



The Carmen® ANPR Engine Reference Manual



This document describes the properties (user parameters) of the CARMEN® software and the cmAnpr engine handler software module.

The Carmen® ANPR Engine

Reference Manual

Document version: 2024.12.04.

Table of Contents

LIST OF THE PROPERTIES AND THEIR IMPORTANCE.....5

DESCRIPTION OF THE PROPERTIES OF THE CMANPR ENGINE HANDLER MODULE7

1. ANPRNAME.....7

POSSIBLE VALUES OF CMANPR ENGINE PROPERTIES.....8

DESCRIPTION OF THE PROPERTIES OF THE CMANPR ENGINES.....10

2. DATAFILE.....10

PROPERTIES RELATED TO PROCESSING TIME11

3. TIMEOUT11

3.1. TIMEOUT11

3.2. TIMEOUT_WALL.....12

4. ADAPT_ENVIRONMENT14

5. RECOGNITIONMODE16

6. DEPTH.....17

PROPERTIES RELATED TO THE GEOMETRY OF PLATES18

7. SIZE.....18

8. SIZE_MAX19

9. SIZE_MIN.....20

10.	SLANT.....	21
11.	SLANT_MAX.....	22
12.	SLANT_MIN.....	22
13.	SLOPE.....	23
14.	SLOPE_MAX.....	24
15.	SLOPE_MIN.....	24
16.	CHARHEIGHTINMM.....	25
17.	PLATESIZE(F).....	25
18.	PLATESIZE(F)_SOURCE.....	26
19.	XTOYRES.....	27
PROPERTIES RELATED TO FILTERING OUTPUTS BASED ON LICENSE PLATE CHARACTERISTICS.....		28
20.	LOCATION.....	28
21.	READ_ADR_TYPE.....	30
22.	READ_BASE_TYPE.....	30
23.	READ_EMPTYADR_TYPE.....	31
24.	READ_LICENSE_PLATE_TYPE.....	32
25.	READ_WITHOUT_TYPE.....	32
26.	GENERAL.....	33
27.	TYPEWEIGHT.....	35
28.	COLORTYPE.....	39
29.	NCHAR_MAX.....	39
30.	NCHAR_MIN.....	39
PROPERTIES RELATED TO CUSTOMIZATION OF RESULTS.....		40
31.	GAPTOSPACE.....	40
32.	PLATETYPEINFOS.....	43
33.	AUTOTYPEMODIFICATION.....	50
34.	CONVERTOTOO.....	50
35.	COUNTRYNAME.....	51
36.	CYRILLIC_STYLE.....	53
37.	LOCAL_CHARACTER_MODIFICATION.....	54
37.1.	LOCAL_CHARACTER_CONVERSION.....	54
37.2.	LOCAL_CHARACTER_PERMUTATION.....	54
38.	UNICODE_IN_TEXT.....	55
PROPERTIES RELATED TO COLOR RECOGNITION.....		56
39.	ANALYZECOLORS.....	56
40.	COLOR.....	58
40.1.	COLOR_SOURCE.....	60
40.2.	COLOR_SOURCE_VALUE.....	61

41. WHITEBALANCE.....	62
PROPERTIES RELATED TO THE POSITION OF LICENSE PLATES IN INPUT IMAGES.....	63
42. ROI/ROU.....	64
42.1. ROI.....	64
42.2. ROU.....	65
43. POSFREQ.....	67
44. POSFREQHALFLIFE.....	69
45. POSFREQHISTXS.....	70
46. POSFREQHISTYS.....	70
47. POSFREQWEIGHT.....	70
PROPERTIES RELATED TO IMAGE QUALITY.....	71
48. CONTRAST_MIN.....	71
49. GAMMA.....	71
PROPERTIES RELATED TO THE CALCULATION OF THE CONFIDENCE LEVEL.....	72
50. CONFIDENCEMODE.....	72
50.1. CONFIDENCEMODE_X.....	77
51. CONFIDENCE_PRECISION.....	77
52. PLATECONF.....	78
53. ZEROCONFIDENCERESULTS.....	79
PROPERTIES RELATED TO MEMORY HANDLING.....	80
54. HEAPFREEFREQ.....	80
PROPERTIES RELATED TO MAKE AND MODEL RECOGNITION (MMR).....	81
APPENDICES.....	88
CONFIDENCE LEVEL CALCULATION DETAILS AND EXAMPLE.....	88
SAMPLE CALCULATION.....	89
ZEROCONFIDENCERESULT VS GENERAL.....	90
CALCULATING THE MINIMAL BOUNDING RECTANGLE OF THE RETURNED LICENSE PLATE.....	92
RETRIEVING COUNTRY NAMES FROM RETURNED PLATE TYPE VALUES.....	93
COUNTRY AND STATE ID'S.....	93
RETRIEVING COUNTRY NAMES FROM RETURNED PLATE TYPE VALUES FOR LEGACY ENGINES.....	117
COUNTRY AND STATE ID'S FOR LEGACY ENGINES.....	119
RESULT STRUCTURE EXAMPLES.....	121
ABBREVIATIONS.....	125
DEFINITIONS.....	126
SYMBOLS.....	126
CONTACT INFORMATION.....	127

This document describes the properties (user parameters) of the CARMEN® software and the cmAnpr engine handler software module.

LIST OF THE PROPERTIES AND THEIR IMPORTANCE

PROPERTIES OF THE CMANPR ENGINE HANDLER MODULE		
anprname		

PROPERTIES OF THE CMANPR ENGINES		IMPORTANCE
Properties related to the identification of the current engine	datafile	low
Properties related to processing time	timeout	high
	timeout_wall	medium
	adapt_environment	medium
	recognitionmode	medium
	depth	low
Properties related to the geometry of plates	size	high
	size_max	high
	size_min	high
	slant	medium
	slant_max	medium
	slant_min	medium
	slope	medium
	slope_max	medium
	slope_min	medium
	charheightinmm	low
	platesize(f)	low
	platesize(f)_source	low
	xtoyres	low
Properties related to filtering outputs based on license plate characteristics	location	high
	read_adr_type	high
	read_base_type	high
	read_emptyadr_type	high
	read_license_plate_type	high
	read_without_type	high
	general	medium
	typeweight	medium
	colortype	low
	nchar_max	low
	nchar_min	low
Properties related to customization of results	gapospace	medium
	local_character_permutation	medium
	platetypeinfos	medium
	autotypemodification	low
	convert0toO	low
	countryname	low
	cyrillic_style	low
	local_character_conversion	low
	unicode_in_text	low

Properties related to color recognition	analyzecolors	low
	color	low
	whitebalance	low
Properties related to the position of license plates in input images	ROI	medium
	ROU	medium
	posfreq	low
	posfreqhalflife	low
	posfreqhistxs	low
	posfreqhistys	low
	posfreqweight	low
Properties related to image quality	contrast_min	medium
	gamma	low
Properties related to the calculation of the confidence level	confidencemode	medium
	confidencemode_x	low
	confidence_precision	low
	plateconf	low
	zeroconfidenceresults	low
Properties related to memory handling	heapfreefreq	medium

DESCRIPTION OF THE PROPERTIES OF THE CMANPR ENGINE HANDLER MODULE

1. ANPRNAME

NAME OF THE CURRENT ENGINE

The name of the current engine can be altered during runtime. Its value can be set according to the following scheme: "engine module name: property group". For example: cmanpr-eur-7.3.12.5_20Q2

Possible value: character string

Default value: varies with each engine release

Suggested value: not applicable

[Back to top](#)

POSSIBLE VALUES OF CMANPR ENGINE PROPERTIES

NAME (in alphabetical order)	POSSIBLE VALUES	DEFAULT VALUE
adapt_environment	{-1,0,1,2,3,4,...8,...16,...32,...64,...127}	engine dependent
analyzecolors	{0,1,2,4,5,8,9}	engine dependent
autotypemodification	{0,1}	engine dependent
charheightinmm	positive integers	empty
colortype	{0,1,2}	0
confidence_precision	positive integers	100
confidencemode	{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}	7
confidencemode_x	{replace "X" with number 0 - 15}	n/a
contrast_min	[0..255]	1
convertOtoQ	{0,1}	0
cyrillic_style	{0,1}	0
datafile	character string	n/a
depth	[0..500]	engine dependent
gamma	{0,1}	0
gapospace	{0,1,2}	0
general	{0,1,3,4,5,6,7,8,9,10,11,12,13,14,15}	engine dependent
heapfreefreq	{0,1}	0
local_character_conversion	{0,1,2,3}	0
local_character_permutation	{0,1,2,3}	0
location	varies with each regional engine	empty
nchar_max	positive integers	engine dependent
nchar_min	positive integers	engine dependent
plateconf	{0,1,2}	2
posfreq	character string	empty string
posfreqhalfife	[0..1048576]	1000
posfreqhistxs	[2..64]	16
posfreqhistys	[2..64]	16
posfreqweight	[0..100]	50
read_adr_type	{0,1}	engine dependent

read_base_type	{0,1}	engine dependent
read_emptyadr_type	{0,1}	0
read_license_plate_type	{0,1}	1
read_without_type	{0,1}	engine dependent
recognitionmode	{0..5}	0
ROI	polygon(s)	empty
ROU	polygon(s)	empty
size	positive integers	engine dependent
size_max	positive integers	engine dependent
size_min	positive integers	engine dependent
slant	[-400..400]	engine dependent
slant_max	[-400..400]	engine dependent
slant_min	[-400..400]	engine dependent
slope	[-400..400]	engine dependent
slope_max	[-400..400]	engine dependent
slope_min	[-400..400]	engine dependent
timeout	non-negative integers	engine dependent
timeout_wall	non-negative integers	0
typeweight	string	empty string
unicode_in_text	{0,1}	1
whitebalance	[0..100]	100
xtoyres	positive integers	100
zeroconfidenceresults	{0,1}	0

DESCRIPTION OF THE PROPERTIES OF THE CMANPR ENGINES

2. DATAFILE

NAME OF THE ENGINE'S DATA FILE

This property specifies the knowledge file (.dat) used by the engine. By default, the name of the datafile corresponds to the name of the engine.

For example, the **cmnpr-eur-7.3.15.67_22Q4** engine's datafile is **cmnpr-15.67-eur.dat**. The role of this property is to query the name of the datafile after initialization and to include this information in the application's log.

Possible values: character string
varies with each engine release; default property value can be queried by the

Default value: GetProperty() function

Suggested value: leave the default value

[Back to top](#)

PROPERTIES RELATED TO PROCESSING TIME

3. TIMEOUT

3.1. TIMEOUT

CPU TIME LIMIT

Sets the maximum working time for the CPU in milliseconds in which the module tries to find license plates.

You can always give the module sufficient time by using this timeout. However, this could result in longer runtimes if the system is busy.

The interval starts when `cm_findfirst()` is called.

At the end of this period, the engine tries to finish searching for new plates and any additional call of `cm_findnext()` will result no plates found. Zero timeout value means no time limit.

By setting the timeout value before the `cm_findnext()` call, the timing will be restarted and the evaluation lasts till the newly specified time interval.

Example:

If the value of the timeout is set to 500

The `cm_findfirst()` returns successfully after 200 ms. In this case after additional `cm_findnext()` call 300 ms would be available. However, if the timeout is set to 500 after `cm_findfirst()`, then 500ms would be available for further `cm_findnext()` calls.

Possible values: non-negative integers

Default value: varies with each regional engine; default property value can be queried by the `GetProperty()` function

Suggested value: around 1000 – on servers and desktop computers

5000 – on slower computers with less than 2GHz CPU clock speed

[Back to top](#)

3.2. TIMEOUT_WALL

REAL TIME LIMIT

Sets the length of the time interval in milliseconds in which the module tries to find license plates. You can ensure that the module won't run much longer than the desired time by using this `timeout_wall` property. However, the results could possibly be worse if the system is busy and the `timeout_wall` is set too low.

The interval starts when `cm_findfirst()` is called.

At the end of this period, the engine tries to finish searching for new plates and any additional call of `cm_findnext()` will result no plates found. Zero `timeout_wall` value means no walltime limit.

By setting the `timeout_wall` value before the `cm_findnext()` call, the timing will be restarted and the evaluation lasts till the newly specified time interval.

Example:

If the value of the `timeout_wall` is set to 500

The `cm_findfirst()` returns successfully after 200 ms. In this case after additional `cm_findnext()` call 300 ms would be available. However, if the `timeout_wall` is set to 500 after `cm_findfirst()`, then 500ms would be available for further `cm_findnext()` calls.

Possible values: non-negative integers

Default value: 0

Suggested value: equal or greater values than the value set for timeout

[Back to top](#)

USAGE

It is possible to use the `timeout` and the `timeout_wall` parameters together or just one or the other. If either timeout is reached, the engine tries to finish searching for new plates and any additional call of `cm_findnext()` will result no plates found. If both parameters are used, the `timeout` value should always be smaller than the `timeout_wall` value or equal to it. Otherwise it has no effect.

It can happen that in case of a busy processor the ANPR process takes a long time (2-4 seconds) even if the `timeout` is set to 1000 ms, because during this 2-4 seconds CARMEN® can only use the processor for around 1000 ms.

In either case, if CARMEN® has found a License Plate close to the set `timeout/timeout_wall` limit, the process will be finished as quickly as possible. So, if you are setting the `timeout_wall` property to 1000 ms the ANPR process cannot take much longer than 1000 ms, but if you are setting only the `timeout` property 1000 ms the ANPR process could take up to 2000-4000 ms depending on the CPU usage.

[Back to top](#)

4. ADAPT_ENVIRONMENT

ACCELERATED MODE BASED ON PHYSICAL LOCATION AND OTHER CHARACTERISTICS OF THE LICENSE PLATE

This property can substantially boost recognition speeds in case that this property is not set to '0'. The engine is capable of statistically adapting to type, location and geometrics of the license plates on the images.

Possible values:

1 – This allows the cmAnpr engine to adapt to its environment by applying a real-time statistical prediction algorithm to quickly assess the physical location of the particular deployment. The engine will begin to anticipate the license plate types of those countries/states that it encounters most frequently. With this feature enabled, it often takes only a few plate recognitions for the software to start delivering results up to twice as fast.

To turn off this feature, set value to 0.

2 – This enables [posfreq](#) and at the same time, [posfreqhalflife](#) is set to 1000. Collection of statistical information is started on the image divided to 16x16 equal sections (if this is not overwritten previously by unique [posfreq](#), [posfreqhalflife](#), [posfreqhistxs](#), [posfreqhistys](#) settings) This works as before, as it would have been set individually by those 2 parameters.

4, 8, 16, 32, 64 – By setting these values, collection of statistical information and adaptation includes also some geometric parameters and the contrast of the license plates. Continuous runtime adaptation of the engine makes recognition more accurate and faster.

These geometric parameters are: [size](#), [size_min](#), [size_max](#), [slant](#), [slant_min](#), [slant_max](#), [slope](#), [slope_min](#), [slope_max](#), [contrast_min](#) parameters. This makes manual setting of those parameters unnecessary.

4 – height of the characters

8 – width of the characters

16 – slant

32 – slope

64 – contrast

By setting the adapt_environment parameter to **"-1"**, all the collection and adaptation of all the information – like license plate location, character size (height, width), slant, slope, contrast is turned on.

Please read the notes on the next page!

 Important!**Notes for using adapt_environment:**

Timeout: Avoid setting this value too low! Some engines have been trained for a large plate type variety, but a specific deployment may only encounter a few different country/state plates on a regular basis (e.g. Arabic engine used in Bahrain). Although the adapt_environment feature may dramatically improve recognition speeds for those country/state plates that are frequent at a specific location, "infrequent" plate types will still require about the same recognition times as without this feature enabled. Consequently, using a [timeout](#) value that is too low (e.g. based on the mean recognition time calculated when this acceleration feature is enabled) may cause the system to skip "infrequent" plates without providing a result, as the recognition cycle would time out before being able to provide a result.

Recognition accuracy: Our tests indicated that enabling the adapt_environment feature does not negatively impact text reading accuracy, while at the same time substantially increasing recognition speeds. However, the accuracy of country/state identification is more sensitive to this feature. Consequently, this feature might present some issues that would require careful selection of an appropriate [timeout](#) value or a secondary screening for occasional false positives of country/state identification. For example, no result or misread might be provided for a previously recognized license plate, following a sequence of different plate images. An issue like this, could be the result of either a low [timeout](#) value (read previous note), or a unique anomaly, where the internal statistical analysis expected a "frequent" country/state plate at the specific location, while the actual plate was a similar but "infrequent" type.

Error reporting: Using adapt_environment makes it difficult to reproduce errors or misreads, because the engine continuously changes recognition parameters. Before reporting an error, also process the image with this acceleration feature disabled to help identify whether it is an acceleration-related issue.

Possible values: Possible values: {-1,0,1,2,3,4,8,16,32,64 – and all combinations of the positive numbers}

Default value: -1 – but varies by the currently set recognition mode

Suggested value: -1 – if the images are collected in the same physical location

[Back to top](#)

5. RECOGNITIONMODE

(available from version 7.3.11.242)

Some engines contain more than one recognition possibility. You can check the actual values and options of the current engine as explained below.

The `GetProperty(„recognitionmodes“)` function will give you back the possible values with their number and name:

0 - Fix_Normal

1 - Fix_Quick

2 - Mobile_Normal

3 - Mobile_Quick

4 - MovingFix_Normal

5 - MovingFix_Quick



Recognitionmodes from 2 to 5 are available from 7.3.14.127.

The ANPR engine can be used in different ways depending on the type of use.

For an installed fixed camera, the Fix..., for a camera attached to a moving vehicle, the MovingFix..., while for a completely general use (e.g., mobile phone) the Mobile... parameter settings are recommended.

