

**mav** HOME OF ANPR CAMERAS

MAV IQ



## MAV IQ Intelligent ANPR Camera

APPLICATION PROGRAM INTERFACE (API)

## Contents

<b>About this Guide .....</b>	<b>2</b>
Icons Used in this Manual.....	2
Copyright Notice .....	2
Trademarks .....	2
Technical Support.....	2
<b>API Overview .....</b>	<b>3</b>
Web Service API.....	3
Push Service API.....	3
<b>TCP/IP Connection Requirements.....</b>	<b>4</b>
Limited TCP Ports.....	4
Quiet Link Detection .....	4
IP Address .....	5
Port Number .....	6
<b>Web Service API.....</b>	<b>7</b>
Camera Status.....	8
Camera Control .....	9
Camera Images .....	10
ANPR Decodes.....	11
Configuration Parameters.....	16
Retrieving Parameters.....	16
Setting Parameter Values.....	16
Security.....	17
<b>Push Service API .....</b>	<b>18</b>
Connection Mode.....	18
Server mode .....	18
Client Mode .....	18
Decode Messages .....	18
<b>Demo Client.....</b>	<b>19</b>

## About this Guide

### ICONS USED IN THIS MANUAL

These special messages refer to noteworthy information, and include a symbol for quick identification:



**Alert:** Important information that cautions about features affecting performance, security features, or causing potential problems with your Intelligent ANPR Software.



**Tip:** Useful information about the configuration of your Intelligent ANPR Firmware.



**Note:** Important information on a feature that requires callout for special attention.



**Cross Reference:** Provides a pointer to related information in this or other documentation.

### COPYRIGHT NOTICE

This document is Copyright © MAV Systems Ltd, 2015. All rights reserved. The content of this document is licensed from Rudstone Technologies Ltd. Reproduction of any part of this guide in any form whatsoever requires express written permission from MAV. The contents of the guide are subject to change, and MAV can accept no responsibility for any errors, or their consequences.

### TRADEMARKS

MAV Systems Ltd have a number of products and components as registered trademarks and the use of any of these registered trademarks for any purpose must be confirmed with MAV prior to release.

### TECHNICAL SUPPORT

For technical support, please contact MAV using [anpr@anprcameras.com](mailto:anpr@anprcameras.com) for email requests or contact +44 01344 859753 for assignment of a point of contact.

## API Overview

The MAV IQ has firmware referred to as the Intelligent ANPR Module that exposes two API connections:

The Web Service Interface – which supports camera configuration, obtaining images and obtaining decode information.

The Push Service – which supports live decode information in real-time.

### WEB SERVICE API

The Intelligent ANPR Module contains a webserver that serves HTML web pages and is able to respond to specific requests with JSON objects and JPEG Images.

For example, requesting: [http://ip\\_address/anprmon?func=GetImage&cameraid=1](http://ip_address/anprmon?func=GetImage&cameraid=1) the webserver will return an “image/jpeg” document showing the current image from the overview camera.

The API interface can be tested directly from a browser by entering the URL information directly into the address field. The browser will make the best attempt at displaying the returned document which is usually adequate.

More examples of how the Web Service API interface operates can be found in the default web pages loaded on the camera’s webserver. By accessing the camera’s default page, there is a complete web application for displaying the camera’s current images, ANPR Decodes and allowing camera control and configuration of various parameters.

The default web application is implemented with basic HTML and a combination of JavaScript and JQuery performing the underlying requests for information from the camera. A suitably experienced programmer will be able to use these pages as a reference.

### PUSH SERVICE API

The Intelligent ANPR Module contains a Push Server that can be configured as a server socket (to allow inbound connections) or as a client socket (to connect out to a server) and will “Push” decodes to the host as they happen.

The content of the message sent by the Push Service is the same decode response used by the web server except it will only contain 1 decode.

## TCP/IP Connection Requirements



This section contains some requirements above and beyond a typical TCP/IP connection to ensure uninterrupted operation. Please ensure that you have read this section completely.

### LIMITED TCP PORTS

The TCP/IP stack in the Intelligent ANPR Module does not have access to unlimited resources, so it must operate within set limits.

There is a hard limit of 10 client connections on the HTTP Server and 4 client connections on the Push Service when configured in server mode.

When these resources have been exhausted, connection requests are rejected until resources become available again. This is called “port starvation”.

A common cause of “port starvation” is a client that disconnects after each transaction.

