

---

# VaxOCR Genesis

## *ON PC Version*

### User Manual

---

August 2022

**Version 1.1**

# 1. Contents

1. Contents.....	2
2. Introduction .....	4
3. Software License options .....	4
4. Operating system and software requirements.....	5
5. System requirements and PC specification .....	5
6. Licensing.....	6
6.1 30 day trial license .....	6
6.2 Commercial software license.....	6
6.3 Commercial hardware license (USB).....	6
6.4 Additional Notes .....	6
7. VaxOCR Genesis Installation .....	7
8. Requirements for Images.....	9
8.1 Camera Orientation and Placement .....	9
8.2 Examples of camera orientation.....	10
8.3 Character pixel height.....	10
9. Speed of a moving object / camera .....	11
10. VaxOCR Genesis Operation.....	12
11. VaxOCR Genesis Lite .....	12
12. OCR Configuration Main Screen .....	13
13. Configuring Genesis Parameters.....	14
14. Video Configuration .....	20
15. Calculation of the average characters height .....	24
16. Region of Interest Configuration .....	26
17. Results Publishing Configuration Screen .....	28
18. Saving the VaxOCR Genesis Configuration .....	32
19. VaxOCR Genesis Viewer.....	35
19.1 Initial setup .....	36
20. Main Screen .....	39
21. Action Buttons .....	40
22. Running the Program .....	40
23. More on VaxOCR Genesis Results Integration.....	41
23.1 TCP/IP sockets.....	41
23.2 TCP/IP sockets – sending plain results.....	44
23.3 HTTP-POST XML or JSON.....	44
23.4 Send Results to the Optix N <sup>x</sup> VMS System .....	46
24. Words Available for Dynamic Text Replacement .....	47
25. Genesis Grammar Rules Definition.....	49
25.1 Grammar rules information structure .....	49

25.2	Tokens, operators, and definitions .....	49
25.2.1	Replacement and deletion rules section .....	49
25.2.2	Definition of variables section .....	49
25.2.3	Definition of variables and grammar rules sections .....	49
25.2.4	Grammar rules section.....	49
25.2.5	Common to all sections.....	49
25.3	Replacement rules section.....	50
25.4	Grammar rules section.....	52
25.5	Example, extracting information from fast-food ticket restaurant .....	55
25.6	Example, extracting digits from a long code/date .....	56
25.1	Example, defining an optimum character set for a label.....	58
	If you have the opportunity to design say a packaging label for optimum recognition, then use a font as specified above. E.g. for the PC version use Charles Wright and select the Charles Wright optimisation button. ....	58
26.	ASCII SPECIAL CODES .....	59
27.	Changelog.....	60
27.1	Version 23-12-2021.....	60
27.2	Version 1.1 08-08-2022.....	60

## 2. Introduction

**VaxOCR Genesis** is a high-performance generic OCR reader developed by Vaxtor Technologies to read any combination of uppercase Latin characters and/or numbers arranged in up to three lines. Operating under any lighting conditions, it is unaffected by image quality, print degradation and font shape variations.

The software can be user-configured to read a variety of items including: ID cards, passports, stock labels, shipping labels, delivery notes, tickets & receipts, product codes, fast food till receipts, bank notes, vehicle signage, VIN codes, video OSD text etc.

Complex syntax rules can be defined to read and report the text correctly.

The generic OCR engine takes advantage of many of the current integration and publishing capabilities that have been developed by Vaxtor over many years. The reads can be written to the Axis internal SD card (if fitted), a shared network drive, sent to Helix (Vaxtor's comprehensive Back Office program which can be PC or cloud based), or data can be sent to many other destinations and VMS systems such as Milestone and Genetec using the comprehensive list of reporting options.

This manual will guide you through the installation, configuration and these result publishing procedures. It also includes some tips about camera setup and configuration to maximize the OCR analytics performance and read accuracy.

*Note that a separate version is also available which runs on certain intelligent cameras including Axis, Mobotix, Hanwha and i-Pro amongst others. Contact Vaxtor for more details.*

### Main functions

- Users may review real time video whilst performing OCR analytics
- May be run as an application or silently as a Windows service
- Runs in free-flow mode, no external signals are required to trigger the OCR
- Supports a wide variety of default camera manufacturers and models
- Generic ONVIF and RTSP support for legacy camera connections
- Support for recorded video processing
- Code reads are saved directly to a local hard disk
- Results can be transmitted in real time through TCP/IP sockets
- Integrates with the VaxOCR Genesis real time results transmission mechanism
- Uses the proprietary Vaxtor OCR Engine designed and developed by Vaxtor Recognition Technologies over many years.

## 3. Software License options

Once installed, VaxOCR Genesis activates a default 30 days trial license.

In order to obtain a commercial perpetual license, please contact [info.eu@vaxtor.com](mailto:info.eu@vaxtor.com)

## 4. Operating system and software requirements

- Operating System: Microsoft Windows 7 or above (64 bit)
- Microsoft Visual C++ 2015 x64 Runtime Package
- MS Framework 4.5.2 or above
- A valid VaxOCR license installed (trial or perpetual)

## 5. System requirements and PC specification

Computer specifications will heavily depend on the type of camera scenarios you are going to manage and the number of cameras to be managed simultaneously from the PC. Please contact us for advice.

### RAM memory usage

Your system should have a minimum of 16GB RAM with an additional 1 GB RAM per camera instance.

### Computer Processor

Taking Intel as a benchmark, the processor required depends on the vehicle speed, camera horizontal angle and the camera vertical angle. All angles should be minimal for optimum recognition. As a general guide, one physical core is required above 3.0GHz per camera for slow or stopped scenarios and one physical core is required above 4.0GHz per camera for faster moving codes / text. **Note that the processor MUST support AVX2.**

### Examples:

**Intel Core i7-9700T, (2.0-4.3) GHz, 8 Cores – 8 Threads** – can process 8 cameras.

The **Intel i9-9900T has 8 cores & 16 threads** and can process 8 cameras or 16 very slow-moving objects.

See separately available document: 'VaxALPR – CPU Recommendations'

This guide is indicative only. It would be possible to achieve many other valid combinations, for example using an Intel i7-7700T to manage 4 cameras.

## 6. Licensing

### 6.1 30 day trial license

Run **license\_trial.bat** (located in Program Files/Vaxtor Technologies/Vax OCR Genesis 1.0 ) to install the SafeNet environment and VaxOCR Genesis trial license.

(SafeNet is a commercially available software protection system)

### 6.2 Commercial software license

Commercial licenses are available in both hardware key (USB dongle) or software formats.

- On your PC go to the directory where VaxOCR Genesis has been installed and run the application **license\_perpetual.exe**
- Select the main Tab '**Collect Status Information**' and click on the '**Collect Information**' button at the bottom and when prompted and save as the suggested .c2v file somewhere on your hard drive. E.g. Desktop
- E-mail this file to your support / sales contact or support@vaxtor.es
- You will receive a permanent commercial software license (.v2c file) in 1-2 working days. Copy this to your PC (for example onto the Desktop or the Documents directory)
- To install this file and license the software, run the application **license\_perpetual.exe** once more and select the '**Apply License File**' main tab and upload the "v2c" file you received by clicking on the three dots bottom right to select the directory where it is stored and click Open. This will convert your demonstration license into a commercial one.
- Alternatively, you may be sent a license code. To redeem this, go to <https://licensing.vaxtor.com>

### 6.3 Commercial hardware license (USB)

- Request a hardware license by e-mail [info.eu@vaxtor.com](mailto:info.eu@vaxtor.com)
- You will receive a USB key by courier in approximately 3 working days  
*Note that there is a charge for the USB security device.*

### 6.4 Additional Notes

- Trial licenses are not available for a virtual machine environment
- Commercial licenses for virtual machines are only available in USB dongle format
- Once the trial license period expires, is **not** possible to deploy another trial license on the same PC nor extend the trial license period

***We strongly recommend you to back up your system before installing a software-based key.***

Note that you may also use this program to transfer a bought software license key to another machine by selecting the main Tab 'Transfer License'. A hardware USB key may simply be plugged into another suitable computer to be able to run the software there.

## 7. VaxOCR Genesis Installation

Note that some of the install dialogue boxes are common to all our OCR products and may refer to VaxALPR instead of VaxOCR Genesis.

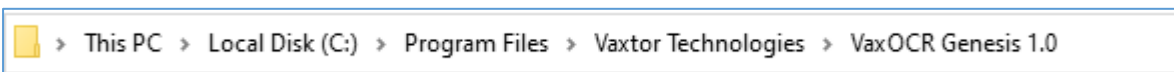
Run **VaxOCR Genesis Installer** to install the Vaxtor software components onto your PC.

- You must have administrator permission to allow the installer to register some of the software components. You normally can right click and 'Run as administrator'.
- Accept the License Agreement to start the installation.
- Ensure the target installation directory has read/write permission.  
(VaxOCR Genesis saves the configuration files inside its installation directory and this must not be read only).

The package VaxOCR Genesis is made up of 2 different software components that will be installed:

- **VaxOCR Genesis Requirements:** [VaxOCR Genesis Requirements.exe](#)
- **Vaxtor OCR Genesis:** [VaxOcrGenesis.exe](#)

The normal program location will be:

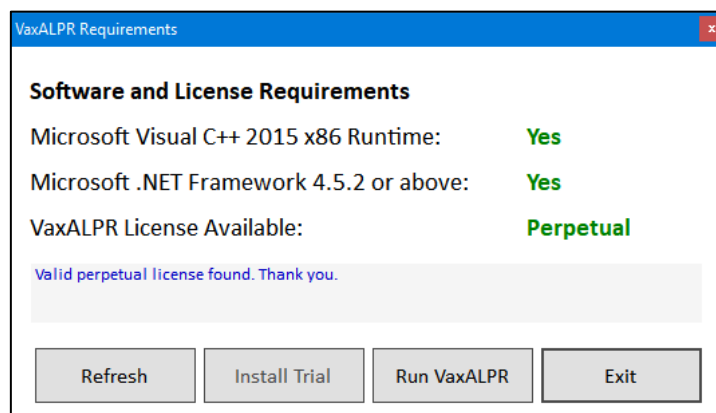


### VaxOCR Genesis Requirements

This software checks if the third-party software requirements are met including the existence of a valid VaxOCR Genesis License.

You can optionally deploy the trial license directly from here.

You must have administrator privileges to run this software.



### VaxOCR Genesis

This is the main application and is where the video sources are setup (e.g. an IP camera) and the OCR parameters adjusted. The user can then setup where the results are to be transmitted to.



## 8. Requirements for Images

OCR (Optical Character Recognition) is an image processing technology used to locate and read text which can be printed onto a variety of substances such as paper, card, thermal paper, plastics, wood, metals etc. using a variety of methods and fonts. However, recognizing the characters is more challenging if the images have any of the following characteristics or artifacts:

- Over / Under exposed
- Blurred or distorted text
- Unevenly lit
- Very Low contrast
- Damaged or worn text
- Bad weather conditions if outdoor deployment on say vehicles or containers.



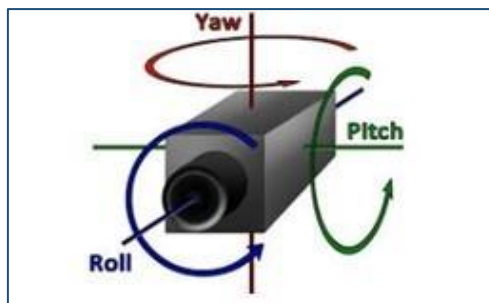
Badly printed and damaged text is impossible to read correctly

The less clear that the text images are, then the more likely it is that the OCR software will report them incorrectly -- Garbage In --> Garbage Out.

As a general guide, if a human has difficulty reading the codes, then so will a computer. It is important to note that OCR technology not Plug and Play and it is important to take into consideration all of these elements together when setting up your system.

### 8.1 Camera Orientation and Placement

The image orientation is a key factor to achieve the maximum OCR efficiency. It is recommended that you stay below the following thresholds:



**Roll Rotation:** Below 25°

**Yaw Rotation:** Below 25°

**Pitch Rotation:** Below 25°

In other words, keep the camera as perpendicular to the target object as possible. Accuracy will decrease significantly if these guidelines are not adhered to.

## 8.2 Examples of camera orientation

The following examples illustrate various camera positions.

The first example is perfect but not always possible, however the software is very tolerant to real-world conditions.



## 8.3 Character pixel height

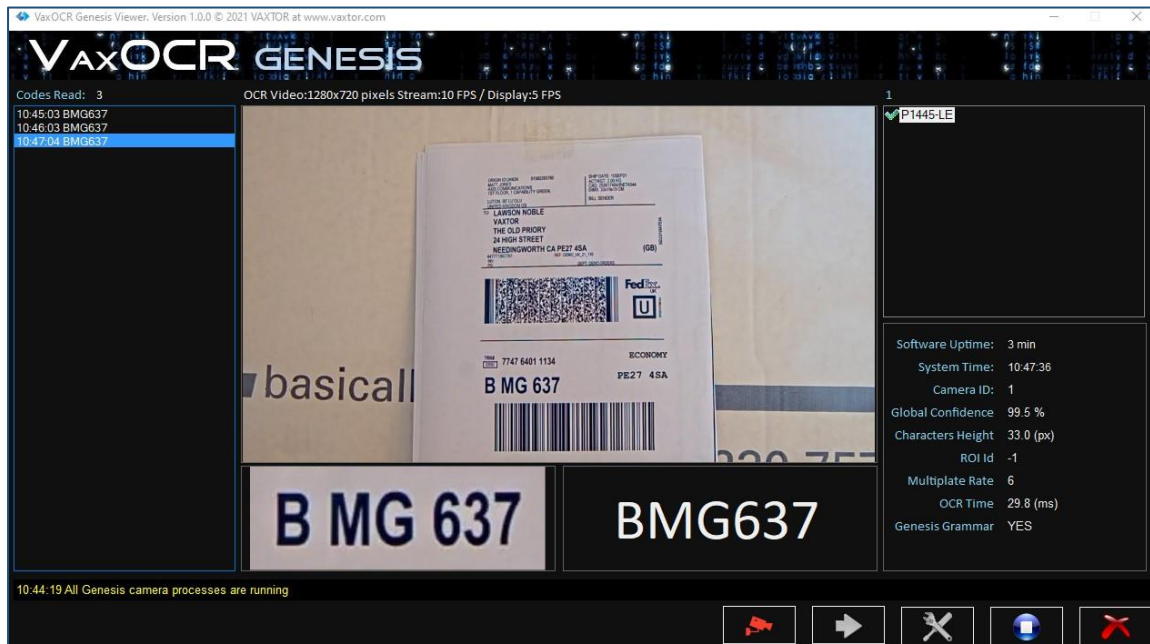
The most accurate way of measuring the size of the characters on an object is to use the height of each character. The optimal range is from 20 to 30 pixels high. There is a tool to help determine the correct lens to use to attain this height described later in this manual.