If your ANPR engine supports it you can also choose between "Normal" and "Quick" modes.

The Normal mode provides the best recognition rate, while the Quick mode is significantly faster, albeit at the cost of slightly less correct readings.

The `GetProperty(„recognitionmode“)` function will give you back the name of the actual recognitionmode as a string.

With `SetProperty(„recognitionmode“, 0)` or `SetProperty(„recognitionmode“, „Fix_Normal“)` you can set a desired recognitionmode.

Please note that this parameter is not saved into gxsd.dat.

Possible values: {0..5}

Default value: 0

[Back to top](#)

6. DEPTH

HEURISTIC SENSITIVITY



Note

This property is deprecated/not available from engine version 7.3.15.128.

cmAnpr engines with country/state recognition use more sophisticated algorithms than general ones, which not only provides country/state recognition but also improves text accuracy in most cases. If the recognition time is too long for the application, the depth of search may be lowered using this parameter; although the accuracy of both text and country/state reading will be affected. If the application is not time-sensitive, then the depth value may be increased to get higher recognition accuracy. [1..300]: depth of the heuristic sensitivity.

Possible values: [1..300].

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value

[Back to top](#)

PROPERTIES RELATED TO THE GEOMETRY OF PLATES

7. SIZE

AVERAGE HEIGHT OF THE CHARACTERS

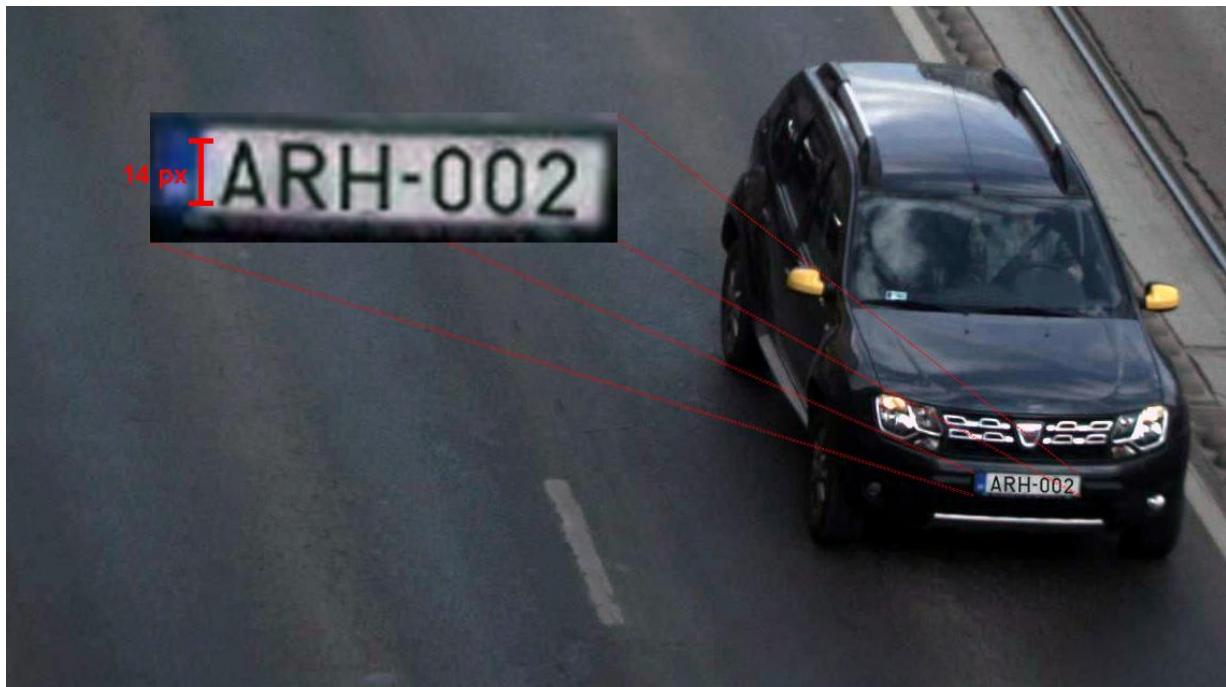
The average height of the number plate characters in the image in pixels.

Possible values: positive integers (between 10 and [size_max](#))

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: [20..50] (between 20 and 50), depending on the character heights in the input images

[Back to top](#)



8. SIZE_MAX

MAXIMUM HEIGHT OF CHARACTERS

The maximum height of the number plate characters in the image in pixels.

Possible values: positive integers greater or equal to the actual [size](#) value or '-1'

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value

Note

If you set the value of [size_max](#) to '-1' the ANPR engine will try to find a number plate on the image no matter its size. We recommend using this setting if you are absolute unsure about the size of the characters on the plates occurring on the image.

Note

If you set the value of [size_max](#) property above 80 or to '-1' the ANPR running time could increase significantly so only use these values if you have to.

[Back to top](#)

9. SIZE_MIN

MINIMUM HEIGHT OF THE CHARACTERS

The minimum height of the number plate characters in the image in pixels.

Possible values: positive integers (between 10 and the actual [size](#) value)

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value

[Back to top](#)

Note

$$\text{size_min} \leq \text{size} \leq \text{size_max}$$

The [size](#) value has to be equal to or greater than [size_min](#) and equal to or less than [size_max](#) otherwise the engine returns no data.

If you set [size_max](#) to -1, you don't and shouldn't have to adjust [size](#) and [size_min](#) accordingly. So, if you generally have characters in the 300 pixel range but sometimes the engine encounters smaller and bigger ones as well you should set size to 300, [size_min](#) to the minimum size that occurs on the images and size max to -1.

10. SLANT

AVERAGE SLANT OF THE NUMBER PLATE

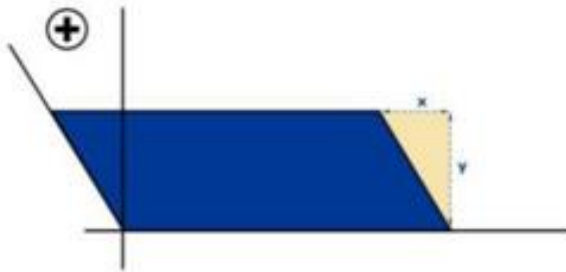
The average slant of the number plate characters in the image. This value is represented in percent (%) and it is positive if the vertical axis of the characters slants to the left viewing from bottom to top.

Possible values: integer numbers (between -100 and 100)

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value

positive SLANT



negative SLANT



$$\text{Slant} = \frac{x}{y} * 100$$

[Back to top](#)

11. SLANT_MAX

MAXIMUM SLANT OF THE NUMBER PLATE

The maximum slant of the number plate characters in the image. This value is represented in percent (%) and it is positive if the vertical axis of the characters slants to the left viewing from bottom to top.

Possible values: integer numbers (between the actual [slant](#) value and 100)

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value

[Back to top](#)

12. SLANT_MIN

MINIMUM SLANT OF THE NUMBER PLATE

The minimum slant of the number plate characters in the image. This value is represented in percent (%) and it is positive if the vertical axis of the characters slants to the left viewing from bottom to top.

Possible values: integer numbers (between -100 and the actual [slant](#) value)

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value



Note

$$\text{slant_min} \leq \text{slant} \leq \text{slant_max}$$

The [slant](#) value has to be equal to or greater than [slant_min](#) and equal to or less than [slant_max](#) otherwise the engine returns no data.

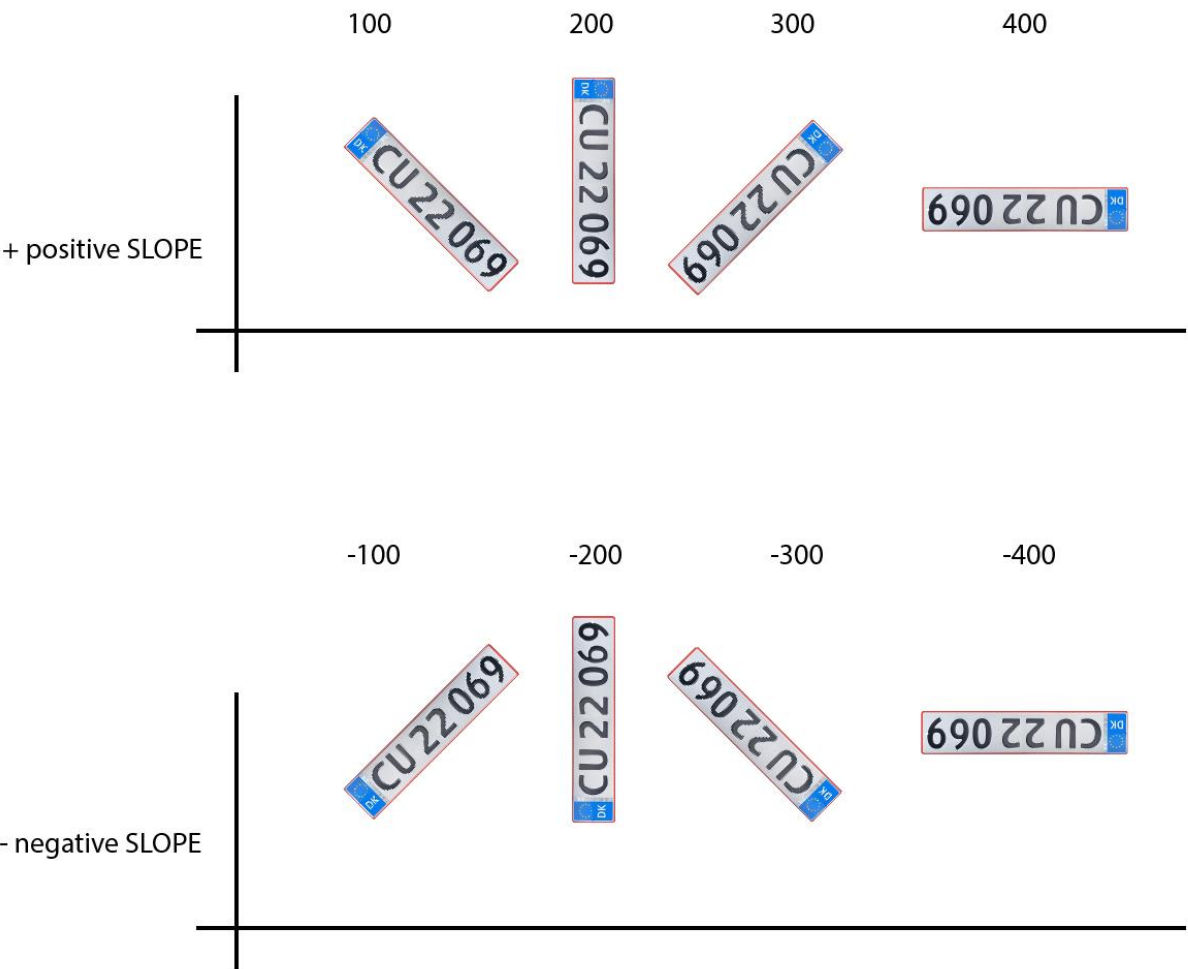
[Back to top](#)

13. SLOPE

AVERAGE SLOPE OF THE NUMBER PLATE

The average slope of the number plate in the image. This value is represented in percent (%) and it is positive if the number plate is rotated clockwise and negative if it is rotated counter clockwise.

- Possible values:** integer numbers (between -400 and 400)
- Default value:** varies with each engine release; default property value can be queried by the `GetProperty()` function
- Suggested value:** leave the default value
- [Back to top](#)



14. SLOPE_MAX

MAXIMUM SLOPE OF THE NUMBER PLATE

The maximum slope of the number plate in the image. This value is represented in percent (%) and it is positive if the number plate is rotated clockwise and negative if it is rotated counter clockwise.

Possible values: integer numbers (between the actual [slope](#) value and 400)

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value

[Back to top](#)

Note

If you set the value of `slope_max` above 100 the ANPR processing time could increase significantly so only use these values if you have to.

15. SLOPE_MIN

MINIMUM SLOPE OF THE NUMBER PLATE

The minimum slope of the number plate in the image. This value is represented in percent (%) and it is positive if the number plate is rotated clockwise and negative if it is rotated counter clockwise.

Possible values: integer numbers (between -400 and the actual [slope](#) value)

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value

[Back to top](#)

Note

If you set the value of `slope_min` below -100 the ANPR processing time could increase significantly so only use these values if you have to.

Note

$$\text{slope_min} \leq \text{slope} \leq \text{slope_max}$$

The slope value has to be equal to or greater than [slope_min](#) and equal to or less than [slope_max](#) otherwise the engine returns no data.

16. CHARHEIGHTINMM

POSSIBILITY TO SET THE CHARACTER HEIGHT IN MM

You can set the typical height of the characters on the plate in mm. The engine uses this information to return more precise statistical platesize and platesizef values.

For plate types we have the platesize data in our database this parameter has no effect.

Possible values: positive integer numbers

Default value: empty

the typical size of the characters of the license plates the engine encounters in

Suggested value: your setting

[Back to top](#)

17. PLATESIZE(F)

SIZE OF THE PLATE

The engine can return the physical dimensions of the license plate it found measured in mm.

The returned string has a **width*height** structure.

Property value can be queried like this:

- GetProperty("platesize").
- GetProperty("platesizef").

"platesize" will return in integers in the string value, like: **520*110**

"platesizef" will return in floats in the string value, like: **519.89*110.41**

You can only get this property; it is not possible to set it!

[Back to top](#)

18. PLATESIZE(F)_SOURCE

source of the platesize(F) value

The exact measurements of many plate types are known to us, but there are also many plate types for which we lack precise measurement information.

Possible values:

- database (exact)
- statistical (estimated, based on the size of the characters on the plate)
- no_data

HOW TO USE PLATESIZE(F) AND PLATESIZE(F)_SOURCE PROPERTIES

- `GetProperty(„platesize“)` – in this case you will get back the information for the type of the last result
- `GetProperty(„platesizef_typevalue_source“)` – in this case you will get back the information for an exact type

Examples:

1. `GetProperty(„platesize“)`

In this case you will get back the “platesize” in integers in the string value for the type of the last result.

2. `GetProperty(„platesizef_212147“)`

In this case you will get back the “platesizef” in floats in the string value for type 212147.

3. `GetProperty(„platesizef_source“)`

In this case you will get back the “platesizef” source information for the type of the last result.

4. `GetProperty(„platesize_212147_source“)`

In this case you will get back the “platesize” source information for type 212147.

You can only get this property; it is not possible to set it!

[Back to top](#)

19. XTOYRES

THE RATIO OF HORIZONTAL AND VERTICAL DIMENSIONS OF PLATES

X to Y resolution – only for analog input!

This value is represented in percentages (%). This parameter uses 2 values for calculation: the ratio of horizontal and vertical resolution of the plate in the image and the real life horizontal and vertical size ratio of the plate.

This parameter can be set manually using images that contain plates of the same type. Calculating the average of the ratio "*r*" of the width and the height of the plates on images and by calculating the "*R*" ratio of the real width and height of the actual plate:

$$xtoyres = [(100 \cdot r) / R + 0.5]$$

Example 1 (metric):

Physical dimensions: The width of Hungarian plates is 51 cm, the height is 11 cm, so the ratio of the width and the height is $51/11=4.6363$.

Dimensions on images: On 100 images with Hungarian plates, the average ratio of the width and the height of plates is 5.25.

In this case, using the formula:

$$xtoyres = ((100 \cdot 5.25) / 4.6363) + 0.5 = \sim 114.$$

Example 2 (imperial):

Physical dimensions: The width of North American standard plates is 12 inches, the height is 6 inches so the ratio of the width and the height is $12/6=2$.

Dimensions on images: On 100 images with USA plates, the average of the width and the height of plates is 1.79. In this case, using the formula:

$$xtoyres = ((100 \cdot 2) / 1.79) + 0.5 = \sim 112.$$

Note

If you have regular digital camera that does not change the ratio of the width and the height of the objects, simply use 100.

This property has a role only if there is significant distortion in the dimensions of the objects, for example in case of half-frames taken by analog cameras.

Possible values: positive integers

Default value: 100

Suggested value: 100

(Zero value setting of *xtoyres* means automatic re-setting of *xtoyres* to 100.)

[Back to top](#)

PROPERTIES RELATED TO FILTERING OUTPUTS BASED ON LICENSE PLATE CHARACTERISTICS

20. LOCATION

WHERE THE IMAGES ARE COMING FROM

(available from version 7.3.14.13)

If you set this parameter the ANPR engine will modify its search based on the type of plates that generally occurs at the set location.

So, it modifies the weights of the types just like the [typeweight](#) parameter, but automatically and typically in a more detailed way than what you would do yourself.

It can also change some other settings depending on the set location to improve the recognition rate.

Usage:

-> SetProperty(„location” , "HUN") -> sets the location to HUN (Hungary)

If you try to set a location not known to the engine it will return an unknown property error.

-> GetProperty(„location”) -> gets the actual location

If the location hasn't been set prior it will return "unset".

-> GetProperty(„location_all”) -> gets the list of all locations known by the engine

-> SetProperty(„location” , „0”) -> resets the location, so the engine changes all of the [typeweights](#) and other location related settings back to the default

 Note

This property is a better alternative to [typeweight](#) in most cases so we encourage you to use it instead of [typeweight](#). It is easier to use and produces better results.

It is also important to note that the location and typeweight parameters can be used simultaneously.

What is important to note is that setting the location will reset all the previously set [typeweights](#), so, if you would like to manually set some [typeweights](#) in addition to using the location parameter you should set the location first and set the additional [typeweights](#) after.

Be aware that setting the location also modifies the [typeweights](#) so in most cases you do not (or even should not) have to set them additionally.