Upon disconnect, the TCP port enters a state called “TIME\_WAIT” where it is unavailable for a period of time (in the order of 4 minutes). The background to this is in RFC793.

To avoid this situation, it is recommended that the Host establishes and maintains a connection to the Intelligent ANPR module without closing the connection between transactions.



If you are using a browser to access the camera, whilst many older browsers placed a limit of 2 simultaneous connections to a server, that no longer seems to be the case. Some newer browsers seem to exploit many more connections, so there is a possibility of running out of sockets if multiple browsers connect at the same time.

## QUIET LINK DETECTION

It is possible for a TCP/IP link to be disconnected silently such that both end-points are unaware of the problem.

These situations would occur if some intermediate equipment was reset and abandoned its NAT routing table, for example, or if some mobile equipment was involved.

The quiet link would not be discovered until one end sent some data. At that point the socket would report an error.

Both the TCP Server socket and the TCP Client socket in the Intelligent ANPR Module will perform "Quiet Link Detection".

If there has been no traffic for around 2½ minutes, the session will be closed.

The strategy for working with this mechanism is different for the Web Service and the Push Service.

Push Service Client Mode	The Push Service will create a "<CR><LF>" sequence every minute. The host service will see a zero length message (because it is expecting <CR> to terminate a message) and should ignore zero length messages.
Push Service Server Mode	The Client (Host) must send a "<CR>" or "<CR><LF>" sequence every minute. The Push Service ignores any data sent to it.
Web Server	Generally not an issue because of the polling design on the interface; however, if you have not polled for 2½ minutes, the port could be disconnected.  To generate "passive traffic", send a Camera Status Request every minute.

## IP ADDRESS

The camera has three mechanisms for obtaining an IP address:-

- The IP Address can be statically configured and, as such, the camera will have a fixed IP address.
- The IP Address is requested from a DHCP server.
- The IP Address is locally assigned using the method documented in RFC 3927, i.e. a random 169.254.0.0/16 address.



In the case of the latter, an attempt to get a DHCP assigned address will happen periodically and, if successfully, the camera will change address as per RFC 3927.

Only IP V4 is currently supported.

## PORT NUMBER

The default port to communicate with the camera's web server is port 80/TCP.

This can be changed by configuration if required.

Changing the port is not something that is expected to be a common event, so it is changed using the request in a browser <http://<<cameraAddr>>/Admin/Config?func=setdata&30004=81>

The above example will change the HTTP Port number to port 81. It requires a power cycle to reset the HTTP Server.

## Web Service API

The Intelligent ANPR Module contains a webserver that can serve HTML pages and is able to respond to specific requests with JSON objects and JPEG Images.

In doing so, it can:-

- Control aspects of the ANPR (Mono) camera and the Overview (Colour) camera.
- Retrieve images from either camera.
- Obtain decode information.
- Change configuration parameters.

## Camera Status

The current camera status can be obtained by sending the following request:

<http://<<cameraAddr>>/cameracontrol?func=getstatus>

The expected response to this message will be a *Camera Status* message formatted as a JSON Object.