The software can be configured to read characters as low as 14 pixels high – or as high as 70 pixels high, but in order to maximize reading accuracy and not waste processor time looking for very large characters you should try to keep to this 20-30 range.

## 9. Speed of a moving object / camera

Labels and identification codes can be quite complex and take longer to read than for simple objects such as license plates. This can be due to the huge variance in possible locations, qualities and orientation of text and can also depend upon the complexity of the background. This should be taken into consideration when specifying the speed of a PC if the objects will be moving such as items on a conveyor belt – or identification codes on vehicles.

Simpler labels can be read much faster. In the following example the ticket took only 40ms to read on an i9 computer and so it could have been moving quite fast along a conveyor belt:



## 10. VaxOCR Genesis Operation

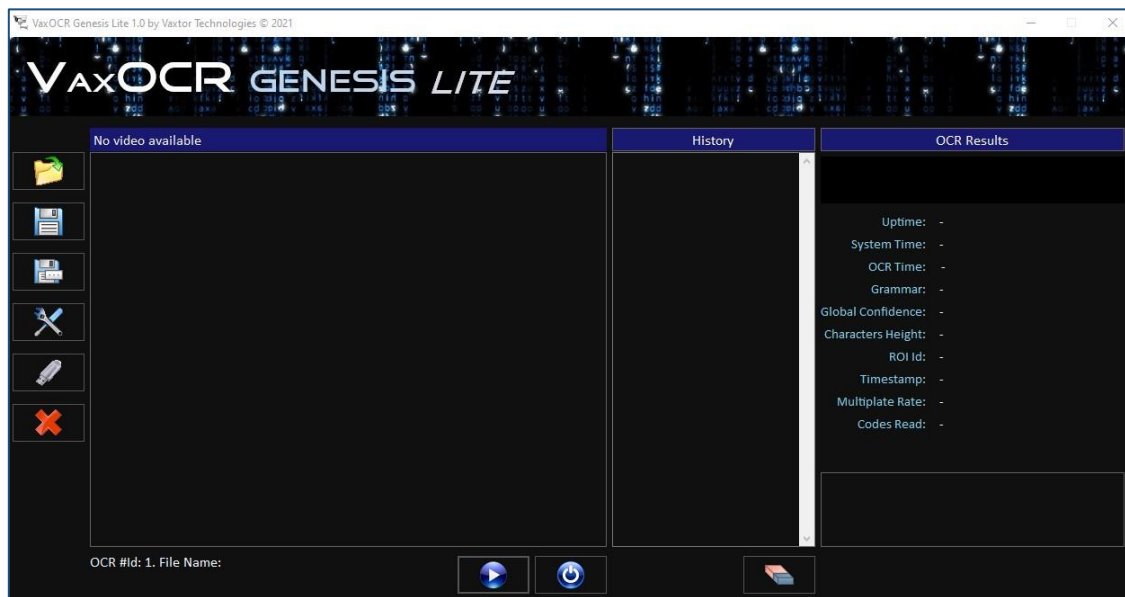
There should be two installed programs on your desktop: VaxOcrGenesisLite and the main program VaxOCR Genesis 1.0.



## 11. VaxOCR Genesis Lite

This is where each camera is connected and the recognition parameters setup. Once all required cameras have been set up then the main program can be run.

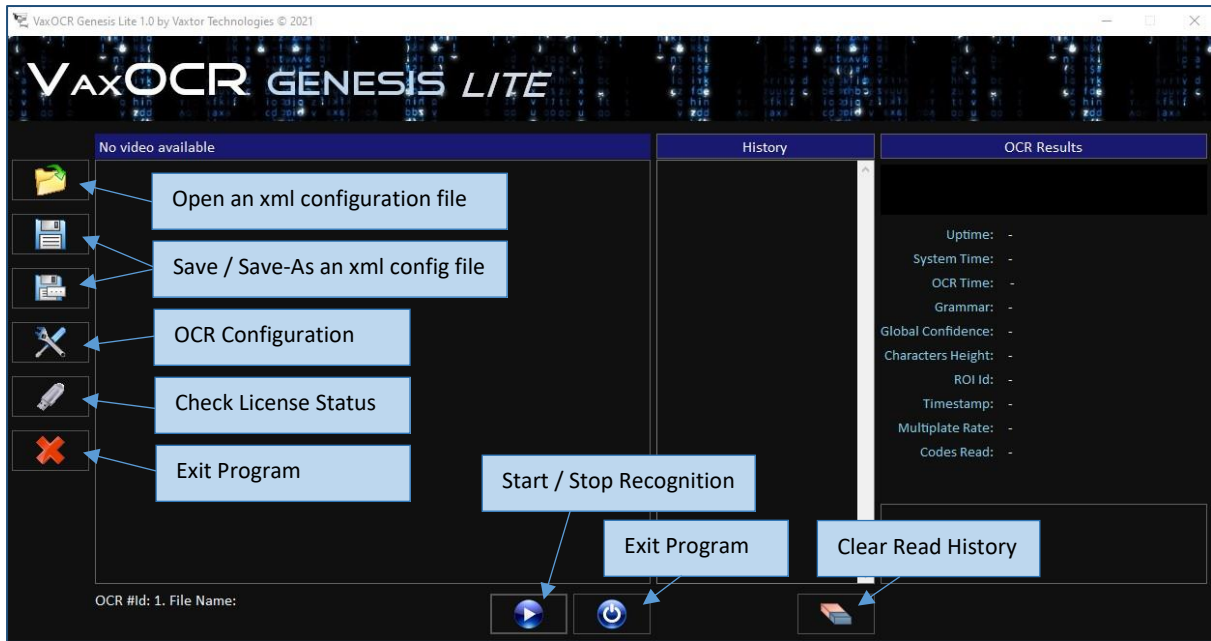
On launching Lite, the main screen is displayed.



VaxOCR Genesis Lite Main Screen

The Settings icons on the Left are where you configure all of the OCR parameters and add a camera (the Tools symbol). These settings can then be saved as an .xml file in the cfg directory using the Save or Save-As icons above. e.g. **C:\Program Files\Vaxtor Technologies\VaxOCR Genesis 1.0\cfg**

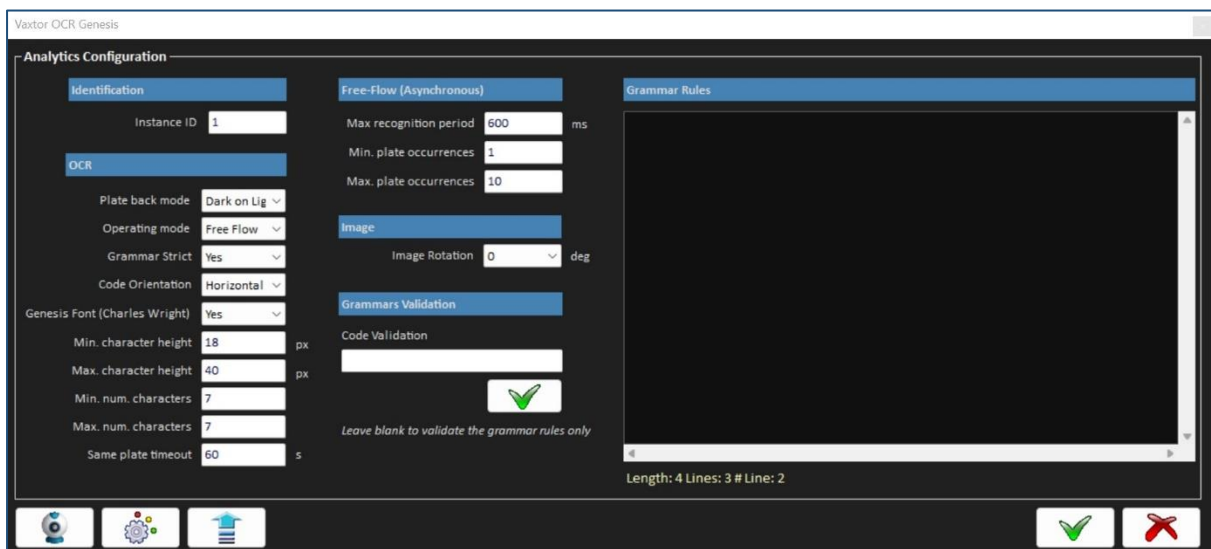
The Open icon re-loads this file for further editing.



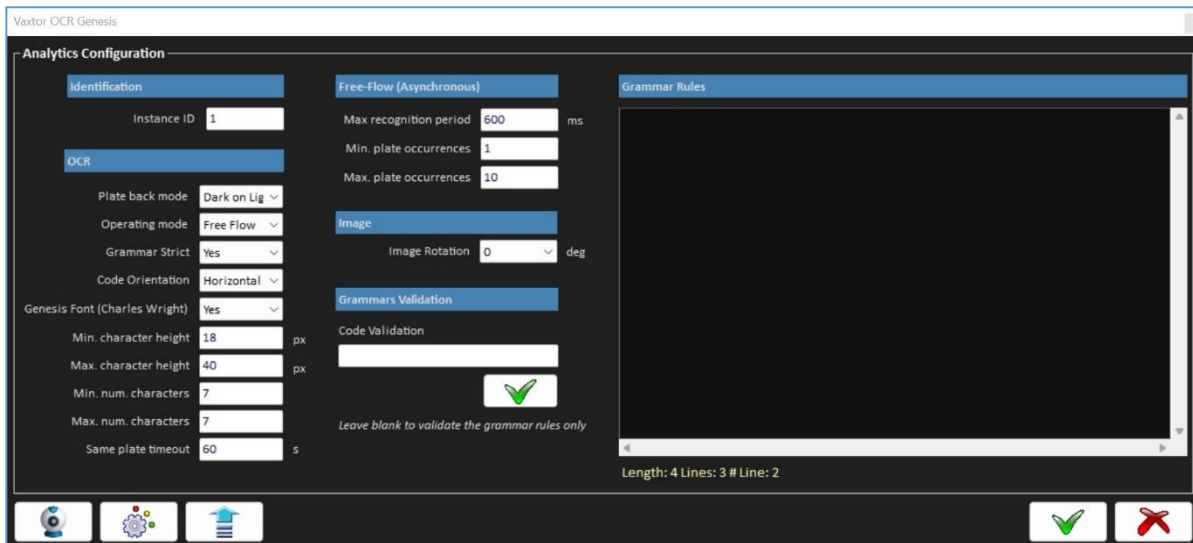
## 12. OCR Configuration Main Screen

The main configuration screen is where all the OCR parameters are setup.

At the bottom of the window the icons take you to different configuration screens: *Video Configuration, Regions of Interest and Results Publishing options.*



## 13. Configuring Genesis Parameters



### Instance ID (1 or above)

This is the ID (Identifying number) of the OCR camera and should be set to one or above. In a system with several cameras, then each time a camera is setup it should be given a unique ID. This ID is passed on the VaxOCR Viewer (Main program) to differentiate between cameras.

### Plate (text) background mode

This indicates the contrast between the text color to be read and its background (dark on light, light on dark, or both). Only use 'both' if required as this will increase the OCR time.

### Operating mode

Free-flow: the OCR software continuously searches for text without needing any external triggers.

Signaled ( or Triggered): the OCR requires an external trigger in order to read some text. This can be generated from say a beam being broken or a pressure pad for example.

When using signaled mode on a PC, the software listens to port 19000 by default (but this is configurable).

You should then open the socket connection, send a message (TCP/IP packet) with the reader ID you want to trigger and then close the socket.

When the socket is closed, the OCR takes the current frame from the video and scans it looking for a plate. If a plate is found then it is reported as normal and if no plate is present it reports 'NONE' as the plate.

Contact Vaxtor for more information on setting up a trigger.

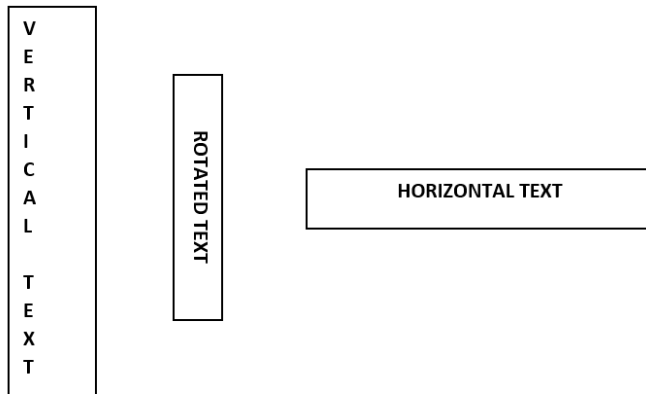
### Grammar Strict

Select this to force the Engine to only use the grammar defined by you. Text that does not match these rules will **not** be reported. This is the recommended option for using VaxOCR Genesis where exact matches are required.

Do not select this option when you want to report ALL text found in the field of view.  
See Grammar Rules later.

### Code Orientation

Genesis can be programmed to read Horizontal text, Vertical text or Both. (Default is horizontal).  
Note that reading a vertical code is not the same as a 90 degree rotated code.



### Genesis Font (Charles Wright) Y/N

Genesis is a font independent OCR engine but has been optimized for the Charles Wright font. If you are reading text or labels written in Charles Wright, then select that option here. (Using this option with non-Charles Wright text will lower the recognition rate)

Charles Write is the UK Licence Plate font and may be downloaded from: <https://www.k-type.com/fonts/charles-wright/> (A 5-off commercial licence is available for \$15.)

This is an easy-to read font Genesis has been optimized for this font (standard bold).

# Charles Wright ( UK Licence Plates)

## ABCDEFGHIJKLMNOPQRSTUVWXYZ

## 1234567890

### **Minimum character height (14-70 pixels)**

This is the minimum height that a code's characters should be before being read. If the lens (zoom) is setup correctly then the characters should be about 20-30 pixels high in the area of the field of view where they should be read. Small characters will cause misreads.

### **Maximum Character Height (14-70 pixels)**

Set the maximum height of the code's characters in pixels.

**NOTE:** The recommended difference between the min and max heights is about 10 pixels.

### **Minimum and Maximum number of characters**

Range (4-12): This is where you specify the minimum and maximum number of characters to scan for. So, if you are looking to read stock codes which could be say ABC1 up to ABC9999 – then you would specify 4-7 as the number of characters range.