Possible values: varies with each regional engine

Default value: empty string

Suggested value: the present site where the images were taken

[Back to top](#)

21. READ_ADR_TYPE

ENABLE / DISABLE THE READING OF ADR PLATES

(available from version 7.3.16.190)

You have the option to specify via this property whether you would like to read ADR plates from the images.

Possible values: {0,1}

Default value: varies with each engine

Suggested value: depending on your needs

[Back to top](#)

22. READ_BASE_TYPE

enable / disable the reading of exact typeless, so-called base_TYPE

(available from version 7.3.16.190)

You have the option to specify via this property whether you would like to read plates with BASE_type from the images. This allows the engine to potentially provide the text of the license plate with a type aligned with the country and/or the state of the plate (the so called BASE_type). Typically, it is simpler for the engine to ascertain the country/state information rather than the precise type of the plate. Nevertheless, in the majority of instances, the engine will still provide the exact type of the identified plate.

Possible values: {0,1}

Default value: varies with each engine

Suggested value: depending on your needs

[Back to top](#)

23. READ_EMPTYADR_TYPE

ENABLE / DISABLE THE READING OF EMPTY-ADR PLATES

(available from version 7.3.16.190)

You have the option to specify via this property whether you would like to read "Empty ADR" plates from the images, subject to availability in the engine.

Possible values: {0,1}

Default value: 0

Suggested value: depending on your needs

[Back to top](#)

24. READ_LICENSE_PLATE_TYPE

enable / disable the reading of THE KNOWN LICENSE plates

(available from version 7.3.16.190)

You have the option to specify via this property whether you would like to read the known license plates from the images.

Possible values: {0,1}

Default value: 1

Suggested value: depending on your needs

[Back to top](#)

25. READ_WITHOUT_TYPE

ENABLE / DISABLE THE READING OF UNIDENTIFIED LICENSE PLATE TYPES

(available from version 7.3.16.190)

You have the option to specify via this property whether you would like to read unknown license plates from the images. This enables the engine to return plates solely with the text of the license plate, without a recognized (and known) type.

Possible values: {0,1}

Default value: varies with each engine

Suggested value: depending on your needs

[Back to top](#)


26. GENERAL

SELECTING GENERAL OR SPECIAL RESULT DELIVERY MODE

This property lets you select the desired reading modes of license plates, ADR plates, EADR plates (Empty ADR, ADR plates without text), BASE types and plates with unknown type (type refers to cmNP::type).

This property can filter 5 kinds of LPR results:


1. Text results with unidentified license plate type (type=0)
2. Results with identified license plate type (type range [0..997000])
3. Results with identified ADR plate type (type range [997000..997999])
4. Results with EmptyADR plate type (type=999980..999999)
5. Results with BASE plate type (type = XYZ000)

 Note

BASE type: this type allows the engine to potentially provide the text of the license plate with a type aligned with the country and/or the state of the plate. Typically, it is simpler for the engine to ascertain the country/state information rather than the precise type of the plate. Nevertheless, in the majority of instances, the engine will still provide the exact type of the identified plate.

For example:

101000 means that the result is a Hungarian plate, but a new one or an uncertain exact type.

 Note

Value 2 is intentionally omitted from the below table. If general is set to 2, no result will be returned!

The following table shows what kinds of results are returned for each possible value of general property:

values of general	Expected type of the result				
	unidentified license plate type (type=0)	IDENTIFIED license plate type (0<type<997000)	ADR plate (997000<=type<=997999)	EmptyADR plate (999980<=type<=999999)	BASE plate (type = XYZ000)
0		x			
1	x	x			
2 (do not use)					
3	x				
4		x	x		
5	x	x	x		
6			x		
7	x		x		
8		x		x	
9	x	x		x	
10				x	
11	x			x	
12		x	x	x	
13	x	x	x	x	
14			x	x	
15	x		x	x	
16		x			x
17	x	x			x
18					x
19	x				x
20		x	x		x
21	x	x	x		x
22			x		x
23	x		x		x
24		x		x	x
25	x	x		x	x
26				x	x
27	x			x	x
28		x	x	x	x
29	x	x	x	x	x
30			x	x	x
31	x		x	x	x

Possible values: {0,1,3,4,5,6,7,8,9,10,11,12,13,14,15}

Default value: varies with each engine release; default property value can be queried by the GetProperty() function

Suggested value: 1

To see what is the connection between [general](#) and [zeroconficenceresults](#) property, please check [this](#) chapter.

Please note, that you can control what kind of license plates you would like to read with the engine by setting the **read_*_type** properties one by one. They are setting the general property directly.

[Back to top](#)

27. TYPEWEIGHT

PRIORITIZING OR OMITTING LICENSE PLATE TYPES OR COUNTRIES/STATES/PROVINCES

(available from version 7.2.8.46)

CARMEN® engines recognize the country based on the so-called license plate types. Two license plates fall into the same type if they differ only in their texts keeping the syntax, layout, font type, delimiters, and all graphical elements. The above implies that each country/state has several different license plate types. Each type is assigned to an ID, and the ID's are arranged into intervals for easier handling of countries/states. These intervals are defined as follows:

From **ccode*1000** to **ccode*1000+999**; where ccode stands for the "country code" or "state code" as listed in chapter "[Country and State IDs](#)"

For example:

The country code of Hungary is 101. The valid Hungarian types are between 101000 and 101999. By default, all types are handled equally by the engine when the type (country/state) is recognized. The engines are usually released for certain regions (continents), which may include several countries/states from a larger area. The more countries/states supported by the engine, the more types it needs to handle. If the site where the engine is used is not at a border-crossing point, 90% of known plate types never occur. In order to give users the possibility to handle this, the engine assigns weight values to each type.

By configuring these, the priority of certain types or countries/states can be changed. This leads to a more specific engine configuration, which can perform more efficiently in that site than the general settings.

The default weight is 1000 for each type. The weight values can be integer numbers from minus 10.000 to positive 10.000.

How to enable / disable certain types, countries, or states?

There are two factors of the weight values:

- the absolute value
- and the sign

The absolute value defines the priority of a specific type among the set of all known license plate types. The higher the absolute value, the higher the priority of the type.

The sign defines whether or not the type should be returned by the engine. Negative weights will disable the type, and upon finding, the engine will not return it.

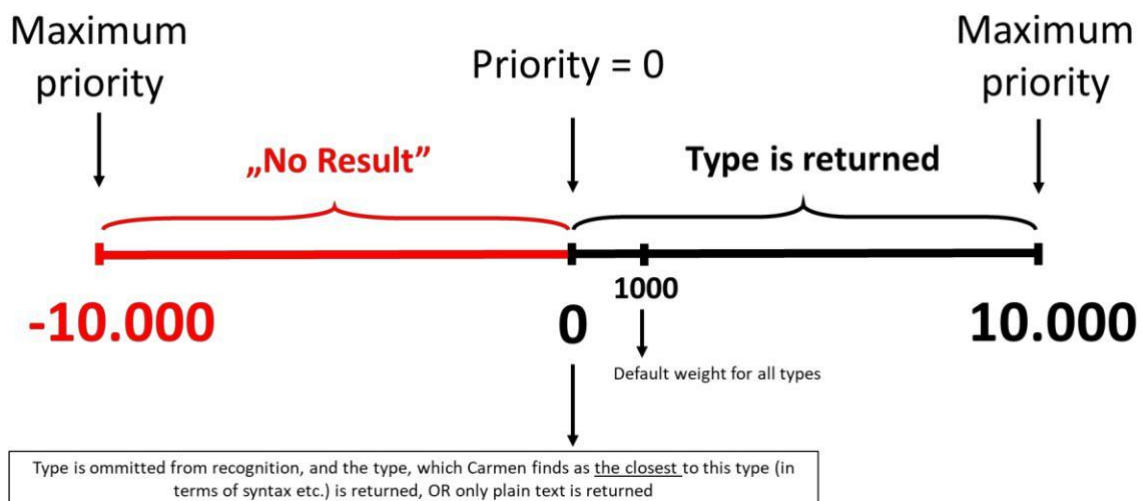
Possible values: string

Default value: empty string – filtering is disabled; all types are considered with the same weight

Suggested value: raise the weight of the desired types, or simply leave the default value to skip this feature

Weight value range: -10.000 to 10.000

Default weight for all type is 1000.



For example:

Assume that in a certain border crossing point the trucks have two license plates, one from each country. The project requires reading only one type, but the engine is trained for both. In this case, a negative value with high absolute value can disable the unwanted type range and prevent them from being returned.

Weight value examples (after OCR part of the process is finished):

"-10000" to "-1": type recognized but result is dropped

"0": omitted from the type set and: matched to closest similar type or plain text returned or no result returned

"1" to "10000" – recognized and returned

Advantages of using typeweight

In case of regions where there are many different license plate types, the engine has to choose the type of the license plates from a large set, even thousands of different versions. By using typeweight, the engine can be configured to be more specific for a smaller area. This can improve both the accuracy and the processing time.

Syntax

The string value of typeweight should contain ordered pairs of type (type ranges) and weight values. For example:

typeweight="101,-1000" disables all license plate types of Hungary

typeweight="1,-1000" disables all license plate types of Europe

typeweight="101,-1000;122,-1000" disables all license plates of Hungary and Germany

typeweight="1,500;101,1000" sets the weight of all Hungarian plates to 1000, and decreases the weight of all other European types to 500

typeweight="123,0" omits all Belgian license plate types from the recognition, meaning the plates that seem to appear as Belgian plates are returned matched with the type closest to the Belgian syntax

typeweight="101,5000" increases the weight of all license plate types of Hungary.

Note

The specified weights are applied one after the other. So in case of typeweight="1,500;101,1000", all European license plates are set with weight of 500, and after that the types of Hungary are set with weight=1000.

Examples

1. European engine used in Germany, for monitoring and categorizing German vehicles. License plates from foreign countries are not relevant.
For an efficient solution, all countries of Europe should be ignored except for Germany. Solution: set the weight of all European types to -1000 and then set a high positive value for Germany:
`typeweight="1,-1000;122,1000"`
2. USA engine used in Florida, the occurrence of vehicles from neighbour states are expected, but license plates from states and even countries that are far away are very rare. Solution: initialize two instances of the engine, and use one of them as default which can detect FL, SC, GA, AL, MS, LA license plates and the other as secondary which can cope with the rare cases.
For the default configuration:
`typeweight="5,0;550,1000;548,1000;549,1000;530,1000;533,1000;532,100"`
ignores all USA types except for the above-mentioned states (see state code list)
For the secondary configuration:
`typeweight="550,0;548,0;549,0;530,0;533,0;532,0"`
ignores the above covered states and therefore the engine can focus on the others

Note

In this kind of use cases, we are suggesting to use location property, which is much easier and you can get similar, or most of the time even better results.

Note

This configuration will ensure higher speed for most of the cases, while the rare license plates can also be detected with a slightly longer extra processing time.

Relation with [general](#)

Note that the [general](#) property is also related with enabling/disabling certain types of license plates.

Note

The function `GetProperty()` can return the current value of the **typeweight** property.

By defining a new string for typeweight all previous typeweight settings will be reset.

[Back to top](#)

28. COLORTYPE

COLOR SCHEME ON THE PLATE (BACKGROUND VS CHARACTER COLORS)

Module property to exclusively read plates with either standard color schemes, or inverted color schemes, or both. (Background vs character colors)

- 0: Module searches for all plates in the image regardless of the type of color scheme.
- 1: Module searches for plates only with dark characters on light background.
- 2: Module searches for plates only with light characters on dark background.

Possible values: {0,1,2}

Default value: 0

Suggested value: 0

[Back to top](#)

29. NCHAR_MAX

MAXIMUM NUMBER OF CHARACTERS IN THE PLATE

The maximum number of characters in the license plate. The engine does not return plates on which the number of characters is more than the nchar_max value **unless** they have an identified type (cmNP::type > 0).

Possible values: positive integers (bigger than [nchar_min](#))

Default value: varies with each engine release; default property value can be queried by the GetProperty() function

Suggested value: leave the default value

[Back to top](#)

30. NCHAR_MIN

MINIMUM NUMBER OF THE CHARACTERS IN THE PLATE

The minimum number of characters in the license plate. The engine does not return plates on which the number of characters is less than the nchar_min value **unless** they have an identified type (cmNP::type > 0).

Possible values: positive integers (between 0 and [nchar_max](#))

Default value: varies with each engine release; default property value can be queried by the GetProperty() function

Suggested value: leave the default value

[Back to top](#)

PROPERTIES RELATED TO CUSTOMIZATION OF RESULTS

31. GAPTOSPACE

GAPS BETWEEN CHARACTERS

If this parameter is set to a positive value, the module searches for gaps between characters that can be spaces. Space characters are inserted into these positions in the **text** and **wtext** fields of the result structure. If the value is zero, no spaces will appear in the result.

Meaning of the possible values:

0 – the text is returned in one block (no spaces will appear in the result)

1 – spaces are inserted instead of gaps into **text** and **wtext** members of the **cmNP** structure. The cmAnpr engine calculates the positions of the gaps based on the frames of the characters. If the gap is wide enough, spaces are inserted into **text** and **wtext**

Note

In case of plates with both Arabic and corresponding Latin characters [gaptospace=1](#) won't work properly unless the [local_character_permutation](#) is set to 3.

If only the Arabic or only the Latin character sequences are wanted using [gaptospace=2](#) is suggested instead.

2 – This feature is only available for engine version 7.2.8.6 and higher. In this case, spaces are inserted into the corresponding positions by default. But by the following parameters, the user can define the mode of the return of the text:

symbolnewrow	beginning of a new row
symbollowcharstart	beginning of the small characters
symbollowcharend	end of the small characters
symbolarms	coat of arms
symbolhyphen	hyphen
symbolspace	space
symbolverticalline	vertical line

First, the engine identifies the country/state of the license plate, which defines the layout of characters and symbols. The symbols are then substituted with the ASCII characters specified in the corresponding property as shown in the table above.

The following license plate is a sample, which contains the most possible text positions:



If the gaptospace parameter is set to 0 the returned text will be: AB123DF234GH

- if it is set to 1, the text will be: AB 123 DF 234 GH
- if it is set to 2 the text will be the same, but by the following parameters the returned text can be defined more precisely:

```
symbolnewrow=47
symbollowcharstart=40
symbollowcharend=41
symbolarms=42
symbolhyphen=45
symbolverticalline=124
symbolspace=95
```

With these settings the user receives the following text: */AB|(1/2/3)DF_234-GH

Sample gxsd.dat entry:

```
<cmanpr-7.2.8.8>
...
  <gapospace value="2"/>
  <symbolnewrow value="47"/>
  <symbolowcharstart value="40"/>
  <symbolowcharend value="41"/>
  <symbolarms value="42"/>
  <symbolhyphen value="45"/>
  <symbolverticalline value="124"/>
  <symbolspace value="95"/>
...
</cmanpr-7.2.8.8>
```

Possible values: {0,1,2}

Default value: 0

Suggested value: 2 – if the geometry of the license plate needs to appear in the **text** and **wtext**

[Back to top](#)

32. PLATETYPEINFOS

RETRIEVING VARIOUS TYPE OF INFORMATION ABOUT PLATES

(available from engine version 7.3.12.193)

This property enables the user to directly retrieve all kind of information for the recognized images or even for an exact plate type.

These are the information which can be get currently:

- countryname (same as the already existed [countryname](#) property result: „US-TX“)
- CountryS (Country Short, official 3 letter ISO code: „USA“)
- CountryL (Country Long, the full name of the country: „United States Of America“)
- StateS (State Short: „TEX“)
- StateL (State Long: „Texas“)
- category (License Plate category: „TAXI“)

HOW TO USE THIS PROPERTY

- `GetProperty(„platetypeinfos_informationname“)` – in this case you will get back the information for the type of the last result
- `GetProperty(„platetypeinfos_typevalue_informationname“)` – in this case you will get back the information for an exact type

Examples:

1. `GetProperty(„platetypeinfos_category“)`



Note

Please note that this command will return the value in text for the actual read.

2. `GetProperty(„platetypeinfos_category_value“)`



Note

Please note that this command will return the numerical value for the current read.

3. **GetProperty(„platetypeinfos_StateL“)**
4. **GetProperty(„platetypeinfos_212147_category“)** -> The result will be „TAXI“

Note

Please note that an exact type can contain multiple categories, and in this case, the 4th command prints out all the possibilities separated by ','.

Similarly, if the category for the current read (1st command) contains multiple values, the category could be POLICE,MOTORCYCLE for example.

Possible outcome of the requests:

Result	Meaning
„Unknown property“	Wrong string was used after "platetypeinfos"
„No type result“	Type was not given back by the engine
„Invalid type“	Type does not exist
„Valid type, no info“	Type is valid, but the requested info is not available in this engine
"N/A"	In case of category, it means that for the resulting type the category is not yet recorded
Valid result	Type is valid and the information is available

HOW TO MODIFY THIS PROPERTY

It is possible to personalize the information if you would like to see something else than the default values. You can do it with the following command:

SetProperty(„platetypeinfos_informationname_originalname“, „newname“).

Example:

SetProperty(„platetypeinfos_category_TAXI“, „CAB“) -> after this command if the category would be „TAXI“ then the engine will give back the result as „CAB“

If you would like to set them back, you can do it one by one with the above command, or there is an option to set back all the default values at once:

SetProperty(„platetypeinfos“, „default“).

Please read the possible category values on the next page.

Note

Please note that from the below list, CARMEN® will return the requested values in text by using the following command: **GetProperty(„platetypeinfos_category“)**.

If the user would like to retrieve the numerical value, it can be queried using the following command: **GetProperty(„platetypeinfos_category_value“)**.