<pre> {   "ANPRCameraStatus":   {     "zoomPos": 0,     "focusPos": 20,     "rawZoomPos": 0,     "rawFocusPos": 1000,     "focusMode":     "manual",     "shutterTuning":     2625,     "irisTuning": 584,     "agcTuning": 192,     "shutter":     "unknown",     "ircMode": "mono"   },   "OVRCameraStatus":   {     "zoomPos": 0,     "focusPos": 20,     "rawZoomPos": 0,     "rawFocusPos": 1000,     "focusMode":     "manual",     "shutterTuning":     1984,     "irisTuning": 500,     "agcTuning": 300,     "shutter": "1/100",     "ircMode": "colour"   } }                 </pre>	<table border="1"> <tr> <td style="background-color: #f2f2f2;"><b>zoomPos</b></td> <td>Current zoom position in scaled units between 0 and 1000.</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>focusPos</b></td> <td>Current focus position in scaled units between 0 and 1000.</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>rawZoomPos</b></td> <td>Current zoom position in raw camera units</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>rawFocusPos</b></td> <td>Current focus position in raw camera units</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>focusMode</b></td> <td>Current focus mode:- "auto" "manual"</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>shutterTuning</b></td> <td>Current shutter value in raw camera units.</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>irisTuning</b></td> <td>Current iris value in raw camera units.</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>agcTuning</b></td> <td>Current agc value in raw camera units.</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>Shutter</b></td> <td>The current shutter speed. Only relevant in shutter priority mode.</td> </tr> <tr> <td style="background-color: #f2f2f2;"><b>ircMode</b></td> <td>Current irc mode:- "mono" "colour" "unknown"</td> </tr> </table>	<b>zoomPos</b>	Current zoom position in scaled units between 0 and 1000.	<b>focusPos</b>	Current focus position in scaled units between 0 and 1000.	<b>rawZoomPos</b>	Current zoom position in raw camera units	<b>rawFocusPos</b>	Current focus position in raw camera units	<b>focusMode</b>	Current focus mode:- "auto" "manual"	<b>shutterTuning</b>	Current shutter value in raw camera units.	<b>irisTuning</b>	Current iris value in raw camera units.	<b>agcTuning</b>	Current agc value in raw camera units.	<b>Shutter</b>	The current shutter speed. Only relevant in shutter priority mode.	<b>ircMode</b>	Current irc mode:- "mono" "colour" "unknown"
<b>zoomPos</b>	Current zoom position in scaled units between 0 and 1000.																				
<b>focusPos</b>	Current focus position in scaled units between 0 and 1000.																				
<b>rawZoomPos</b>	Current zoom position in raw camera units																				
<b>rawFocusPos</b>	Current focus position in raw camera units																				
<b>focusMode</b>	Current focus mode:- "auto" "manual"																				
<b>shutterTuning</b>	Current shutter value in raw camera units.																				
<b>irisTuning</b>	Current iris value in raw camera units.																				
<b>agcTuning</b>	Current agc value in raw camera units.																				
<b>Shutter</b>	The current shutter speed. Only relevant in shutter priority mode.																				
<b>ircMode</b>	Current irc mode:- "mono" "colour" "unknown"																				

## Camera Control

All camera control requests are sent to <http://<<cameraAddr>>/cameracontrol> where [<<cameraAddr>>](http://<<cameraAddr>>) is the IP Address of the target camera.

The operation is routed to the ANPR Camera by suffixing the request with the parameter "[cameraid=0](http://<<cameraAddr>>/cameracontrol?func=<func>&cmd=<cmd>&cameraid=0)" and to the Overview Camera by adding "[cameraid=1](http://<<cameraAddr>>/cameracontrol?func=<func>&cmd=<cmd>&cameraid=1)".

An example would be: <http://192.168.1.211/cameracontrol?func=zoom&cmd=wide&cameraid=0>

In the tables below add the prefix <http://<<cameraAddr>>/cameracontrol> and the suffix "[cameraid=0](http://<<cameraAddr>>/cameracontrol?func=<func>&cmd=<cmd>&cameraid=0)" or "[cameraid=1](http://<<cameraAddr>>/cameracontrol?func=<func>&cmd=<cmd>&cameraid=1)" as required.

Zoom Wide	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=zoom&amp;cmd=wide...">...?func=zoom&amp;cmd=wide...</a>
Zoom Tele	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=zoom&amp;cmd=tele...">...?func=zoom&amp;cmd=tele...</a>
Zoom Stop	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=zoom&amp;cmd=stop...">...?func=zoom&amp;cmd=stop...</a>
Zoom Direct	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=zoom&amp;cmd=&lt;&lt;some number in camera units&gt;&gt;">...?func=zoom&amp;cmd=&lt;&lt;some number in camera units&gt;&gt;</a>
Focus Near	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=focus&amp;cmd=near...">...?func=focus&amp;cmd=near...</a>
Focus Far	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=focus&amp;cmd=far...">...?func=focus&amp;cmd=far...</a>
Focus Direct	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=focus&amp;cmd=&lt;&lt;some number in camera units&gt;&gt;">...?func=focus&amp;cmd=&lt;&lt;some number in camera units&gt;&gt;</a>
Focus Stop	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=focus&amp;cmd=stop...">...?func=focus&amp;cmd=stop...</a>
Focus Manual	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=focus&amp;cmd&gt;manual...">...?func=focus&amp;cmd&gt;manual...</a>
Focus Auto	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=focus&amp;cmd=auto...">...?func=focus&amp;cmd=auto...</a>
Focus Oneshot	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=focus&amp;cmd=oneshot...">...?func=focus&amp;cmd=oneshot...</a>
IRC Mono	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=irc&amp;cmd=mono...">...?func=irc&amp;cmd=mono...</a>
IRC Colour	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=irc&amp;cmd=colour...">...?func=irc&amp;cmd=colour...</a>
IRC Auto	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=irc&amp;cmd=auto...">...?func=irc&amp;cmd=auto...</a>
Preset Save	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=preset&amp;cmd=save...">...?func=preset&amp;cmd=save...</a>
Preset Recall	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=preset&amp;cmd=recall...">...?func=preset&amp;cmd=recall...</a>
Get Status	<a href="http://&lt;&lt;cameraAddr&gt;&gt;/cameracontrol?func=getstatus">...?func=getstatus</a> [no camera id required on this function]