### **Same plate timeout (seconds)**

Set the number of seconds that should elapse before reading the same code twice. This is to prevent multiple reporting of the same code in situations when the code or text is moving slowly or is stationary. For example, if a package stops on a conveyor belt and the code is reported but the package doesn't move for 40 seconds, then this delay should be set to say 60 seconds or more to prevent a duplicate read.

*NOTE: When using signaled (triggered) mode, it is recommended that you set this delay to 0 seconds.*

## **Free-Flow (Asynchronous mode) Settings**

### **Maximum recognition period (milliseconds)**

Set the number of milliseconds that the engine should spend analyzing a code or portion of text.

*(1000 milliseconds = 1 second)*

In free-flow mode the engine continuously analyses video frames and reads and reports codes. It makes a final decision on the code read after an interval of time - the Maximum recognition period. There is a dedicated time counter for every code read which starts counting after the first read. When it reaches the preset timeout it stops, checks the number of samples read of the same code and returns the "best" result.

*If an instantaneous code is not needed then set this timer to say 1500ms (1.5 seconds) so that the engine continues to look for the same code to read again for as long as possible. Note that if a new code is spotted during this time, the old one will be reported and a new code-trace started.*

We call the number of times the same code has been read within the multicode timeout period the **multicode rate**. Several reads of each code are good and produce better results.



This value is one of the metrics displayed in the main screen after each plate is read and is very useful when tuning your system. Several reads of each plate are good and produce better results.



*In this case the code BMG637 has been read 5 times before being reported*

### **Minimum plate (code) occurrences**

This is the minimum number of times the code should be read within the maximum recognition period before reporting. i.e. report the result only if the minimum number of occurrences has been met. E.g. if set to 2, then only report the code if two identical reads of the plate candidate have been made.

The default value is 1, but if the scenario is good and you are getting multiplate rates above 3 on average, - then by increasing this value to 2 may help reduce false positives.

*If you are getting high multiplate rate average then increase this minimum value.*

### **Maximum plate (code) occurrences**

This is the maximum number of times the code should be read within the maximum recognition period. If this value is reached before the maximum recognition period has elapsed, the OCR will force the result to be output.

Example:

If the OCR engine reads every 100ms (10 times per second. This depends on the resolution of your camera, the quality of image (low noise) and the speed of the PC), and say the maximum recognition period is set to 5000ms (5 seconds) and the maximum plate occurrences is set to 10, the result will be output after about 1 second because the maximum plate occurrences (10) was reached before the 5 second timeout.

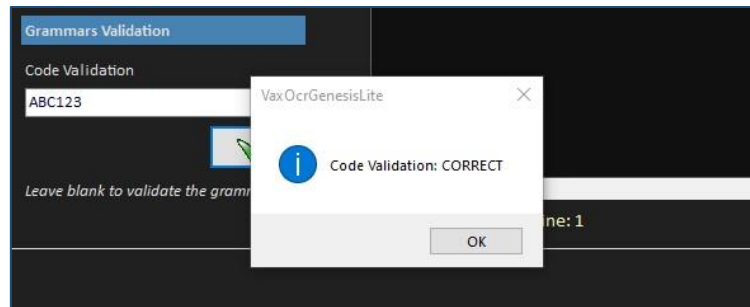
### **Image rotation (degrees)**

You may apply a specific 90° rotation to the input image. The default is 0 degrees. Other valid values are 90, 180 and 270 degrees. This can be used if a code or marking is written vertically on a label or object and needs to be read).

### **Grammars (Code) Validation**

See next section on setting the grammar rules.

Once these rules have been defined you can test them here by entering a code and pressing the green tick box next to it:



A dialogue box will appear confirming if the entered text matches the defined grammar or not.

## Grammar Rules

In this dialogue box you can define very complex grammar rules for reading multi-line text strings. (At present up to 3 lines). These rules can be set up to find particular strings (e.g. 17 alphanumeric or 5 vowels) or change certain characters (such as 1 to l).

For example, you might have a complex alphanumeric sequence (in this example 17 characters associated with a QR code):



In this case the code contains both letters and digits and so grammar rules can be used to specify for example that the code is always formed thus:

**DLDDLDDLDDDDDD** where D=Digit and L=Alphabetic (Letter)

So, the next to last character shown in the QR code image above should always be interpreted as a numeric zero and not an alphabetic O.

Firstly, you specify what characters are to be replaced followed by the grammar definition. Special characters are used to mark character positions for replacement and / or deletion.

### Simple example

If a character string is something like ABC123 – (so 3 letters followed by 3 numbers and you want to set a rule that if the first character is a 0 (zero) then replace with an O (oh) then the two lines of grammar required could be:

```
^0%L%L%D%D~^O%L%L%D%D  
%L%L%D%D
```

..where the first line specifies the replacement and the second the basic syntax or grammar i.e. the string you are looking for.

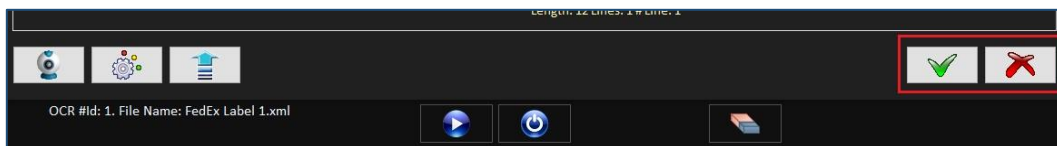
There are special characters to start at the beginning or end of a string and you can even define variables to represent a group of alphanumerics or words to be replaced.

A comprehensive guide to the grammar and its use can be found towards the back of this manual [here](#).

The guide contains several complex examples.

Enter the grammar rules and **definition** in the main dialogue box. There can be as many lines as you will need.

Once the OCR settings and Grammar have been defined, save the settings by pressing the green tick at the bottom right. Press the red cross to cancel your changes.



### OCR Engine Operation in Stages

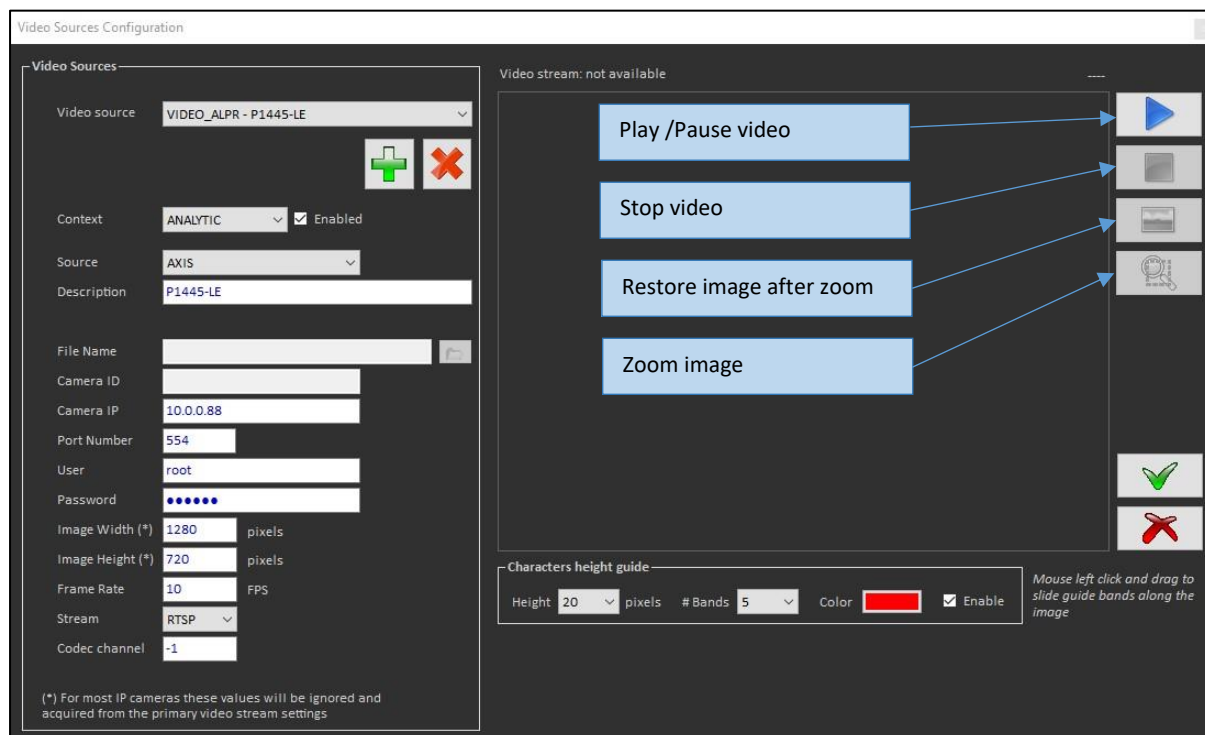
*Understanding this helps you to setup the OCR System correctly. The Engine works in several stages before finally reporting a code. The first stage is the Code finder which searches each image for code candidates before choosing one. The Code reading stage consist of de-skewing and rotating the text image. Skewed or rotated text is normalized (transformed into a standard size and rotation). This is followed by the actual OCR (Optical Character Recognition) of the plate and then syntax rules are applied to this code (so an 0 may be corrected to a O for example). The final stage is the repetition of this process (the code trace) as a vehicle moves through the field of view until a final code is output.*

## 14. Video Configuration



Select the camera icon to setup an image source:

This is where the video source used by the OCR and optionally the video contextual source (or environment camera) linked to the OCR camera are added and setup.



The left-hand side of the window is where the video source used by the OCR is added and setup.

### Video source

First use the green + icon to add a video source.

### Context

Normally select 'Analytic' here.

There are two types of video sources: Analytic and Environment.

Once an analytic camera has been set up to read text / codes, you can then set up an associated Environment camera – or a contextual / overview camera. The OCR will be performed on the Analytic camera and every time a code or piece of text is read, - an image is taken from the second environment camera and both images are saved or sent to a back office such as Helix.

In the case of say a label on an item in a warehouse being read by a camera on a fork lift truck, Genesis could read the stock number and a linked environment camera could capture the whole scene and identify where the item actually is.

## Source

The sources can be classified into different groups:

- IP based cameras.
- USB or non-IP cameras
- Video files
- Generic RTSP streams

On selecting the drop-down menu you will see a list of well-known cameras such as Axis, Sony etc. Select the appropriate camera if shown.

If the camera you are using is not in this list, then select **GENERIC** and enter the IP address of the camera's RTSP stream, port number, user name and password (if used) for your camera.

Each camera has its own RTSP format, examples are:

Hanwha: `rtsp://admin:PassworD!@192.168.16.59:554/profile1/media.smp`

Milesight: `rtsp://admin:PassworD!@192.168.16.59:554//main`

Hikvision (some models): `rtsp://user:pass@192.168.1.12:554`

Mobotix: `rtsp://<Camera IP>:<RTSP Port>/mobotix.h264`

Axis: `rtsp://root:pass@10.0.0.44/axis-media/media.amp?resolution=1920x1080&fps=10`  
etc.

(Select **VIDEO\_FILE** if you wish to process a recorded video clip and attempt to read the plates seen)

Depending on the type of video camera selected, some of the following text boxes will be enabled or disabled.

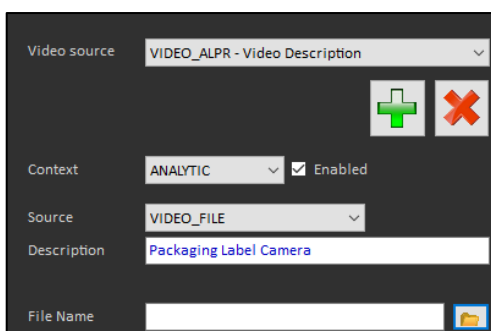
## Description

Enter a description for your camera. E.g. Conveyer Belt 20

## File name

If a recorded file of traffic is to be processed then locate the file here.

The video source can be a media clip. Note that if the clip won't play then you will probably need to install a standard codec pack such as K-Lite.



The screenshot shows a configuration window with the following fields and controls:

- Video source:** A dropdown menu currently showing "VIDEO\_ALPR - Video Description". To its right are two buttons: a green plus sign and a red minus sign.
- Context:** A dropdown menu showing "ANALYTIC" and a checked checkbox labeled "Enabled".
- Source:** A dropdown menu showing "VIDEO\_FILE".
- Description:** A text input field containing "Packaging Label Camera".
- File Name:** An empty text input field with a folder icon button to its right.

## Camera ID

Non-IP cameras by certain manufacturers such as GigE cameras from IDS or Basler

## Camera IP

When selecting an IP camera from video source you should set the camera's IP address

When using a GENERIC video source you should enter an RTSP or ONVIF connection string as follows:

### **RTSP, RTMP, HTTP, TCP, UDP, MSSH Protocols**

- [protocol]://[user:password]@[IP address or host name]/[URL params]
- Example: rtsp://UserName:Password@IPAddress:RtspPort/Streaming/channels/XXYY  
( see examples above)

### **ONVIF RTSP streams**

- The RTSP stream of the first Onvif media profile (default)
  - onvif://[onvifuser]:[onvifpassword]@[IP address or host name]:[onvif HTTP port]
  - Example: onvif://user:pass@192.168.2.55:8080
- RTSP stream selected by the index of the Onvif media profile
  - The index of the media profile must be in the 0..n-1 range
  - onvif://[onvifuser]:[onvifpassword]@[IP address or host name]:[onvif HTTP port]/[index of the onvif profile]
  - Example: onvif://user:pass@192.168.2.55:8080/1
- RTSP stream selected by the name of the Onvif media profile
  - This is the name of the media profile as it has been configured in the IP camera settings
  - onvif://[onvifuser]:[onvifpassword]@[IP address or host name]:[onvif HTTP port]/[name of the onvif media profile]
  - Example: supposing the name of the profile is "high quality" it would be onvif://user:pass@192.168.2.55:8080/high quality

### **Non-managed IP camera parameters**

Port number, user, password, image width, image width and height.

Most of the IP cameras will ignore these values and the settings from the camera's primary stream will be used. (When using most Axis cameras then the resolution CAN be changed here)

**Frame Rate** - The maximum frame rate to be used.

**IMPORTANT:** The camera will normally allow you to set up a target frame rate to output images. E.g. 10. If the Frame rate is set here to a lower value, then OCR engine will take this value as its maximum operational frame rate.

Example: If the camera is operating at 20 FPS and the OCR engine can operate at 5 FPS (200ms frame analysis period – depending on processor power) and we set this parameter to 3 FPS, the OCR engine will operate 3 FPS maximum.

This value represents the maximum frequency of the OCR analytics cycle.

## Stream

Can be either MJPEG or RTSP. Not all IP cameras will have MJPEG available so it is recommended to use RTSP by default.