To check the possible values please check the following pages.

Possible values of the category information:

- 1 - UNKNOWN
- 2 - COMMON
- 3 - CUSTOM
- 4 - TAXI
- 5 - DIPLOMATIC
- 6 - EXPORT
- 7 - OLD_TIMER
- 8 - MOTORCYCLE
- 9 - PUBLIC_TRANSPORT
- 10 - COMMERCIAL
- 11 - POLICE
- 12 - GOVERNMENT
- 13 - PROBATION
- 14 - DRIVING_SCHOOL
- 15 - TRAILER
- 16 - ARMY
- 17 - CONSULAR
- 18 - EQUIPMENT
- 19 - SECURITY_FORCES
- 20 - PRIVATE_TRANSPORT
- 21 - UNDER_EXPERIMENT
- 22 - UN
- 23 - LIMOUSINE
- 24 - TRANSIT
- 25 - TEMPORARY
- 26 - VETERAN
- 27 - UNIVERSITY
- 28 - DISABLED
- 29 - TRACTOR
- 30 - BUS
- 31 - SCHOOL_BUS
- 32 - AMBULANCE
- 33 - FIREFIGHTER
- 34 - TRUCK
- 35 - TOW_TRUCK
- 36 - QUAD
- 37 - NATIONAL_PARK
- 38 - ELECTRIC
- 39 - PICKUP_TRUCK
- 40 - VAN
- 41 - LIGHT_GOODS_VEHICLE
- 42 - HEAVY_GOODS_VEHICLE
- 43 - GENDARMERIE
- 44 - IMPORT

- 45 - ROYAL
- 46 - MUNICIPAL
- 47 - BORDER_GUARD
- 48 - CUSTOMS
- 49 - TRANSPORT
- 50 - TRADE
- 51 - DIPLOMATIC_PERSONNEL_WITH_NO_DIPLOMATIC_IMMUNITY
- 52 - NON-DIPLOMATIC_STAFF
- 53 - RED_CROSS
- 54 - SERVICE_STAFF
- 55 - TECHNICAL_STAFF
- 56 - EMBASSY
- 57 - CIVIL_PROTECTION
- 58 - NAVY
- 59 - AIR_FORCE
- 60 - GROUND_FORCES
- 61 - FINANCIAL_POLICE
- 62 - MINISTRY
- 63 - TOURIST
- 64 - INTERNATIONAL_ORGANIZATIONS
- 65 - MINIBUS
- 66 - RICKSHAW
- 67 - SUV
- 68 - SALVAGE_YARD
- 69 - REPAIR_TOWING
- 70 - SHERIFF
- 71 - EXEMPT
- 72 - PARCEL_DELIVERY
- 73 - POWER_COMPANY
- 74 - FUNERAL_SERVICES
- 75 - RAILWAYS
- 76 - POLICE_JUSTICE
- 77 - POST
- 78 - CITY_ADMINISTRATOR
- 79 - SPORTSCAR
- 80 - SLOW_MOVING_VEHICLE
- 81 - UNDER_CUSTOMS_CLEARANCE
- 82 - BICYCLE_CARRIER
- 83 - MOPED
- 84 - NATIONAL_TAX_AND_CUSTOMS_ADMINISTRATION
- 85 - OVERSIZED

Possible values of special categories by countries / states:

201	- EMIRI_GUARDS
202	- APPORTIONED
203	- COMBINATION
204	- DEALER
205	- DISTRIBUTOR
206	- TRUCK_TRAILER
207	- SEMI_TRAILER
208	- PERMANENT_TRAILER
209	- WEIGHTED
210	- UNDER_10000_LBS
211	- UNDER_26000_LBS
212	- LIVERY
213	- INTERSTATE
214	- STATE_OWNED
215	- FOR_HIRE
216	- TLC
217	- AGRICULTURE
218	- CAMPER
219	- PHYSICIAN
220	- DENTIST
221	- PHARMACIST
222	- ARCHITECT
223	- ENGINEER
224	- REPAIR
225	- TRANSPORTER
226	- VOLUNTEER_AMBULANCE_SERVICE
227	- VOLUNTEER_FIREFIGHTER
228	- HISTORICAL
229	- OFFICIAL
230	- PHYSICIAN_THERAPIST
231	- PSYCHOLOGIST
233	- CHIROPRACTOR
234	- PRESS
235	- CIVIL_AIR_PATROL
236	- JUDGE
237	- PURPLE_HEART
238	- PHYSICIAN_ASSISTANT
239	- SURVIVORS_OF_THE_SHIELD
240	- OLYMPIC_SPIRIT
241	- EMERGENCY_MEDICAL_TECHNICIAN
242	- ARMY_NATIONAL_GUARD
243	- SUPREME_COURT
244	- TRIBOROUGH_BRIDGE_AND_TUNNEL_AUTHORITY
245	- NURSE



246	- VISITING_NURSE
247	- VISION_SPECIALIST
248	- COMMUTER_VAN
249	- MOBILITY_ASSISTANCE_VEHICLE
250	- FIRST_AIDER
251	- PARAMEDIC
252	- VAN_POOL
253	- LAW_ENFORCEMENT
254	- FARMER
255	- RETIRED
256	- VEHICLE_MANUFACTURER
257	- FRATERNAL_ORDER_OF_POLICE
258	- NURSE_PRACTITIONER
259	- MAYOR_EMERITUS
260	- US_MARINE
261	- SMALL_VEHICLE
262	- EMERGENCY_VEHICLE
263	- FREE_ZONE
264	- RESTRICTED
265	- SCHOOL_VEHICLE
266	- REPOSSESSION
267	- COAST_GUARD
268	- FLEET
269	- TOKEN_TRAILER
270	- STATE_TROOPER
271	- PERMITTED
272	- PUPILS
273	- BRONZE_STAR
901	- NON_GOVERNMENTAL_ORGANIZATION
902	- ERITREAN_DEFENCE_FORCES
903	- HORSE_DRAWN_VEHICLE
904	- AFRICAN_UNION
905	- CIVIL_ORGANIZATION
906	- AID_AGENCY
501	- ABSENTEE-SHAWNEE_TRIBE_OF_INDIANS_OF_OKLAHOMA
502	- APACHE_TRIBE_OF_OKLAHOMA
503	- CADDO_NATION_OF_OKLAHOMA
504	- CHEROKEE_NATION
505	- CHEYENNE-ARAPAHO_TRIBES
506	- CHICKASAW_TRIBE_OF_OKLAHOMA
507	- CHOCTAW_TRIBE_OF_OKLAHOMA
508	- CITIZEN_POTAWATOMI_NATION
509	- COMANCHE_NATION
510	- DELAWARE_NATION
511	- EASTERN-SHAWNEE_TRIBE_OF_OKLAHOMA



512 - IOWA_TRIBE_OF_OKLAHOMA
513 - KAW_NATION
514 - KICKAPOO_TRIBE_OF_OKLAHOMA
515 - KIOWA_INDIAN_TRIBE_OF_OKLAHOMA
516 - MIAMI_TRIBE_OF_OKLAHOMA
517 - MODOC_TRIBE_OF_OKLAHOMA
518 - MUSCOGEE_CREEK_NATION
519 - OSAGE_NATION
520 - OTTAWA_TRIBE_OF_OKLAHOMA
521 - OTOE-MISSOURIA_TRIBE_OF_INDIANS
522 - PAWNEE_NATION_OF_OKLAHOMA
523 - PEORIA_TRIBE_OF_INDIANS_OF_OKLAHOMA
524 - PONCA_TRIBE_OF_INDIANS_OF_OKLAHOMA
525 - QUAPAW_TRIBE_OF_INDIANS
526 - SAC_AND_FOX_NATION
527 - SHAWNEE_TRIBE
528 - SEMINOLE_NATION_OF_OKLAHOMA
529 - SENECA-CAYUGA_TRIBE_OF_OKLAHOMA
530 - TONKAWA_TRIBE_OF_OKLAHOMA
531 - UNITED_KEETOOWAH_BAND_OF_CHEROKEE_INDIANS_IN_OKLAHOMA
532 - WICHITA_AND_AFFILIATED_TRIBES
533 - WYANDOTTE_NATION
601 - EXECUTIVE_BRANCH
602 - LEGISLATIVE_BRANCH
603 - JUDICIAL_BRANCH
604 - AQABA_FREE_TRADE_ZONE
611 - MINISTRY_OF_DEFENCE
622 - COMANDO_UNITA_FORESTALI
630 - FOREIGN_TRAILER
641 - FOREIGNER
651 - PUBLIC_SERVICE_VEHICLE
661 - COMMERCIAL_BUT_PRIVATELY_USABLE

OTHER:

900 - MIXED

[Back to top](#)

33. AUTOTYPEMODIFICATION

This parameter allows the state recognition purely by plate text (only available in the engines which contains Mexico – CAM, NAM). The regions are determined by the predefined range of plate types. This overrules the recognised region if any.

Possible values: {0,1}

Default value: 0 – except in NAM and CAM engines, where 1

[Back to top](#)

34. CONVERTOTO

If this parameter is set to "1" all 0 (zeros) will be converted to "O" – only in case of general engine where no syntax check is available.

Possible values: {0,1}

Default value: 0

[Back to top](#)

35. COUNTRYNAME

RETRIEVING COUNTRY/STATE NAMES DIRECTLY

(available from engine version 7.3.9.71)

Note

This property is deprecated/not available from engine version 7.3.16.159.

This property enables the user to directly retrieve country and state information from the recognized images.

Possible values: string

Default value: „default” – returns the country information in the form of country codes listed in [Appendix](#).

Suggested value: „default”

Engine versions 7.3.9.70 and earlier, (including all versions 7.2.8.x and 7.2.7.x), returned the country and state information encoded in the type field of the cmNP structure. A full description on country and state information retrieval in earlier engine versions can be found in the [Retrieving country names from returned plate type values](#) section.

From version 7.3.9.71, the engine can return the country or state code itself.

The engine will, in most cases, return either an ISO 3166-1 alpha-3 or a 3166-2 code. However, there are some countries and/or regions that the engine can recognize but are not present in the official ISO registry and therefore have no official ISO codes. In these cases, easily identifiable but non-official ISO codes were used. For the complete country code table please see [Appendix](#).

The countryname property can be used to:

- Retrieve country codes.
- Customize country codes.
- Retrieve country codes with type values.

1. RETRIEVING COUNTRY CODES

After successfully processing an image (if a previous call of `cmAnpr.FindFirst` or `cmAnpr.FindNext` returned true), the value of this property will contain the country that issued the recognized license plate.

For example:

```
if (cmAnpr.FindFirst(image))  
{ string country = cmAnpr.GetProperty(„countryname“); }
```

The string „country“ will contain the code of the country as listed in [Appendix](#).

2. CUSTOMIZING AND RESETTING:

Before calling `cmAnpr.FindFirst`, set the substitution text to a certain country ID by creating a new property name: „countryname_“ + country_ID, where country_ID can be anything from the „Country – State (CountryName)“ column of the table in [Appendix](#).

```
cmAnpr.SetProperty(„countryname_US-TX“, „TEXAS“);
```

After this call, in all cases when the recognized plate is from Texas, the engine will return `country=„TEXAS“` instead of `country=„US-TX“`.

To reset every previously set custom country name, call:

```
cmAnpr.SetProperty(„countryname“, „default“);
```

3. RETRIEVING COUNTRY CODES FROM TYPE VALUES

With the help of this property it is also possible to retrieve the country code of a certain license plate type if only the information returned in the type field of a cmNP data structure is given.

Suppose that a license plate has the following type value,101112, but you do not know what country it represents. In this case it is possible to feed this value to the engine and it will return the country code:

```
string country = cmAnpr.GetProperty(„countryname_101112”);
```

After this call, the engine will return that the country=“HUN”

Note

This call is independent from FindFirst and FindNext.

If the specified number does not represent a possible license plate type, the GetProperty() call above will return „Invalid type”.

Note

Another direct retrieval of Country and State text is also available in the SDK through the “cm_getcountrycode()” function and “CC_TYPE enumeration.”. For more information: please see the SDK Programmers Manual.

36. CYRILLIC_STYLE

Successor of this property is [local_character_conversion](#).

Note

This property is deprecated/not available from engine version 7.3.15.124.

This parameter allows the user to show the results in Cyrillic characters. In case of 1, the engine will return the Russian plates in Cyrillic format.

Possible values: {0,1}

Default value: 0

[Back to top](#)

37. LOCAL_CHARACTER_MODIFICATION

37.1. LOCAL_CHARACTER_CONVERSION

This parameter allows the user to show the results in local language characters.

0 – The engine will decide which character sequence to return as a default

1 – the engine will return the Russian plates in Cyrillic format

2 – the engine will return the Nepali plate results in Nepali characters

3 – (1+2) – both Nepali and Russian plates shown in local language characters

Possible values: {0,1,2,3}

Default value: 2

[Back to top](#)

37.2. LOCAL_CHARACTER_PERMUTATION

(available from engine version 7.3.13.33)

This parameter allows the user to change the behaviour of the engine in case of character sequence pairs: Arabic character sequences with their Latin pairs also present on the plate.

Characters without a pair will be returned normally.

0 – The engine will decide which character sequence to return for every Arabic - Latin character sequence pair on the plate.

1 – Arabic to Latin: the engine will return the Latin character sequence for every Arabic - Latin character sequence pair on the plate.

2 – Latin to Arabic: the engine will return the Arabic character sequence for every Arabic - Latin character sequence pair on the plate.

3 – all characters: the engine will return both character sequences for every Arabic - Latin character sequence pair on the plate.

Possible values: {0, 1,2,3}

Default value: 0

 Note

Should you set this parameter to 0, 1 or 2 we strongly suggest **not to set** the [gapospace](#) parameter to 1. It won't function properly on plates with Arabic - Latin character sequence pairs.

Using [gapospace](#) = 2 is preferred in this case.

 Note

Using this parameter is preferred instead of setting the [unicode_in_text](#) parameter to 2.

(If the [unicode_in_text](#) is set to 2 the engine tries to "translate" the Arabic characters to Latin characters but the translation is often imprecise or even impossible.)

[Back to top](#)

38. UNICODE_IN_TEXT

Representation of the non-ASCII characters in the text parameter of the cmNP structure.

Meaning of the possible values:

0 – The non-ASCII characters are replaced with an exclamation mark ('!')

1 – Each non-ASCII character is represented with 6 ASCII characters: '(xxxx)', where xxxx is the Unicode code of the character in hexadecimal form including the leading zeros

 Note

This will give a rough translation. If both the Arabic and the corresponding Latin characters are present on the plate using the [local_character_permutation](#) parameter is suggested instead.

Possible values: {0,1}

Default value: 1

Suggested value: 1

[Back to top](#)

PROPERTIES RELATED TO COLOR RECOGNITION

Color names and codes what Carmen® can return

Color	BGR Code	Decimal (returned value)
Black	0x000000	0
Red	0x0000FF	255
Brown	0x004080	16512
Orange	0x0080FF	33023
Green	0x00FF00	65280
Yellow	0x00FFFF	65535
Gray	0x808080	8421504
Blue	0xFF0000	16711680
Cyan	0xFFFF00	16776960
White	0xFFFFFFFF	16777215

39. ANALYZECOLORS

COLOR RECOGNITION MODE

This property selects the color recognition mode for license plates, but only when the plate color is an additional distinguishing feature that carries extra information. Meaning of the possible values:

0 – Color identification is disabled

1 – cmAnpr engine returns discrete color values – see table above. During processing, the engine reads exact BGR values, and the statistically most probable results are given.

Note

Due to statistical reasons, results based on discrete values may be incorrect in case of plates with previously "unseen" colors (e.g. blue plates will not be identified if there were only red and orange ones available when the cmAnpr engine was released).

2 – cmAnpr engine returns the exact BGR (Blue, Green, Red) values read (further processing may be required by the user in order to define the exact color)

3 – (not available value)

4 – In case of certain contemporary Saudi Arabian plates, where the band color is also marked with a symbol (circle, or isosceles triangle pointing to various directions), a discrete color value is returned according to the marker symbol.

5 – Both color identification and symbol reading are applied to recognize color. Discrete color values are returned such as in case of value 1.

6-7 (not available values)

8 – Color identification based purely on plate text color indication (currently only in case of Thailand plates)

9 – Color identification based on both plate text color indication and exact color recognition (currently only in case of Thailand plates)

The cmAnpr engine can recognize the color of a license plate only when the following requirements are fulfilled:

- the country/state recognition feature must be enabled ([depth](#) > 0)

 Note

We are using BGR (Blue,Green,Red) values everywhere, except in our Demo applications (ADI, ADV), where RGB is in use.

 Note

Set an adequate [timeout](#) value according to instructions in the [timeout](#) section above, which will allow full processing of the country/state recognition.

- the engine must correctly identify the country/state since the layout defines the location of the color area
- the analyzecolors property must be set to {1,2,4,5,8,9}

 Note

The color recognition feature is defined only for the values listed above. When setting values 4 or 5, color information may be returned from grayscale images as well for the contemporary plate types of Saudi Arabia.

Possible values: {0,1,2,4,5,8,9}

Default value: 0, except ARAB engine where it is 5

Suggested value: 5 – in case color recognition is required, otherwise leave the default value

[Back to top](#)

40. COLOR

DIFFERENT COLORS ON THE PLATE

(available from version 7.3.13.193)

In case of color recognition on License Plates there are 3 colors which we are distinguish (Background, Text and Dedicated Area color).

For example:



On the above plate the main background color is white, the main text color is black and the dedicated area color is red.

