoseDetectResult.

sample code:

```
auto det = xilinx::posedetect::PoseDetect::create();
auto image = cv::imread("sample.jpg");
auto results = det->run(image);
for(auto result: results.pose14pt) {
    std::cout << result << std::endl;
}
```

Display of the posedetect model results:

Figure 3.4: pose detect image

3.16.2 Member Function Documentation

3.16.2.1 static std::unique_ptr<PoseDetect> xilinx::posedetect::PoseDetect::create (bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class PoseDetect.

Parameters

need_mean_scale_process	Normalize with mean/scale or not, default value is true.

Returns

An instance of PoseDetect class.

3.16.2.2 virtual int xilinx::posedetect::PoseDetect::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the PoseDetect network (input image cols).

Returns

InputWidth of the PoseDetect network.

3.16.2.3 virtual int xilinx::posedetect::PoseDetect::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the PoseDetect network (input image rows).

Returns

InputHeight of the PoseDetect network.

3.16.2.4 virtual PoseDetectResult xilinx::posedetect::PoseDetect::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the posedetect neuron network.

Parameters

img	Input data of input image (cv::Mat).

Returns

PoseDetectResult.

3.17 xilinx::posedetect::PoseDetectResult Struct Reference

Struct of the result returned by the posedetect network.

`#include <xilinx/posedetect/posedetect.hpp>`

Classes

- struct Point

  Struct of a coordinate point.

Public Types

- using Pose14Pt = std::array< Point, 14 >

Public Attributes

- int width

  width of input image.
- int height

  height of input image.
- Pose14Pt pose14pt

  the pose of input image.

3.17.1 Detailed Description

Struct of the result returned by the posedetect network.

3.17.2 Member Typedef Documentation

3.17.2.1 using xilinx::posedetect::PoseDetectResult::Pose14Pt = std::array<Point, 14>

a pose , represented by 14 coordinate points. 1: R_shoulder, 2: R_elbow, 3: R_wrist, 4: L_shoulder, 5: L_elbow, 6: L_wrist, 7: R_hip, 8: R_knee, 9: R_ankle, 10: L_hip, 11: L_knee, 12: L_ankle, 13: head, 14: neck

3.18 xilinx::posedetect::PoseDetectResult::Point Struct Reference

Struct of a coordinate point.

`#include <xilinx/posedetect/posedetect.hpp>`

Public Attributes

- float x
- float y

3.18.1 Detailed Description

Struct of a coordinate point.

3.18.2 Member Data Documentation

3.18.2.1 float xilinx::posedetect::PoseDetectResult::Point::x

x-coordinate, x is normalized relative to the input image cols ,the value range from 0 to 1.

3.18.2.2 float xilinx::posedetect::PoseDetectResult::Point::y

y-coordinate, y is normalized relative to the input image rows ,the value range from 0 to 1.

3.19 xilinx::refinedet::RefineDet Class Reference

Base class for detecting pedestrian in the input image (cv::Mat).

`#include <xilinx/refinedet/refinedet.hpp>`

Public Member Functions

- **RefineDet** (const RefineDet &)=delete
- virtual RefineDetResult run (const cv::Mat &image)=0

  Function of get running result of the RefineDet neuron network.
- virtual int getInputWidth () const =0

  Function to get InputWidth of the refinedet network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the refinedet network (input image rows).

Static Public Member Functions

- static std::unique_ptr< RefineDet > create (Type type=REFINEDET_480x360, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class RefineDet.
- static std::unique_ptr< RefineDet > create_ex (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class RefineDet.

3.19.1 Detailed Description

Base class for detecting pedestrian in the input image (cv::Mat).

Input is a image (cv::Mat).

Output is position and score of pedestrian in the input image.

sample code:

```
auto image = cv::imread("sample_refinedet.jpg");
auto network = deephi::refinedet::RefineDet::create(
               deephi::refinedet::RefineDet::REFINEDET_640x480,
               true);
auto results = det->run(image);
for (const auto &r : results.bboxes) {
   auto score = r.score;
   auto x = r.x * image.cols;
   auto y = r.y * image.rows;
   auto width = r.width * image.cols;
   auto height = r.height * image.rows;
}
```

Display of the refinedet_REFINEDET_640x480 model results:

Figure 3.5: REFINEDET_640x360 detect result

3.19.2 Member Function Documentation

3.19.2.1 static std::unique_ptr<RefineDet> xilinx::refinedet::RefineDet::create (Type *type* = REFINEDET_480x360, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class RefineDet.