## Codec channel

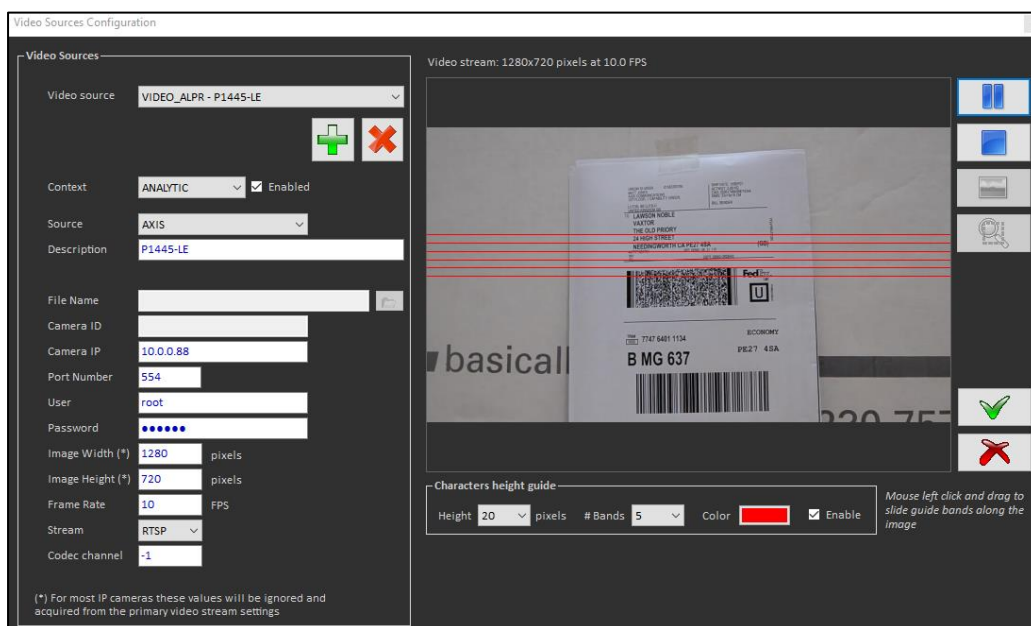
This Indicates the stream channel to acquire the video from. Most IP cameras have 2 or 3 streams which can be set to different resolutions. You might use one stream to send to a remote VMS system and another for the Container Code processing.

For most IP cameras this value can be set to 1, 2 or 3. (It actually adds these numbers to the RTSP parameters). The default value is set to -1 – which selects the default stream from the camera – i.e. stream 1.

Select the green tick to save your configuration so far and return to the main screen.

You can then use the 'Save As' icon on the left to save your configuration to the cfg directory using a meaningful name.

Press the Play Icon to test if your video stream is working correctly.



Once the video stream is visible you can use the red measurement bars to determine the height of the characters in pixels and if necessary adjust the camera lens to bring them within the min and max pixels heights that were set earlier.

## 15. Calculation of the average characters height

One of the key tasks during Genesis configuration is to estimate the average character height in pixels to set up a valid range for the OCR to work on. The correct range will help to achieve better performance and higher accuracy.

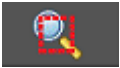
Sometimes it is hard to estimate this height just by watching the live video screen, furthermore, it's even harder to verify that the character height is optimal, - not too small, not too big.

The program fortunately contains a visual tool that will help the user to quickly and easily get a good estimate of the height range of the plates seen in the live video. The tool is essentially a tape measure.

In the "Characters height guide" we can configure the spacing of the calibration stripes:

- Height: The gap between any 2 lines, in pixels
- # Bands: The number of lines to display. (this just helps you estimate more accurately)
- Color: The color of the lines may be changed for clarity.
- Enable: Turn the calibration stripes on or off.

1. Click the triangular Play icon ( top right) to play the live video and press **PAUSE** when you have a label or text in the frame that you wish to calibrate for.
2. Move the mouse and hold the left button to slide the red horizontal lines up and down over the frame. The separation between any two stripes is equal to the height defined in the "Characters height guide" section at the bottom of the screen, in pixels.  
*(Here we have set the number on bands to 5 and the height between each band to 20)*

3. Use the Zoom icon  to define an area around the plate and zoom into that area.



4. We can now change the height (gap between the red bands) and/or the number of bands and then slide the bands up and down to accurately determine the height of each character. In this



case, after zooming in we can clearly see that height of the characters takes up nearly two gaps separation - which would be around 38 pixels high. Optimum recognition occurs at a character height of between 20 & 30 pixels. Less than 20 can affect recognition on small character details – but a bit more is fine. The only disadvantage of having larger characters is that it can take slightly more processing power to look for and analyze the plates.

If the characters are not within a reasonable range, then adjust the lens on the camera.

You can use the panoramic icon above the zoom icon to return to the default view.

When finished click on the green tick to save all the video settings and return to the Configuration screen.

After setting up and previewing the main OCR camera, if you wish to add an associated Environment camera then select ENVIRONMENT in the Context box and repeat the setup.

*Remember to tick 'Enabled' next to the Context box.*

## 16. Region of Interest Configuration



Select the cog icon to select the region of interest within the video frame.

The Region of Interest (ROI) is used to define an area within the video frame where the OCR analytics takes place. The user can define a polygon and choose whether the area to look for text inside or outside this region. The user can then set multiple regions, i.e. multiple ROIs, in complex situations although this is rare.

Using an ROI can decrease OCR processing time and also reduce false positives.

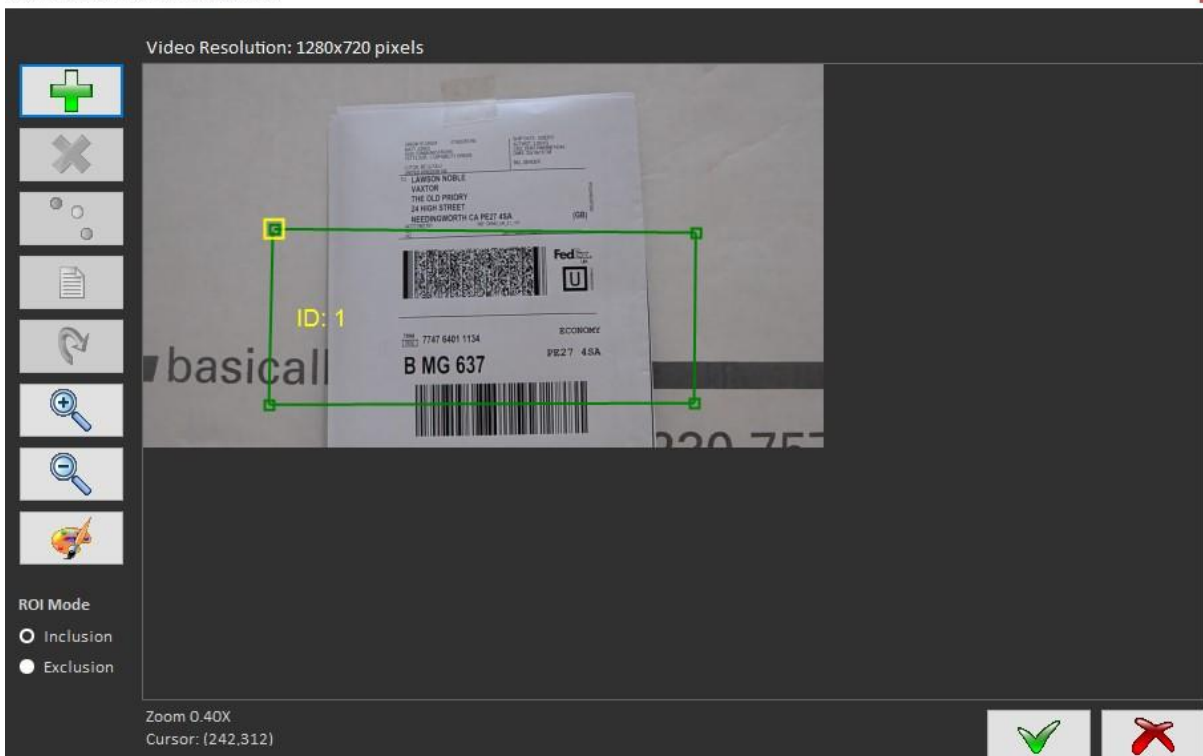
So, if the camera is looking at a complex label as in the example below, the ROI can be used to limit the OCR to the area near to the delivery code thus reducing the processor load.

If a similar piece of text is within the camera's field of view and keeps getting mistaken for the target code, then these false positives can be eliminated by creating an ROI to exclude this part of the image.

Each ROI must be given a unique numeric Identifier. E.g. '1'.

*Note that the whole target text field must be in or out the ROI to pass the test.*









Region of Interest (ROI) Configuration



The **ROI Mode** determines the behavior of the ROI, it can be set to either include or exclude the area around it. We cannot mix both types for logic reasons.

- Inclusion: The OCR will apply the analytics only inside the polygons defining the ROI
- Exclusion: The OCR will apply the analytics only outside the polygon defining the ROI

The icons to perform the ROI functions are:

	Add a new region of interest and save a region ID (numeric)		Remove all ROIs
	Remove an existing region of interest		Zoom In to edit the ROI
	Move ROI polygons points i.e. change the polygon shape		Zoom Out
	Edit the ROI ID number		Change the polygon color

The functions are self-explanatory.

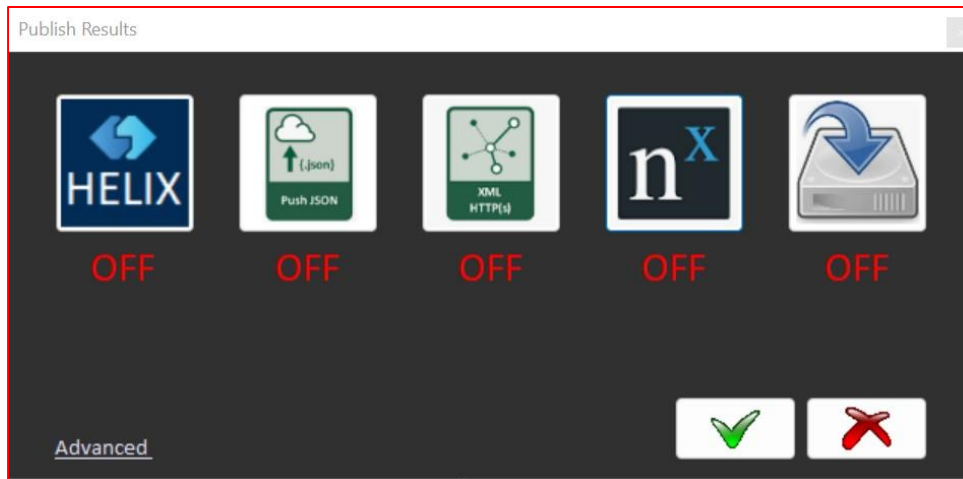
Once defined, click on the green tick to save the ROIs.

## 17. Results Publishing Configuration Screen



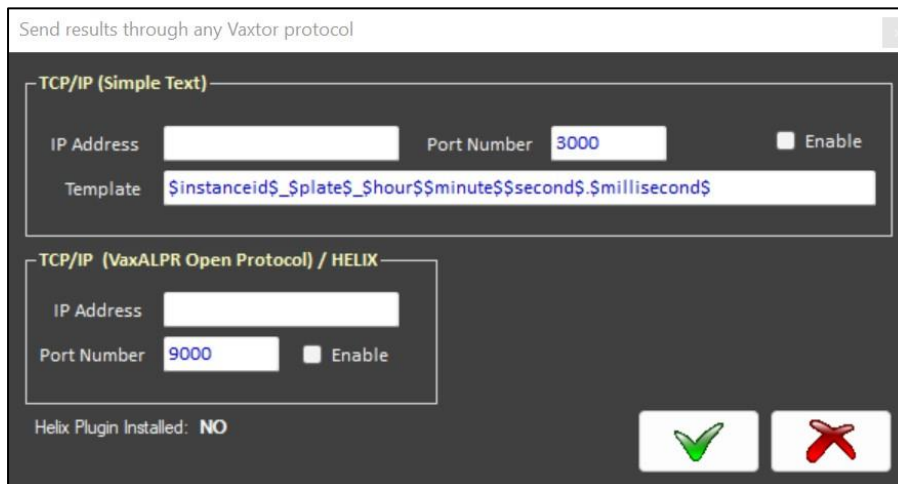
Clicking on this icon:  takes you to the Results Publishing Screen.

This menu allows the user to configure the software to publish the reads in real time, saving or sending the reads to other software or locations. This should be done for each camera setup.



### Results Publishing

#### Sending reads to Helix



Helix is Vaxtor's back office and can accept data from hundreds of cameras saving them to disk. Code reads can be searched and results displayed graphically using Grafana. Contact Vaxtor for more details of visit our website.

If Helix (the Vaxtor Back Office or BOF) is running on the same PC as VaxOCR Genesis then there is no need to configure any of these outputs. ( You can see if "Helix Plugin Installed" is shown). However, if Helix is running on another PC on your network then you should enable TCP/IP, enter the IP address of the PC where Helix is running and **use Port 9000**.

## Other Destinations

VaxALPR provides several mechanisms to allow third party integration and allows external programs to receive results in real time. Since this software is an application and not an SDK, VaxALPR can publish results as follows:

- **TCP/IP (Simple Text):** Each result is encoded using the Template Definition of which fields to send to the server **Port Number** as specified here.
- **HTTP-POST XML:** Each result is formatted in XML and sent to the specified URL.
- **HTTP(s)-POST JSON.** Each result is formatted in JSON and sent to the specified URL.
- **Send results to Optix NX:** Sends data to the Network Optix n<sup>x</sup> Witness VMS as Events & Bookmarks using http. (Note that the latest Optix NX enforces https)

*See the **OCR Results Integration** section later in this manual for more details on configuring these outputs and a list of reserved words for the plain TCP/IP output.*

## Saving plate read results to a directory

In addition to transmitting plates remotely you can enable '**Save results in directory**' to record all of the plate reads and associated images. Select a directory using the Folder icon. This can be useful if you want to perform some testing using VaxALPR Lite and verify the results later.

You can choose to save any of the following images:

The OCR image (the full image that was used to read the code)

The Plate crop – or OCR Patch. (This is the small extracted image of the actual text read.)

The Overview image

If an associated color contextual camera is used then this image is saved also

Finally, you can choose whether to write everything in to the target directory or alternatively create a daily folder which contains all of the selected data.

The **Templates Definition** section is where you can configure exactly what information is written to the .csv file and define the name of the images. In addition, you can configure what information is written into the .csv file such as Confidence, Plate Height, Country etc.