With the usage of this property now it is possible to get all 3 color back at the same time. The user can get them back from code with GetProperty() function as follows:

- "color_bkcolor" -> main background color
- "color_textcolor" -> main font color
- "color_dacolor" -> If there is a dedicated area on the plate, it returns its color in hexadecimal format. If there is no dedicated area on the plate, it returns -2.
- "color" or "color_color" -> Dedicated Area color if exist, otherwise, main background color

The following values can also be returned for each of the above properties:

- 1: No ANPR result
- 3: Color recognition is not possible
- 4: Color recognition is unsuccessful

Note

Furthermore, specifically for color_dacolor, the frame of the dedicated area also can be queried with color_dacolor_frame. This property can also return the above error codes (-1; -2; -3; -4).

It is also possible to get back the text and background color for each character individually. You can do that if you go through the result of the characters one-by-one and read these properties for the characters: "bkcolor" and "color".

In the case on the above license plate the result would be the following:

	Text color	Background color
4	White	Red
8	Black	White
2	Black	White
2	Black	White
8	Black	White
9	Black	White

 Note

The engine returns the colours like it is mentioned in [this](#) table.

[Back to top](#)

40.1. COLOR_SOURCE

SOURCE INFORMATION OF THE COLOR

(available from version 7.3.14.194)

With this parameter, you can find out based on what information/tools did the engine return the given color.

The source information is returned in value-name pairs where the value will never change for the specific source but the name itself can.

Possible returned sources:

- **ONLY_COLOR_RELATION:** Only the color relation of the plate is known (light on dark or dark on light) because the image or the plate has no color information.
- **DISCRETE:** The engine learned the possible colors of the type and chose the returned color based on the comparison to these.
- **EXACT:** It returns the exact color (specific BGR shade) based on the pixels of the plate/characters.
- **EMBLEM:** In cases where an emblem on the plate encodes the color, we can also read it based on that.
- **DISCRETE_AND_EMBLEM:** Color determination is based on considering both criteria.
- **TEXT:** In certain rare cases, the text can also determine the color.
- **DISCRETE_AND_TEXT:** Color determination is based on considering both criteria.

Note

Starting from version 7.3.16.62, the name is no longer included in the string; instead, it can be accessed through the new 'color_source_value' property.

[Back to top](#)

40.2. COLOR_SOURCE _VALUE

NUMBER VALUE FOR THE COLOR SOURCE PROEPTY

(available from version 7.3.16.62)

Each above color_source has an integer value. While the name of the color source might change (for example we changed the SYMBOL name to EMBLEM recently) over time, the value and the meaning remains the same.

- **0:** ONLY_COLOR_RELATION
- **1:** DISCRETE
- **2:** EXACT
- **4:** EMBLEM
- **5:** DISCRETE_AND_EMBLEM
- **8:** TEXT
- **9:** DISCRETE_AND_TEXT

[Back to top](#)

41. WHITEBALANCE

WHITE BALANCE CORRECTION

When applying color recognition, the white balance is set automatically by the engine. In some special cases (e.g. special camera settings, extreme weather or light conditions) the returned colors may not be correct. To avoid misrecognition of colors the **whitebalance** property can be used.

- If it is set to 0 then white balance correction is not used.
- If it is set to 100 then maximal white balance correction is applied

24/7 use of white balance correction can also be applied. Incorrect results may return only when the camera is used in very different environmental conditions in a very short time (e.g. the camera is used at night with **whitebalance 100**, turned off and then turned on in the morning with the same settings).

Possible values: [0..100]

Default value: 100

Suggested value: 100

[Back to top](#)

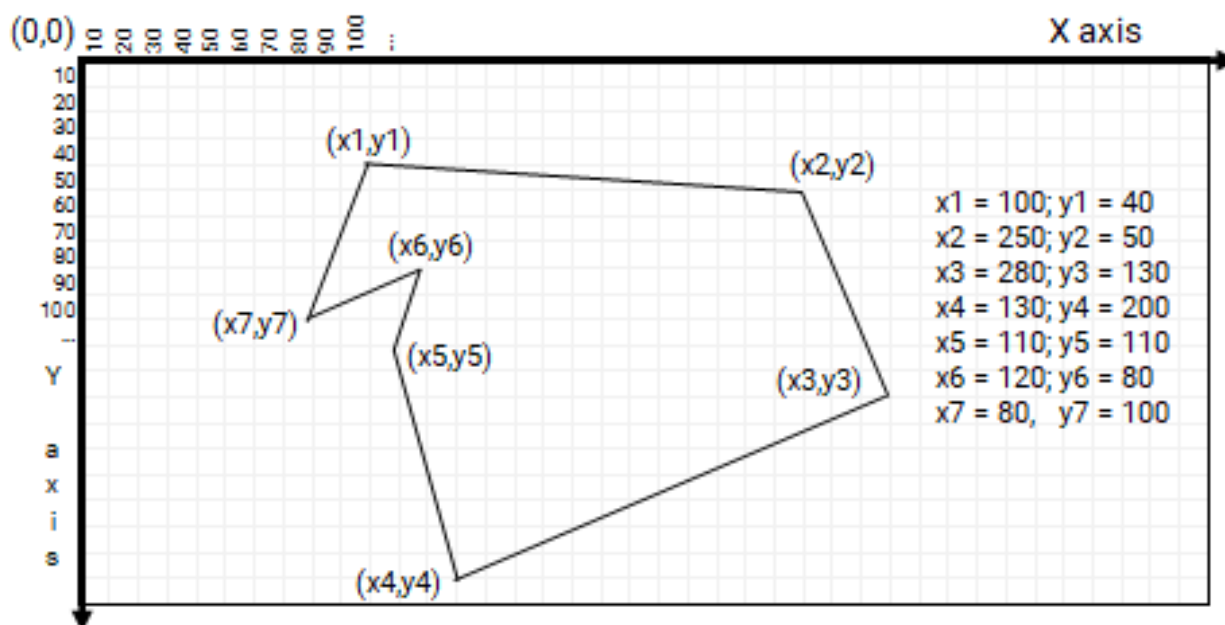
PROPERTIES RELATED TO THE POSITION OF LICENSE PLATES IN INPUT IMAGES

How to define a polygon

For the below properties we are using polygons as follows:

One Polygon is: $P = x_1, y_1; x_2, y_2; x_3, y_3; \dots; x_i, y_i; \dots; x_n, y_n$

Where (x_i, y_i) means one point on a picture in axis parallel coordinate system. The origo (0,0) is the top left corner. On the X axis the values are increasing rightwards, on the Y axis the values are increasing downwards. You have to set the polygon points clockwise.



42. ROI/ROU

REGION OF INTEREST / REGION OF UNINTEREST

(available from version 7.3.11.189)

Note

From CARMEN® 7.3.1.26 there is a possibility to set the ROI/ROU properties in ADI Demo application. If you are interested, please check page #10 in [this](#) document.

42.1. ROI

REGION OF INTEREST

Please check how to [define a polygon](#) at the beginning of this section.

You can set with this property, the polygon(s) where CARMEN® should search for the license plates.

Usage:

- P: CARMEN® will search the LP inside the polygon (delete the previous ROI settings)
- P1 + P2 + ... + Pn: set more polygons at the same time (delete the previous ROI settings)
- +P: add one more polygon to the existing ones, it will NOT delete the previous settings
- del: delete the previous ROI settings



This parameter can be set, get, its values saved into gxsd.dat, and it's readable from there. When ROI is set (without using the ROU parameter), then CARMEN® will search for license plates only in the selected ROI zone.

If you would like to set the above polygon as ROI: "P1x1,P1y1;P1x2,P1y2;P1x3,P1y3;P1x4,P1y4"

42.2. ROU

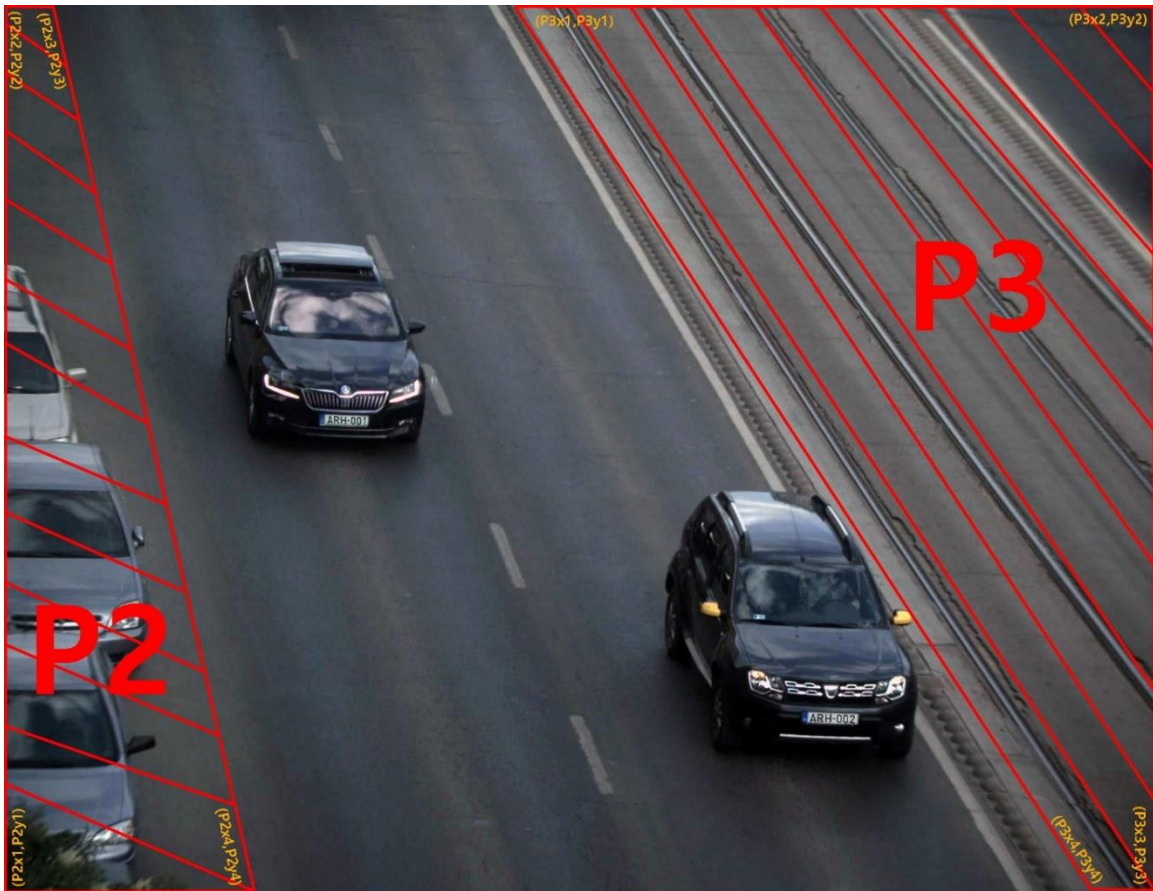
REGION OF UNINTEREST

Please check how to [define a polygon](#) at the beginning of this section.

You can set with this property the polygon(s) where CARMEN® shouldn't search for the license plates.

Usage:

- P: CARMEN® will NOT search the LP inside the polygon
- P1 + P2 + ... + Pn: set more polygons at the same time (delete the previous ROU settings)
- +P: add one more polygon to the existing ones, it will NOT delete the previous settings
- del: delete the previous ROU settings



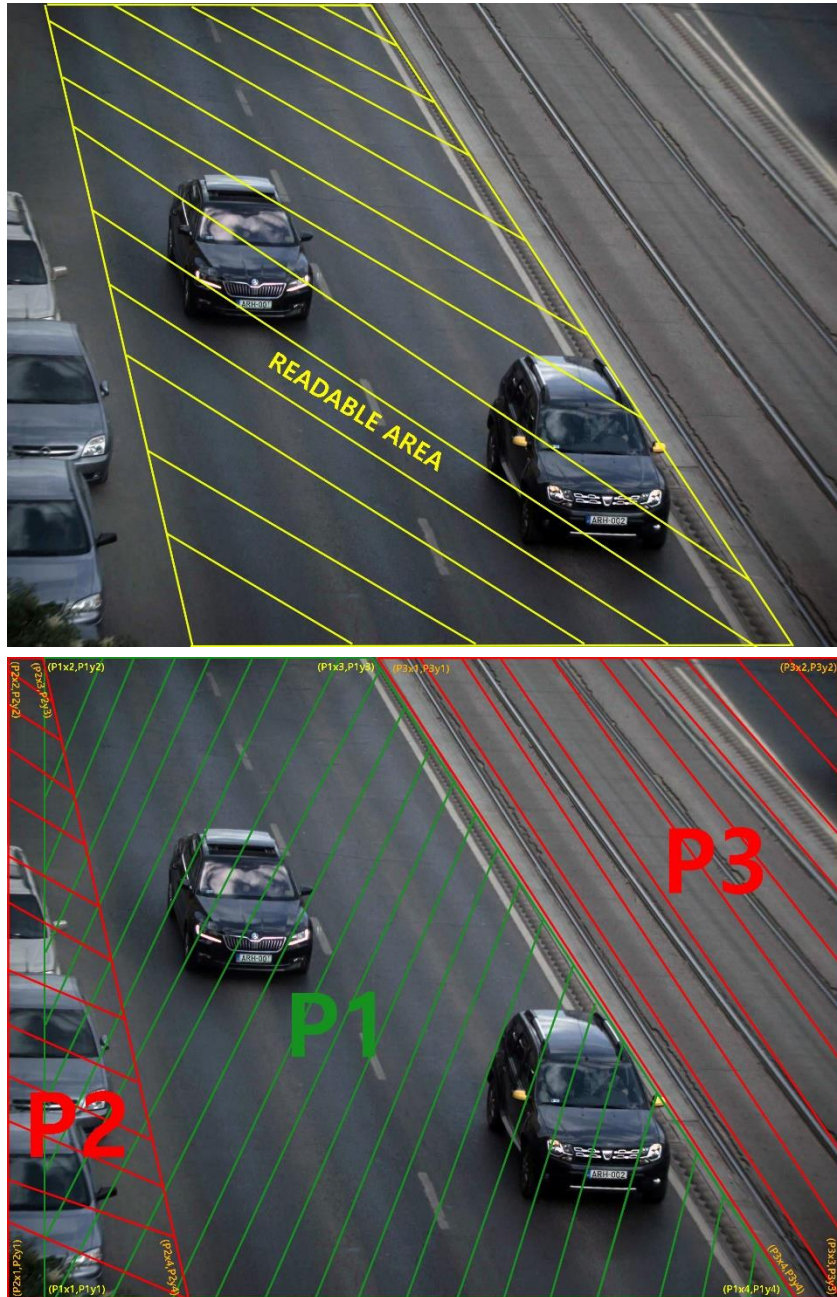
This parameter can be set, get, its values saved into gxsd.dat, and it's readable from there.

If you set ROU only then CARMEN® will search LP outside ROU.

If you would like to set the above polygons as ROU:

"P2x1,P2y1;P2x2,P2y2;P2x3,P2y3;P2x4,P2y4+P3x1,P3y1;P3x2,P3y2;P3x3,P3y3;P3x4,P3y4"

Carmen® will never search license plates from a ROU zone, even when it is interfering with a ROI zone (ROU is stronger than ROI)



Note

If you set [posfreq](#) as a polygon then it will delete the ROI and ROU and overwrite the ROI with the [posfreq](#) polygon.

Note

We strongly suggest not to use the [posfreq](#) and the ROI/ROU parameters simultaneously to avoid future difficulties.

[Back to top](#)

43. POSFREQ

POSITION FREQUENCY

By using the position frequency, certain areas can be specified for the ANPR engine, which are more superior to the others.

With the following parameters, the ANPR algorithm can be set to search for license plates on specific parts of the image. Moreover, some parts can be differentiated according to probability of the license plate occurrence. The essence of the method is that the image is divided into equal zones and each zone is provided with a weight.

The correct value assignment of the weight increases the effectiveness of the searching process. Giving a larger weight of the appointed zone increases the probability of finding the plate and decreases the plate reading time.

The weight assignment is possible in three ways:

- by **uniform distribution**: the weight of each zone will be the same positive number,
- **defining zones**: zones can be defined by the user,
- **defining a polygon**: the polygon should contain all the plates to be read.

The weights may be calculated in a self-adaptive way, as well. In this case, the engine calculates the weights by itself based on the incoming images: each found plate increases the weight of that zone which contains the plate.

The property contains a string of characters. It consists of numbers separated by ',' and '\n', where ',' separates the numbers and the '\n' stands for the line wrap.

If the string is empty, it initializes the grid with uniform distribution (the weight of each zone will be the same positive number). If there are exactly two numbers and at least three number pairs (two columns and three rows), the string defines a polygon. Otherwise, it defines zones, where the given numbers represent the starting weights of each zone.

The data is invalid if the rows are not the same length or if there are less than two columns, or less than three rows.

Zero-weight zones are omitted from the search for license plates. However, the engine will provide results even for plates partially located in a zero-weight zone, if the other parts of the license plate can be found in a non-zero weight zone.