Parameters

type	Enum Type
@param	need_mean_scale_process Normalize with mean/scale or not, default value is true.

Returns

An instance of RefineDet class.

3.19.2.2 static std::unique_ptr<RefineDet> xilinx::refinedet::RefineDet::create_ex (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class RefineDet.

Note

for internal use

Parameters

type	string
@param	need_mean_scale_process Normalize with mean/scale or not, default value is true.

Returns

An instance of RefineDet class.

3.19.2.3 virtual RefineDetResult xilinx::refinedet::RefineDet::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the RefineDet neuron network.

Parameters

img	Input data of input image (cv::Mat).

Returns

A vector of RefineDetResult.

3.19.2.4 virtual int xilinx::refinedet::RefineDet::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the refinedet network (input image cols).

Returns

InputWidth of the refinedet network

3.19.2.5 virtual int xilinx::refinedet::RefineDet::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the refinedet network (input image rows).

Returns

InputHeight of the refinedet network.

3.20 xilinx::refinedet::RefineDetResult Struct Reference

Struct of the result with the refinedet network.

`#include <xilinx/refinedet/refinedet.hpp>`

Classes

- struct BoundingBox

  Struct of a object coordinate and confidence.

Public Attributes

- int width

  width of the input image.
- int height

  height of the input image.
- std::vector< BoundingBox > bboxes

  the vector of BoundingBox.

3.20.1 Detailed Description

Struct of the result with the refinedet network.

3.21 xilinx::refinedet::RefineDetResult::BoundingBox Struct Reference

Struct of a object coordinate and confidence.

`#include <xilinx/refinedet/refinedet.hpp>`

Public Attributes

- float x
- float y
- float width
- float height
- float score

  body detection confidence, the value range from 0 to 1.

3.21.1 Detailed Description

Struct of a object coordinate and confidence.

3.21.2 Member Data Documentation

3.21.2.1 float xilinx::refinedet::RefineDetResult::BoundingBox::x

x-coordinate , x is normalized relative to the input image cols ,the value range from 0 to 1.

3.21.2.2 float xilinx::refinedet::RefineDetResult::BoundingBox::y

y-coordinate , y is normalized relative to the input image rows ,the value range from 0 to 1.

3.21.2.3 float xilinx::refinedet::RefineDetResult::BoundingBox::width

body width , width is normalized relative to the input image cols , the value range from 0 to 1.

3.21.2.4 float xilinx::refinedet::RefineDetResult::BoundingBox::height

body height , heigth is normalized relative to the input image rows , the value range from 0 to 1.

3.22 xilinx::roadline::RoadLine Class Reference

Base class for detecting roadline from a image (cv::Mat).

`#include <xilinx/roadline/roadline.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the roadline network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHight of the roadline network (input image rows).
- virtual RoadLineResult run (const cv::Mat &image)=0

  Function of get running result of the RoadLine network.

Static Public Member Functions

- static std::unique_ptr< RoadLine > create (bool need_preprocess=true)

  Factory function to get a instance of derived classes of class RoadLine.

Protected Member Functions

- **RoadLine** (const RoadLine &)=delete

3.22.1 Detailed Description

Base class for detecting roadline from a image (cv::Mat).

Input is a image (cv::Mat).

Output road line type and points maked road line.

Note

The input image size is 640x480

sample code:

```
auto det = xilinx::roadline::RoadLine::create();
auto image = cv::imread(argv[1]);
if(image.empty()) {
    cerr << "cannot load " << argv[1] << endl;
    abort();
}

std::vector<int> color1 = {0, 255, 0, 0, 100, 255};
std::vector<int> color2 = {0, 0, 255, 0, 100, 255};
std::vector<int> color3 = {0, 0, 0, 255, 100, 255};

RoadLineResult results = det->run(image);
for(auto line : result.lines){
  std::vector<cv::Point> points_poly = line.points_cluster;
  int type == line.type;
  if(type == 2 && points_poly[0].x < image.rows * 0.5)
      continue;
   cv::polylines(image, points_poly, false, Scalar(color1[type], color2[type], color3[type]), \
3, CV_AA, 0);
}
cv::imwrite("results.jpg",image);
```

Display of the roadline model results:

Figure 3.6: roadline result image

3.22.2 Member Function Documentation

3.22.2.1 static std::unique_ptr<RoadLine> xilinx::roadline::RoadLine::create (bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class RoadLine.

