



version: 202.0

## 1 SDK 導論

---

### 摘要



QDEEP SDK技術為用戶提供了一種使用深度學習（AI）的簡單方法，可以使用不同的模型來檢測不同的對象。

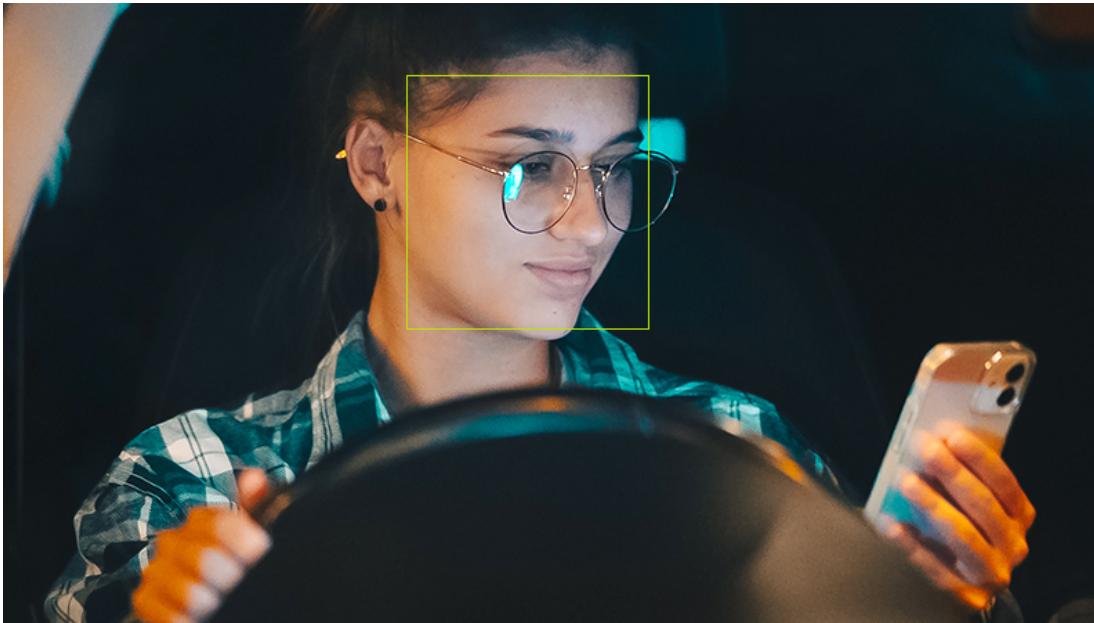
## 1.1 簡介

QDEEP SDK 的 API 可供開發人員輕鬆使用我們的採集卡做深入學習並開發豐富的客戶端應用程序。它支持以下功能：

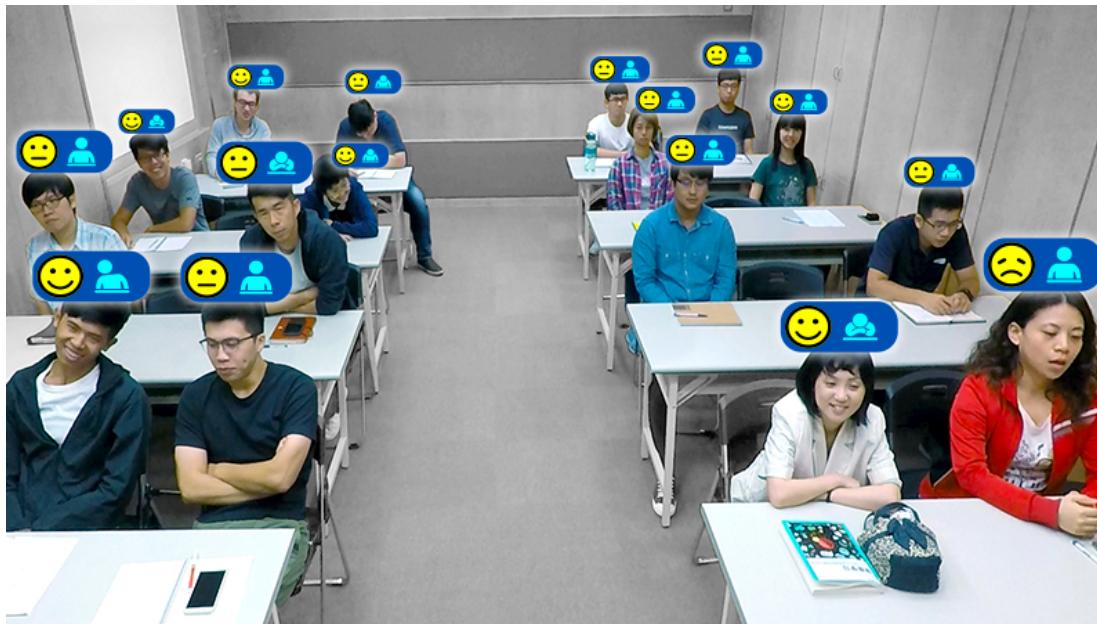
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_AOI\_GENERAL\_DEFECT\_DETECTION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_AOI\_PCB\_DEFECT\_DETECTION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_AOI\_GAUGE\_READER\_DETECTION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_BOAT
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_DEPTH\_MAP\_3D\_EX
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_DRIVING\_DISTRACTION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_EDUCATION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_HEAD\_BODY
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_LANDMARK\_5\_KEYPOINTS
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_LANDMARK\_68\_KEYPOINTS
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_LANDMARK\_68\_KEYPOINTS\_3D\_EX
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_LANDMARK\_68\_KEYPOINTS\_FACE\_BEAUTY\_EX
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_LANDMARK\_68\_KEYPOINTS\_MODAL\_ANALYTICS\_EX
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FLAME
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HAND\_LANDMARK\_21\_KEYPOINTS
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_BACKGROUND\_BLURRING
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_BACKGROUND\_REMOVAL
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_EPTZ\_AUTO\_FRAMING
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_EPTZ\_FACE\_LAYOUT
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_EPTZ\_SPEAKER\_TRACKING
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_HANDWRITE\_EXTRACTION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_SAFETY\_INSPECTION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_SKELETON\_17\_KEYPOINTS
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_SKELETON\_136\_KEYPOINTS
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_LICENSE\_PLATE\_RECOGNITION\_PARKING
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_LICENSE\_PLATE\_RECOGNITION\_LAW\_ENFORCEMENT
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_MISSING\_OBJECT
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_OPTICAL\_CHARACTER\_RECOGNITION

- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_PHOTO\_RETOUCHING
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_PHOTO\_SUPER\_RESOLUTION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_RETAIL\_PRODUCT\_RECOGNITION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_SEGMENTATION
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_TRAFFIC
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_TRAFFIC\_TAIWAN\_04
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_TRAFFIC\_TAIWAN\_08
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_UNATTENDED\_OBJECT
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_XRAY\_INSPECTION\_SYSTEM
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_CUSTOMIZED
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_CUSTOMIZED\_LITE
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_CUSTOMIZED\_MEDICAL\_GRADE
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_CUSTOMIZED\_MULTI\_LABELS
- QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_NVIDIA\_CLARA\_AX

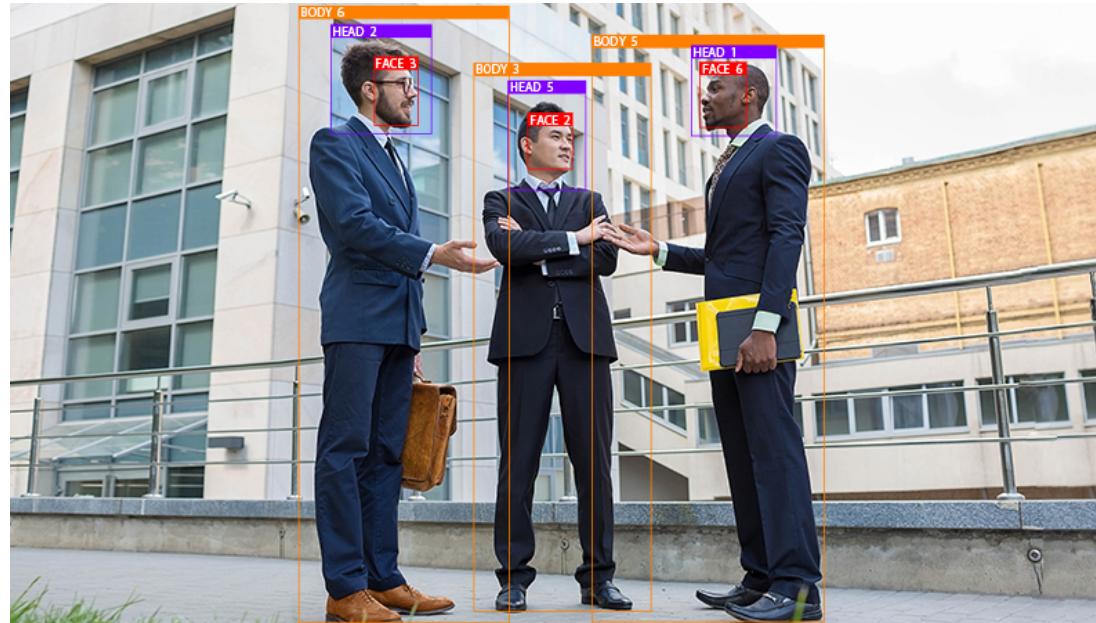
**1.1.1 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_DRIVING\_DISTRACTION**



