



**PC VCA**

**Integration Quick Start  
Guide**

# Table of Contents

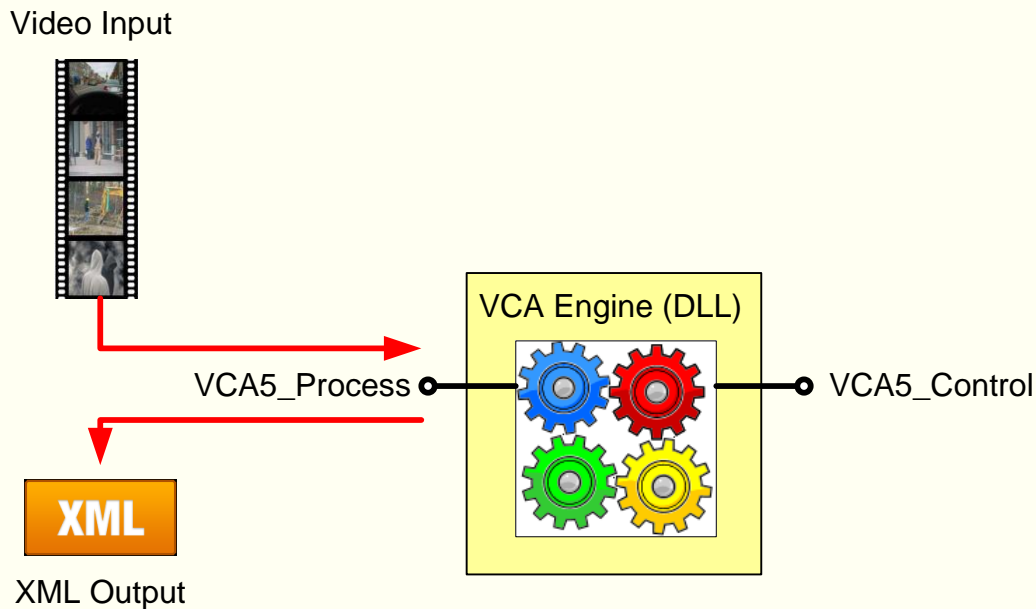
1. Introduction .....	3
2. System Overview .....	4
2.1 VCA5_Process .....	5
2.2 VCA5_Control .....	7
2.3 Activation .....	8
3. VCA Integration Scenarios .....	9
3.1 "I just want to know when a VCA rule is triggered, I don't care about rendering the VCA annotation on top of the video" .....	9
3.2 "I want to store all the VCA metadata in a database for forensic analysis later, but I'm not keen on having to write lots of code to perform the rendering of the metadata or zones/rules configuration on the video". .....	9
3.3 "I want to render the metadata (tracks/trails) myself, but the zone and 3D calibration rendering sounds like a lot of work". .....	9
3.4 "I want full control over everything to do with metadata, zones and calibration rendering. The final look is more important than the speed of implementation". .....	10
3.5 "I want to render the metadata stream and I am using VCApresence. At least one installation of the system will also be using VCApresence". .....	10
REVISION HISTORY .....	11

# 1. Introduction

---

This document aims to provide an overview of the steps involved when integrating with the VCA5 video analytics SDK. This document serves as a quick introduction to the whole integration process, including control and streaming.

## 2. System Overview



The core of the VCA5 SDK is the VCA Engine. This is packaged in a shared library (DLL or Dynamic Link Library on Windows), and is linked to a client application at run time.

The VCA5\_Process function is the most frequently called function – it transforms video input data to XML output metadata that describes the objects and events detected in the scene.

In most integration scenarios, the VCA Engine DLL would be integrated as part of the 3<sup>rd</sup> party application. The 3<sup>rd</sup> party application is responsible for calling VCA5\_Process and VCA5\_Control to process the video and configure the engine, respectively.

A selection of demonstration applications are included with the SDK to illustrate how the VCA Engine can be integrated into a 3<sup>rd</sup> party application.

## 2.1 VCA5\_Process

### Overview

VCA5\_Process is the main function in the VCA5 SDK. It is called to perform the actual video analysis. The client application passes in a raw (uncompressed) video frame, and receives an XML document in return. The XML document completely describes everything detected in that frame including objects, events and counts.

### Metadata Streams

The VCA metadata is an uncompressed human-readable XML stream that provides a full description of every tracked object, every event that has been triggered as well as a host of other detailed VCA data.

### Rendering the Metadata Stream

There are a number of options for rendering the VCA annotation (tracking boxes and trails, alarms and counters). The choice is as follows:

- Retrieve the XML stream from the output of VCA5\_Process, decode the XML and render the data in a custom manner. This method would likely be most appropriate if the platform for integration already supports optimized rendering of lines and rectangles on top of the video. This is also the most flexible method with the developer having ultimate control over the whole rendering process. On the other hand, this is also the most time consuming method. Note that with a VCApresence license, portions of the metadata are encrypted, so this method cannot be used: the VCA Metadata Renderer DLL must be used instead.
- Retrieve the XML stream from the output of VCA5\_Process, and pass the XML metadata directly to the VCA Metadata Rendering DLL (supplied as part of the SDK). The VCA Metadata Rendering DLL takes care of rendering all VCA related data on top of a DirectDraw surface provided by the client application. The DLL can render the full metadata (tracks, trails, counts, alarms), and also supports the rendering of zones, lines and calibration setup. This method provides the advantage of allowing the developer to get the integration going quickly. The complicated rendering of transparent polygons and the 3D calibration scene is handled inside the DLL and exposed via a simple API. When using a VCApresence license, this is the only way to render the encrypted metadata stream.