Parameters

need_process	normalize with mean/scale or not, default value is true.

Returns

An instance of RoadLine class.

3.22.2.2 virtual int xilinx::roadline::RoadLine::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the roadline network (input image cols).

Returns

InputWidth of the roadline network.

3.22.2.3 virtual int xilinx::roadline::RoadLine::getInputHeight ( ) const [pure virtual]

Function to get InputHight of the roadline network (input image rows).

Returns

InputHeight of the roadline network.

3.22.2.4 virtual RoadLineResult xilinx::roadline::RoadLine::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the RoadLine network.

Parameters

img	Input data , input image (cv::Mat) need to resized as 640x480.

Returns

The struct of RoadLineResult

3.23 xilinx::roadline::RoadLineResult Struct Reference

Struct of the result returned by the roadline network.

`#include <xilinx/roadline/roadline.hpp>`

Classes

- struct Line

  Struct of the result returned by the roadline network.

Public Attributes

- int width

  width of input image.
- int height

  height of input image.
- std::vector< Line > lines

  the vector of line

3.23.1 Detailed Description

Struct of the result returned by the roadline network.

3.24 xilinx::roadline::RoadLineResult::Line Struct Reference

Struct of the result returned by the roadline network.

`#include <xilinx/roadline/roadline.hpp>`

Public Attributes

- int type
- std::vector< cv::Point > points_cluster

  point clusters, make line from these.

3.24.1 Detailed Description

Struct of the result returned by the roadline network.

3.24.2 Member Data Documentation

3.24.2.1 int xilinx::roadline::RoadLineResult::Line::type

road line type, the value range from 0 to 3.

- `0` : background
- `1` : white dotted line
- `2` : white solid line
- `3` : yollow line

3.25 xilinx::segmentation::Segmentation Class Reference

Base class for Segmentation.

`#include <xilinx/segmentation/segmentation.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the segmentation network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHight of the segmentation network (input image rows).
- virtual SegmentationResult run_8UC1 (const cv::Mat &image)=0

  Function of get running result of the segmentation network.
- virtual SegmentationResult run_8UC3 (const cv::Mat &image)=0

  Function of get running result of the segmentation network.

Static Public Member Functions

- static std::unique_ptr
  < Segmentation > create (Type type, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Segmentation.

Protected Member Functions

- **Segmentation** (const Segmentation &)=delete

3.25.1 Detailed Description

Base class for Segmentation.

Input is a image (cv:Mat).

Output is struct SegmentationResultShow define before.

sample code :

```
auto det = xilinx::segmentation::Segmentation::create(
      xilinx::segmentation::FPN);
auto img = cv::imread("sample_segmentation.jpg");

int width = det->getInputWidth();
int height = det->getInputHeight();
cv::Mat image;
cv::resize(img, image, cv::Size(width, height), 0, 0,
           cv::INTER_LINEAR);
auto result = det->run_8UC1(image);
for (auto y = 0; y < result.segmentation.rows; y++) {
  for (auto x = 0; x < result.segmentation.cols; x++) {
        result.segmentation.at<uchar>(y,x) *= 10;
    }
}
cv::imwrite("segres.jpg",result.segmentation);

auto resultshow = det->run_8UC3(image);
resize(resultshow.segmentation, resultshow.segmentation, cv::Size(resultshow.cols * 2, resultshow.rows * 2)
      );
cv::imwrite("sample_segmentation_visualization_result.jpg",resultshow.segmentation);
```

Figure 3.7: segmentation visulization result image

3.25.2 Member Function Documentation

3.25.2.1 static std::unique_ptr<Segmentation> xilinx::segmentation::Segmentation::create (Type *type*, bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class Segmentation.