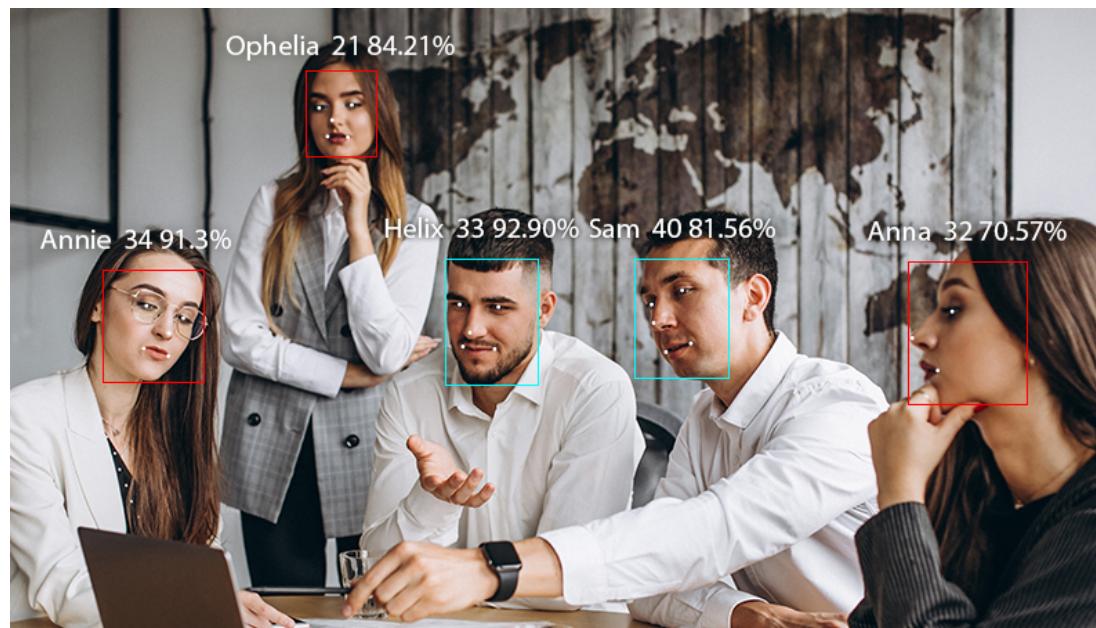
### 1.1.2 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_EDUCATION



### 1.1.3 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_HEAD\_BODY



### 1.1.4 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_LANDMARK\_5\_KEYPOINTS



1.1.5 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_LANDMARK\_68\_KEYPOINTS



1.1.6 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_FACE\_LANDMARK\_68\_KEYPOINTS\_FACE\_BEAUTY\_EX



1.1.7 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_BACKGROUND\_BLURRING



1.1.8 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_BACKGROUND\_REMOVAL



**1.1.9 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_EPTZ\_AUTO\_FRAMING**



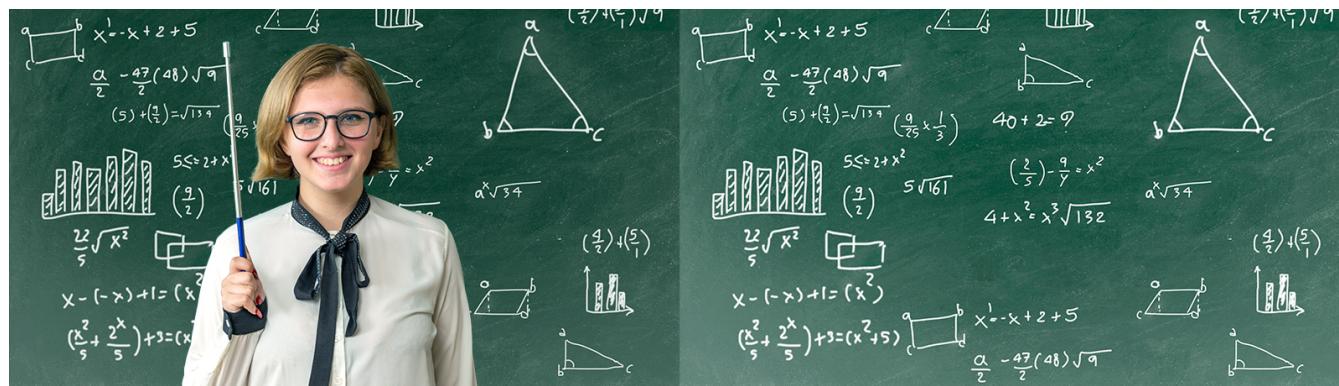
**1.1.10 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_EPTZ\_FACE\_LAYOUT**



### 1.1.11 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_EPTZ\_SPEAKER\_TRACKING



### 1.1.12 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_HANDWRITE\_EXTRACTION



1.1.13 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_HUMAN\_SKELETON\_17\_KEYPOINTS



1.1.14 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_LICENSE\_PLATE\_RECOGNITION\_PARKING



1.1.15 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_LICENSE\_PLATE\_RECOGNITION\_LAW\_ENFORCEMENT



1.1.16 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_MISSING\_OBJECT



1.1.17 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_OPTICAL\_CHARACTER\_RECOGNITION



100% ORGANIC  
CANNABIS  
FOR MEDICAL USE ONLY

1.1.18 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_PHOTO\_RETTOUCHING



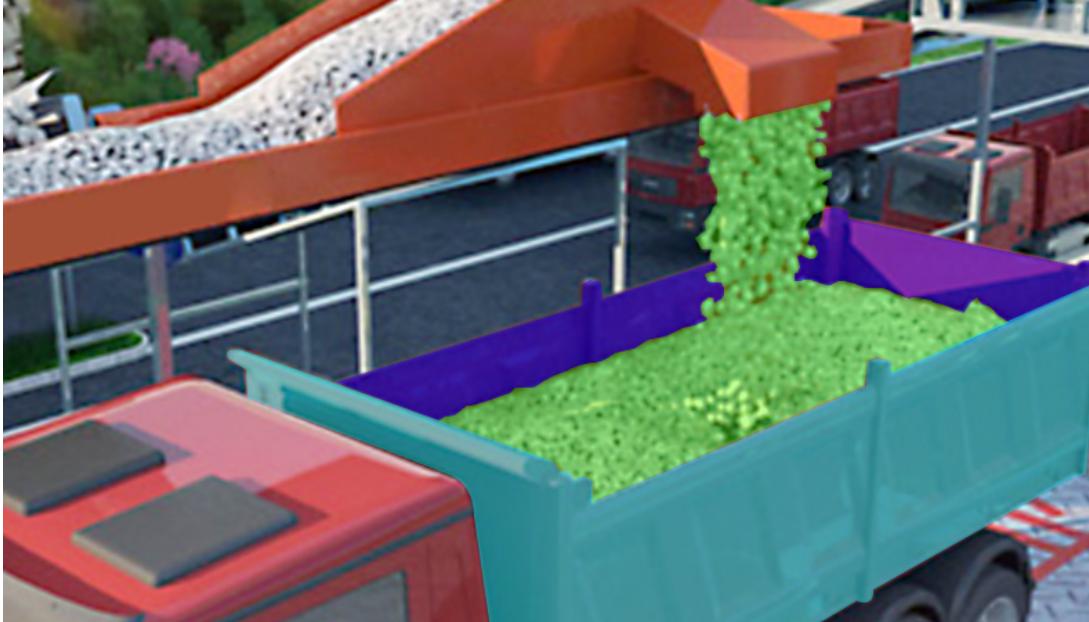
1.1.19 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_PHOTO\_SUPER\_RESOLUTION



1.1.20 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_RETAIL\_PRODUCT\_RECOGNITION



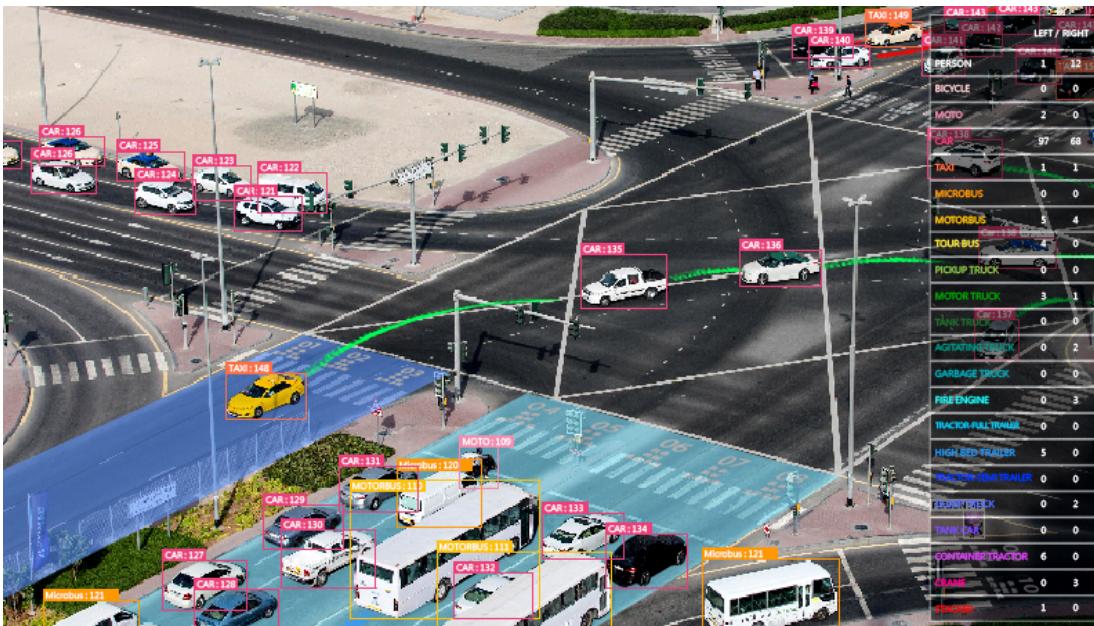
1.1.21 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_SEGMENTATION