The response to the camera control request is a camera status message. See the previous "Camera Status" section.



Whilst a OneShot focus is in progress, the focus mode will change to "Auto" then back to "Manual".

## Camera Images

A JPEG image can be requested from either camera.

To request a JPEG image, use one of the following HTTP Requests:-

ANPR <http://<<cameraAddr>>/ANPRMon?func=GetImage&cameraid=0>

Overview <http://<<cameraAddr>>/ANPRMon?func=GetImage&cameraid=1>

The response will be a JPEG image.

Whilst these images can be requested constantly to create a pseudo video stream, the frame rate is paced to about 4 frames per second so as not to overburden the system and will vary depending on performance and load.

An optional MJPEG add-on is available for video streaming.

## ANPR Decodes

The Intelligent ANPR Module stores a rolling buffer of the last 8 decodes seen.

Each decode has its own unique reference number (starting at 1 on each reboot).

When making a request for the decode list, a "Last Reference" must be given in the request and only decodes since the "Last Reference" will be returned.

If a reference of 0 is sent or no reference at all, then all held decodes are returned.

To request the ANPR Decode, send the following:

<http://<<cameraAddr>>/ANPRMon?func=decodes&ref=<<reference>>>

Where <<reference>> is the last reference seen or is initially "0".

Because of the amount of data returned, it is not helpful to describe the object in its raw form.

A pseudo BNF notation is used with further objects being described between"«»" characters.

Returned JSON Object:=

```
{ "decodes" :
  [ «between zero and eight decode object»
  ]
}
```

**Decode Object:=**

```
{ "ref" : 43989,
  "vrm" : "AV11LPR",
  "frameID" : 12029991,
  "spacedVRM" : "AV11 LPR",
  "seenCount" : 18,
  "serialNo" : 2568084344,
  "MAC" : "00:04:A3:4F:0D:79",
  "cameraName" : "Test_MAV_IQ",
  "decodeID" : 44606,
  "charHeight" : 15,
  "location" :
    { "latitude" : 0.00000000,
      "longitude" : 0.00000000,
      "name" : "",
      "source" : 0},
  "repeatedPlate" : false,
  "timeStamp" :
    { "Time" : 1473670749,
      "ms" : 640,
      "LocalTime" : 1473677949},
  "timeSync" :
    { "timeMode" : 0,
```

## Application program interface (api)

```

    "lastSyncTime" : 1473062065,
    "lastSyncSource" : "SNTP",
    "lastSyncInfo" : "192.53.103.108"},
"confidence" : 94,
"correctSpacing" : true,
"countryCode" : "UK",
"preferredCountry" : true,
"preferredFormat" : true,
"syntax" : "1994-",
"direction" : 126,
"motion" : "towards",
"laneInfo" :
    {"ID" : 1} ,
"accessoryStatus" :
    {"12VOut" : false,
     "relayOut" : false,
     "input" : true,
     "12VSense" : false} ,
"frameTimeRef" : 606284767,
"nightModeActive" : false,
"frameHeight" : 720,
"frameWidth" : 1280,
"plateRect" :
    {"left" : 401,"top"
     : 346,"right"
     : 520,"bottom"
     : 372} ,
"zoomLocked" : true,
"overviewSource" : 2,
"overviewScale" : 2,
"overviewRect" :
    {"left" : 84,
     "top" : 0,
     "right" : 804,
     "bottom" : 576} ,
"overviewPlateRect" :
    {"left" : 300,"top"
     : 338,"right"
     : 420,"bottom"
     : 364} ,

```

Plate Track Object:=

```

"charHeight" : 13,
"frameID" : 12029973,
"platePos" : {"x" : 275, "y" : 224},