Each Network has their own scale,such as FPN is 256 ∗ 512;

Parameters

Type,the	tpye of segmentation network(FPN, ENET, ESPNET)
need_-preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of segmentation class.

3.25.2.2 virtual int xilinx::segmentation::Segmentation::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the segmentation network (input image cols).

Returns

InputWidth of the segmentation network.

3.25.2.3 virtual int xilinx::segmentation::Segmentation::getInputHeight ( ) const [pure virtual]

Function to get InputHight of the segmentation network (input image rows).

Returns

InputHeight of the segmentation network.

3.25.2.4 virtual SegmentationResult xilinx::segmentation::Segmentation::run_8UC1 (const cv::Mat & *image*) [pure virtual]

Function of get running result of the segmentation network.

Note

The type of CV_8UC1 of the Reuslt's segmentation.

Parameters

img	Input data of input image (cv::Mat).

Returns

a result include segmentation output data.

3.25.2.5 virtual SegmentationResult xilinx::segmentation::Segmentation::run_8UC3 (const cv::Mat & *image*) [pure virtual]

Function of get running result of the segmentation network.

Note

The type of CV_8UC3 of the Reuslt's segmentation.

Parameters

img	Input data of input image (cv::Mat).

Returns

a result include segmentation image and shape;.

3.26 xilinx::segmentation::Segmentation8UC1 Class Reference

The Class of Segmentation8UC1, this class run function return a cv::Mat with the type is cv_8UC1.

`#include <xilinx/segmentation/segmentation.hpp>`

Public Member Functions

- virtual int getInputWidth () const

  Function to get InputWidth of the segmentation network (input image cols).
- virtual int getInputHeight () const

  Function to get InputHight of the segmentation network (input image cols).
- virtual SegmentationResult run (const cv::Mat &image)

  Function of get running result of the segmentation network.

Static Public Member Functions

- static std::unique_ptr
  < Segmentation8UC1 > create (Type type, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Segmentation8UC1.

Protected Member Functions

- **Segmentation8UC1** (std::unique_ptr< Segmentation > segmentation)
- **Segmentation8UC1** (const Segmentation8UC1 &)=delete

3.26.1 Detailed Description

The Class of Segmentation8UC1, this class run function return a cv::Mat with the type is cv_8UC1.

- sample code :

```
auto det = xilinx::segmentation::Segmentation8UC1::create(
      xilinx::segmentation::FPN);
auto img = cv::imread("sample_segmentation.jpg");

int width = det->getInputWidth();
int height = det->getInputHeight();
cv::Mat image;
cv::resize(img, image, cv::Size(width, height), 0, 0,
           cv::INTER_LINEAR);
auto result = det->run(image);
for (auto y = 0; y < result.segmentation.rows; y++) {
  for (auto x = 0; x < result.segmentation.cols; x++) {
        result.segmentation.at<uchar>(y,x) *= 10;
    }
}
cv::imwrite("segres.jpg",result.segmentation);
```

3.26.2 Member Function Documentation

3.26.2.1 static std::unique_ptr<Segmentation8UC1> xilinx::segmentation::Segmentation8UC1::create (Type *type*, bool *need_preprocess* = true) [inline], [static]

Factory function to get a instance of derived classes of class Segmentation8UC1.

```
@param Type, the tpye of segmentation network(FPN, ENET, ESPNET)
@param need_preprocess Normalize with mean/scale or not, default value
```

is true.

Returns

An instance of segmentation8UC1 class.

3.26.2.2 virtual int xilinx::segmentation::Segmentation8UC1::getInputWidth ( ) const [inline], [virtual]

Function to get InputWidth of the segmentation network (input image cols).

Returns

InputWidth of the segmentation network.

3.26.2.3 virtual int xilinx::segmentation::Segmentation8UC1::getInputHeight ( ) const [inline], [virtual]

Function to get InputHight of the segmentation network (input image cols).

Returns

InputHeight of the segmentation network.

3.26.2.4 virtual SegmentationResult xilinx::segmentation::Segmentation8UC1::run (const cv::Mat & *image*) [inline], [virtual]

Function of get running result of the segmentation network.