1.1.22 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_SMOG



### 1.1.23 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_TRAFFIC



### 1.1.24 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_TRAFFIC\_TAIWAN\_04



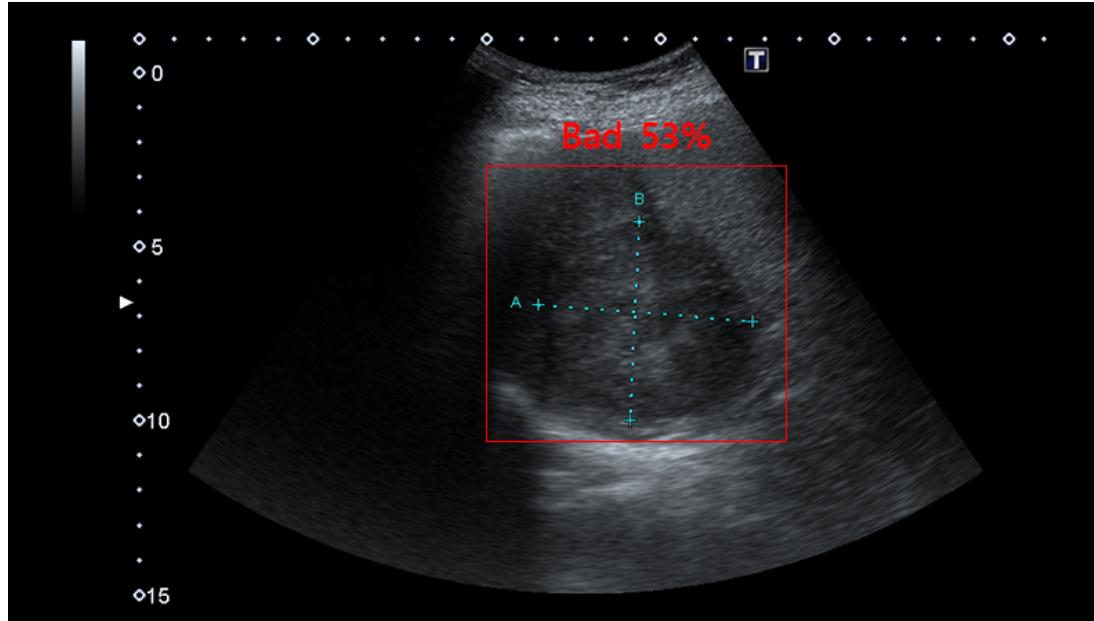
### 1.1.25 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_TRAFFIC\_TAIWAN\_08



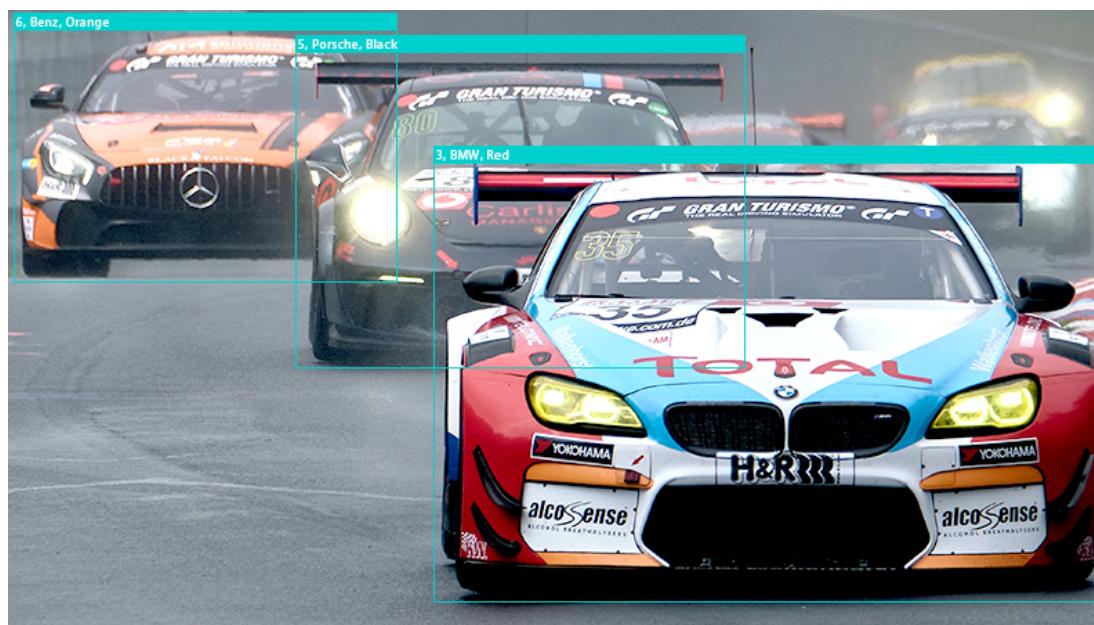
### 1.1.26 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_UNATTENDED\_OBJECT



1.1.27 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_CUSTOMIZED\_MEDICAL\_GRADE



1.1.28 QDEEP\_OBJECT\_DETECT\_CONFIG\_MODEL\_CUSTOMIZED\_MULTI\_LABELS



## 1.2 SDK 功能模塊

QDEEP SDK 包括 **內容分析、生物識別、行為分析** 集成一身。

**Note :** 開發者使用QDEEP SDK 開發時，必須是X64的OS系統。



# 四大功能模塊

採集, 錄影, 串流, 分析 一氣呵成

2023.10.3, 1.1.0.202.0

■ 數據回呼  
■ 事件回呼

高效能繪圖引擎 高速顯示, 轉轉換像, 虛擬顯示, 3D 顯示, HDR 顯示		
影像抓拍 連續, 壓切, 縮放 BMP, JPG, PNG, TIF, DICOM/HL7/WL		■
除交錯演算法 Motion Adapter, Blending, Filter Triangle, Advanced 3D Adapter, RGB Repack		■
視頻疊加引擎 文字, 跑馬燈, 圖片, 內存緩衝, 關鍵色去背		
RAW, MPEG2, H264 <sup>3D/VC</sup> , H265 <sup>10</sup> , AAC Intel® Media SDK   NVIDIA® CUDA/NVDEC™   AMD® UVHD™		
播放, 計停, 停止, 單格, 搜索, 快慢速回放, 逐幀播放, 同步回放		

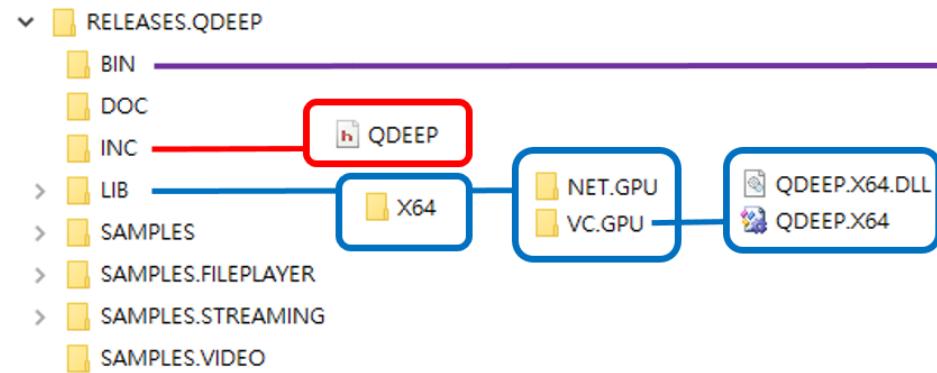
高精度計時器	
RS232/RS422/RS485/VISCA/PELCO	■
NVIDIA® CUDA/NVDEC™	■
檔案轉碼	
Intel® Media SDK	■
檔案診斷與修復 MP4	
段落剪裁輸出 多段剪裁	■
檔案合併 片頭片尾添加, 多檔合併	■
光碟燒錄 CD, DVD, DVD-RW	
輔助裝置採集 音效卡, USB 攝像機, 虛擬桌面, 網路串流	
虛擬攝像機輸出 Virtual AVStream Capture Driver	