## CSV Content

The default file structure produces a record for our test label as follows:

```
1 10:30:27.039 07/01/2022 BMG637 99.67 33 40 6 GEN 1 B(100.00)
M(99.00) G(100.00) 6(100.00) 3(100.00) 7(99.00)
```

This includes the Instance ID, time and date, the text read, the global confidence, the text height, OCR time in milliseconds, the multicode rate and the confidence for the individual text characters.

However, you can use certain reserved words here to customise the data fields.

The available reserved words are:

<code>\$instanceid\$</code>	: ID configured (00001, 00002, ....) will pad with zeros up to 5 chars.
<code>\$hour\$</code>	: 2 digit hour (localtime).
<code>\$minute\$</code>	: 2 digit minutes (localtime).
<code>\$second\$</code>	: 2 digit seconds (localtime).
<code>\$millisecond\$</code>	: 3 digit milliseconds (localtime).
<code>\$day\$</code>	: 2 digit day
<code>\$month\$</code>	: 2 digit month
<code>\$year\$</code>	: 4 digit year
<code>\$plate\$</code>	: Plate number
<code>\$charheight\$</code>	: Average height in pixels of the characters in the plate number.
<code>\$left\$, \$top\$, \$right\$, \$bottom\$</code>	: Pixel coordinates of the top left corner and the bottom right corner of the license plate in the image.
<code>\$confidence\$</code>	: Confidence with 2 decimals, separator is a "." Always.
<code>\$processingtime\$</code>	: In milliseconds
<code>\$country\$</code>	: Country
<code>\$roiid\$</code>	: Region of Interest Id where the plate was detected (-1 if none).
<code>\$direction\$</code>	: 0, 1, 2: Unknown, Getting closer, Getting farther.
<code>\$imagefile\$</code>	: Full path to the OCR image (full frame).
<code>\$plateimagefile\$</code>	: Full path to the Plate image (crop).
<code>\$envimagefile\$</code>	: Full path to the Overview image.

### Image File Name

The default image file name is:

`$instanceid$_$plate$_$hour$$minute$$second$. $millisecond$_$type$.jpg`

Which produces files such as:

00001\_ABC123\_170816.949\_ocr\_img and  
00001\_ABC123\_170816.949\_ocr\_crop

### Advanced

Use this section to specify a port number and JPEG quality

#### Internal port number

Set a port number to allow VaxALPR intercommunication i.e. between each lane and the main VaxALPR Viewer.

#### JPEG quality (1-100)

Set the required compression ratio for the saved images. The lower the number, the higher the compression ratio but the quality of images will be lower. The default is 90 which produces vary acceptable images.

Save all the settings by clicking on the green tick, bottom right. This returns you to the main Configuration screen.

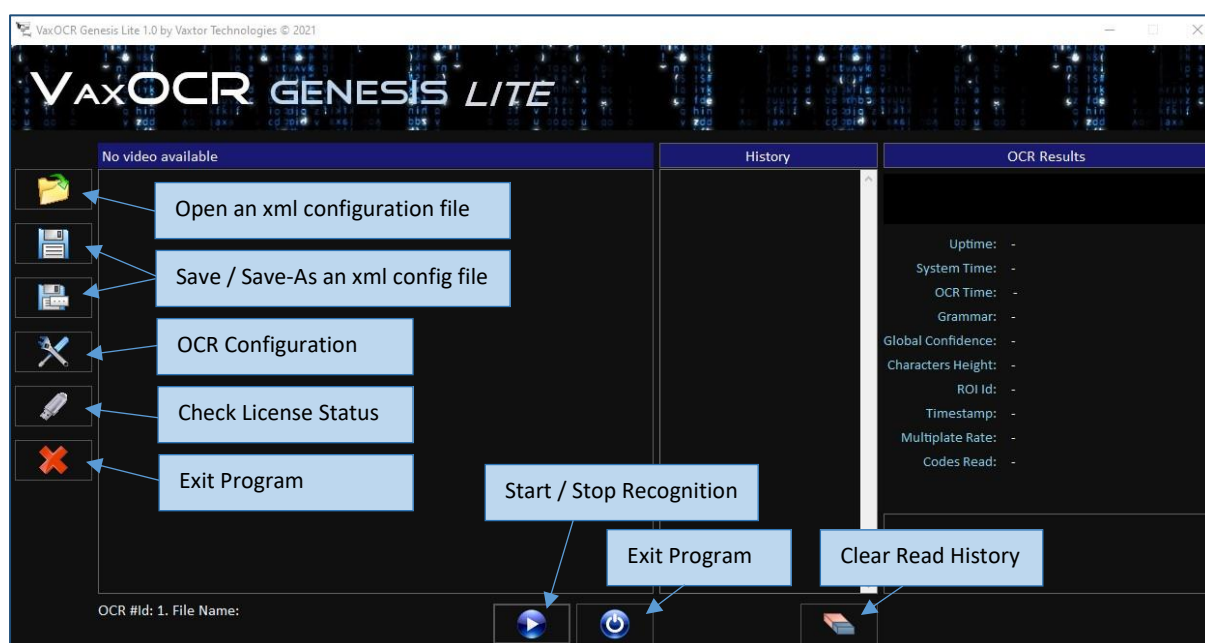
Once you have completed all configuration, press the green tick, bottom right to return to the main Genesis Lite screen.

## 18. Saving the VaxOCR Genesis Configuration

After setting all of the parameters for a camera, you must save them as an .xml file in the \cfg folder using the **Save-As** icon as shown below. Once saved, you can continue editing and re-save to the same file name using just the Save icon.

e.g. *C:\Program Files\Vaxtor Technologies\VaxOCR Genesis 1.0\cfg*


Choose an appropriate file name such as 'Camera 1' or 'Conveyor Belt 1'.



Repeat this configuration process for every camera you wish to add to your Genesis system.

Once all of the cameras are setup and their respective configuration files saved, exit the program and load the **VaxOCR Genesis** main program (the VaxOCR Viewer) to link all the cameras together to create our Genesis system by loading these files one by one.

See next section.

The open icon  will open any previously saved configuration files to edit or copy for a new camera. Once opened and edited these will now be re-saved as .xml files when the Save-As icons is used.





The Check License Status icon will display what Vaxtor On-PC products you currently have licensed:

ENTITY DESCRIPTION	TYPE	License
Helix-6 Base	Product	Perpetual
Helix-6 Fines Management	Plugin	Unavailable
Helix-6 Fines Exporter	Plugin	Unavailable
VaxALPR - OCR Engine	Product	Perpetual
ADR - Dangerous goods	Plugin	Perpetual
ALPR with 3 line plates	Plugin	Unavailable
VaxOCR Containers (ISO 6346)	Product	Perpetual
VaxOCR Railway Vehicles (UIC)	Product	Perpetual
VaxOCR USDOT	Product	Unavailable
iSpeed plugin	Plugin	Perpetual
MMC plugin	Plugin	Perpetual
Vehicle Class plugin	Plugin	Perpetual
VaxALPR Stop & Go	Plugin	Unavailable
VaxOCR Genesis	Product	Perpetual
VaxALPR RedLight	Product	Unavailable
Helix-6 Pro	Product	Unavailable

Press the Play icon to test your camera / OCR configuration:

The screenshot shows the VaxOCR Genesis Lite interface. The main video stream displays a FedEx shipping label with the text "B MG 637" and a barcode. The OCR Results panel on the right shows the following details:

OCR Results
<b>BMG637</b>
Uptime: <1 min
System Time: 07-01-2022 13:28:24
OCR Time: 55.9 (ms)
Grammar: YES
Global Confidence: 99.5 %
Characters Height: 33.7 (px)
ROI Id: 1
Timestamp: 13:28:24
Multiplate Rate: 5
Codes Read: 1

The History panel shows a list of reads: 13:27:51 BMG637 and 13:28:24 BMG637. The Play icon (a blue square with a white play symbol) is highlighted with a red box at the bottom of the interface.

Results will appear in the History panel. Click on each read to view the read details (OCR Results panel). You will see all relevant technical metadata here such as the OCR time taken to read the code etc.

OCR Results

**BMG637**

Uptime: 3 min  
System Time: 07-01-2022 13:31:25  
OCR Time: 74.7 (ms)  
Grammar: YES  
Global Confidence: 99.7 %  
Characters Height: 33.7 (px)  
ROI Id: 1  
Timestamp: 13:31:25  
Multiplate Rate: 5  
Codes Read: 8

**B MG 637**



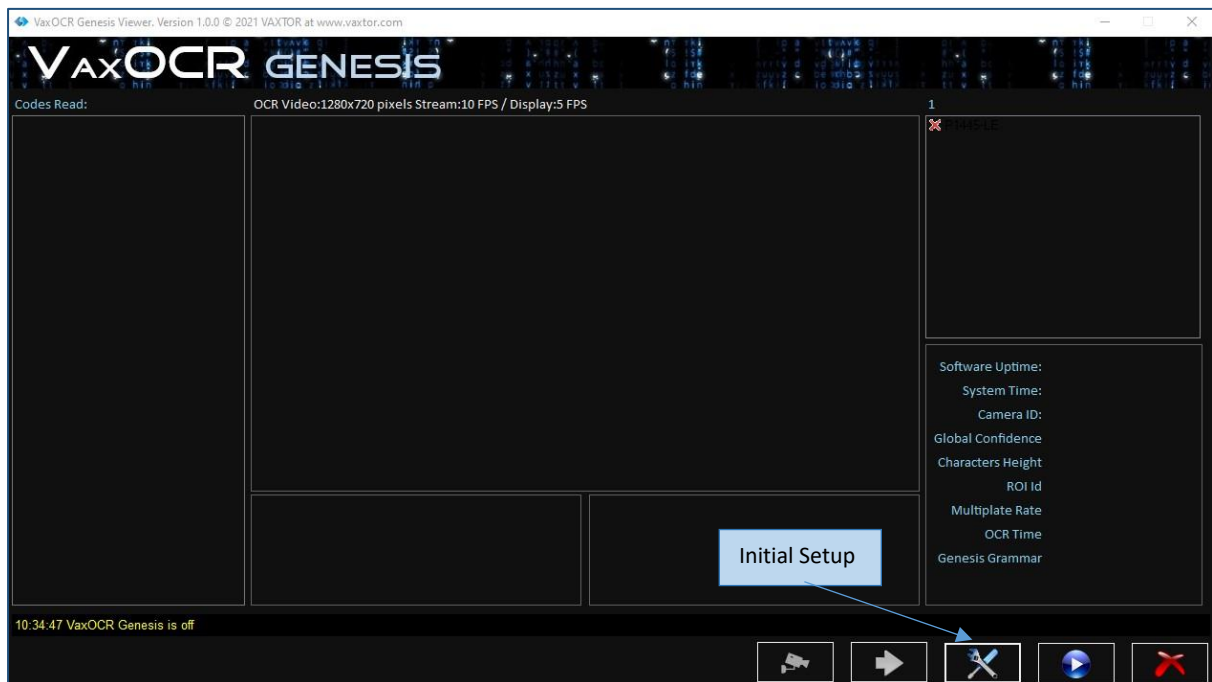
You can clear these results using the eraser icon to continue testing.



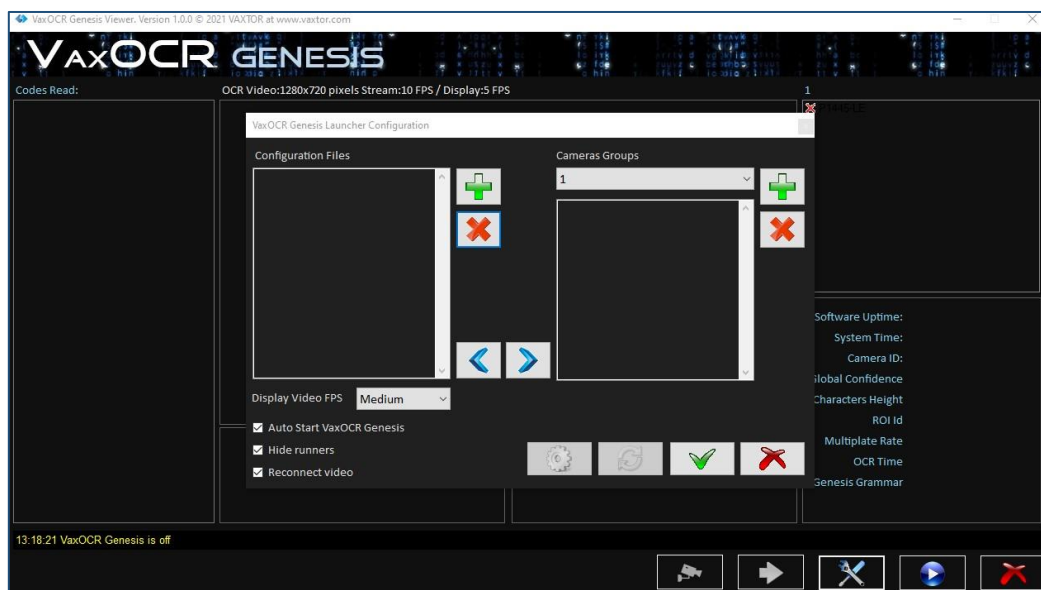
Once all settings have been finalized exit the program using the blue button icon:

## 19. VaxOCR Genesis Viewer

VaxOCR Genesis, also known as the VaxOCR Viewer, is the software that brings together and runs one or more OCR cameras that have been configured by VaxOCR Setup. The application operates unattended and requires no user interaction other than the initial setup which is basically where you select which OCR files to run.