Note

The result cv::Mat of the type is CV_8UC1.

Parameters

image	Input data of the image (cv::Mat)

Returns

SegmentationResult The result of segmentation network.

3.27 xilinx::segmentation::Segmentation8UC3 Class Reference

The Class of Segmentation8UC3, this class run function return a cv::Mat with the type is cv_8UC3.

```
#include <xilinx/segmentation/segmentation.hpp>
```

Public Member Functions

- virtual int getInputWidth () const

  Function to get InputWidth of the segmentation network (input image cols).
- virtual int getInputHeight () const

  Function to get InputWidth of the segmentation network (input image cols).
- virtual SegmentationResult run (const cv::Mat &image)

  Function of get running result of the segmentation network.

Static Public Member Functions

- static std::unique_ptr
  < Segmentation8UC3 > create (Type type, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Segmentation8UC3.

Protected Member Functions

- **Segmentation8UC3** (std::unique_ptr< Segmentation > segmentation)
- **Segmentation8UC3** (const Segmentation8UC3 &)=delete

3.27.1 Detailed Description

The Class of Segmentation8UC3, this class run function return a cv::Mat with the type is cv_8UC3.

- sample code :

```
auto det = xilinx::segmentation::Segmentation8UC3::create(
      xilinx::segmentation::FPN);
auto img = cv::imread("sample_segmentation.jpg");

int width = det->getInputWidth();
int height = det->getInputHeight();
cv::Mat image;
cv::resize(img, image, cv::Size(width, height), 0, 0,
           cv::INTER_LINEAR);
auto result = det->run(image);
cv::imwrite("segres.jpg",result.segmentation);
```

3.27.2 Member Function Documentation

3.27.2.1 static std::unique_ptr<Segmentation8UC3> xilinx::segmentation::Segmentation8UC3::create (Type *type*, bool *need_preprocess* = `true`) [inline], [static]

Factory function to get a instance of derived classes of class Segmentation8UC3.

Parameters

Type,the	tpye of segmentation network(FPN, ENET, ESPNET)
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of segmentation8UC3 class.

3.27.2.2 virtual int xilinx::segmentation::Segmentation8UC3::getInputWidth ( ) const [inline], [virtual]

Function to get InputWidth of the segmentation network (input image cols).

Returns

InputWidth of the segmentation network.

3.27.2.3 virtual int xilinx::segmentation::Segmentation8UC3::getInputHeight ( ) const [inline], [virtual]

Function to get InputWidth of the segmentation network (input image cols).

Returns

InputWidth of the segmentation network.

3.27.2.4 virtual SegmentationResult xilinx::segmentation::Segmentation8UC3::run (const cv::Mat & *image*) [inline], [virtual]

Function of get running result of the segmentation network.

Note

The result cv::Mat of the type is CV_8UC1.

Parameters

image	Input data of the image (cv::Mat)

Returns

SegmentationResult The result of segmentation network.

3.28 xilinx::segmentation::SegmentationResult Struct Reference

Struct of the result returned by the segementation network.

`#include <xilinx/segmentation/segmentation.hpp>`

Public Attributes

- int **width**
- int **height**
- cv::Mat **segmentation**

3.28.1 Detailed Description

Struct of the result returned by the segementation network.

3.29 xilinx::ssd::SSD Class Reference

Base class for detecting position of vehicle,pedestrian and so on.

`#include <xilinx/ssd/ssd.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the SSD network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the SSD network (input image rows).
- virtual SSDResult run (const cv::Mat &img)=0

  Function of get running result of the ssd neuron network.

Static Public Member Functions

- static std::unique_ptr< SSD > create (Type type, bool need_mean_scale_process=true)

  Factory function to get a instance of derived classes of class SSD.
- static std::unique_ptr< SSD > create_ex (const std::string &model_name, bool need_mean_scale_process=true)

  Factory function to get a instance of derived classes of class SSD.

Protected Member Functions

- **SSD** (const SSD &)=delete

3.29.1 Detailed Description

Base class for detecting position of vehicle,pedestrian and so on.

Input is a image (cv:Mat).