回放	編輯	周邊工具
----	----	------

## 1.3 SDK 的開發步驟

這本節中，我們將會一步一步引導你環境的設置，然後使用 QDEEP SDK，請參考下列設置步驟：

- 執行並安裝 **CODECS 1.1.0.exe** -
  - 設備管理列表，請確保顯示卡裝置沒有問號，是正常地執行工作
- 解壓縮 **RELEASES.QDEEP 1.1.0.7z** 至 **QDEEP\_SDK\**
  - 請使用 **7-Zip** ([www.7-zip.org](http://www.7-zip.org)) 去解壓縮副檔名為**.7z** 檔案文件
- 並在範例目錄下設置 QDEEP Header/Static/Run-time 函式庫
  - 從 **QDEEP\_SDK\INC** 複製 **QDEEP.H** 到你的專案目錄下
  - 從 **QDEEP\_SDK\LIB\X64\VC.GPU** 複製 **QDEEP.X64.LIB** 到你的專案目錄下
- 編譯範例程序並產生出執行檔
  - 從 **QDEEP\_SDK\BIN** 複製所有的檔案到你的執行檔相同目錄下
  - 根據所需的功能從 **RELEASES.QDEEP.MODEL.X** 複製 **.CFG**、**.MODEL** 到你的執行檔相同目錄下
  - 從 **QDEEP\_SDK\LIB\X64\VC.GPU** 複製 **QDEEP.X64.DLL** 到你的執行檔相同目錄下
  - **QDEEP.X64.DLL** 必須要與執行檔在同一個目錄下！
- 執行執行檔



A list of DLL files contained within a purple-outlined rounded rectangle, likely representing a system library or plugin directory. The files listed include:

- cldnn\_global\_custom\_kernels.dll
- Movidius.dll
- c10.dll
- c10\_cuda.dll
- caffe2\_detectron\_ops\_gpu.dll
- caffe2\_module\_test\_dynamic.dll
- caffe2\_nvrtc.dll
- cldnnplugin.dll
- cublas64\_92.dll
- cuda64\_92.dll
- cudnn64\_7.dll
- cufft64\_92.dll
- cufftw64\_92.dll
- curand64\_92.dll
- cusolver64\_92.dll
- cusparse64\_92.dll
- hddlplugin.dll
- inference\_engine.dll
- inference\_engine\_c\_api.dll
- inference\_engine\_legacy.dll
- inference\_engine\_lp\_transformations.dll
- inference\_engine\_nn\_builder.dll
- inference\_engine\_preproc.dll
- inference\_engine\_transformations.dll
- libomp5md.dll
- libompstubs5md.dll
- libmmd.dll
- mklDnnPlugin.dll
- myriadPlugin.dll
- ngraph.dll
- nvrtc64\_92.dll
- nvrtc-builtins64\_92.dll
- nvToolsExt64\_1.dll
- plugins.xml
- svm\_dispmd.dll
- tbb.dll
- torch.dll
- usb-ma2x8x.mvcmd
- vc2015.redist.x64.exe



## 1.4 作業系統與執行緒安全

SDK支持 X64 的 Windows Vista、Windows 7、Windows 8、Windows 10 和 Windows Server 2008 下的作業環境。所有的 QDEEP API 符合執行性安全，可被使用在多執行緒、多進程。



GPU 類別的 Intel CPU/ GPU / VPU MOVIDIUS 只作用於 **Windows 10**

## 1.5 SDK 程式教學速覽

本章節將快速介紹使用者一些典型常用的 QDEEP SDK 範例程序

### 1.5.1 物體偵測裝置 Initialize APIs

```
PVOID m_pDetector = NULL;  
  
QDEEP_CREATE_OBJECT_DETECT(    QDEEP_GPU_TYPE_NVIDIA,  
                                0,  
                                QDEEP_OBJECT_DETECT_CONFIG_MODEL_FACE_HEAD_BODY,  
                                "QDEEP.OD.FACE.HEAD.BODY.CFG",  
                                &m_pDetector);  
  
QDEEP_START_OBJECT_DETECT(m_pDetector);
```

C

### 1.5.2 物體偵測裝置 Uninitialize APIs

```
QDEEP_STOP_OBJECT_DETECT(m_pDetector);  
  
QDEEP_DESTROY_OBJECT_DETECT(m_pDetector);
```

C

### 1.5.3 置入物體偵測引擎 APIs

```
ULONG m_pObjectSize = 1000;

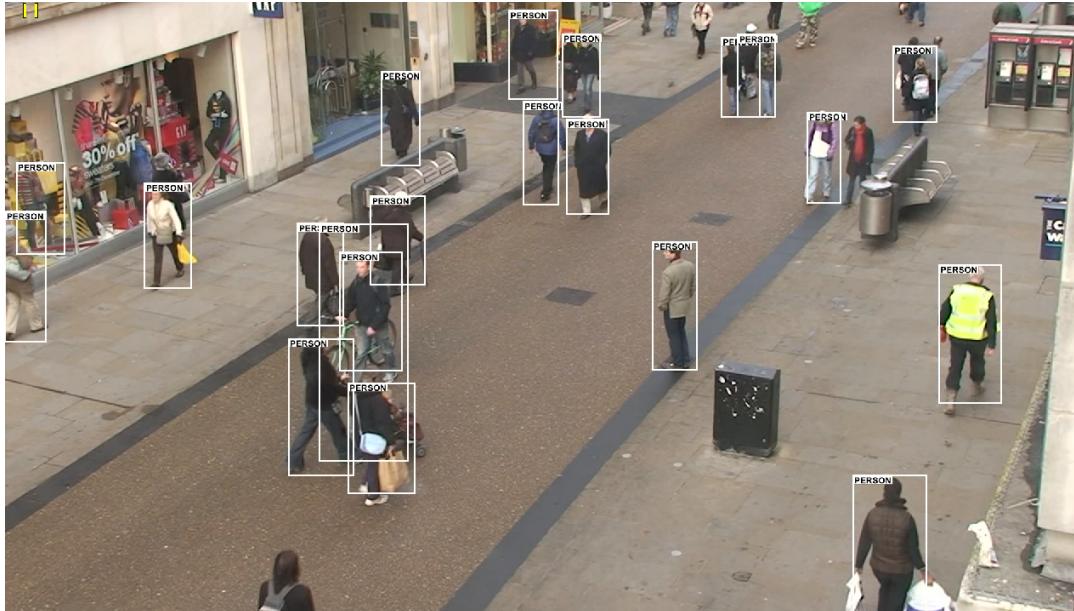
QDEEP_OBJECT_DETECT_BOUNDING_BOX* m_pObjectList =
    (QDEEP_OBJECT_DETECT_BOUNDING_BOX*)malloc( m_pObjectSize * sizeof(QDEEP_OBJECT_DETECT_BOUNDING_BOX) );

QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER( m_pDetector,
    QDEEP_COLORSPACE_TYPE_YV12,
    nVideoWidth,
    nVideoHeight,
    pFrameBuffer,
    nFrameBufferLen,
    m_pObjectList,
    &m_pObjectSize );
```

## 2 偵測 和 追蹤函式 API

---

### 摘要



本章 API 介面提供使用者使用QDEEP 進行檢測和追蹤。

## 2.1 Object Detect Creating Function

### 2.1.1 QDEEP\_CREATE\_OBJECT\_DETECT

### 2.1.2 QDEEP\_CREATE\_OBJECT\_DETECT\_EX

說明

使用者透過 QDEEP\_CREATE\_OBJECT\_DETECT 建立一個偵測裝置介面。



GPU 類別的 Intel CPU / GPU / VPU MOVIDIUS 只作用於 物件偵測

參數

型別	參數名稱	輸出入	描述
ULONG	nGpuType	輸入	指定 GPU 類別 QDEEP_GPU_TYPE_DEFAULT QDEEP_GPU_TYPE_NVIDIA QDEEP_GPU_TYPE_INTEL_CPU QDEEP_GPU_TYPE_INTEL_GPU QDEEP_GPU_TYPE_INTEL_VPU_MOVIDIUS
UINT	iGpuNum	輸入	指定顯示卡，從 0 開始