## More Information

For more information about the VCA5\_Process function, consult the **UDA5 VCA5 SDK Manual**.

Further information about the VCA XML metadata, including an XML schema is available in the **VCAsys Metadata Format** manual. For more information about the VCA Metadata Rendering DLL, please consult the **VCA MetaRender API Manual**.

All documents are located in the documentation section of the SDK.

## Demo Applications

The SDK also includes a number of demo applications that enable the integrator to get started from a tested codebase. These are as follows:

- **StartVCA.exe:** A collection of simple demo applications that illustrate how to start up the SDK and use it to process video frames.
- **SimpleVCA.exe:** More complicated program that shows VCA configure save/load, parsing metadata and simple metadata rendering.
- **DemoVCA.exe:** A fully featured demo application that illustrates how to configure detection filters, display video and render the results of VCA processing.

## 2.2 VCA5\_Control

### Overview

The VCA5\_Control function is used to configure and control the VCA engine. This function is used to perform such functions as configuring a detection rule or zone.

### More Information

For more information about the VCA5\_Process function, consult the **UDA5 VCA5 SDK Manual**, located in the documentation section of the SDK.

### Demo Applications

The SDK includes a number of demo applications, complete with source code that illustrates the use of VCA5\_Control. These are as follows:

- **StartVCA.exe:** A collection of simple demo applications that illustrate how to start up the SDK and use it to process video frames.
- **SimpleVCA.exe:** More complicated program that show VCA configure save/load, parsing metadata and show simple draw metadata.
- **DemoVCA.exe:** A fully featured demo application that illustrates how to configure detection filters, display video and render the results of VCA processing.

## 2.3 Activation

### Overview

The VCA5 SDK is a licensed product. Therefore, it's necessary to activate the engine before any VCA processing can take place. This is a simple procedure that's clearly documented in the **UDA5 VCA5 SDK Manual**.

During activation, the VCA5 SDK generates a hardware key (so-called Globally Unique Identifier, or GUID) that's unique to your hardware. This hardware key can be generated using the following sources as a seed for the hash:

- **A capture card.** A GUID can be generated based on a capture card, if installed. In this case, the activation code will be locked to the capture card. The VCA5 SDK will only work if that capture card is installed in the PC.
- **The PC hardware.** A GUID can be generated based on a selection of the PC's hardware. Typically, up to 3 items of the PC's hardware can be changed before the VCA engine will need to be re-activated. In this mode, the activation key is independent of the capture card and the VCA engine will only work on the PC on which it was licensed.

### More Information

For more information about activating the VCA5 SDK function, consult the **UDA5 VCA5 SDK Manual**, located in the documentation section of the SDK.



## 3. VCA Integration Scenarios

---

### **3.1 “I just want to know when a VCA rule is triggered, I don’t care about rendering the VCA annotation on top of the video”**

**Solution:**

Set up the detection rule(s) using the VCA5\_Control function. Push video into the VCA5\_Process function and examine the metadata that’s returned. Look for the <events> section in the metadata document and parse the XML to determine the events that have been generated as a result of rules being triggered.

**Ease of implementation: Easy**

### **3.2 “I want to store all the VCA metadata in a database for forensic analysis later, but I’m not keen on having to write lots of code to perform the rendering of the metadata or zones/rules configuration on the video”.**

**Solution:**

Retrieve the full XML metadata stream from the output of VCA5\_Process. Archive the XML into your database by whichever method is appropriate. Pass the XML metadata to the VCA Metadata Rendering DLL and allow it to render the metadata on top of the video stream. Use the VCA Metadata Rendering DLL to perform zone drawing/display and for 3D calibration setup.

**Ease of implementation: Medium**

### **3.3 “I want to render the metadata (tracks/trails) myself, but the zone and 3D calibration rendering sounds like a lot of work”.**

**Solution:**

Retrieve the full XML metadata stream from the output of VCA5\_Process. Parse the XML and use it to display the metadata in whichever format desired. Use the VCA Metadata Render DLL to render the zone and 3D calibration on top of the video.

**Ease of implementation: Medium**

### **3.4 “I want full control over everything to do with metadata, zones and calibration rendering. The final look is more important than the speed of implementation”.**

#### **Solution:**

Retrieve the full XML metadata stream from the output of VCA5\_Process. Parse the XML and use it to display the metadata in whichever format desired. Use whichever method desired to render the zones and calibration configuration interfaces.

**Ease of implementation: Difficult**

### **3.5 “I want to render the metadata stream and I am using VCApresence. At least one installation of the system will also be using VCApresence”.**

#### **Solution:**

Since VCApresence encrypts portions of the metadata (VCAsurveillance does not have this limitation), the only way to render the metadata stream is with the use of the VCA Metadata Rendering DLL. The VCA Metadata Rendering DLL internally decrypts the stream and renders it onto the video. Attempting to use the metadata stream directly will result in portions of missing annotation.

**Ease of implementation: Medium**

# REVISION HISTORY

MANUAL#	DATE (M/D/Y)	COMMENTS
0.1	30/03/10	First draft
1.0	04/01/2010	Reviewed
1.2	11/07/2011	V1.2.0 Updated
02-2017-A	02/27/2017	Correction on typos Manual name changed (VCA5 SDK Integration Quick Start Guide -> PC VCA Integration Quick Start Guide)