For example:

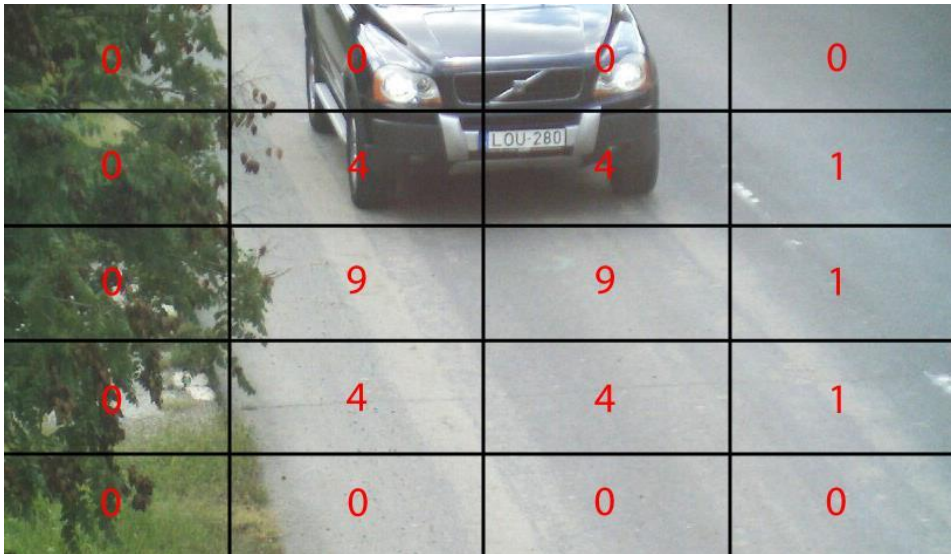
Defining zones:

```
<posfreq value="1,1,1;4,9,4;1,1,1"/>
```

The image is divided into 3x3 zone with the given starting weights.

1	1	1
4	9	4
1	1	1

Please check the following sample image, which indicates superior areas in the centre of the image:



```
<posfreq value="0,0,0,0;0,4,4,1;0,9,9,1;0,4,4,1;0,0,0,0"/>
```

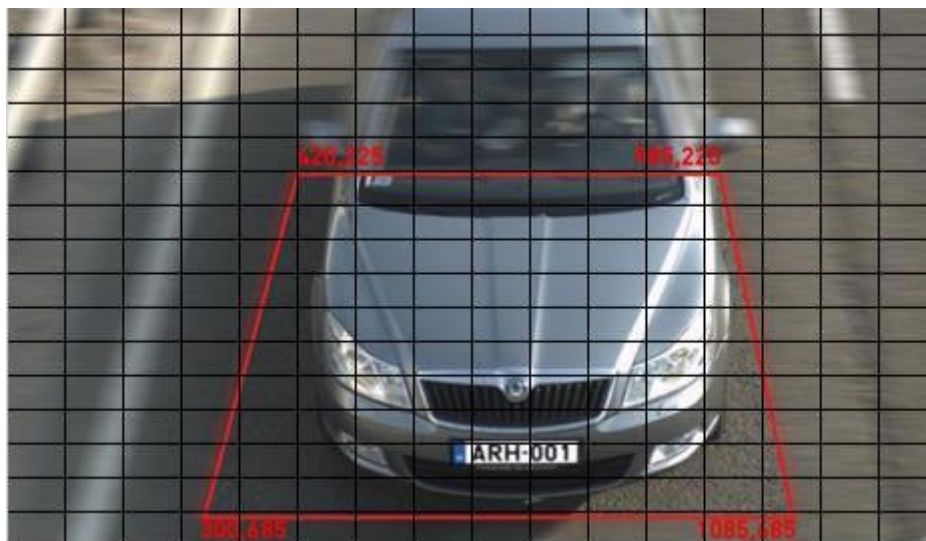
Defining a polygon:

```
<posfreq value="420,225;985,225;1085,685;300,685">
```

Note

The order of the coordinates must be set clockwise (see below). A polygon is designated in the center of the image, on the score of which the algorithm produces the zones (posfreqhistxs*posfreqhistys is the number of zones, 16*16 by default) in such a way that the starting weight of the zones – contained by the polygon – will be maximal. The weight of the zones intersected by the polygon will be lower in proportion to the intersection. Finally, the weight of the outer (untouched by the polygon) zones will be 0. It does not return any character from the zones with 0 weight.

Example for a polygon posfreq setting:



Possible usage of posfreq polygon

(points are represented in pixel coordinates on the 16x16 default grid)

Possible values: character string

Default value: "" (empty string)

Suggested value: leave the default value

[Back to top](#)

44. POSFREQHALFLIFE

LEVEL OF WEIGHT ADAPTATION

If its value is 0, the weights will not be adapted (it does not learn from the previous cases). It will use the original settings all the time.

Otherwise, after 'posfreqhalflife' number of evaluations, the starting information will be half lapsed and the new information will be half freshened.

Half-life: after the evaluation of so many images, the total weight of the histogram will be twice as much.

Possible values: [0..1048576]

Default value: 1000

Suggested value: 1000

[Back to top](#)

45. POSFREQHISTXS

HORIZONTAL RESOLUTION FOR DEFINING POSITION FREQUENCY

In case of setting a polygon or zones, the number of columns can be set by this property.

Possible values: [2..64]

Default value: 16

Suggested value: 16

[Back to top](#)

46. POSFREQHISTYS

VERTICAL RESOLUTION FOR DEFINING POSITION FREQUENCY

In case of setting a polygon or zones, the number of rows can be set by this property.

Possible values: [2..64]

Default value: 16

Suggested value: 16

[Back to top](#)

47. POSFREQWEIGHT

WEIGHT FOR THE POSITION OF THE LICENSE PLATES

This parameter defines the extent the system has to take into account the position of the license plates.

If this parameter is 0, the system does not distinguish between the non-0 weight zones. In this case, the searching does not exploit the distribution of the position of license plates.

If this parameter is 100, the system tries to exploit maximally the distribution of the position of license plates.

Possible values: [0..100]

Default value: 50

Suggested value: 50

[Back to top](#)

PROPERTIES RELATED TO IMAGE QUALITY

48. CONTRAST_MIN

MINIMUM CONTRAST DIFFERENCE

The minimum difference between the grayscale value of the number plate characters and the plate background.

Note

No license plate result will be returned where the grayscale contrast is smaller than the specified value.

Possible values: [1..255]

Default value: varies with each engine release; default property value can be queried by the `GetProperty()` function

Suggested value: leave the default value

[Back to top](#)

49. GAMMA

TURNING ON/OFF GAMMA CORRECTION ON INPUT IMAGES (available from version 7.2.7.87)

This property provides the option to apply gamma correction, which makes input images brighter.

Meaning of the possible values:

- 0: Gamma correction is disabled.
- 1: Gamma correction is applied with the standard correction value of 2.2, which can result in better recognition rates in case of images with lower contrast.



Image without gamma correction



Image with gamma correction

Possible values: {0,1}

Default value: 0

Suggested value: 0 – image quality issues should be resolved in the camera

[Back to top](#)

PROPERTIES RELATED TO THE CALCULATION OF THE CONFIDENCE LEVEL

50. CONFIDENCEMODE

CALCULATION OF PLATE CONFIDENCE, (SUCCESSOR OF PLATECONF)

(available from version 7.2.8.41 – **Confidence modes 8-15 are available from 7.3.10.204**)

Unlike [plateconf](#), [confidencemode](#) has several options to fine-tune the calculation method of the returned overall confidence level. From version 7.2.8.41, use this property instead of [plateconf](#). The possible values and their meanings are listed in the table on the next page.

The confidence level can be calculated in several ways but we can choose from three basic principles:

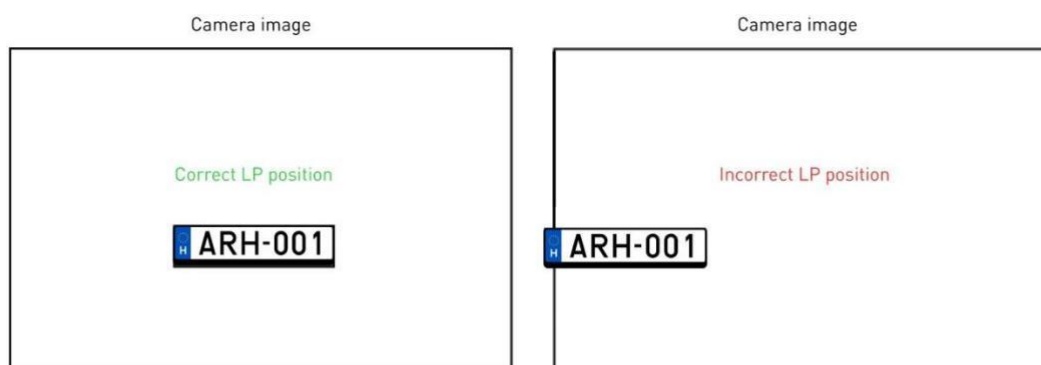
- Having the product of the factor(s), (text, type and position), as the final confidence level. (Confidence modes 0-7)

OR

- Having the arithmetic mean, (average), of the factor(s) as the final confidence. (Confidence modes 8-15)

The latter will show a higher confidence rate (for the same reading) but will be less scalable.

The following image demonstrates the required license plate position on the camera image:



- Using a normalized confidence value (see below the tables)

Meaning of the possible values:

confidencemode	Formula for the returned confidence				Equivalent to plateconf value	Comments
	Factors			Formula		
	Text confidence	Type confidence	Position confidence			
0	100			Constant	-	Confidence level calculation is disabled, the engine will always return 100 as overall confidence
1	Qtext			$\prod Qc_n$	0	The text confidence will be returned by multiplying each character's confidence
2		Qtype		Qtype	-	The type confidence will be returned
3	Qtext	Qtype		Qtext * Qtype	1	The product of the text confidence and the type confidence will be returned
4			Qpos	Qpos	-	The position confidence will be returned
5	Qtext		Qpos	Qtext * Qpos	-	The product of the text confidence and the position confidence will be returned
6		Qtype	Qpos	Qtype * Qpos	-	the product of the position confidence and the type confidence will be returned
7	Qtext	Qtype	Qpos	Qtext * Qtype * Qpos	2	The product of the position confidence, the type confidence and the text confidence will be returned

confidencemode	Formula for the returned confidence				Equivalent to plateconf value	Comments
	Factors			Formula		
	Text confidence	Type confidence	Position confidence			
8	100			Constant	-	The engine will always return 100 as overall confidence
9	Qtext			$\overline{Q_{\text{text}}}$	-	The text confidence will be returned by taking the arithmetic mean (average) of the character's confidence
10		Qtype		Qtype	-	The type confidence will be returned
11	Qtext	Qtype		$\frac{(\overline{Q_{\text{text}}} + Q_{\text{type}})}{2}$	-	The arithmetic mean (average) of the text confidence and the type confidence will be returned
12			Qpos	Qpos	-	The position confidence will be returned
13	Qtext		Qpos	$\frac{(\overline{Q_{\text{text}}} + Q_{\text{pos}})}{2}$	-	The arithmetic mean (average) of the text confidence and the position confidence will be returned
14		Qtype	Qpos	$\frac{(Q_{\text{type}} + Q_{\text{pos}})}{2}$	-	The arithmetic mean (average) of the position confidence and the type confidence will be returned
15	Qtext	Qtype	Qpos	$\frac{(\overline{Q_{\text{Text}}} + Q_{\text{type}} + Q_{\text{pos}})}{3}$	-	The arithmetic mean (average) of the position confidence, the type confidence and the text confidence will be returned

We have introduced two types of **normalized confidences** (from version 7.3.16.79, but only in selected regions):

- text-based normalized confidence (applies to the accuracy of the returned text)
- state-based normalized confidence (applies to the accuracy of the returned country / state)
- textstate-based normalized confidence [relates to the accuracy of the text, country, and state triplet (collective correctness)]

Note

The normalized confidences provided by the engines are remarkably precise. Therefore, if the utilized engine offers this feature, its usage is highly recommended.

Normalized confidences have been incorporated into the engines with the following `confidencemode` values:

- **33: normalized text** confidence
- **48: normalized state** confidence
- **49: normalized textstate** confidence

The meaning of normalized confidence is the following: if the normalized confidence value is x , then the probability of the returned result being correct is $x\%$. For example, if a result is returned with a normalized text confidence value of 80, there is an 80% chance that the text is accurate. With a normalized textstate confidence of 40 there is a 60% chance that the text and/or state may be incorrect.

These new `confidencemode` options can be used within the engine similarly to the existing `confidencemode` options, through the relevant properties (`confidencemode`, [confidencemode_X](#)).

For more information on the above calculation, see the appendix [Confidence Level Calculation](#).

Possible values: {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15, 33, 49}

Default value: 49 (if the engine supports normalised confidences), 33 in GEN engine, 7 for the rest of the engines

Suggested value: 49 (if the engine supports normalised confidences), 33 in GEN engine, 7 for the rest of the engines.

 Note

If you set [plateconf](#) and [confidencemode](#) as well, then that one will take affect which you set later and it is possible that it may overwrite the other value.

For example:

- If confidencemode is set to 4 and later you set plateconf to 2, then confidencemode will be overwritten to 7
- If plateconf is set to 1 and later you set confidencemode to 12, then confidencemode will take affect and plateconf will not be changed

[Back to top](#)

50.1. CONFIDENCEMODE_X

(available from version 7.3.12.5)

After a successful ANPR you can get additional confidence results. Property value can be queried by the `GetProperty()` function like this: `GetProperty("confidencemode_X")`. You have to replace „X" with the appropriate number (0 - 15) from the above table. For example, `confidencemode_5` will provide you the text confidence and the position confidence multiplied together.

You can only get this property; it is not possible to set it!

[Back to top](#)

51. CONFIDENCE_PRECISION

(available from version 7.3.15.112)

This parameter allows the user to set the precision of the confidence.

If you set it to 'n', the engine will give back the confidence based on the following formula:

$\text{calculated_confidence} = \text{Round}(\text{anpr_confidence} * n),$

where `anpr_confidence` is the engine's inner calculated confidence, which is a float number between 0 and 1. So, with a bigger `confidence_precision` the engine returns more precise confidence values.

For example, with the `anpr_confidence` of 0.7851, the returned confidence is 1, 8, 79 or 785 should you set `confidence_precision` to 1, 10, 100 or 1000 respectively which are 27%, 2%, 0.6% and 0.01% off the precise value respectively.

Possible values: positive integer

Default value: 100

Suggested value: In most cases the default 100 is enough, but sometimes it is useful to set it to 1000 or even more

[Back to top](#)

52. PLATECONF

CALCULATION OF THE PLATE CONFIDENCE

(available from version 7.2.7.106)

Successor of this property is [confidencemode](#).

Note

This property is deprecated/not available from engine version 7.3.15.124.

In case of *plateconf* =0 the text confidence (Q_{text}) will be returned.

In case of *plateconf* =1 text and type confidence ($Q_{\text{type}} * Q_{\text{text}}$) will be returned.

In case of *plateconf* =2 the overall confidence ($Q_{\text{pos}} * Q_{\text{type}} * Q_{\text{text}}$) will be returned.

For more information on the above calculation, see the appendix [Confidence Level Calculation](#)

Possible values: {0,1,2}

Default value: 2

Suggested value: 2 - Please use the successor parameter: [confidencemode](#).

[Back to top](#)

Note

If you set [plateconf](#) and [confidencemode](#) as well, then that one will take affect which you set later and it is possible that it may overwrite the other value.

For example:

- If [confidencemode](#) is set to 4 and later you set [plateconf](#) to 2, then [confidencemode](#) will be overwritten to 7
- If [plateconf](#) is set to 1 and later you set [confidencemode](#) to 12, then [confidencemode](#) will take affect and [plateconf](#) will not be changed

53. ZEROCONFIDENCERESULTS

ENABLE ALL RESULTS

In case of dual or 2 level engines, you can turn on this feature to enable all results in case of very **low false positive rate** engines.

For example – **FRQ_FPO** or **PST_FPO** two-step engines:

value 0 – no result in case of unknown type or text

value 1 – the engine will provide a result (text and type) with 0 confidence value

- you can accept all results (higher than 0 confidence level) to still get the super low false positive results
- and send all 0 confidence level results to manual review or to level 2 engine

Possible values: {0,1}

Default value: 0

To see what is the connection between [general](#) and [zeroconfidenceresults](#) property, please check [this](#) chapter.

[Back to top](#)

PROPERTIES RELATED TO MEMORY HANDLING

54. HEAPFREEFREQ

During the run of an engine, it reserves some memory resource for itself. It can happen that for a large image it reserves a bigger amount of memory (this is a normal behaviour). This may remain reserved for CARMEN® even if later it uses only smaller amount of the memory.

You are able to free up the memory which is allocated to CARMEN® with this property. If you set this property to 30 then it means after every 30 FindFirst() calls, it will free up the memory. We suggest you to use this property if you have 2 GB memory or less.

Possible values: {0, every positive integer}

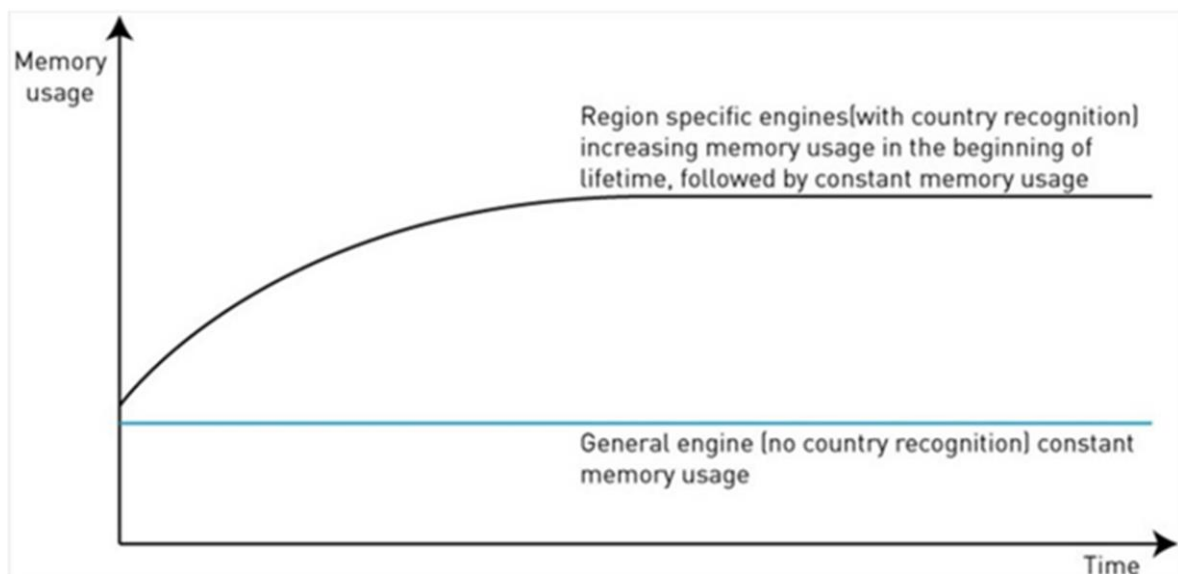
Default value: 0 (means this feature is turned off)

[Back to top](#)



Note

THE RELATIONSHIP OF MEMORY CONSUMPTION AND TIME IN CARMEN® ENGINES



Region specific CARMEN® engines require more memory than the general engine. Each instance uses around 300MB of memory, you can calculate with 500 MB memory for each one. CARMEN® instances do not share the memory area, so this requirement should be multiplied by the number of instances in a system.