型別	參數名稱	輸出入	描述
ULONG	nConfigModel	輸入	<p>指定模型配置</p> <p>QDEEP_OBJECT_DETECT_CONFIG_MODEL_AOI_GENERAL_DEFECT_DETECTION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_AOI_PCB_DEFECT_DETECTION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_AOI_GAUGE_READER_DETECTION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_BOAT          QDEEP_OBJECT_DETECT_CONFIG_MODEL_DEPTH_MAP_3D_EX          QDEEP_OBJECT_DETECT_CONFIG_MODEL_DRIVING_DISTRACTION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_EDUCATION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_FACE_HEAD_BODY          QDEEP_OBJECT_DETECT_CONFIG_MODEL_FACE_LANDMARK_5_KEYPOINTS          QDEEP_OBJECT_DETECT_CONFIG_MODEL_FACE_LANDMARK_68_KEYPOINTS          QDEEP_OBJECT_DETECT_CONFIG_MODEL_FACE_LANDMARK_68_KEYPOINTS_3D_EX          QDEEP_OBJECT_DETECT_CONFIG_MODEL_FACE_LANDMARK_68_KEYPOINTS_FACE_BEAUTY_EX          QDEEP_OBJECT_DETECT_CONFIG_MODEL_FACE_LANDMARK_68_KEYPOINTS_MODAL_ANALYTICS_EX          QDEEP_OBJECT_DETECT_CONFIG_MODEL_FLAME          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HAND_LANDMARK_21_KEYPOINTS          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_BACKGROUND_BLURRING          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_BACKGROUND_REMOVAL          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_EPTZ_AUTO_FRAMING          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_EPTZ_FACE_LAYOUT          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_EPTZ_SPEAKER_TRACKING          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_HANDWRITE_EXTRACTION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_SAFETY_INSPECTION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_SKELETON_17_KEYPOINTS          QDEEP_OBJECT_DETECT_CONFIG_MODEL_HUMAN_SKELETON_136_KEYPOINTS          QDEEP_OBJECT_DETECT_CONFIG_MODEL_LICENSE_PLATE_RECOGNITION_PARKING          QDEEP_OBJECT_DETECT_CONFIG_MODEL_LICENSE_PLATE_RECOGNITION_LAW_ENFORCEMENT          QDEEP_OBJECT_DETECT_CONFIG_MODEL_MISSING_OBJECT          QDEEP_OBJECT_DETECT_CONFIG_MODEL_OPTICAL_CHARACTER_RECOGNITION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_PHOTO_RETTOUCHING          QDEEP_OBJECT_DETECT_CONFIG_MODEL_PHOTO_SUPER_RESOLUTION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_RETAIL_PRODUCT_RECOGNITION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_SEGMENTATION          QDEEP_OBJECT_DETECT_CONFIG_MODEL_TRAFFIC          QDEEP_OBJECT_DETECT_CONFIG_MODEL_TRAFFIC_TAIWAN_04          QDEEP_OBJECT_DETECT_CONFIG_MODEL_TRAFFIC_TAIWAN_08          QDEEP_OBJECT_DETECT_CONFIG_MODEL_UNATTENDED_OBJECT          QDEEP_OBJECT_DETECT_CONFIG_MODEL_XRAY_INSPECTION_SYSTEM</p>
			<p>指定模型配置</p> <p>QDEEP_OBJECT_DETECT_CONFIG_MODEL_CUSTOMIZED          QDEEP_OBJECT_DETECT_CONFIG_MODEL_CUSTOMIZED_LITE          QDEEP_OBJECT_DETECT_CONFIG_MODEL_CUSTOMIZED_MEDICAL_GRADE          QDEEP_OBJECT_DETECT_CONFIG_MODEL_CUSTOMIZED_MULTI_LABELS          QDEEP_OBJECT_DETECT_CONFIG_MODEL_NVIDIA_CLARA_AX</p>
CHAR *	pszConfigFileName	輸入	<p>預設 NULL</p> <p>指定模型配置</p>
PVOID *	ppDetector	輸出	返回一個物體偵測裝置介面
float *	pModeVersion	輸出	返回模型的版本 <p><b>Only in QDEEP_CREATE_OBJECT_DETECT_EX()</b></p>

型別	參數名稱	輸出入	描述
DWORD	dwFlags	輸入	預設 <b>QDEEP_OBJECT_DETECT_FLAG_FULL</b> 彈性開關子功能
PVOID	pUserData	輸入	預設 <b>NULL</b> 使用者自定資料
CHAR *	pszEncryptionKey	輸入	模型的解密金鑰 <b>Only in QDEEP_CREATE_OBJECT_DETECT_EX()</b>

#### 返回值

代碼	描述
QDEEP_RS_SUCCESSFUL	成功
QDEEP_RS_ERROR_GENERAL	失敗原因：其它錯誤
QDEEP_RS_ERROR_OUT_OF_MEMORY	失敗原因：系統記憶體不足以配置裝置內容
QDEEP_RS_ERROR_OUT_OF_RESOURCE	失敗原因：錯誤的輸出資源
QDEEP_RS_ERROR_INVALID_DEVICE	失敗原因：無法開啟影像或音訊擷取裝置
QDEEP_RS_ERROR_INVALID_PARAMETER	失敗原因：不正確的輸入參數
QDEEP_RS_ERROR_NON_SUPPORT	失敗原因：錯誤編碼格示無法支援

#### 範例程式

```
PVOID m_pDetector = NULL;

QDEEP_CREATE_OBJECT_DETECT(QDEEP_GPU_TYPE_NVIDIA, 0, QDEEP_OBJECT_DETECT_CONFIG_MODEL_TRAFFIC, "QDEEP.OD.TRAFFIC.CFG", &m_pDetector);
```

### 2.1.3 CFG File Introduction

#### CFG 檔案參數

參數名稱	描述
SCOPE.SIZE	指定物件偵側大小
CLASS.SIZE	指定偵測類別數量
EXTRA.CFG.FILEPATH	指定特別設定檔

## 2.2 QDEEP\_START\_OBJECT\_DETECT

### 說明

使用者可利用此函式啟動物體偵測裝置。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_START_OBJECT_DETECT(m_pDetector);
```

C

## 2.3 QDEEP\_STOP\_OBJECT\_DETECT

### 說明

使用者可利用此函式停止物體偵測裝置。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_STOP_OBJECT_DETECT(m_pDetector);
```

C

## 2.4 QDEEP\_DESTROY\_OBJECT\_DETECT

### 說明

使用者可以使用此函式結束物體偵測裝置，並釋放系統裝置的資源

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_DESTROY_OBJECT_DETECT(m_pDetector);
```

C

## 2.5 Object Detect Buffer Function

### 2.5.1 QDEEP\_SET\_VIDEO\_OBJECT\_DETECT\_UNCOMPRESSION\_BUFFER

### 2.5.2 QDEEP\_SET\_VIDEO\_OBJECT\_DETECT\_UNCOMPRESSION\_BUFFER\_EX

#### 說明

使用者可以使用此函示將未編碼的影像數據置入物體偵測引擎

#### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
ULONG	nColorSpaceType	輸入	指定來源數據的色彩空間的格式: QDEEP_COLORSPACE_TYEP_RGB24 QDEEP_COLORSPACE_TYEP_BGR24 QDEEP_COLORSPACE_TYEP_ARGB32 QDEEP_COLORSPACE_TYEP_ABGR32 QDEEP_COLORSPACE_TYEP_YUY2 QDEEP_COLORSPACE_TYEP_YV12
ULONG	nWidth	輸入	指定來源數據的高度
ULONG	nHeight	輸入	指定來源數據的寬度
BYTE *	pFrameBuffer	輸入	指定來源數據的緩存
ULONG	nFrameBufferLen	輸入	指定來源數據的大小
QDEEP_OBJECT_DETECT_BOUNDING_BOX *	pObjectList	輸出	返回物體數據
ULONG	pObjectSize	輸入/輸出	返回物體數據數量
ULONG	nCropX	輸入	指定裁切區域的起始 x 座標 <b>Only in</b> <b>QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_EX()</b>
ULONG	nCropY	輸入	指定裁切區域的起始 y 座標 <b>Only in</b> <b>QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_EX()</b>
ULONG	nCropW	輸入	指定裁切區域的水平寬度 <b>Only in</b> <b>QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_EX()</b>
ULONG	nCropH	輸入	指定裁切區域的垂直高度 <b>Only in</b> <b>QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_EX()</b>
DWORD	dwPostFlags	輸入	預設 QDEEP_OBJECT_DETECT_FLAG_FULL 子功能動態開關參數

#### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

#### 範例程式

```

ULONG pObjectSize = 1000;

QDEEP_OBJECT_DETECT_BOUNDING_BOX* pObjectList = (QDEEP_OBJECT_DETECT_BOUNDING_BOX*)
    malloc( m_DetBoxLen * sizeof(QDEEP_OBJECT_DETECT_BOUNDING_BOX) );

QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER( pDetector,
    QDEEP_COLORSPACE_TYEP_YV12,
    nVideoWidth,
    nVideoHeight,
    pFrameBuffer,
    nFrameBufferLen,
    pObjectList,
    &pObjectSize );

QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_EX( pDetector ,
    QDEEP_COLORSPACE_TYEP_YV12,
    nVideoWidth,
    nVideoHeight,
    pFrameBuffer,
    nFrameBufferLen,
    pObjectList,
    &pObjectSize,
    0,
    0,
    960,
    540);

```

### 2.5.3 QDEEP\_SET\_VIDEO\_OBJECT\_DETECT\_UNCOMPRESSION\_BUFFER\_3D

### 2.5.4 QDEEP\_SET\_VIDEO\_OBJECT\_DETECT\_UNCOMPRESSION\_BUFFER\_3D\_EX

說明

使用者可以使用此函示將未編碼的影像數據置入物體偵測引擎

參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
ULONG	nColorSpaceType	輸入	指定來源數據的色彩空間的格式: QDEEP_COLORSPACE_TYEP_RGB24 QDEEP_COLORSPACE_TYEP_BGR24 QDEEP_COLORSPACE_TYEP_ARGB32 QDEEP_COLORSPACE_TYEP_ABGR32 QDEEP_COLORSPACE_TYEP_YUY2 QDEEP_COLORSPACE_TYEP_YV12
ULONG	nWidth	輸入	指定來源數據的高度
ULONG	nHeight	輸入	指定來源數據的寬度
BYTE *	pFrameBufferL	輸入	指定左邊來源數據的緩存

型別	參數名稱	輸出入	描述
ULONG	nFrameBufferLenL	輸入	指定左邊來源數據的大小
BYTE *	pFrameBufferR	輸入	指定右邊來源數據的緩存
ULONG	nFrameBufferLenR	輸入	指定右邊來源數據的大小
QDEEP_OBJECT_DETECT_BOUNDING_BOX *	pObjectList	輸出	返回物體數據
ULONG	pObjectSize	輸入/輸出	返回物體數據數量
ULONG	nCropX	輸入	指定裁切區域的起始 x 座標 <b>Only in QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_3D_EX()</b>
ULONG	nCropY	輸入	指定裁切區域的起始 y 座標 <b>Only in QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_3D_EX()</b>
ULONG	nCropW	輸入	指定裁切區域的水平寬度 <b>Only in QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_3D_EX()</b>
ULONG	nCropH	輸入	指定裁切區域的垂直高度 <b>Only in QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_3D_EX()</b>
DWORD	dwPostFlags	輸入	預設 QDEEP_OBJECT_DETECT_FLAG_FULL 子功能動態開關參數

#### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

#### 範例程式