```

## Application program interface (api)

```

    "plateWidth" : 69
    ** repeat by number of tracks

"rawReads" :=
    "[AV11LPR],
    [AV11 LPR],
    ** repeat by number of reads

"speed" :=
    {"radarSpeed" : 0,
     "radarSpeedUnits" : "NoSpeed",
"radarRawSpeeds" :
    [12029973,0,0],
    [12029974,0,0],
    ** repeat by number of reads

"radarDiags" : ""}

"plate" : "«base64encodedJPEG»",
"overview" : "«base64encodedJPEG»"

END

```

The plate and overview fields have been edited out and would normally contain a base 64 encoded JPEG image.

The following table contains a description of the fields in the decode information:

## DECODE OBJECT

ref	An incremental number assigned to each individual decode. Resets to 1 on reboot.
cameraName	The Camera Name as defined in the configuration.
serialNo	The serial number of the unit.
MAC	The MAC address of the network interface.
plate	A base64 encoded JPEG image of the plate patch decoded
overview	A base64 encoded JPEG image of the overview selected
repeatedPlate	Is false if this is the first decode of the VRM. Is true if the re-admit timer has expired and the plate remained in view.

decodeID	<p>An incremental ID of each VRM Decode.</p> <p><i>Note. To be used in conjunction with repeatedPlate value.</i></p> <p><i>If repeatedPlate is false, then it is the first instance of a decode for the plate.</i></p> <p><i>If repeatedPlate is true, then the plate has been in view for longer than the re-admit time and is a repeat.</i></p>
frameHeight	The height of the raw image from the camera
frameWidth	The width of the raw image from the camera
Location	A location object identifying the location of the camera
source	<p>See “Location Object” below for more details</p> <p>0 - “Unknown”</p> <p>1 – Manual</p> <p>2 - last GPS Fix</p> <p>3 - GPS Fix</p>
name	A text string entered in the camera configuration to describe the location.
latitude	The latitude reported in decimal degrees.
longitude	The longitude reported in decimal degrees.
timestamp	
Time	The time in seconds since 1/1/1970 00:00 (CTIME).
Ms	The number of milliseconds within the seconds of Time.
charHeight	The height of the characters in the selected plate patch in pixels.
confidence	A confidence factor. This is the average of confidence factor returned by the neural net for each character. Because the confidence factors from the neural net have a tendency to be high for any match, the overall confidence factor should not be used for any decision making.
correctSpacing	The VRM not only matches the characters in a syntax rule, but also the correct spacing requirement too.
countryCode	The country code from the syntax that was matched.
direction	The vector angle of the plate track taken between first read and last read. Zero degrees is at the top of the image and increments in a clockwise direction. A value of -1 indicates that there is not enough x or y change to calculate the angle, usually a stationary plate.
frameID	The frame number from the camera from which the plate patch was taken.
vrn	The Vehicle Registration Mark (number/license plate) without any spacing.
frameTimeRef	The time reference for the overview frame.
nightModeActive	A boolean value reflecting the Night Mode of the unit.
plateRect	The bounded rectangle of the plate in camera pixel units (in raw image).
preferredCountry	The VRM matched a syntax rule marked as a preferred country.

# Application program interface (api)

preferredFormat	The VRM matched a syntax rule marked as a preferred format.
spacedVRM	The VRM represented with the detected spacing.
syntax	The name of the rule that the VRM matches in the enabled syntax rules.
plateTrack	An array of between 1 and 20 plate track objects. There is one for each frame the plate was decoded in.
charHeight	The character height in the decode frame.
frameID	The frame number the plate was decoded from.
platePos	The X,Y position of the centre of the plate rectangle (in raw image).
plateWidth	The distance between the left of the first character to the right of the last character in the decoded frame – Note: not the plate background that the characters are surrounded by.

## Configuration Parameters



This interface is designed to be used by the web interface for setting configuration options. Developers intending to use this interface should examine how the built-in web pages use the parameters to ensure that appropriate values are being set. Incorrect values may have unintended consequences and may affect the operation of the module.

The configuration parameters accessed by the web pages are accessed over the Web Service API.

There is a *GetData* syntax for retrieving parameters and a corresponding *SetData* syntax for writing new values.

The parameters are identified by a hexadecimal value which can be found in the table below.