PROPERTIES RELATED TO MAKE AND MODEL RECOGNITION (MMR)

(minimal ANPR engine version for MMR: 7.3.11.164 – MMR will not work with lower versions)

The CARMEN® ANPR products from version 7.3.1.21 are equipped with the new Make and Model Recognition (MMR) feature.

If the CARMEN® system contains a typed (not the general) CARMEN® engine and an MMR engine (mmr-x.x.x.x.dll on Windows or mmr-x.x.x.x.so on Linux), the CMANPR module can recognize make, model, category, and the color of the vehicles.

Note

MMR engines need separate MMR licenses and must be the same region as ANPR license to be able to work together. GEN MMR engine can work with every ANPR engine (except GEN ANPR engine, which is not capable to run with any MMR engine).

The CMANPR module can get the MMR information in two ways: auto and manual.

Using the **auto mode**, the CMANPR module generate MMR information after every number plate reading (FindFirst, FindNext).

This mode is recommended if you want to recognize every single image (e.g., each image from a video/image sequence, or from multiple standalone images).

The **manual mode** is useful if you have many images or image sequences (e.g., Highway events or video) and you can define the exact image(s) from which the MMR should create a result. In this case you can get MMR information only from the best image of the sequence.

To get the results use the getMMR function of the CMANPR module. You need the image and the number plate reading result (cmNP structure) to call getMMR function.

In both cases (auto and manual) you can get the MMR result through the properties below.

Available property names to get the MMR result (from CARMEN® ANPR version 7.3.1.24)

NOTE: All properties are read only!

- **mmr/result/make:** the 'make' from the make&model of the recognized vehicle.
- **mmr/result/make/conf:** the confidence of the recognized 'make'.
- **mmr/result/model:** the 'model' from the make&model of the recognized vehicle.
- **mmr/result/model/conf:** the confidence of the recognized 'model'.
- **mmr/result/makemodel:** the 'make model' from the make&model of the recognized vehicle.
- **mmr/result/makemodel/conf:** the confidence of the recognized 'make&model'.
- **mmr/result/heading:** the 'view angle' of the last result.
- **mmr/result/heading/conf:** the confidence of the recognized 'view angle'.
- **mmr/result/category:** the 'category' of the recognized vehicle. Short name.

Known categories (full name):

- BUS (Bus)
- CAR (Car)
- CARAVAN (Caravan)
- HVT (Heavy truck)
- LGT (Light truck)
- MTB (Motorbike)
- PICK-UP (Pick-up)
- TRAILER (Trailer)
- TRUCK (Truck)
- UNK (Unknown)
- VAN (van)
- **mmr/result/category/name:** the full name of the category
- **mmr/result/category/conf:** the confidence of the recognized 'category'.
- **mmr/result/color:** the 'color' code of the recognized vehicle.
- **mmr/result/color/name:** the 'color name' of the recognized vehicle.

Known color names:

- BLACK
- BLUE
- BRONZE
- BROWN
- BURGUNDY
- CREAM
- GOLD
- GRAY
- GREEN
- LIME
- ORANGE
- PINK
- PURPLE
- RED
- SILVER
- TURQUOISE
- WHITE
- YELLOW

 **Note**

If CARMEN® receive a grayscale image, the MMR engine will give back "GRAYSCALE" as color, which meant for the image, not for the vehicle color.

- **mmr/result/color/conf**: the confidence of the recognized 'color'.
- **mmr/result/viewpoint**: the view point

Possible viewpoint values:

- front
- rear
- front_side*
- rear_side*
- top*
- top_side*
- side*

Note

* can be used if triggermode is set to 0.

- **mmr/result/viewpoint/conf:** the confidence of the recognized 'view point'.
- **mmr/result/bodytype:** the 'body style name' of the recognized vehicle.

Known bodytype names:

- Sedan
 - SUV
 - Hatchback
 - Combi/Wagon
 - MPV
 - Coupe
 - Convertible
 - Liftback
- **mmr/result/bodytype/conf:** the confidence of the recognized 'body style'.
 - **mmr/result/time:** the time of the last MMR recognition in msec.

Note

You can read more about MMR properties in our [MMR Brief description](#).

New properties for the CARMEN® ANPR products from version v7.3.1.24 (regarding ANPR and MMR engines):

NOTE: All properties are read only except 'engpropstype', 'anpr/fullname', 'mmr/fullname' and 'mmrmode'!

- **engpropstype** - Engine type for property handling functions. The type of the engine for the property handling functions (SetProperty, GetProperty, SaveProperties, GetEngineProperties). The available values are: anpr, mmr.

NOTE: In case of SaveProperties, the module is not capable of saving the property value to the gxsd.dat file.

- **anpr/fullname**: The name of the actual (current) ANPR engine. Same as the anprname property.
- **anpr/module**: The name of the actual (current) ANPR engine's module (cmanpr-x.x.x.dll or cmanpr-x.x.x.so).
- **anpr/region**: The name of the actual (current) ANPR engine's region (property group from the full name, the string after '.').
- **anpr/status/code**: The status value of the actual ANPR engine.

Available status values:

- **0** (no status): No status info
- **1** (valid): The ANPR engine is valid and there is a license for the engine
- **-1** (no engine): The default ANPR engine isn't defined
- **-2** (invalid): 'anprname' or 'anpr/fullname' property defines an invalid ANPR engine
- **-3** (no license): The ANPR engine is valid but there is no license for the engine
- **-4** (expired license): There is a license for the engine but it's expired
- **anpr/status/msg**: The ANPR engine status information in string format.
- **anpr/supportmmr**: 1 if the ANPR engine supports the MMR feature.

- **anpr/result/charheight**: the average height of the characters in the last ANPR result.
- **anpr/result/proctime_ms**: the process time of the last ANPR reading in msec.
- **mmr/fullname**: The name of the actual (current) MMR engine. It can be altered during runtime. Its value can be set according to the following: **MMR engine module name : property group**. If there is no specified property group then the cmnpr module handles it as 'default'.
- **mmr/fullname/auto**: The name of the best MMR engine for the current ANPR engine.
- **mmr/module**: The name of the actual (current) MMR engine's module (mmr-x.x.x.x.dll or mmr-x.x.x.x.so) .
- **mmr/region**: The name of the actual (current) MMR engine's region (property group from the full name, the string after ':').
- **mmr/status/code**: Status value of the actual MMR engine.

Available status values:

- **0** (no status): No status info
- **1** (valid): The MMR engine is valid and there is a license for the engine
- **-1** (no engine): The default MMR engine isn't defined
- **-2** (invalid): 'mmr/fullname' property defines an invalid MMR engine
- **-3** (no license): The MMR engine is valid but there is no license for the engine
- **-4** (expired license) : There is a license for the MMR engine but it's expired
- **mmr/status/msg**: The MMR engine status information in string format.
- **mmrmode**: Set the MMR mode (default value: 0)

Available modes:

- **0** (off) : no MMR recognizing
- **1** (auto) : automatic MMR recognizing every FindFirst() call
- **2** (manual) : MMR recognizing by calling the cm_getmmr (getMMR) function

OBSOLETE property names to get the MMR result (before CARMEN® ANPR version 7.3.1.24)

NOTE: All properties are read only!

- **mmr/mm/make:** the 'make' from the make&model of the recognized vehicle.
- **mmr/mm/model:** the 'model' from the make&model of the recognized vehicle.
- **mmr/mm/submodel:** the 'sub model' from the make&model of the recognized vehicle.
- **mmr/mm/fullmm:** the 'make model submodel' from the make&model of the recognized vehicle.
- **mmr/mm/conf:** the confidence of the 'make&model' recognition.
- **mmr/category:** the 'category' of the recognized vehicle. Three characters code.

Known categories (full name):

- BUS (Bus)
- CAR (Car)
- HVT (Heavy truck)
- LGT (Light truck)
- UNK (Unknown)
- VAN (van)
- **mmr/category/fullname:** the full name of the category
- **mmr/category/conf:** the confidence of the 'category' recognition.
- **mmr/color/code:** the 'color' code of the recognized vehicle.
- **mmr/color/name:** the 'color name' of the recognized vehicle.

Known color names:

- BLACK
- BLUE
- BROWN
- GRAY
- GREEN
- ORANGE
- PURPLE
- RED
- WHITE
- YELLOW
- **mmr/color/conf:** the confidence of the 'color' recognition.
- **mmr/time:** the time of the last MMR recognition in msec.

[Back to top](#)

APPENDICES

CONFIDENCE LEVEL CALCULATION DETAILS AND EXAMPLE

CARMEN® engines return a confidence level value for each result. This confidence level is always an integer from 0 to 100 and is a measure of the correctness of the result. In other words, it represents how sure the engine is that the result is correct. Higher confidence level means a more reliable result. It is important to mention that even if the confidence level is low, the result can be correct.

Note

The confidence level has no correspondence with the recognition rate. The confidence level is always related to one individual OCR result, (either a single character, or a complete license plate, etc.), while the recognition rate, (the percentage of the license plates that are read correctly,) is always related to a set of images and a reference result set.

CARMEN® calculates separate confidence levels for each subtask of the license plate recognition.

These are:

- q_i confidence of the i th character (*values 0 to 100*)
- Q_{text} Where $Q_{\text{text}} = \text{prod}(q_i), i=1, \dots, n$ **OR** $Q_{\text{text}} = \text{avg}(q_i), i=1, \dots, n$
- Q_{type} confidence of the country/state recognition
- Q_{pos} confidence of the plate position in the image – if it is at the edge of the image, then there is a chance for missing characters
- Q_{all} overall confidence level for the whole license plate – derived from the above

Derive overall confidence level from the separate confidence levels.

Example:

Consider a 6-character license plate with a "C" that is physically damaged (e.g. because of a screw). The recognition of this character is less reliable than of any others, and this weak point should reflect in the overall text confidence.

The formula for calculating the overall confidence level using all possible factors

(*confidencemode=7*):

$$Q_{\text{all}} = Q_{\text{text}} * Q_{\text{type}} * Q_{\text{pos}}$$

See also [confidencemode](#), [plateconf](#).

SAMPLE CALCULATION

(CONFIDENCE MODE=7)

returned values



	returned values					
code	A	R	H	0	0	1
confidence	100	99	99	98	99	99
ncharacter	6					

	returned values
text	ARH001 (in ASCII string format)
wtext	ARH001 (in Unicode string format)
type	111 (defines Hungarian EU standard plate)
confidence	90

Text confidence:

$$Q_{\text{text}} = \prod_{i=1}^n q_i$$

$$\prod_{i=1}^n q_i = q_1 * q_2 * q_3 * q_4 * q_5 * q_6 \quad - (q_i \text{ values represent the confidence of the plate characters})$$

Since these values represent percentages, each q value has to be divided by 100:

$$= (q_1/100) * (q_2/100) * (q_3/100) * (q_4/100) * (q_5/100) * (q_6/100)$$

$$= (100/100) * (99/100) * (99/100) * (98/100) * (99/100) * (99/100)$$

$$\text{So: } \prod_{i=1}^n q_i = 94$$

$$Q_{\text{text}} \approx 94\%$$

Overall confidence of the returned plate type

$$Q_{\text{all}} = Q_{\text{type}} * Q_{\text{pos}} * Q_{\text{text}} = 90 \text{ ('confidence' value above)}$$

$$Q_{\text{type}} * Q_{\text{pos}} = \frac{Q_{\text{all}}}{\prod_{i=1}^n q_i} = \frac{Q_{\text{all}}}{Q_{\text{text}}}$$

$$Q_{\text{type}} * Q_{\text{pos}} = 90/94$$

$$Q_{\text{type}} * Q_{\text{pos}} \approx 95,74\%$$

[Back to top](#)

ZEROCONFIDENCERESULT VS GENERAL

	zeroconfidenceresults = 0	zeroconfidenceresults = 1
general is set to one of the values which is not returning the unidentified types (0, 4, 6, 8, 10, 12,14)	the result will contain only the identified types, without tips type = not ,0', confidence > ,0'	the result will contain only the identified types, with tips. type = not ,0' if confidence > ,0' -> not tip if confidence = ,0' -> tip
general is set to one of the values which is returning the unidentified types (1, 3, 5, 7, 9, 11, 13, 15)	The result will contain the identified and the unidentified types as well, without tips. if type = ,0' -> unidentified type, which is not in the database yet if type = not ,0' -> known type, which is in the database confidence > ,0'	The result will contain the identified and the unidentified types as well, with tips. if type = ,0' -> unidentified type, which is not in the database yet if type = not ,0' -> known type, which is in the database if confidence > ,0' -> not tip if confidence = ,0' -> tip

There is an inside confidence in CARMEN®, which is not seen by the users. This confidence decides if the reading is a tip or not. It can happen that a result which has "20" as confidence still it is not a tip, and on the other side if a result which has "30" as confidence but will be returned as a tip!

Possible results in ADI Demo (confidencemode = 7; general = ['1', '3', '5', '7', '9', '11', '13', '15']; zeroconfidenceresults = '1')

GC = get_confidence (this is the value which is visible in ADI; conf_model = text (C1); conf_mode2 = type (C2); conf_mode4 = position (C7); conf_mode7= product of the previous)

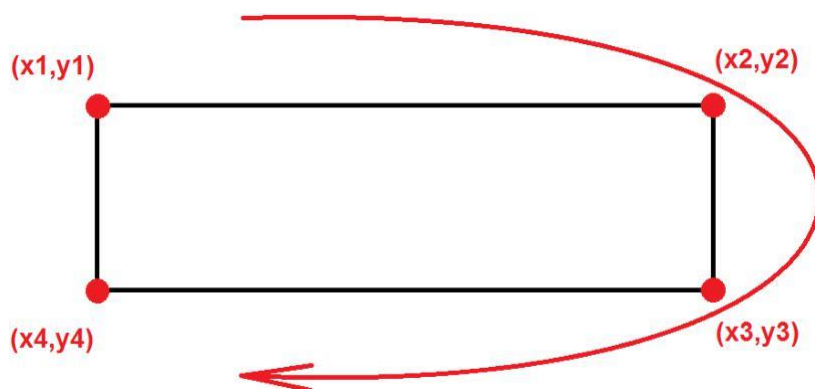
Country/State	Confidence	What it means?	Exact values from code
(0)	0	tip for an unidentified type	GC = '0'; C1 = ""; C2 = ""; C4 = ""; C7 = "";
(0)	43	unidentified type, but not a tip	GC = '43'; C1 = '62'; C2 = '69'; C4 = '100'; C7 = '43'; -> in this case C2 means that the engine is 69% sure that this type is not part of the database!
USA (508115)	0	known type, but only a tip	GC = '0'; C1 = '67'; C2 = '32'; C4 = '100'; C7 = '21'; -> as GC = '0', this is a tip, C2 means that the engine is 32% sure that the result is that type, but the inside confidence return this value only as a tip. C7 is not '0', because GC became '0' because of the inside confidence decided to make this only a tip!
USA (508115)	20	known type, not a tip	GC = '20'; C1 = '84'; C2 = '24'; C4 = '100'; C7 = '20'; -> the C7 is smaller than in the previous case, but still not a tip, based on the inside confidence decision

[Back to top](#)

CALCULATING THE MINIMAL BOUNDING RECTANGLE OF THE RETURNED LICENSE PLATE

The engine returns a 4-point polygon as the frame of the license plate. This is not necessarily rectangular, and there can be cases **when some points are out of the image** (for example if the license plate is at the very edge of the image), if this happens, **negative numbers will be returned**.

The type of the cmNP::frame is gxPG4. It consists of 4 coordinate pairs (x1,y1; x2,y2; x3,y3; x4,y4), which are situated clockwise on the rectangle starting with the upper left corner:



Therefore, 4 points will be received, each referred by x and y coordinates. The coordinates of the top left and bottom right corners of the minimal bounding rectangle can be calculated, as explained in the following drawing:

Top left corner: $x_min = \min(\max(0, \min(x1, x2, x3, x4)), \text{image.xsize})$

$y_min = \min(\max(0, \min(y1, y2, y3, y4)), \text{image.ysize})$



Bottom right corner: $x_max = \max(0, \min(\max(x1, x2, x3, x4), \text{image.xsize}))$

$y_max = \max(0, \min(\max(y1, y2, y3, y4), \text{image.ysize}))$

Based on these formulas, width and height of the bounding rectangle can also be calculated.

[Back to top](#)

RETRIEVING COUNTRY NAMES FROM RETURNED PLATE TYPE VALUES

This chapter only applies to region-specific engines.