```

ULONG pObjectSize = 1000;

QDEEP_OBJECT_DETECT_BOUNDING_BOX* pObjectList = (QDEEP_OBJECT_DETECT_BOUNDING_BOX*)
    malloc( m_DetBOXLen * sizeof(QDEEP_OBJECT_DETECT_BOUNDING_BOX) );

QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_3D( pDetector,
    QDEEP_COLORSPACE_TYEP_YV12,
    nVideoWidth,
    nVideoHeight,
    pFrameBufferL,
    nFrameBufferLenL,
    pFrameBufferR,
    nFrameBufferLenR,
    pObjectlist,
    &pObjectSize );

QDEEP_SET_VIDEO_OBJECT_DETECT_UNCOMPRESSION_BUFFER_3D_EX( pDetector ,
    QDEEP_COLORSPACE_TYEP_YV12,
    nVideoWidth,
    nVideoHeight,
    pFrameBufferL,
    nFrameBufferLenL,
    pFrameBufferR,
    nFrameBufferLenR,
    pObjectList,
    &pObjectSize,
    0,
    0,
    960,
    540);

```

## 2.6 Object Detect Tracking Property

### 2.6.1 QDEEP\_SET\_OBJECT\_DETECT\_PROPERTY

#### 說明

使用者可使用此函式設定物件偵測的門檻值。

#### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
float	fObjectDetectionThreshold	輸入	預設 -1.0 指定物件偵測的門檻值

#### 返回值

返回 QDEEP\_RS\_SUCCESSFUL 表示成功, 否則發生錯誤.

範例程式

```
QDEEP_SET_OBJECT_DETECT_PROPERTY( m_pDetector , -1.0);
```

C

## 2.6.2 QDEEP\_GET\_OBJECT\_DETECT\_PROPERTY

### 說明

使用者可使用此函式獲取物件偵測的門檻值。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
float *	pObjectDetectionThreshold	輸出	預設 NULL 返回目前指定物件偵測的門檻值

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
float m_pObjectDetectionThreshold = 0;  
  
QDEEP_GET_OBJECT_DETECT_PROPERTY ( m_pDetector, &m_pObjectDetectionThreshold );
```

C

## 2.6.3 QDEEP\_SET\_OBJECT\_DETECT\_TRACKING\_PROPERTY

### 說明

使用者可使用此函式設定追蹤的生命週期及範圍參數。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
float	fLifeCycle	輸入	預設 5 指定物體追蹤生命週期
float	fHitThreshold	輸入	預設 2 指定物體追蹤門檻值
float	fScoreThreshold	輸入	預設 0.5 指定物體分數門檻值
BOOL	pFeatureVectorUpdate	輸入	預設 FALSE 指定是否更新特徵向量

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_SET_OBJECT_DETECT_TRACKING_PROPERTY( m_pDetector, 5, 2, 0.5, FALSE);
```

C

## 2.6.4 QDEEP\_GET\_OBJECT\_DETECT\_TRACKING\_PROPERTY

### 說明

使用者可使用此函式獲取追蹤的生命週期及範圍參數。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
float *	pLifeCycle	輸出	預設 NULL 返回目前指定物體追蹤生命週期
float *	pHitThreshold	輸出	預設 NULL 返回目前指定物體追蹤門檻值
float *	pScoreThreshold	輸出	預設 NULL 返回目前指定物體分數門檻值
BOOL *	pFeatureVectorUpdate	輸出	預設 NULL 返回目前是否更新特徵向量

### 返回值

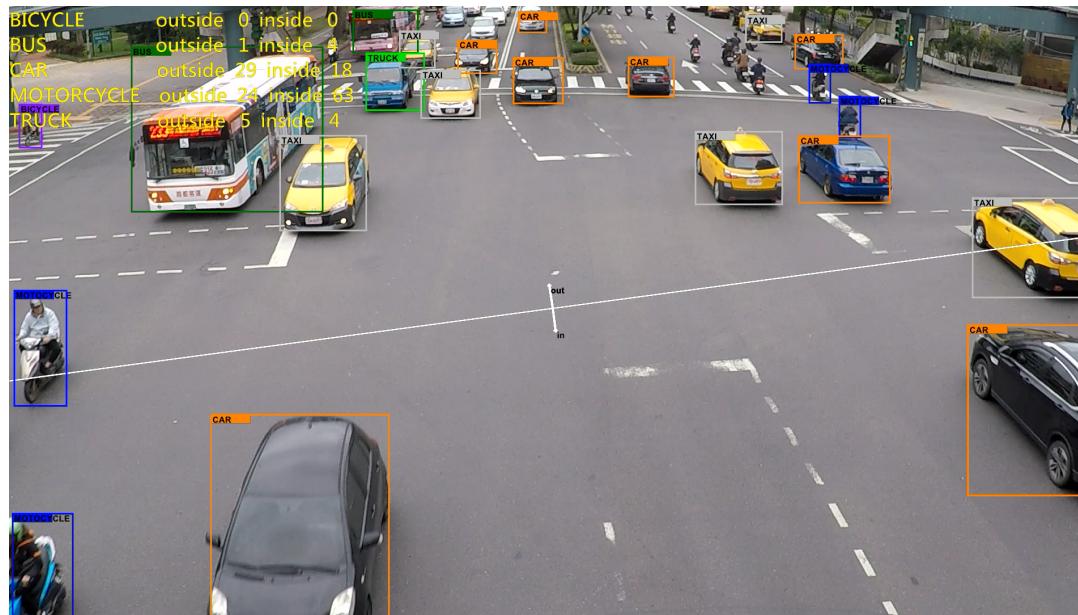
返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
float m_pLifeCycle = 0;  
  
float m_pHitThreshold = 0;  
  
float m_pScoreThreshold = 0;  
  
BOOL pFeatureVectorUpdate = FALSE;  
  
QDEEP_GET_OBJECT_DETECT_TRACKING_PROPERTY( m_pDetector, &m_pLifeCycle, &m_pHitThreshold, &m_pScoreThreshold, &pFeatureVectorUpdate );
```

### 3 記數函式 API

#### 摘要



本章 API 介面提供使用者使用QDEEP 進行線段計數、區域進出記數、區域內記數。

### 3.1 QDEEP\_ADD\_OBJECT\_DETECT\_COUNTING\_LINE

#### 說明

使用者透過 QDEEP\_ADD\_OBJECT\_DETECT\_COUNTING\_LINE 新增一條線段進行計數。

#### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iLineNum	輸入	指定線段編號
ULONG	nPoint1_X	輸入	設定起點 X 座標
ULONG	nPoint1_Y	輸入	設定起點 Y 座標
ULONG	nPoint2_X	輸入	設定終點 X 座標
ULONG	nPoint2_Y	輸入	設定終點 Y 座標
ULONG	nIntersectionMode	輸入	指定穿越線的物體的哪一點被視為物體已經穿越了該線 QDEEP_COUNTING_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_COUNTING_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_COUNTING_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_COUNTING_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

#### 返回值

返回 QDEEP\_RS\_SUCCESSFUL 表示成功, 否則發生錯誤.

#### 範例程式

```
QDEEP_ADD_OBJECT_DETECT_COUNTING_LINE(m_pDetector, 0, 0, 0, 1920, 1080, QDEEP_COUNTING_PROP_INTERSECTION_MODE_BOTTOM_POINT );
```

## 3.2 QDEEP\_DEL\_OBJECT\_DETECT\_COUNTING\_LINE

### 說明

使用者可利用此函式刪除線段計數。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iLineNum	輸入	指定線段編號

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_DEL_OBJECT_DETECT_COUNTING_LINE(m_pDetector, 0 );
```

C

### 3.3 QDEEP\_RESET\_OBJECT\_DETECT\_COUNTING\_LINE

#### 說明

使用者可利用此函式清除線段計數。

#### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iLineNum	輸入	指定線段編號

#### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

#### 範例程式

```
QDEEP_RESET_OBJECT_DETECT_COUNTING_LINE(m_pDetector, 0 );
```

C

## 3.4 QDEEP\_GET\_OBJECT\_DETECT\_COUNTING\_LINE\_STATUS

### 說明

使用者可利用此函式得到物體種類的線段計數，及總計數。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iLineNum	輸入	指定線段編號
ULONG	nClassID	輸入	指定物體種類
ULONG	pClassCrossCount_IN	輸出	返回方向A的種類計數
ULONG	pClassCrossCount_OUT	輸出	返回方向B的種類計數
ULONG	pTotalCrossCount_IN	輸出	返回方向A總計數
ULONG	pTotalCrossCount_OUT	輸出	返回方向B總計數

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### 範例程式