### Retrieving Parameters

The syntax for retrieving parameters is as follows:- <http://<<cameraAddr>>/Admin/Config?func=GetData>

Parameters can be retrieved by appending the parameter numbers to the above request. Multiple parameters can be retrieved in one request.

For example: <http://127.0.0.1/Admin/Config?func=getData&2&4&4&030004>

The response is a “Config” JSON object described as follows:-

```
{ "Config" : [
  { "ID": "2", "Value" : "1.0.0.0" },
  { "ID": "4", "Value" : "MAV Systems Ltd.<br/>01234 123456" },
  { "ID": "30001", "Value" : "3232236098" }
] }
```

The response is a class that contains a “Config” item that is an array of Config Items.

Each item is an object containing an “ID” (that has been requested) and its current “Value”.

Returned values can be any JSON object type, but are commonly limited to String, Number, Boolean. See [www.json.org](http://www.json.org) for more information.

### Setting Parameter Values

The syntax for setting parameters is as follows:- <http://<<cameraAddr>>/Admin/Config?func=SetData>

Parameters can be set by appending the parameter numbers and new values to the above request. Multiple parameters can be set in one request.

E.g: <http://127.0.0.1/Admin/Config?func=SetData&1=My%20Camera&2=1.2.3.4>

This will set parameter 1 to “My Camera” and parameter 2 to “1.2.3.4”.

It is not necessary to quote strings as it is taken care of in the HTTP Protocol.

The response is normally used for displaying directly to an interactive user in a dialog box and as such contains formatted HTML.

The response is either simply “Configuration settings saved.” or is a list of error messages which relate to failed validation of parameters.

## Security

The configuration parameters are protected by a username/password combination. These credentials (see the Web Interface User Guide) will need to be provided to the HTTP Server

### PARAMETER LIST



The parameter list is described in a separate document “Configuration Parameters” reference PRJ001-DOC-0004. Additional information on the parameters is described in the “Web Interface User Guide”, reference PRJ001-DOC-0002.

## Push Service API

Unlike the Web Server API, the Push Service offers a real-time interface over which decodes will be sent as they occur.

### CONNECTION MODE

The push service offers two different connection modes:-

- Server Mode
- Client Mode

### SERVER MODE

When in Server Mode, the Intelligent ANPR Module will expect inbound client connections from the host.

This mode of operation is useful for planning redundant configurations.



Please ensure that you have read the *TCP/IP Connection Requirements* section.

### CLIENT MODE

In Client Mode the Push Service makes an outbound connection to the configured address and port.

If the connection fails at any time, the Push service will attempt to reconnect at regular intervals (normally 5 seconds).

This mode of operation is useful when the IP address of the camera is not known or is behind a firewall.



Please ensure that you have read the *TCP/IP Connection Requirements* section.

## DECODE MESSAGES

Once a connection has been established, the Push Service will send a JSON decodes object containing a single decode object as described in the [ANPR Decodes](#) section.

The decode object will be terminated with a carriage return/linefeed (0x0D 0x0A) at the end of the object followed by a further carriage return/linefeed (to create the effect of a blank line).

Host software must be able to cope with multiple blank lines. In Client Mode, the client will periodically send a carriage return/linefeed to monitor the link. See



If you are using a browser to access the camera, whilst many older browsers placed a limit of 2 simultaneous connections to a server, that no longer seems to be the case. Some newer browsers seem to exploit many more connections, so there is a possibility of running out of sockets if multiple browsers connect at the same time.

### Quiet Link Detection.

There is no catch-up of decodes that have been missed. Only decodes that happen after the link has been established are sent to the Host Server.

## DEMO CLIENT

A C# Application is available as a reference.

The code only contains the 20% of useful code that works when everything is as expected. The 80% of code that does error checking and exception handling is missing by design.

It is intended to demonstrate techniques for working with the Web Service API and the Push Service API.

It is not uncommon in C# and the .NET framework to have several different ways of completing the same task, so the Demo Client is not intended to be the best way to implement your code nor a recommended method of operation, merely a reference for something that works and that can be used as a test harness for trying out things.

No support is offered for the Demo Client and it is not guaranteed to work with future changes that Microsoft may make to their software. It will be maintained however and is available as part of a maintenance programme.

# Application program interface (api)

Project	PRJ001
Name	Application Program Interface
Number	PRJ001-DOC-0003
Issue	1.2
Firmware	1.2.4

# Application program interface (api)

Approved by

EMP01