Output is a vector<SSDResult>.

sample code :

```
Mat img = cv::imread("sample_ssd_TRAFFIC_480x360.jpg");
auto ssd = xilinx::ssd::SSD::create(xilinx::ssd::TRAFFIC_480x360,true);
auto results = ssd->run(img);
for(const auto &r : results.bboxes){
   auto label = r.label;
   auto x = r.x * img.cols;
   auto y = r.y * img.rows;
   auto width = r.width * img.cols;
   auto heigth = r.height * img.rows;
   auto score = r.score;
   std::cout << "RESULT: " << label << "\t" << x << "\t" << y << "\t" << width
      << "\t" << height << "\t" << score << std::endl;
}
```

Display of the ssd_TRAFFIC_480x360 model results:

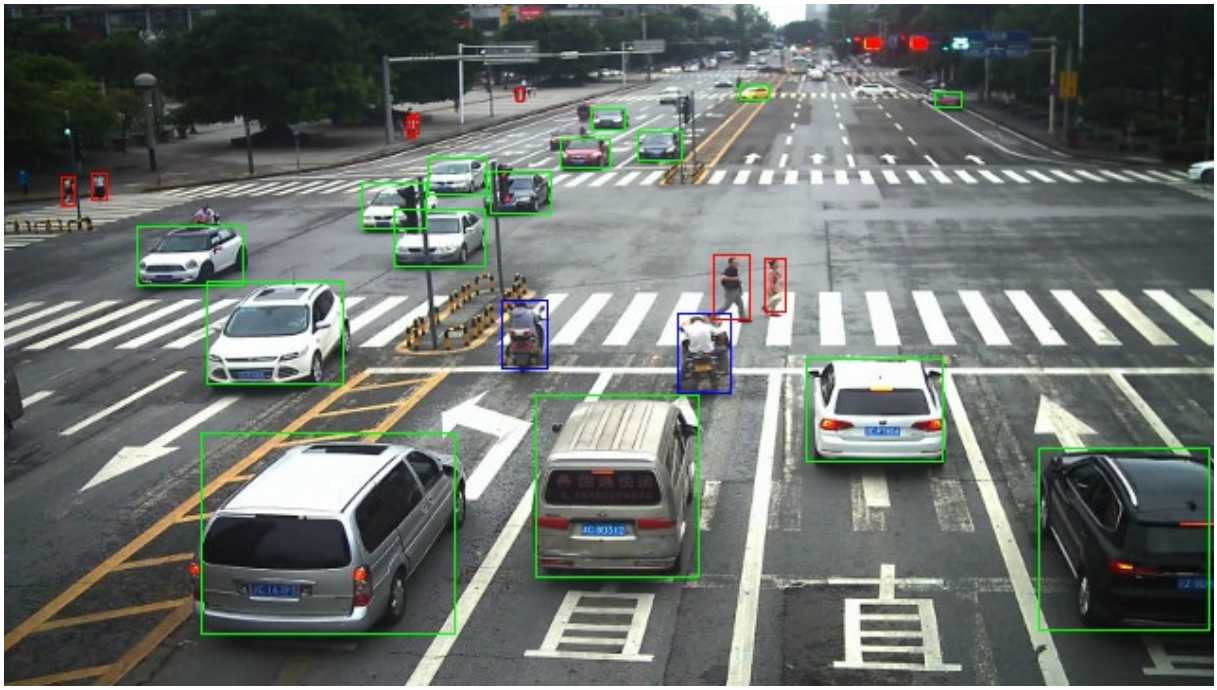

Figure 3.8: out image

Display of the ADAS_VEHICLE_V3_480x360 model results:

Figure 3.9: out image

3.29.2 Member Function Documentation

3.29.2.1 static std::unique_ptr<SSD> xilinx::ssd::SSD::create (Type *type*, bool *need_mean_scale_process* = `true`) `[static]`

Factory function to get a instance of derived classes of class SSD.

Parameters

type	Enum Type
@param	need_mean_scale_process Normalize with mean/scale or not, default value is true.

Returns

An instance of SSD class.

3.29.2.2 static std::unique_ptr<SSD> xilinx::ssd::SSD::create_ex (const std::string & *model_name*, bool *need_mean_scale_process* = `true`) `[static]`

Factory function to get a instance of derived classes of class SSD.