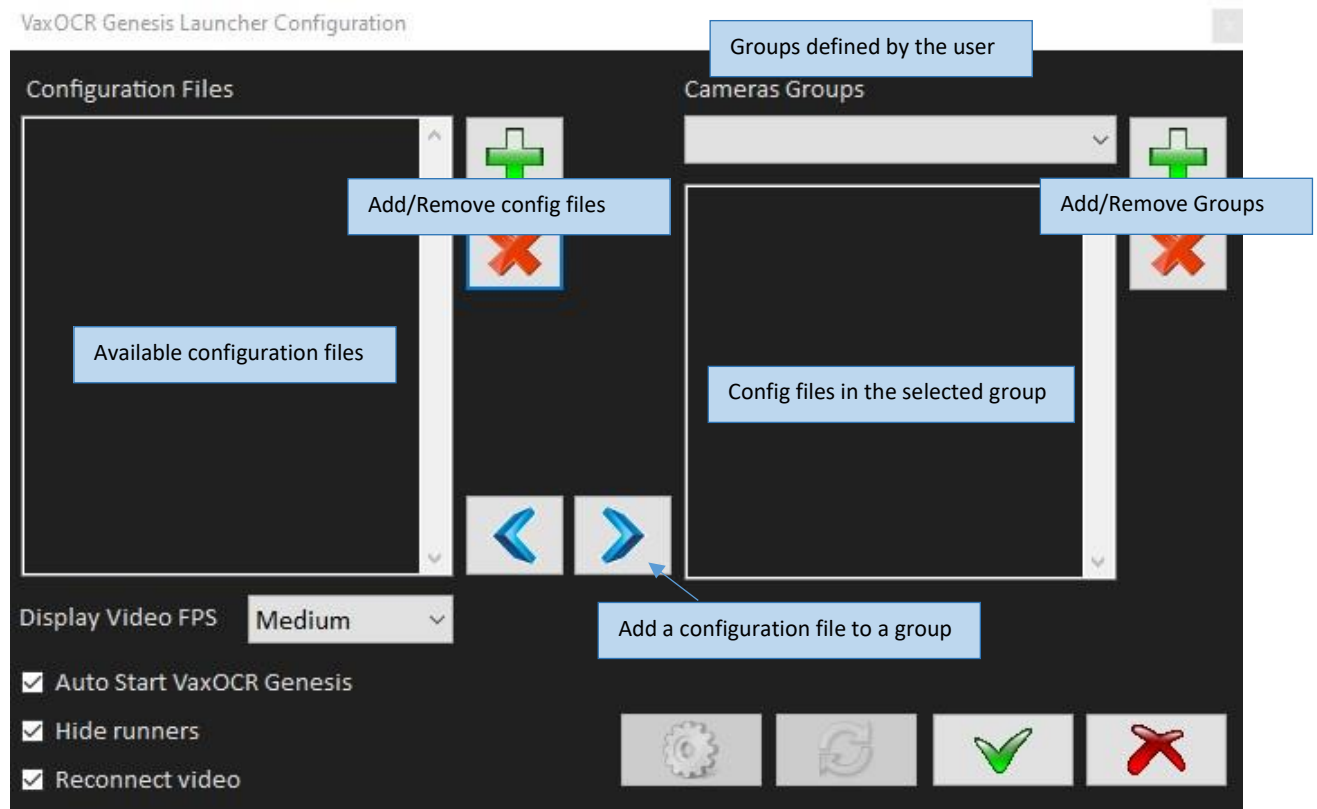


The Setup icon takes you to the following screen:



## 19.1 Initial setup

Initial setup takes place just once and here we select the one or more VaxOCR Configuration files (.xml) configured by VaxOCR Genesis Lite that will become our Genesis OCR system.



### VaxOCR Genesis Configuration files

This is the location where all OCR configuration files defined by VaxOCR Genesis Lite are located, which corresponds with the “cfg” folder inside VaxOCR Genesis installation directory.

*Note that the fact of being listed here does not mean that they are going to be processed by VaxOCR Viewer, the user must choose which files should be included in the process manually.*

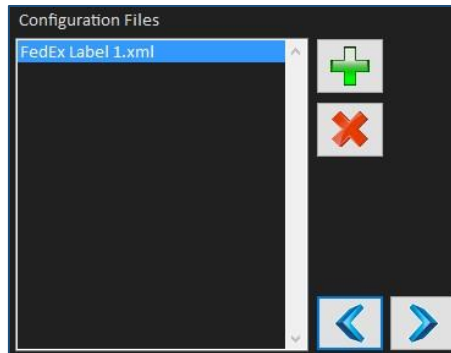
### Camera Groups

The way VaxOCR Viewer manages the cameras is by groups. A group is a list of Configuration files and only one group can run at a time so only the configuration files of the group run simultaneously. Thus, VaxOCR Viewer manages a list of groups each one containing a list of VaxOCR Configuration files.

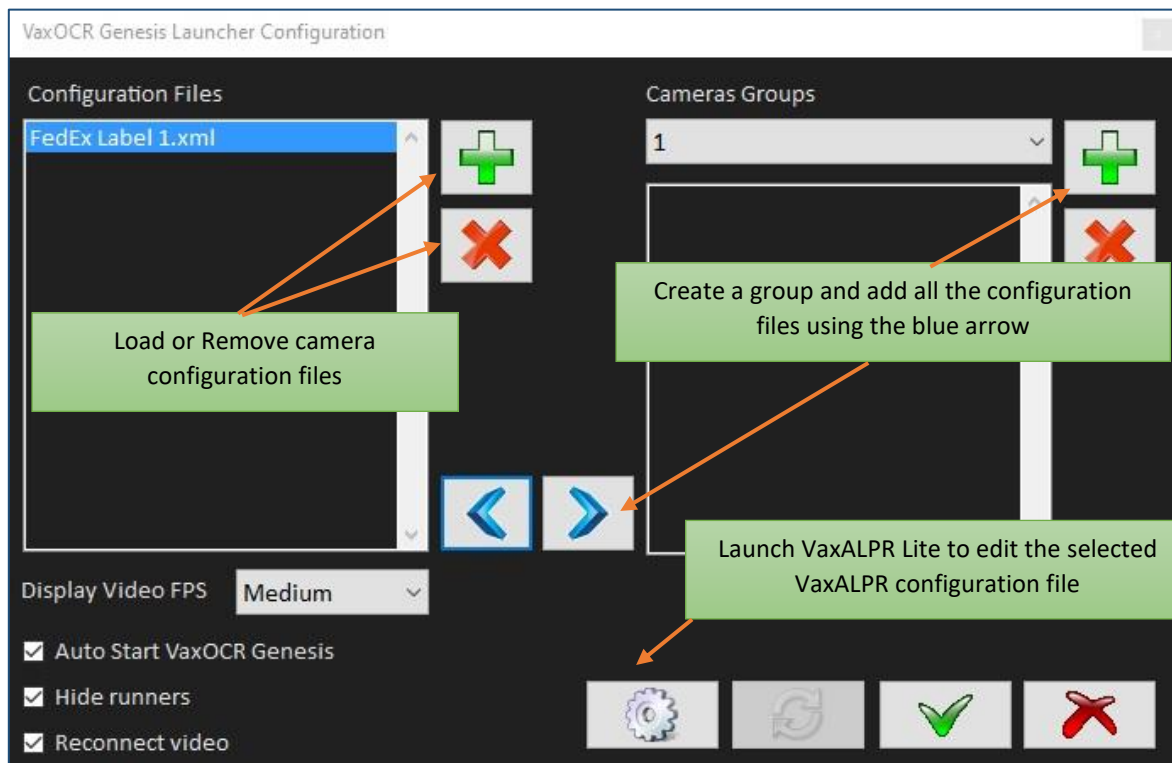
Normally, we will have only one group containing all of our VaxOCR Configuration files. However, the group structure is useful in some scenarios where sometimes you may want to easily switch between camera groups.

## How it works step by step

Use the **Plus** Icon on the left to load the configuration files previously created by VaxALPR Lite located in the “cfg” folder in the VaxALPR installation directory. The red Cross Icon removes them. Repeat for all the cameras you want to integrate.



If you do not need to create more than one group you can press the button to create a default group and automatically add all your files, otherwise you'll have to manually create the groups and add the files using the blue arrow buttons.



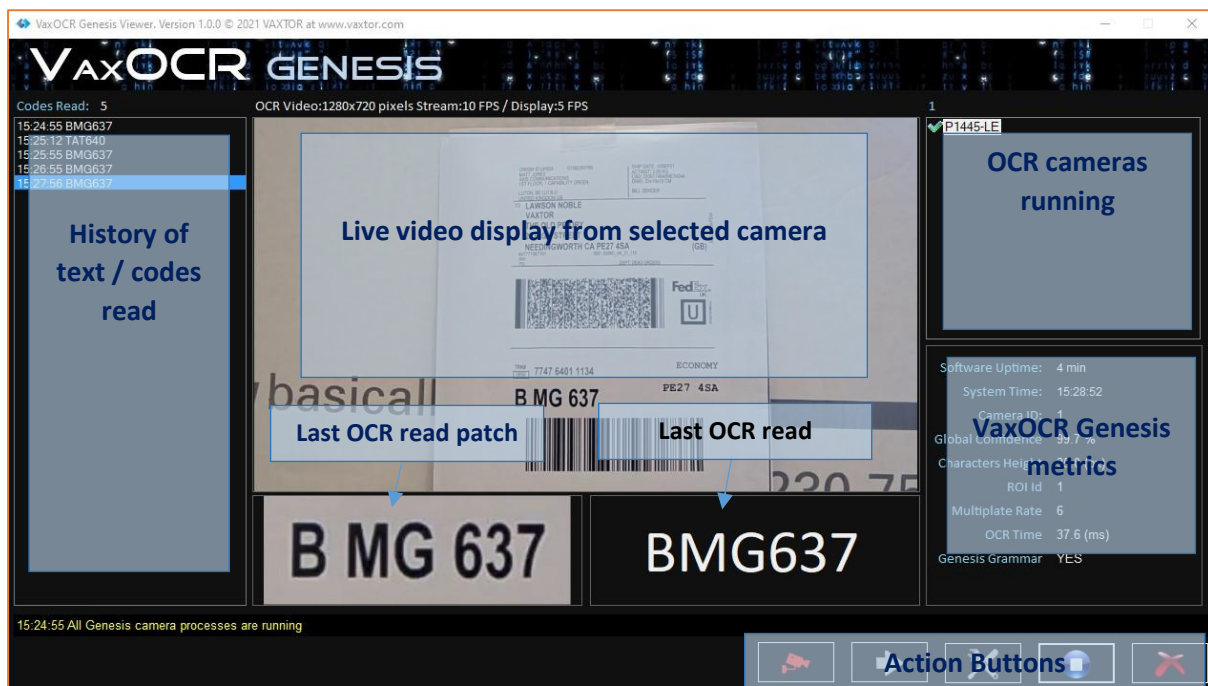
All groups must contain at least one VaxALPR Configuration File and all files loaded (in the left window) must be allocated to a group before exiting. If some are not needed, remove them with the red cross icon, – they will not be deleted.

## Other options

- **Display Video FPS:** This sets the frame rate at which to display the live video on the PC screen. This feature does not affect the OCR operating frame rate, it is more of a visual or cosmetic feature. Refreshing video frames consumes processor resources and so we recommend setting this to Low.
- **Auto Start VaxOCR Genesis:** The software will automatically load your configuration at start up and will automatically launch the OCR processes. Note that this option does not launch VaxOCR Genesis after rebooting the computer, it just restarts the selected VaxOCR Configuration files after starting the application. *To auto start VaxOCR Genesis, add the program to your Windows Startup folder.*
- **Hide runners:** Show or hide the console windows in the task bar running the OCR processes.
- **Reconnect video:** Activate or deactivate automatic video reconnection in case of losing the camera connectivity. If this option is disabled the OCR will stop running if connectivity is lost. If enabled then once the connectivity is recovered, VaxOCR will continue running automatically. This option is normally enabled unless the system is configured to process video files (instead of live cameras) as we don't normally want the program to reload the media clip after finishing.

## 20. Main Screen

After setting up the ALPR files groups we're ready to launch VaxOCR Genesis from the main screen.



### Main Screen Components

- **OCR Cameras running:** This window displays the list of the connected cameras in the active group, the name corresponds with the camera description we setup in VaxOCR Genesis Lite.
- **Live Video Display:** This window shows the live video stream from either the OCR or associated environment camera ( if present) selected from the cameras list on the right. You can see the display frame rate from the VaxOCR Genesis Lite configuration screen described in the previous sections.  
Use the camera icon at the bottom to switch between OCR and Environment cameras (if available).
- **History of Codes Read:** This shows the 100 most recent reads and timestamps read by the currently selected camera.
- **Last OCR read Details:** This is the OCR result of the plate that has just been read.
- **Last OCR read Patch:** This is the cropped text from the image - the code Patch.
- **VaxOCR Genesis Metrics:** The program displays various metrics from the OCR Engine from the last code read. These include data such as the confidence of the last read and the average height of the characters read.
- **Action Buttons:** The main Action Buttons along the bottom of the display to the following:

## 21. Action Buttons



This toggles between the main OCR camera and an associated environment camera if present.



This stops the running OCR camera group and switches to next one (if multiple groups have been setup)



VaxOCR Genesis Viewer Configuration. The cameras should be stopped first.



Start/Stop VaxOCR Genesis. This stops the OCR process so that you can edit parameters.



Exit the program.

## 22. Running the Program



Press the PLAY button  to start / stop the recognition.

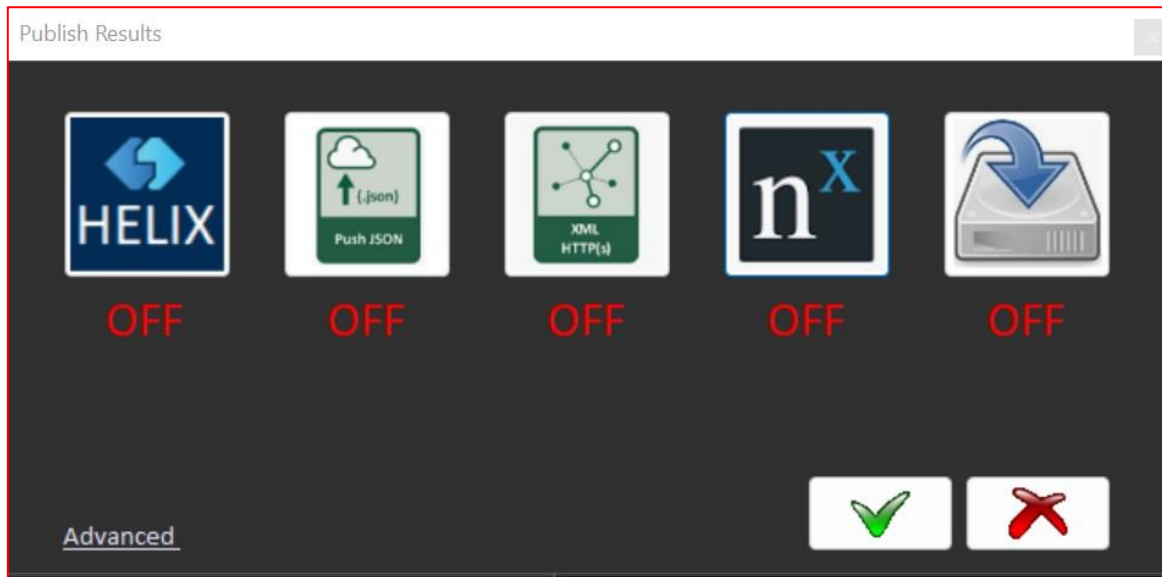
All code reads will be shown on the main screen along with the live video. (History)

The most recent read is displayed underneath the live view.

The right-hand side of the display will show data from each read.