```
QDEEP_GET_OBJECT_DETECT_COUNTING_LINE_STATUS(m_pDetector,
    0,
    QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_PERSON,
    pClassCrossCount_IN,
    pClassCrossCount_OUT,
    pTotalCrossCount_IN,
    pTotalCrossCount_OUT);
```

## 3.5 QDEEP\_ADD\_OBJECT\_DETECT\_COUNTING\_AREA

### 說明

使用者透過 QDEEP\_ADD\_OBJECT\_DETECT\_COUNTING\_LINE 新增一塊區域進行計數。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iAreaNum	輸入	指定區域編號
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG*	pPointsArray_X	輸入	輸入 X 座標數據
ULONG*	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nIntersectionMode	輸入	指定物體穿越區域時，物體的哪個點被視為物體已經穿越了該區域 QDEEP_COUNTING_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_COUNTING_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_COUNTING_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_COUNTING_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 QDEEP\_RS\_SUCCESSFUL 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};  
  
ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};  
  
QDEEP_ADD_OBJECT_DETECT_COUNTING_AREA(m_pDetector, 0, 4, pPointsArray_X, pPointsArray_Y, QDEEP_COUNTING_PROP_INTERSECTION_MODE_BOTTOM_POINT );
```

## 3.6 QDEEP\_DEL\_OBJECT\_DETECT\_COUNTING\_AREA

### 說明

使用者可利用此函式刪除區域計數。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iAreaNum	輸入	指定區域編號

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_DEL_OBJECT_DETECT_COUNTING_AREA(m_pDetector, 0 );
```

C

## 3.7 QDEEP\_RESET\_OBJECT\_DETECT\_COUNTING\_AREA

### 說明

使用者可利用此函式清除區域計數。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iAreaNum	輸入	指定區域編號

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_RESET_OBJECT_DETECT_COUNTING_AREA(m_pDetector, 0 );
```

C

## 3.8 QDEEP\_GET\_OBJECT\_DETECT\_COUNTING\_AREA\_STATUS

### 說明

使用者可利用此函式得到物體種類的區域計數，及總計數。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iAreaNum	輸入	指定區域編號
ULONG	nClassID	輸入	指定物體種類
ULONG	pClassCrossCount_IN	輸出	返回進入區域種類計數
ULONG	pClassCrossCount_OUT	輸出	返回離開區域種類計數
ULONG	pTotalCrossCount_IN	輸出	返回進入總計數
ULONG	pTotalCrossCount_OUT	輸出	返回離開總計數

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### 範例程式

```
QDEEP_GET_OBJECT_DETECT_COUNTING_AREA_STATUS(m_pDetector,
    0,
    QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_PERSON,
    pClassCrossCount_IN,
    pClassCrossCount_OUT,
    pTotalCrossCount_IN,
    pTotalCrossCount_OUT);
```

### 3.9 Counting Callback

#### 3.9.1 QDEEP\_REGISTER\_OBJECT\_DETECT\_COUNTING\_LINE\_CALLBACK

#### 3.9.2 QDEEP\_REGISTER\_OBJECT\_DETECT\_COUNTING\_AREA\_CALLBACK

說明

使用者可使用註冊一個回調函式，當物件跨過線或進出區域時，此回調函式將會被呼叫。使用者也可以從回調函式取得物件資訊及記數。

參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
PF_OBJECT_DETECT_COUNTING_LINE_CALLBACK PF_OBJECT_DETECT_COUNTING_AREA_CALLBACK	pCB	輸入	指向一個自定義回調函式
PVOID	pUserData	輸入	使用者自定資料

返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

#### PF\_OBJECT\_DETECT\_COUNTING\_LINE\_CALLBACK

PF\_OBJECT\_DETECT\_COUNTING\_AREA\_CALLBACK

回調函式的參數

型別	參數名稱	描述
PVOID	pDetector	物體偵測裝置的介面
UINT	iLineNum iAreaNum	指定線段或區域編號
ULONG	nCrossDirection	指定方向
ULONG	nClassID	指定物體種類
ULONG	nObjectID	指定物體
ULONG	nClassCrossCount_IN	返回方向A或區域內的種類計數
ULONG	nClassCrossCount_OUT	返回方向B或區域外的種類計數
ULONG	nTotalCrossCount_IN	返回方向A或區域內的總計數
ULONG	nTotalCrossCount_OUT	返回方向B或區域外的總計數
PVOID	pUserData	使用者自定資料

返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

範例程式

```
// COUNT CALLBACK
QRETURN OBJECT_DETECT_COUNTING_LINE_CALLBACK( PVOID pDetector, UINT iLineNum, ULONG nCrossDirection, ULONG nClassID, ULONG nObjectID, ULONG
nClassCrossCount_IN, ULONG nClassCrossCount_OUT, ULONG nTotalCrossCount_IN, ULONG nTotalCrossCount_OUT, PVOID pUserData );
{
    m_nLineCount[ iLineNum * m_pMainDialog->m_nTotalClass * 2 + nClassID * 2 + nCrossDirection ] ++ ;
}

void test_callback()
{
    QDEEP_REGISTER_OBJECT_DETECT_COUNTING_LINE_CALLBACK (m_pDetector, OBJECT_DETECT_COUNTING_LINE_CALLBACK, pUserData );
}
```

C

## 4 人臉識別函式 API

---

本章 API 介面提供使用者使用 QDEEP 進行人臉識別偵測。

## 4.1 QDEEP\_GET\_OBJECT\_RECOGNITION\_COMPARISON

### 說明

使用者透過這個API來進行人臉偵測以及識別。

### 參數

型別	參數名稱	輸出入	描述
float	pFeatureVectorA	輸入	指定物件 A 的特徵向量
float	pFeatureVectorB	輸入	指定物件 B 的特徵向量
float	pSimilarity	輸出	返回物件 A 與物件 B 的相似度

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
float m_pSimilarity = 0;  
  
QDEEP_GET_OBJECT_RECOGNITION_COMPARISON(m_pFeatureVectorA, m_pFeatureVectorB, &m_pSimilarity);
```

C

## 4.2 QDEEP\_SET\_OBJECT\_DETECT\_POST\_PROCESSING\_UPDATE

### 說明

使用者透過這個API來進行人臉異步更新功能。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置介面
ULONG	nGpuType	輸入	指定 GPU 類別 QDEEP_GPU_TYPE_NVIDIA QDEEP_GPU_TYPE_INTEL_CPU QDEEP_GPU_TYPE_INTEL_GPU QDEEP_GPU_TYPE_INTEL_VPU_Movidius
UINT	iGpuNum	輸入	指定顯示卡：從 0 開始
QDEEP_OBJECT_DETECT_BOUNDING_BOX *	pObjectList	輸入	指定物體數據
ULONG	nObjectSize	輸入	指定物體數據數量
DWORD	dwPostFlags	輸入	預設 QDEEP_OBJECT_DETECT_FLAG_FEATURE_VECTOR 子功能開關參數

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### 範例程式

```
QDEEP_SET_OBJECT_DETECT_POST_PROCESSING_UPDATE(m_pDetector, QDEEP_GPU_TYPE_NVIDIA, 0,  
                                                pObjectList, pObjectSize,  
                                                QDEEP_OBJECT_DETECT_FLAG_FEATURE_VECTOR);
```

## 4.3 Post Processing Update Callback

### 4.3.1 QDEEP\_REGISTER\_OBJECT\_DETECT\_POST\_PROCESSING\_UPDATE\_CALLBACK

#### 說明

使用者可使用註冊一個回調函式，當物件更新人臉資訊的時候，此回調函式將會被呼叫。

#### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
PF_OBJECT_DETECT_POST_PROCESSING_UPDATE_CALLBACK	pCB	輸入	指向一個自定義回調函式
PVOID	pUserData	輸入	使用者自定資料

#### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### PF\_OBJECT\_DETECT\_POST\_PROCESSING\_UPDATE\_CALLBACK

#### 回調函式的參數

型別	參數名稱	描述
PVOID	pDetector	物體偵測裝置的介面
QDEEP_OBJECT_DETECT_BOUNDING_BOX *	pObject	返回指定物體數據
PVOID	pUserData	使用者自定資料

#### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

## 5 事件函式 API

---

本章 API 介面提供使用者使用 QDEEP 進行事件偵測。

## 5.1 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECTS\_CROSSING\_LINE

### 說明

使用者透過這個API來進行物件跨線的事件，當跨線物件的數量大於指定數量門檻，即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nPoint1_X	輸入	設定起點 X 座標
ULONG	nPoint1_Y	輸入	設定起點 Y 座標
ULONG	nPoint2_X	輸入	設定終點 X 座標
ULONG	nPoint2_Y	輸入	設定終點 Y 座標
ULONG	nCrossDirection	輸入	指定方向(0:逆時鐘跨線, 1:順時鐘跨線)
ULONG	nObjectSizeThreshold	輸入	指定數量門檻
ULONG	nDurationThreshold	輸入	指定穿越線的物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定穿越線的物體的哪一點被視為物體已經穿越了該線 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 QDEEP\_RS\_SUCCESSFUL 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECTS_CROSSING_LINE(pDetector, 0, pSelectClassIDs, 3, 0, 0, 1920, 1080, 0, 1, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT );
```