Note

for internal use

Parameters

model_name	String of model name
@param	need_mean_scale_process Normalize with mean/scale or not, default value is true.

Returns

An instance of SSD class.

3.29.2.3 virtual int xilinx::ssd::SSD::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the SSD network (input image cols).

Returns

InputWidth of the SSD network.

3.29.2.4 virtual int xilinx::ssd::SSD::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the SSD network (input image rows).

Returns

InputHeight of the SSD network.

3.29.2.5 virtual SSDResult xilinx::ssd::SSD::run (const cv::Mat & *img*) [pure virtual]

Function of get running result of the ssd neuron network.

Parameters

img	Input data of input image (cv::Mat).

Returns

SSDResult.

3.30 xilinx::ssd::SSDResult Struct Reference

Struct of the result returned by the ssd neuron network.

`#include <xilinx/ssd/ssd.hpp>`

Classes

- struct BoundingBox

  Struct of a object coordinate ,confidence and classification.

Public Attributes

- int width

  Width of input image.
- int height

Height of input image.

- std::vector< BoundingBox > bboxes

  all objects, a vector of BoundingBox

3.30.1 Detailed Description

Struct of the result returned by the ssd neuron network.

3.31 xilinx::ssd::SSDResult::BoundingBox Struct Reference

Struct of a object coordinate ,confidence and classification.

`#include <xilinx/ssd/ssd.hpp>`

Public Attributes

- int label

  classification
- float score

  confidence
- float x
- float y
- float width
- float height

3.31.1 Detailed Description

Struct of a object coordinate ,confidence and classification.

3.31.2 Member Data Documentation

3.31.2.1 float xilinx::ssd::SSDResult::BoundingBox::x

x-coordinate, x is normalized relative to the input image cols ,the value range from 0 to 1.

3.31.2.2 float xilinx::ssd::SSDResult::BoundingBox::y

y-coordinate ,y is normalized relative to the input image rows ,the value range from 0 to 1.

3.31.2.3 float xilinx::ssd::SSDResult::BoundingBox::width

width, width is normalized relative to the input image cols ,the value range from 0 to 1.

3.31.2.4 float xilinx::ssd::SSDResult::BoundingBox::height

height, height is normalized relative to the input image rows ,the value range from 0 to 1.

3.32 xilinx::yolov3::YOLOv3 Class Reference

Base class for detecting objects in a image (cv::Mat).

`#include <xilinx/yolov3/yolov3.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the YOLOv3 network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the YOLOv3 network (input image rows).
- virtual YOLOv3Result run (const cv::Mat &image)=0

  Function of get running result of the YOLOv3 neuron network.

Static Public Member Functions

- static std::unique_ptr< YOLOv3 > create (Type type, bool need_mean_scale_process=true)

  Factory function to get a instance of derived classes of class YOLOv3.
- static std::unique_ptr< YOLOv3 > create_ex (const std::string &model_name, bool need_mean_scale_process=true)

  Factory function to get a instance of derived classes of class YOLOv3.

Protected Member Functions

- **YOLOv3** (const YOLOv3 &)=delete

3.32.1 Detailed Description

Base class for detecting objects in a image (cv::Mat).

Input is a image (cv::Mat).

Output is position of the pedestrians in the input image.

sample code:

```
   auto yolo = xilinx::yolov3::YOLOv3::create(xilinx::yolov3::ADAS_512x256,
true);
   Mat img = cv::imread("test.jpg");

   auto results = yolo->run(img);

   for(auto &box : results.bboxes){
     int label = box.label;
     float xmin = box.x * img.cols + 1;
     float ymin = box.y * img.rows + 1;
     float xmax = xmin + box.width * img.cols;
     float ymax = ymin + box.height * img.rows;
     if(xmin < 0.) xmin = 1.;
     if(ymin < 0.) ymin = 1.;
     if(xmax > img.cols) xmax = img.cols;
     if(ymax > img.rows) ymax = img.rows;
     float confidence = box.score;

     cout << "RESULT: " << label << "\t" << xmin << "\t" << ymin << "\t"
          << xmax << "\t" << ymax << "\t" << confidence << "\n";
     if (label == 0) {
       rectangle(img, Point(xmin, ymin), Point(xmax, ymax), Scalar(0, 255, 0),
                 1, 1, 0);
     } else if (label == 1) {
       rectangle(img, Point(xmin, ymin), Point(xmax, ymax), Scalar(255, 0, 0),
                 1, 1, 0);
     } else if (label == 2) {
```

```
      rectangle(img, Point(xmin, ymin), Point(xmax, ymax), Scalar(0, 0, 255),
                1, 1, 0);
    } else if (label == 3) {
      rectangle(img, Point(xmin, ymin), Point(xmax, ymax),
                Scalar(0, 255, 255), 1, 1, 0);
    }
  }
  imwrite("result.jpg", img);
```