## 23. More on VaxOCR Genesis Results Integration



VaxOCR Genesis can publish real time results using TCP/IP Sockets and the software publishes a result every time a code is read. Some of the information below is common to our ALPR system and so is named accordingly.

### 23.1 TCP/IP sockets

Enter the IP address of the host machine receiving the results and enter the port number. You will have to “listen” to the same port number.

**IMPORTANT:** Multiple OCR instances (cameras) can publish to the same port number. The message (results) includes the camera ID to identify the message sender (or OCR source). i.e. the camera.

From a software development perspective, you’ll have open a “listener” on the port number and wait for results. VaxOCR Genesis opens a connection for every new message, transmits the information and closes the connection.

#### .NET integration

In case of using a .NET platform Vaxtor provides a Class Library (Assembly) named **VaxAlprRemotePlateParser.dll** that can do this job for you and it is very easy to use.

This is automatically installed into your VaxALPR main directory:

Uninstall	06/01/2022 10:24	Application	112 KB
VaxAlprManager.dll	05/01/2022 14:22	Application exten...	56 KB
VaxAlprRemotePlateParser.dll	05/01/2022 14:22	Application exten...	31 KB

#### Visual Basic .NET example

1. Declare the main class of the class library

## Imports VaxOCR Genesis RemotePlateParser

```
Private _server As AlprResultsServer = Nothing
```

2. Instance the main class and provide 2 callbacks to receive plate results and any possible errors from the library. Next start the server and provide the same port number defined in VaxOCR Genesis configuration.

```
_server = New AlprResultsServer(AddressOf Me.OnNewPlate, AddressOf Me.OnAlprServerError)  
_server.Start(port_number)
```

3. Shutdown the server when you have finished

```
_server.Shutdown()
```

### Functions callback definition

```
Public Sub OnNewPlate(ByVal alpr_plate As AlprPlateResult)
```

' Here you can get all ALPR results contined in 'alpr\_plate' instance.

' AlprPlateResult is defined and available in VaxOCR Genesis RemotePlatesParser.dll

```
End Sub
```

```
Public Sub OnAlprServerError(ByVal err_msg As String)
```

MsgBox(err\_msg, MsgBoxStyle.Exclamation)

```
End Sub
```

### Raw integration and other platforms

In case of not using .NET you will have to parse the results yourself. This is the format of the data you'll get from the socket every time VaxALPR publishes a result:

- 4 bytes, int32 => 0xCAFEBABE is the start message header identifier (fixed)
- 4 bytes, int32 => 0xBABE10 result message (fixed)
- 4 bytes, int32 => VaxALPR instance ID, matches the tag InstanceID from VaxALPR configuration file
- 8 bytes, int64 => Time Stamp, current date in milliseconds
- 4 bytes, int32 => NC1:number of ASCII characters of the plate number
- NC1 bytes, byte => ASCII bytes array
- 4 bytes, int32 => NC2:number of ASCII characters of the plate origin
- NC2 bytes, byte => ASCII bytes array
- 4 bytes, int32 => ROI ID containing the license plate (1...n). 0 value means plate out of any ROI
- 8 bytes, double => Global confidence of plate recognition: ]0...100]
- 8 bytes, double => Characters height, in pixels
- NC1 bytes, float => Per character recognition confidence ]0...100]
- 8 bytes, double => OCR processing time, in milliseconds
- 4 bytes, int32 => 'x0' coordinate of the plate left-top corner, in pixels
- 4 bytes, int32 => 'y0' coordinate of the plate left-top corner, in pixels
- 4 bytes, int32 => 'x1' coordinate of the plate right-bottom corner, in pixels
- 4 bytes, int32 => 'y1' coordinate of the plate right-bottom corner, in pixels
- 8 bytes, int64 => OCR image timestamp, in milliseconds

- 4 bytes, int32 => SF: size of the OCR image, in bytes
- SF bytes, byte => OCR image data in JPEG format
- 4 bytes, int32 => R1: size of image reserved data, in bytes
- R1 bytes, byte => reserved data, R1 bytes
- 8 bytes, int64 => Environment image timestamp, in milliseconds
- 4 bytes, int32 => SF: size of the environment image, in bytes
- SF bytes, byte => Environment image data in JPEG format
- 4 bytes, int32 => R2: size of image reserved data, in bytes
- R1 bytes, byte => reserved data, R2 bytes
- 4 bytes, int32 => vehicle direction, if available: 0=>unknown, 1=>closer, 2=>farther
- 4 bytes, int32 => ALPR multiplate rate
- 4 bytes, int32 => 1: plate reported under SYNCHRO\_SINGALED mode, 0: SYNCHRO\_FREE\_FLOW mode
- **4 bytes, int32 => 0x42F83988 Header indicating optional data follows up**
- -----
- **4 bytes, int32 => 0x50000001 Plate number in wide string format section**
- 4 bytes, int32 => Number of plate characters
- 4 bytes, int32 => NC3: Number of bytes taking the full plate string
- NC3 bytes, byte => Array of bytes representing the string either in UTF8 or UNICODE format
- **4 bytes, int32 => 0x50000002 Hot-List section (\*)**
- 4 bytes, int32 => 1:white list. 2: black list
- 4 bytes, int32 => NC4:number of ASCII characters in the message
- NC4 bytes, byte => ASCII bytes array containing the message
- **4 bytes, int32 => 0x50000003 String code section (\*\*)**
- 4 bytes, int32 => String code = 1:plate country region, 2: vehicle color, 3: vehicle make, 4: vehicle model
- 4 bytes, int32 => NC5: number of ASCII characters of the string
- NC5 bytes, byte => ASCII bytes array containing the characters
- **4 bytes, int32 => 0x50000004 GPS section**
- 8 bytes, double => Latitude in decimal format
- 4 bytes, int32 => 1:N, 2:S
- 8 bytes, double => Longitude in decimal format
- 4 bytes, int32 => 1:W, 2:E
- -----
- 4 bytes, int32 => 0x42F87D89 is the end message header identifier (fixed)

(\*) The hot list block can appear twice, one per list.

(\*\*) The string code block can appear more than once, but each code only can appear once.

## 23.2 TCP/IP sockets – sending plain results

Enter the IP address of the host machine receiving the results and enter the port number as above.

In addition, you can specify a template by using certain reserved words customize the data fields. The available reserved words are:

<code>\$instanceid\$</code>	: ID configured (00001, 00002, ....) will pad with zeros up to 5 chars.
<code>\$hour\$</code>	: 2 digit hour (localtime).
<code>\$minute\$</code>	: 2 digit minutes (localtime).
<code>\$second\$</code>	: 2 digit seconds (localtime).
<code>\$millisecond\$</code>	: 3 digit milliseconds (localtime).
<code>\$day\$</code>	: 2 digit day
<code>\$month\$</code>	: 2 digit month
<code>\$year\$</code>	: 4 digit year
<code>\$plate\$</code>	: Plate number
<code>\$confidence\$</code>	: Confidence with 2 decimals, separator is a “.” Always.
<code>\$processingtime\$</code>	: In milliseconds
<code>\$country\$</code>	: Country
<code>\$roid\$</code>	: Region of Interest Id where the plate was detected (-1 if none).
<code>\$direction\$</code>	: 0, 1, 2: Unknown, Getting closer, Getting farther.
<code>\$imagefile\$</code>	: Full path to the OCR image (full frame).
<code>\$plateimagefile\$</code>	: Full path to the Plate image (crop).
<code>\$envimagefile\$</code>	: Full path to the Overview image.

## 23.3 HTTP-POST XML or JSON

### XML Connector:

VaxALPR will send a HTTP POST to the webserver configured each time that a plate is read. The message Content-type will be “application/xml” and the message body will be the XML object defined in the file “**event.xml**” that you can find in VaxALPR installation directory. This file can be edited by the user to define the desired XML structure.

### JSON connector:

VaxALPR will send a HTTP POST or HTTPS POST to the webserver configured each time that a plate is read. The message Content-type will be “application/json” and the message body will be the JSON object defined in the file “**event.json**” in the VaxALPR installation directory. This file can be edited by the user to define the desired JSON object. The available reserved words are:

### Reserved words you can play with in your ‘event’ configuration file

`$timestamp$`: Date and time of the read in local time. The string is formatted according to ISO8601 standard (yyyy-MM-ddTHH:mm:sszzz). In example: 2007-11-23T13:18:05-03:00

`$plate$`: Plate number

`$country$`: Country of the vehicle using ISO ALPHA-3 Code.

`$cameravid$`: Unique id for the reader. This id is setup by the user on the configuration.

`$confidence$`: Global confidence of the plate (0-100).

**\$processingtime\$**: Milliseconds that the OCR took to analyze this plate number.

**\$charheight\$**: Average height in pixels of the characters in the plate number.

**\$left\$, \$top\$, \$right\$, \$bottom\$**: Pixel coordinates of the top left corner and the bottom right corner of the license plate in the image.

**\$absoluteleft\$, \$absolutetop\$, \$absoluteright\$, \$absolutebottom\$**: Absolute coordinates (0 .. 1) of the top left corner and the bottom right corner of the license plate in the image.

**\$width\$**: Width of the analyzed image.

**\$height\$**: Height of the analyzed image.

**\$region\$**: Plate region. Countries like UAE or USA will report here the internal region (Dubai, Bahrein, New York ...)

**\$category\$**: Plate category. Available for Kuwait.

**\$make\$**: Vehicle maker.

**\$model\$**: Vehicle model.

**\$vehiclecolor\$**: Vehicle color.

**\$direction\$**: Vehicle direction (UNKNOWN, GETTING\_CLOSER, GETTING\_FARTHER)

**\$latitude\$, \$longitude\$**: Coordinate of the read formatted in decimal degrees.

**\$image\$**: JPEG image encoded in base64 format.

**\$sizeinbytes\$**: Size in bytes of the JPEG image once decoded.

Note that there is an example JSON file available in the program's installation directory:

**event.json:**

```
{
  "plate": "$platenumber$",
  "date": "$timestamp$",
  "country": "$country$",
  "confidence": $confidence$,
  "left": $left$,
  "top": $top$,
  "right": $right$,
  "bottom": $bottom$,
  "charheight": $charheight$,
  "processingtime": $processingtime$
}
```

Enter the Server URL and define a User Name and password if needed.

## 23.4 Send Results to the Optix NX VMS System

See separate guide “VaxALPR on Camera - Network Optix & NX VMS Integration”

Send results to Optix NX						
<input type="checkbox"/> Enable	Camera ID	<input type="text"/>	Source	<input type="text" value="LPR"/>		
	URL	<input type="text" value="http://192.168.0.100:7001"/>	Caption	<input type="text" value="\$plate\$"/>		
	User	<input type="text" value="admin"/>	Pass	<input type="text" value="*****"/>	Description	<input type="text" value="\$plate\$"/>

1. Once enabled, enter the URL of the NX host server using port 7001: <http://nxserver:7001>  
e.g. <http://192.168.0.41:7001> - Note that the latest Optix NX enforces https.  
(ensure that the IP addresses of the PC and NX server are on the same subnet)
2. Enter your Username & Password.
3. Enter a “Source” parameter. This is used by the NX Server to reference the events received.  
E.g. ‘LPR’. This will mark all NX events as “LPR” events.
4. Enter the Caption and Description templates. This data will be saved with each event.  
Modify the two messages as required. The message can use Dynamic text replacement.  
See below for a list of words.

You can add your own parameters into the message, so if you want to add say a site ID, your message might look like this:

```
{ "plate": "$plate$", "date": "$date$", "ip": "$ip$", "country": "$country$", "sitecode": 12345 }
```

5. Enter the Camera ID. This is the ID that has been setup in the NX VMS settings. This can be found in Camera Settings.

Results may also be sent to other popular VMS systems such as Milestone, Pelco & March Networks.  
Please contact Vaxtor for more information.

## 24. Words Available for Dynamic Text Replacement

- **\$image\$**: Full JPEG image encoded in base64.
- **\$jpgsize\$**: JPEG size in bytes.
- **\$date\$**: Timestamp in ISO8601 format
- **\$plate\$**: Plate number
- **\$tag\$**: Unique hash for this plate number. Same plate number will always give the same \$tag\$. Format based on UTMC algorithm.
- **\$plateutf8\$**: Plate number in utf8 format.
- **\$country\$**: Full country of origin name.
- **\$countrycode\$**: 3 letter country code.
- **\$state\$**: Plate State for USA.
- **\$category\$**: Plate category for countries that support it.
- **\$blacklist\$**: Description on the blacklist linked to the plate number.
- **\$whitelist\$**: Description on the whitelist linked to the plate number.
- **\$ifblacklist\$ .... \$ifblacklist\$**: If the plate is on the blacklist, the text in the 'if clause' will be displayed.
- **\$ifwhitelist\$ .... \$ifwhitelist\$**: If the plate is on the whitelist, the text in the 'if clause' will be displayed.
- **\$ifnotin\$...\$ifnotin\$**: If the plate is not on a list, the text in the 'if clause' will be displayed.
- **\$confidence\$**: Global confidence (0-100).
- **\$charheight\$**: Average charheight (pixels).
- **\$processingtime\$**: Processing time in milliseconds.
- **\$left\$**: Left coordinate for the plate on the image (pixels).
- **\$top\$**: Top coordinate for the plate on the image (pixels).
- **\$right\$**: Right coordinate for the plate on the image (pixels).
- **\$bottom\$**: Bottom coordinate for the plate on the image (pixels).
- **\$absoluteleft\$**: Plate left position based on the total image width (0-1).
- **\$absolutetop\$**: Plate top position based on the total image height (0-1).
- **\$absoluteright\$**: Plate right position based on the total image width (0-1).
- **\$absolutebottom\$**: Plate bottom position based on the total image height (0-1).
- **\$width\$**: OCR image width.
- **\$height\$**: OCR image height.
- **\$ip\$**: Camera IP address.
- **\$roid\$**: Roi ID where the plate number is found.
- **\$speed\$**: Vehicle speed (Km/h).
- **\$multiplate\$**: Amount of times that the plate has been read before reporting.
- **\$signaled\$**: True if the read has been done due to a trigger.
- **\$id\$**: Database ID for this read.
- **\$direction\$**: Enumerate with the vehicle direction (0: Unknown, 1: Towards, 2: Away, 3: Stopped)
- **\$directionstr\$**: String with the vehicle direction.
- **\$safedate\$**: Date in format %Y%m%d\_%H%M%S in the camera time zone (Useful for filenames).
- **\$localdate\$**: Date in format %d/%m/%Y in the camera time zone
- **\$localtime\$**: Date in format %H:%M:%S in the camera time zone.
- **\$imageid\$**: Signal ID in case of a trigger read.
- **\$plateimage\$**: Plate crop JPEG image encoded in base64.
- **\$platejpgsize\$**: JPEG size in bytes.
- **\$overviewimage\$**: Overview JPEG image encoded in base64.
- **\$overviewjpgsize\$**: JPEG size in bytes.
- **\$epoch\$**: Unix epoch (seconds).
- **\$utcdate\$**: Will report the date at ISO8601 format but always in UTC. (2020-12-31T16:11:30.000Z)
- **\$etx\$**: End transmission character (03)
- **\$stx\$**: Start transmission character (02)

You can add your own parameters into the message, so if you want to add say a site ID, your message might look like this:

```
{"plate":"$plate$", "date":"$date$", "ip":"$ip$", "country":"$country$", "sitecode": 12345}
```



## 25. Genesis Grammar Rules Definition

### 25.1 Grammar rules information structure

Vaxtor OCR Genesis requires a simple ASCII file which includes a set of syntactic and grammar rules to be applied, separated by a new line sequence.