## 5.2 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECTS\_IN\_AREA

### 說明

使用者透過這個API來進行當前幀下在區域內物件數量的事件，當物件的數量大於指定數量門檻，即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

## 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nObjectSizeThreshold	輸入	指定數量門檻
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定物體撞擊區域的哪一點被視為物體在區域內 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

## 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

## 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECTS_IN_AREA(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 1, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT );
```

## 5.3 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECTS\_ENTER\_AREA

### 說明

使用者透過這個API來進行物件跨入區域的事件，當跨入區域物件的數量大於指定數量門檻，即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nObjectSizeThreshold	輸入	指定數量門檻
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定物體的哪個點撞擊區域被視為物體已進入該區域 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECTS_ENTER_AREA(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 1, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.4 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECTS\_LEAVE\_AREA

### 說明

使用者透過這個API來進行物件跨離區域的事件，當跨離區域物件的數量大於指定數量門檻，即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nObjectSizeThreshold	輸入	指定數量門檻
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定物體撞擊區域的哪個點被視為物體已離開該區域 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECTS_LEAVE_AREA(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 1, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.5 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECTS\_STOP\_IN\_AREA

### 說明

使用者透過這個API來進行物件停在該區域的事件，當停在該區域物件的數量大於指定數量門檻，即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nObjectSizeThreshold	輸入	指定數量門檻
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定物體撞擊區域的哪個點被視為物體停在該區域內 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECTS_STOP_IN_AREA(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 1, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.6 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECTS\_STOP\_IN\_AREA\_WITHOUT\_WARNING\_SIGN

### 說明

當該區域沒有停止標誌時，使用者可以利用此事件函數來偵測已停止的物體。當停止的物體數量超過閾值時，將會觸發 PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG *	pWarningSignClassIDs	輸入	指定警告標誌類別的 ID 緩衝區
ULONG	nWarningSignClassSize	輸入	指定警告標誌類別的尺寸
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nObjectSizeThreshold	輸入	指定數量門檻
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	請指定物體撞擊區域的哪個點被視為物體在該區域內停止，且沒有警告標誌 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 QDEEP\_RS\_SUCCESSFUL 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG nWarningSignClassIDs[1] = { 0 };

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECTS_STOP_IN_AREA_WITHOUT_WARNING_SIGN(pDetector, 0, pSelectClassIDs, 3, nWarningSignClassIDs, 1, 4,
pPointsArray_X, pPointsArray_Y, 1, 500, QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.7 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECTS\_COLLIDE\_IN\_AREA

### 說明

使用者可以利用此事件函數來偵測在該區域內發生碰撞的物體。當碰撞的物體數量超過閾值時，將會觸發 PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nObjectSizeThreshold	輸入	指定數量門檻
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定物體撞擊區域的哪個點被視為兩個物體在該區域內碰撞 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 QDEEP\_RS\_SUCCESSFUL 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECTS_COLLIDE_IN_AREA(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 1, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.8 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECT\_APPEAR\_IN\_AREA

### 說明

使用者透過這個API來進行當前幀區域有物件的事件，當區域內有物件出現即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定物體撞擊區域的哪個點被視為物體出現在該區域 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECT_APPEAR_IN_AREA(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.9 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECT\_ABSENT\_IN\_AREA

### 說明

使用者透過這個API來進行當前幀區域無物件的事件，當區域無物件出現即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定物體撞擊區域的哪個點被視為物體不在該區域內 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECT_ABSENT_IN_AREA(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.10 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECT\_LOITERING\_IN\_AREA

### 說明

使用者透過這個API來進行區域內遊蕩物件的事件，當區域內有物件遊蕩超過一定時間，即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
ULONG	nIntersectionMode	輸入	指定物體撞擊區域的哪個點被視為物體在該區域內徘徊 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECT_LOITERING_IN_AREA(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 500,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.11 QDEEP\_ADD\_OBJECT\_DETECT\_EVENT\_OBJECT\_DIRECTION\_REVERSED

### 說明

使用者透過這個API來進行區域內物件有逆向的事件，當區域內有物件不是遵循指定角度的方向，即會觸發PF\_OBJECT\_DETECT\_EVENT\_CALLBACK。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
ULONG *	pSelectClassIDs	輸入	指定類別數據
ULONG	nSelectClassSize	輸入	指定類別數量
ULONG	nTotalPoints	輸入	設定區域的點個數
ULONG *	pPointsArray_X	輸入	輸入 X 座標數據
ULONG *	pPointsArray_Y	輸入	輸入 Y 座標數據
ULONG	nDurationThreshold	輸入	指定物體持續時間閾值
float	fDirAngle	輸入	指定角度(角度: 0.0 ~ 360.0)
float	fToleranceAngle	輸入	指定容許角度(角度: 0.0 ~ 360.0)
ULONG	nIntersectionMode	輸入	指定物體撞擊區域的哪個點被視為物體改變方向 QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT QDEEP_EVENT_PROP_INTERSECTION_MODE_UPPER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_LOWER_BOUNDING_BOX QDEEP_EVENT_PROP_INTERSECTION_MODE_FULL_BOUNDING_BOX

### 返回值

返回 QDEEP\_RS\_SUCCESSFUL 表示成功，否則發生錯誤。

### 範例程式

```
ULONG pSelectClassIDs[3] = {QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_BIKE,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_MOTOR,
                            QDEEP_OBJECT_DETECT_CLASS_TRAFFIC_TAIWAN_08_TAXI};

ULONG pPointsArray_X[4] = {0, 1920, 1920, 0};

ULONG pPointsArray_Y[4] = {0, 0, 1080, 1080};

QDEEP_ADD_OBJECT_DETECT_EVENT_OBJECT_DIRECTION_REVERSED(pDetector, 0, pSelectClassIDs, 3, 4, pPointsArray_X, pPointsArray_Y, 500, 120, 10,
QDEEP_EVENT_PROP_INTERSECTION_MODE_BOTTOM_POINT);
```

## 5.12 QDEEP\_DEL\_OBJECT\_DETECT\_EVENT

### 說明

使用者透過這個API來進行刪除事件。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_DEL_OBJECT_DETECT_EVENT(pDetector, 0);
```

C

## 5.13 QDEEP\_RESET\_OBJECT\_DETECT\_EVENT

### 說明

使用者透過這個API來進行重設事件。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
QDEEP_RESET_OBJECT_DETECT_EVENT(pDetector, 0);
```

C

## 5.14 QDEEP\_GET\_OBJECT\_DETECT\_EVENT\_CURRENT\_STATUS

### 說明

使用者透過這個API來取得當前物件符合指定事件的相關資訊和數量。

### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
UINT	iEventNum	輸入	指定事件編號
QDEEP_OBJECT_DETECT_BOUNDING_BOX *	pObjectList	輸出	返回符合事件的物體數據
ULONG	pObjectSize	輸入/輸出	返回符合事件的物件數量

### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功, 否則發生錯誤.

### 範例程式

```
ULONG pObjectSize = 1000;

QDEEP_OBJECT_DETECT_BOUNDING_BOX* pObjectList = (QDEEP_OBJECT_DETECT_BOUNDING_BOX*)
    malloc( m_DetBOXLen * sizeof(QDEEP_OBJECT_DETECT_BOUNDING_BOX) );

QDEEP_GET_OBJECT_DETECT_EVENT_CURRENT_STATUS(pDetector, 0, pObjectList, &pObjectSize);
```

## 5.15 Object Event Callback

### 5.15.1 QDEEP\_REGISTER\_OBJECT\_DETECT\_EVENT\_CALLBACK

#### 說明

使用者可使用註冊一個回調函式，當物件符合事件的時候，此回調函式將會被呼叫。

#### 參數

型別	參數名稱	輸出入	描述
PVOID	pDetector	輸入	物體偵測裝置的介面
PF_OBJECT_DETECT_EVENT_CALLBACK	pCB	輸入	指向一個自定義回調函式
PVOID	pUserData	輸入	使用者自定資料

#### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

### PF\_OBJECT\_DETECT\_EVENT\_CALLBACK

#### 回調函式的參數

型別	參數名稱	描述
PVOID	pDetector	物體偵測裝置的介面
UINT	iEventNum	返回指定事件
QDEEP_OBJECT_DETECT_BOUNDING_BOX *	pObjectList	返回符合事件的物體數據
ULONG	nObjectSize	返回符合事件的物件數量
PVOID	pUserData	使用者自定資料

#### 返回值

返回 **QDEEP\_RS\_SUCCESSFUL** 表示成功，否則發生錯誤。

#### 範例程式

C

```
// COUNT CALLBACK
QRETURN QDEEP_OBJECT_DETECT_EVENT_CALLBACK( PVOID pDetector, UINT iEventNum, QDEEP_OBJECT_DETECT_BOUNDING_BOX *pObjectList, ULONG nObjectSize,
PVOID pUserData );
{
    printf("[iEventNum %d]", iEventNum);
    for(int i = 0; i < nObjectSize; ++i){
        printf("[%d], nClassID: %d", i, pObjectList[i].nClassID);
    }
}

void test_callback()
{
    QDEEP_REGISTER_OBJECT_DETECT_EVENT_CALLBACK (m_pDetector, QDEEP_OBJECT_DETECT_EVENT_CALLBACK, pUserData );
}
```