Display of the yolov3_ADAS_512x256 model results:

Figure 3.10: out image

3.32.2 Member Function Documentation

3.32.2.1 static std::unique_ptr<YOLOv3> xilinx::yolov3::YOLOv3::create (Type *type*, bool *need_mean_scale_process* = true) [static]

Factory function to get a instance of derived classes of class YOLOv3.

Parameters

type	VOC_416x416 or ADAS_512x256
need_mean_scale_process	Normalize with mean/scale or not, default value is true.

Returns

An instance of YOLOv3 class.

3.32.2.2 static std::unique_ptr<YOLOv3> xilinx::yolov3::YOLOv3::create_ex (const std::string & *model_name*, bool *need_mean_scale_process* = true) [static]

Factory function to get a instance of derived classes of class YOLOv3.

Note

for internal use

Parameters

model_name	
need_mean_-scale_process	Normalize with mean/scale or not, default value is true.

Returns

An instance of YOLOv3 class.

3.32.2.3 virtual int xilinx::yolov3::YOLOv3::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the YOLOv3 network (input image cols).

Returns

InputWidth of the YOLOv3 network

3.32.2.4 virtual int xilinx::yolov3::YOLOv3::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the YOLOv3 network (input image rows).

Returns

InputHeight of the YOLOv3 network.

3.32.2.5 virtual YOLOv3Result xilinx::yolov3::YOLOv3::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the YOLOv3 neuron network.

Parameters

image	Input data of input image (cv::Mat).

Returns

YOLOv3Result.

3.33 xilinx::yolov3::YOLOv3Result Struct Reference

Struct of the result returned by the yolov3 neuron network.

`#include <xilinx/yolov3/yolov3.hpp>`

Classes

- struct BoundingBox

Public Attributes

- int width

  Width of input image.
- int height

Height of output image.

- std::vector< BoundingBox > bboxes

  all objects, The vector of BoundingBox .

3.33.1 Detailed Description

Struct of the result returned by the yolov3 neuron network.

Note

VOC dataset category:string label[20] = {"aeroplane", "bicycle", "bird", "boat", "bottle", "bus","car", "cat", "chair", "cow", "diningtable", "dog", "horse", "motorbike","person", "pottedplant", "sheep", "sofa", "train", "tv-monitor"};
ADAS dataset category : string label[3] = {"car", "person", "cycle"};

3.34 xilinx::yolov3::YOLOv3Result::BoundingBox Struct Reference

`#include <xilinx/yolov3/yolov3.hpp>`

Public Attributes

- int label

  classification.
- float score

  confidence, the range from 0 to 1.
- float x
- float y
- float width
- float height

3.34.1 Detailed Description

Struct of detection result with a object

3.34.2 Member Data Documentation

3.34.2.1 float xilinx::yolov3::YOLOv3Result::BoundingBox::x

x-coordinate, x is normalized relative to the input image cols, its value range from 0 to 1.

3.34.2.2 float xilinx::yolov3::YOLOv3Result::BoundingBox::y

y-coordinate, y is normalized relative to the input image rows, its value range from 0 to 1.

3.34.2.3 float xilinx::yolov3::YOLOv3Result::BoundingBox::width

width, width is normalized relative to the input image cols, its value from 0 to 1.

3.34.2.4 float xilinx::yolov3::YOLOv3Result::BoundingBox::height

height, height is normalized relative to the input image rows, its value range from 0 to 1.