The grammar file structure is made up 3 different sections:

- **Character replacement and deletion rules**
- **Definition of variables**
- **Grammar rules**

To define the components in each section we use three different types of items:

- **Tokens: representing literals of a specific type**
- **Basic operators and definitions**
- **Characters and digits representing their own explicit value**

### 25.2 Tokens, operators, and definitions

*(See Appendix for ASCII codes for the special characters mentioned below)*

#### 25.2.1 Replacement and deletion rules section

- ^ The rule for a replacement applies to the first characters of the sequence
- \$ The rule for a replacement applies to the last characters of the sequence
- ? Mark character for deletion
- ~ Replacement rule separator

#### 25.2.2 Definition of variables section

- = Variable assignment

#### 25.2.3 Definition of variables and grammar rules sections

- [,] OR clause between the statements inside the brackets.

#### 25.2.4 Grammar rules section

- %{} Variable usage
- %0 Void, no character
- N: Rule matches if the character sequence is arranged in N lines where  $1 \leq N \leq 3$

#### 25.2.5 Common to all sections

- %D Numeric character: "0123456789"
- %L Alphabetic character: "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
- %V Vowels "AEIOU"
- %C Consonants: "BCDFGHJKLMNPQRSTVWXYZ"
- \* Alphanumeric character (numeric or alphabetic)
- # Start comment

### 25.3 Replacement rules section

Rules to erase or replace a specific alphanumeric with another one

- The OCR can't differentiate O/0, it will always output 0 (zero)
- The OCR can't differentiate l/1 when the representation is just a vertical line "l"

I I I	The OCR will read 'i'
1 1 l	The OCR will read '1'
0 O 0	The OCR will read '0'

This section is particularly useful to "force" the replacements because the same alphanumeric shape can have different meaning depending on the context or use case.

Format:

`[^]<input pattern>[$]~ [^][^]<output pattern>[$]`

- The token around the pattern indicates the matching position in the sequence, in case of having no token the rule applies to the first match anywhere in the sequence.
- A pattern made up of literals and at least one explicit alphanumeric to be replaced or erased
- The tokens and the literals for the input pattern and the output pattern must be the same, only the explicit alphanumeric to replace or erase must change.
- The replacements and deletions apply sequentially when having more than one replacement or deletion statement.

Example 1:

Grammar rule of our text: 3 letters and 3 digits

Sample texts: ABC123, VHR394, PPW992...

Rule: If the first character is a 0, replace with O

`^0%L~^0%L` => Correct

`^0%L%L%D%D%D~^0%L%L%D%D%D` => Correct, more robust approach

`0%L~0%L` => Incorrect, it could replace the second character

Example 2:

Grammar rule of our text: one or more digits and 1 character (any)

Sample texts: 4364R, 449232S, 2530...

Rule: Replace the last 1 with I and 0 with O. We need 2 replacement rules:

`%D1$~%DI$` => If the last character is 1, replace with I

`%D0$~%DO$` => If the last character is 0, replace with O

Example 3:

Grammar rule of our text: one or more alphabetic characters

Sample texts: AR, PGKDS, GHJSO, SPII34...

Rule: Replace every 1 with I and every 0 with O. We need 2 replacement rules

1~I => Replace every '1' with 'I'

0~O => Replace every '0' with 'O'

Example 4:

Grammar rule of our text: 2 characters and 3 numbers

Sample texts: AO923, KJ145, OI992...

Rule: Replace any '1' with 'I' and '0' with 'O' in the first 2 positions

Valid but weak approach

^1\*~^I\*  
^0\*~^O\*  
^\*1~^\*I  
^\*0~^\*O

Valid and robust approach

^1\*%D%D%D~^I\*%D%D%D  
^0\*%D%D%D~^O\*%D%D%D  
^\*1%D%D%D~^\*I%D%D%D  
^\*0%D%D%D~^\*O%D%D%D

Example 5:

Grammar rule of our text: 5 alphanumeric sequence

Sample texts: AB834, 98HUO, 1299W...

Rule: Replace any 5 alphanumeric sequence with the number 99999

^\*\*\*\*\*\$~^99999\$

Example 6:

Grammar rule of our text: 5 digits

Sample texts: 23425, 44536, 75433...

Rule: Remove every alphabetic character from the sequence

%L~?

%D%D%D%D%D

Text: 23456P Result: 23456 (it matches the grammar)

Text: A82BC89U3Result: 82893 (it matches the grammar)

Text: B8923CB Result: 8923 (it does NOT match the grammar)

## 25.4 Grammar rules section

Set of grammar rules to follow by the characters sequence. The rules will be processed sequentially after executing the replacement section (if any).

### Basic examples, fix length sequences

6 digits %D%D%D%D%D%D

3 digits and 3 consonants %D%D%D%C%C%C

Starts with 'P' followed by 5 digits P%D%D%D%D%D

3 digits and 2 consonants %D%D%D%C%C

### Combining fixed length sequences with replacements

#### 3 digits and 2 characters

1\*\$~I\*\$ # Replace last but one '1' with 'I'

0\*\$~O\*\$ # Replace last but one '0' with 'O'

\*1\$~\*I\$ # Replace last '1' with 'I'

\*0\$~\*O\$ # Replace last '0' with 'O'

%D%D%D%L%L # Apply grammar rule, 3 digits and 2 characters

#### 2 vowels and 4 digits assuming our 'I' letter is represented by 'I' or 6 digits sequence

^0\*%D%D%D%D~^0\*%D%D%D%D # Replace the first '0' with 'O'

^\*0%D%D%D%D~^\*O%D%D%D%D # Replace the second '0' with 'O'

%V%V%D%D%D%D # Apply rule 1

%D%D%D%D%D%D # Apply rule 2

### Variable length sequences: introducing the 'OR' clause and the void literal

5-6 digits %D%D%D%D%D[%D,%0]

5 digits ending in A, B, or C %D%D%D%D%D[A,B,C]

2-4 digits and 1 or 2 consonants %D%D[%D,%0] [%D,%0]%C[%C,%0]

Starting with 'AZ12' and 0-2 alphanumeric AZ12[\*,%0] [\*,%0]

#### 9 digits phone numbers starting with 609 with and without any 1-3 digits prefix

[%D,%0] [%D,%0] [%D,%0]609%D%D%D%D%D%D

#### 3 or 4 digits numbers alone or preceded by 'AA' or 'BB'

[AA,BB,%0]%D%D%D[%D,%0]

## Introducing the variables definition

The definition of variables appears in the second section, between the replacements and the grammar rules definition section. You should include those definitions before specifying any rule. The format is as follows:

`<variable_name>=<grammar_rule>`

The name of the variable includes alphanumeric and non-token characters, it may not include blanks.

### 3 digits and 2 consonants

```
MY_VARIABLE=%D%D%D          # Variable definition
%{MY_VARIABLE}%C%C          # Using the variable in a grammar rule
```

### Combination of 2 characters AA,BB,CC,DD followed by 4 to 6 digits

```
COMBO=[AA,BB,CC,DD]
%{COMBO}%D%D%D%D[%D,%0] [%D,%0]
```

Note that any combination of characters **not** defined by COMBO won't match the rule

### 2 numbers plus AA,BB combination plus 2-4 numbers plus C,D,E,F,G characters

```
VAR1=[AA,BB]
VAR2=[C,D,E,F,G]
%D%D%{VAR1}%D%D[%D,%0] [%D,%0]{VAR2}
```

### Read the following words only: STOP, YIELD, SPEED, and 'speed' is followed by 2 or 3 digits

#### *# Section replacement*

```
^STOP~^STOP          # Replace the 0 with O
^Y1ELD~^YIELD        # Replace the 1 with I
```

#### *# Section variables definition*

```
WORDS=[STOP,YIELD]
```

#### *# Section grammar rules*

```
%{WORDS}             # Rule 1: stop or yield
SPEED%D%D[%D,%0]    # Rule 2: speed plus a 2 or 3 digit number
```

### Read numbers of 5-6 digits where every digit is equal or greater than 6

```
N=[6,7,8,9]          # Numbers allowed
%{N}%{N}%{N}%{N}%{N}[%{N},%0] # 5 or 6 digits number of allowed numbers only
```

### Combination of 2 colors separated by an odd number

0~0

1~1

%L%L%L%L%L%L%L~%L%L%L%L%L%L%L

N=[1,3,5,7,9]

COLOR1=[RED,BLUE,YELLOW,ORANGE,BLACK,WHITE]

COLOR2=[GRAY,GREEN,PINK,BROWN,MAGENTA,VIOLET]

[%{COLOR1},%{COLOR2}]%N[%{COLOR1},%{COLOR2}]

### Selective rules depending on the number of lines the sequence is arranged

The OCR can process 1, 2, or 3 line codes, the rules apply to any of them by default. However, it is possible to constrain the rules to the number of lines of the code.

<number\_of\_lines>:<grammar rule> # Number of lines can be 1, 2 or 3

Rule: %L%L%L%D%D%D%D # Default, it applies to any arrangement

ABC  
1234



ABC1234



Rule: 1:%L%L%L%D%D%D%D

# It applies to 1 line sequences only

ABC  
1234



ABC1234



Rule: 2:%L%L%L%D%D%D%D

# It applies to 2 lines sequences only

ABC  
1234



ABC1234



Rule: 3:%L%L%L%D%D%D%D

# It applies to 3 lines sequences only

ABC  
1234



ABC1234



## 25.5 Example, extracting information from fast-food ticket restaurant

### Real ticket



CAJA 23- 16/01/2018 14:34:48

### Grammar rules

DATE=%D%D%D%D%D%D%D%D

TIME=%D%D%D%D%D%D

CAJA%D[%D,%0]%(DATE)%{TIME}

### OCR results

048  
 Restaurante McDonald's  
 Nuevo Barajas  
 Restaurantes McDonald's S.A.U  
 CIF: A-20506097  
 C/Severa 5, 28023 Madrid

FACTURA SIMPLIFICADA  
 Num: 006530232018011801502  
 Num. Ticket: 00011944

Num Pedido: 41  
 CAJA 23- 16/01/2018 14:34:48

CMR	ARTICULO	TOTAL
1	MENU DE CUARTO	7,10
1	PAI LITE MENU QR	
1	SALSA SWEET	
1	ZERO MENU QR	
1	SEN COMPLEMENTO	
Total (incl IVA)		7,10
ICR		7,10

IIVA CANTIDAD IVA  
 IVA 10,00% de 6,45818 = 0,65

Valora tu visita desde la APP de McU y llévate un segundo McFlurry de regalo  
[www.opiniamcdonalds.com](http://www.opiniamcdonalds.com)  
 @mcu-bvff-brbs

1TU SEGUNDA BEBIDA SIDA 1C1  
 Presenta este Ticket y disfruta durante el día de tu compra de tu segunda bebida por 1C (DESCR, SIDA)

CAJA2316012018143448

CAJA 23- 16/01/2018 14:34:48

## 25.6 Example, extracting digits from a long code/date

Let's imagine that we have this code:



But we're interested in reading only the code inside the red rectangle:



The OCR internally is going to read all of the alphanumerics: **KJ21240140520**

The valid grammar for this in Genesis would be %D%D%D%D%D%D%D

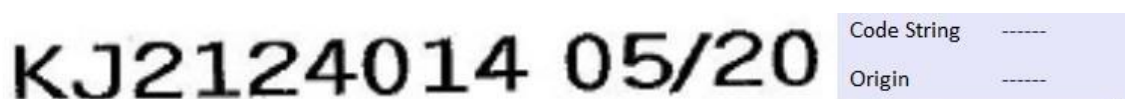
However, this is not enough, we need a mechanism to tell Genesis the proper way to isolate the code we're after.

The grammar rule %D%D%D%D%D%D%D matches the code we want, but if we define this as a grammar rule then Genesis will behave as follows:

**Grammar Strict OFF**



**Grammar Strict ON**



As you can see there is no way to isolate the code by conventional mechanisms.

**SOLUTION:**



Make use of the '?' token to erase characters in the replacement sections, as explained earlier in this manual.

There are several ways to setup the proper rules:

1. Isolate the KJ code at the beginning: **%L~?** This rule will replace all non-numeric characters from the string. We can be more accurate and narrow the replacement to the first 2 characters **^KJ~^??**
2. The second rule causes the removal of the last 4 digits; there are several ways to achieve this too: **%D%D%D%D\$~????\$**  
This rule will remove the last 4 numbers of the code.

So, the full grammar to isolate and read our 7 digits code will be:

**%L~?**  
**%D%D%D%D\$~????\$**  
**%D%D%D%D%D%D%D**

**KJ2124014 05/20**

Code String	2124014
Origin	Genesis

Now, if we wished to isolate these parts of the code:



**KJ2124014 05/20**

The Grammar rules would be:

**^KJ%D%D%D%D%D%D~^KJ???????**  
**KJ%D%D%D%D**

Result:

**KJ2124014 05/20**

Code String	KJ0520
Origin	Genesis



## 26. ASCII SPECIAL CODES

To enter the following special codes, use the ALT key followed by digits on the numeric keypad:

^	ALT + 94
~	ALT + 126
[	ALT + 91
]	ALT + 93
{	ALT + 123
}	ALT + 125

## **27. Changelog**

### **27.1 Version 23-12-2021**

- ✓ Strict Grammar update.

### **27.2 Version 1.1 08-08-2022**

- ✓ Improves levels processing accuracy in code-finder module
- ✓ Includes 'country\_confidence' parameter in the result structure just for VaxALPR compliance
- ✓ Support for Vaxtor font (Charles Wright) for custom labels using UK plates font
- ✓ Automatic ML net processing depending on the label type (Vaxtor or generic)

Ends.