The returned cmNP::type value will be 0 (plate type info is not available) if either of these applies:

- the engine does not include country/state recognition option
- the engine cannot determine the country/state of the plate

The returned cmNP::type value by the engine contains the following information:

- country of the plate
- state information (or province – where applicable)
- subtype of the plate (based on syntax and layout, if detailed information is needed, feel free to contact ARH Support Team)

[Back to top](#)

COUNTRY AND STATE ID'S

(FOR ENGINE VERSION 7.2.8.X; 7.3.9.X; 7.3.10.X, 7.3.11.X, 7.3.12.X AND ABOVE)

(e.g. cmanpr-7.2.8.6-latin or cmanpr-eur-7.3.12.5_20Q2)

Note

The following calculation applies only for **engine versions 7.2.8.x** and above (e.g., cmanpr-7.2.8.6-latin).

The type value (T) is a decimal between 100000 and 999999. Divided this type value by 1000 the result is the code of the country (C). The remainder defines the subtype (S) within the country.

Example: USA engine



cmNP::type: 550549

$C = 550549 / 1000 = 550$ defines USA-FL as country with the state.

The remainder (S) = 549, refers to the above standard, single-row Florida plate format:

'letter number number number letter letter'

Example: Latin engine



cmNP::type: 101011

C=101011/1000=101 defines HUN (Hungary) as country.

The remainder (S) =011, refers to the standard, single-row Hungarian EU-plate



cmNP::type: 122012

C=122012/1000=122 defines DEU (Germany) as country.

The remainder (S) =012, refers to the standard, single-row German EU-plate

Note

NOTE FOR THE BELOW TABLES

Rows with **gray background** indicate that although the type code range is pre-reserved, the given country's licence plates are recognized without type codes.

Upon request, they can be supported with type recognition too. For details contact your sales representative.

[Back to top](#)

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
Europe						
101	101000 - 101999	HUN		HUN	Hungary	HUN
102	102000 - 102999	AUT		AUT	Austria	AUT
103	103000 - 103999	SVK		SVK	Slovakia	SVK
104	104000 - 104999	CZE		CZE	Czech Republic	CZE
105	105000 - 105999	SVN		SVN	Slovenia	SVN
106	106000 - 106999	POL		POL	Poland	POL
107	107000 - 107999	EST		EST	Estonia	EST
108	108000 - 108999	LVA		LVA	Latvia	LVA
109	109000 - 109999	LTU		LTU	Lithuania	LTU
110	110000 - 110999	ROU		ROU	Romania	ROU
111	111000 - 111999	BGR		BGR	Bulgaria	BGR
112	112000 - 112999	HRV		HRV	Croatia	HRV
113	113000 - 113999	BIH		BIH	Bosnia Herzegovina	BIH
114	114000 - 114999	SRB		SRB	Serbia	SRB
115	115000 - 115999	MKD		MKD	North Macedonia	MKD
116	116000 - 116999	MNE		MNE	Montenegro	MNE
117	117000 - 117999	ALB		ALB	Albania	ALB
118	118000 - 118999	GRC		GRC	Greece	GRC
119	119000 - 119999	TUR		TUR	Turkey	TUR
120	120000 - 120999	NLD		NLD	Netherlands	NLD
121	121000 - 121999	LUX		LUX	Luxembourg	LUX
122	122000 - 122999	DEU		DEU	Germany	DEU
123	123000 - 123999	BEL		BEL	Belgium	BEL
124	124000 - 124699	FRA	FRA	FRA	France - French	FRA-FRA
124	124700 - 124999	FRA	OT	FRA_OT	France - Overseas territory	FRA-OT
125	125000 - 125999	CHE		CHE	Switzerland	CHE
126	126000 - 126999	ITA		ITA	Italy	ITA
127	127000 - 127999	PRT		PRT	Portugal	PRT
128	128000 - 128999	ESP		ESP	Spain	ESP
129	129000 - 129999	EUR			European Organization	EUR
131	131000 - 131799	DNK	DNK	DNK	Denmark - Denmark	DNK-DNK
131	131800 - 131899	DNK	FRO	FRO	Denmark - Faroe Islands	DNK-FRO



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
131	131900 - 131999	DNK	GRL	GRL	Denmark - Greenland	DNK-GRL
132	132000 - 132999	NOR		NOR	Norway	NOR
133	133000 - 133999	SWE		SWE	Sweden	SWE
134	134000 - 134899	FIN	FIN	FIN	Finland - Finland	FIN-FIN
134	134900 - 134999	FIN	ALA	FIN	Finland - Aland	FIN-ALA
140	140000 - 140999	GBR	GBR	GBR	Great Britain - Great Britain	GBR-GBR
141	141000 - 141999	GIB		GIB	Gibraltar	GIB
142	142000 - 142999	IMN		IMN	Isle of Man	IMN
143	143000 - 143999	JEY		JEY	Jersey	JEY
144	144000 - 144999	GGY		GGY	Guernsey	GGY
145	145000 - 145999	ALD		ALD	Alderney	ALD
146	146000 - 146999	GBR	NIR	GBR	Great Britain - Northern Ireland	GBR-NIR
149	149000 - 149999	IRL		IRL	Ireland	IRL
171	171000 - 171999	RUS		RUS	Russia	RUS
172	172000 - 172799	UKR		UKR	Ukraine	UKR
172	172800 - 172899	UKR	LPR	UKR	Ukraine - Luhansk	UKR-LPR
172	172900 - 172999	UKR	DPR	UKR	Ukraine - Donetsk	UKR-DPR
173	173000 - 173799	MDA	MDA	MDA	Moldavia - Moldavia	MDA-MDA
173	173800 - 173999	MDA	RMN	MDA	Moldavia - Transnistria	MDA-RMN

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
174	174000 - 174999	BLR		BLR	Belarus	BLR
175	175000 - 175799	GEO	GEO	GEO	Georgia - Georgia	GEO-GEO
175	175800 - 175899	GEO	ABH	GEO	Georgia - Abkhazia	GEO-ABH
175	175900 - 175999	GEO	RSO	GEO	Georgia - South Ossetia	GEO-RSO
176	176000 - 176999	AZE		AZE	Azerbaijan	AZE
177	177000 - 177999	ARM		ARM	Armenia	ARM
178	178000 - 178999	KAZ		KAZ	Kazakhstan	KAZ
180	180000 - 180999	AND		AND	Andorra	AND
181	181000 - 181999	MCO		MCO	Monaco	MCO
182	182000 - 182999	LIE		LIE	Liechtenstein	LIE
183	183000 - 183999	SMR		SMR	San Marino	SMR
184	184000 - 184999	VAT		VAT	Vatican City	VAT
185	185000 - 185999	RKS		RKS	Kosovo	RKS
190	190000 - 190999	ISL		ISL	Iceland	ISL
191	191000 - 191999	MLT		MLT	Malta	MLT
192	192000 - 192799	CYP	SCP	CYP	South Cyprus	CYP-SCP
192	192800 - 192899	CYP	UCP		UN Cyprus	CYP-UCP
192	192900 - 192999	CYP	NCP	CYP	North Cyprus	CYP-NCP
193	193000 - 193999	SJM		SJM	Svalbard and Jan Mayen	SJM

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
Middle East and Asia						
201	201000 - 201999	ISR		ISR	Israel	ISR
202	202000 - 202999	PSE		PSE	Palestine	PSE
206	206000 - 206999	SYR		SYR	Syria	SYR
207	207000 - 207999	LBN		LBN	Lebanon	LBN
208	208000 - 208099	IRQ	MIX	IRQ	Iraq - Iraq MIX	IRQ-MIX
208	208100 - 208149	IRQ	BAG	IRQ	Iraq - Baghdad	IQ-BG
208	208150 - 208199	IRQ	SAL	IRQ	Iraq - Salah ad-Din	IQ-SD
208	208200 - 208249	IRQ	DIY	IRQ	Iraq - Diyala	IQ-DI
208	208250 - 208299	IRQ	WAS	IRQ	Iraq - Wasit	IQ-WA
208	208300 - 208349	IRQ	MAY	IRQ	Iraq - Maysan	IQ-MA
208	208350 - 208399	IRQ	ALB	IRQ	Iraq - Al-Basrah	IQ-BA
208	208400 - 208449	IRQ	BAB	IRQ	Iraq - Babil	IQ-BB
208	208450 - 208499	IRQ	ALA	IRQ	Iraq - Al-Anbar	IQ-AN
208	208500 - 208549	IRQ	KIR	IRQ	Iraq - Kirkuk	IQ-KI
208	208550 - 208599	IRQ	ALK	IRQ	Iraq - Al-Karbala	IQ-KA
208	208600 - 208649	IRQ	NIN	IRQ	Iraq - Niniwa	IQ-NI
208	208650 - 208699	IRQ	ANA	IRQ	Iraq - An-Najaf	IQ-NA
208	208700 - 208749	IRQ	DHI	IRQ	Iraq - Dhi Qar	IQ-DQ
208	208750 - 208799	IRQ	ALQ	IRQ	Iraq - Al-Qadisiyyah	IQ-QA
208	208800 - 208849	IRQ	ALM	IRQ	Iraq - Al-Muthanna	IQ-MU
208	208850 - 208874	IRQ	DIH	IRQ	Iraq - Dahuk	IQ-DA
208	208900 - 208924	IRQ	LSE	IRQ	Iraq - Al-Sulaimaniyah	IQ-SU
208	208925 - 208949	IRQ	COM	IRQ	Iraq - Common	
208	208950 - 208974	IRQ	ARB	IRQ	Iraq - Arbil	IQ-AR
208	208975 - 208999	IRQ	HAL	IRQ	Iraq - Halabja	IQ-HA
209	209000 - 209999	JOR		JOR	Jordan	JOR
210	210000 - 210999	SAU		SAU	Saudi Arabia	SAU
211	211000 - 211999	KWT		KWT	Kuwait	KWT
212	212000 - 212099	ARE	GOV		United Arab Emirates - Government ARE	ARE-GOV
212	212100 - 212199	ARE	DUB	ARE	United Arab Emirates - Dubai	AE-DU
212	212200 - 212299	ARE	AD	ARE	United Arab Emirates - Abu Dhabi	AE-AZ



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
212	212300 - 212399	ARE	AJM	ARE	United Arab Emirates - Ajman	AE-AJ
212	212400 - 212499	ARE	FUJ	ARE	United Arab Emirates - Fujairah	AE-FU
212	212500 - 212599	ARE	SHJ	ARE	United Arab Emirates - Sharjah	AE-SH
212	212600 - 212699	ARE	UAQ	ARE	United Arab Emirates - Um al-Quwain	AE-UQ
212	212700 - 212799	ARE	RAK	ARE	United Arab Emirates - Ras al-Khaimah	AE-RK
212	212800 - 212899	ARE	UNK	ARE	United Arab Emirates - Unknown ARE	ARE-UNK
212	212900 - 212999	ARE	DUB	ARE	United Arab Emirates - Dubai	AE-DU
213	213000 - 213999	QAT		QAT	Qatar	QAT
214	214000 - 214999	BHR		BHR	Bahrain	BHR
215	215000 - 215999	OMN		OMN	Oman	OMN
216	216000 - 216999	YEM		YEM	Yemen	YEM
217	217000 - 217099	ARE	AD	ARE	United Arab Emirates - Abu Dhabi	AE-AZ
217	217100 - 217199	ARE	RAK	ARE	United Arab Emirates - Ras al-Khaimah	AE-RK
220	220000 - 220899	IRN	IRN	IRN	Iran - Iran	IRN-IRN
220	220900 - 220999	IRN	INL	IRN	Iran - International Iran	IRN-INT
221	221000 - 221999	UZB		UZB	Uzbekistan	UZB
222	222000 - 222999	TKM		TKM	Turkmenistan	TKM
223	223000 - 223999	TJK		TJK	Tajikistan	TJK
224	224000 - 224999	KGZ		KGZ	Kyrgyzstan	KGZ
225	225000 - 225999	MNG		MNG	Mongolia	MNG
226	226000 - 226999	AFG		AFG	Afghanistan	AFG
227	227000 - 227999	PAK		PAK	Pakistan	PAK
240	240000 - 240999	CHN		CHN	China	CHN
241	241000 - 241999	HKG		HKG	Hong Kong	HKG
242	242000 - 242999	MAC		MAC	Macau	MAC
243	243000 - 243999	TWN		TWN	Taiwan	TWN
246	246000 - 246999	KOR		KOR	Korea South	KOR
247	247000 - 247999	PRK		PRK	Korea North	PRK
250	250000 - 250999	IND		IND	India	IND
251	251000 - 251999	NPL		NPL	Nepal	NPL
252	252000 - 252999	BTN		BTN	Bhutan	BTN



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
253	253000 - 253030	BGD	DHK	BGD	Bangladesh - Dhaka	BD-13
253	253031 - 253060	BGD	NNJ	BGD	Bangladesh - Narayanganj	BD-40
253	253061 - 253090	BGD	KHN	BGD	Bangladesh - Khulna	BD-27
253	253091 - 253120	BGD	GAZ	BGD	Bangladesh - Gazipur	BD-18
253	253121 - 253150	BGD	KUS	BGD	Bangladesh - Kushtia	BD-30
253	253151 - 253180	BGD	NDI	BGD	Bangladesh - Narsingdi	BD-42
253	253181 - 253210	BGD	RNG	BGD	Bangladesh - Rangamati	BD-56
253	253211 - 253240	BGD	SYL	BGD	Bangladesh - Sylhet	BD-60
253	253241 - 253270	BGD	BAR	BGD	Bangladesh - Barisal	BD-06
253	253271 - 253300	BGD	BOG	BGD	Bangladesh - Bogra	BD-03
253	253301 - 253330	BGD	MYM	BGD	Bangladesh - Mymensingh	BD-34
253	253331 - 253360	BGD	CHI	BGD	Bangladesh - Chittagong	BD-10
253	253361 - 253390	BGD	JES	BGD	Bangladesh - Jessore	BD-22
253	253391 - 253420	BGD	SIR	BGD	Bangladesh - Sirajganj	BD-59
253	253421 - 253450	BGD	BGE	BGD	Bangladesh - Bagerhat	BD-05
253	253451 - 253480	BGD	BND	BGD	Bangladesh - Bandarban	BD-01
253	253481 - 253510	BGD	BRG	BGD	Bangladesh - Barguna	BD-02
253	253511 - 253540	BGD	BHO	BGD	Bangladesh - Bhola	BD-07
253	253541 - 253570	BGD	BRH	BGD	Bangladesh - Brahmanbaria	BD-04
253	253571 - 253600	BGD	CND	BGD	Bangladesh - Chandpur	BD-09
253	253601 - 253630	BGD	CUA	BGD	Bangladesh - Chuadanga	BD-12

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
253	253631 - 253660	BGD	CLA	BGD	Bangladesh - Comilla	BD-08
253	253661 - 253690	BGD	COX	BGD	Bangladesh - Cox's Bazar	BD-11
253	253691 - 253720	BGD	DIN	BGD	Bangladesh - Dinajpur	BD-14
253	253721 - 253750	BGD	FAR	BGD	Bangladesh - Faridpur	BD-15
253	253751 - 253780	BGD	FEN	BGD	Bangladesh - Feni	BD-16
253	253781 - 253810	BGD	GAI	BGD	Bangladesh - Gaibandha	BD-19
253	253811 - 253840	BGD	GOP	BGD	Bangladesh - Gopalganj	BD-17
253	253841 - 253870	BGD	HAB	BGD	Bangladesh - Habiganj	BD-20
253	253871 - 253900	BGD	JAP	BGD	Bangladesh - Jamalpur	BD-21
253	253901 - 253930	BGD	JHA	BGD	Bangladesh - Jhalakati	BD-25
253	253931 - 253960	BGD	JHE	BGD	Bangladesh - Jhenaidah	BD-23
253	253961 - 253990	BGD	JOY	BGD	Bangladesh - Joypurhat	BD-24
253	253991 - 254020	BGD	KGR	BGD	Bangladesh - Khagrachari	BD-29
254	254021 - 254050	BGD	KIS	BGD	Bangladesh - Kishoreganj	BD-26
254	254051 - 254080	BGD	KUR	BGD	Bangladesh - Kurigram	BD-28
254	254081 - 254110	BGD	LAK	BGD	Bangladesh - Lakshmipur	BD-31
254	254111 - 254140	BGD	LAL	BGD	Bangladesh - Lalmonirhat	BD-32
254	254141 - 254170	BGD	MAD	BGD	Bangladesh - Madaripur	BD-36
254	254171 - 254200	BGD	MAG	BGD	Bangladesh - Magura	BD-37
254	254201 - 254230	BGD	MAN	BGD	Bangladesh - Manikganj	BD-33
254	254231 - 254260	BGD	MEH	BGD	Bangladesh - Meherpur	BD-39
254	254261 - 254290	BGD	MOU	BGD	Bangladesh - Moulvibazar	BD-38
254	254291 - 254320	BGD	MUN	BGD	Bangladesh - Munshiganj	BD-35
254	254321 - 254350	BGD	NAO	BGD	Bangladesh - Naogaon	BD-48
254	254351 - 254380	BGD	NRI	BGD	Bangladesh - Narail	BD-43
254	254381 - 254410	BGD	NAT	BGD	Bangladesh - Natore	BD-44
254	254411 - 254440	BGD	NAW	BGD	Bangladesh - Nawabganj	BD-45
254	254441 - 254470	BGD	NET	BGD	Bangladesh - Netrakona	BD-41
254	254471 - 254500	BGD	NIL	BGD	Bangladesh - Nilphamari	BD-46
254	254501 - 254530	BGD	NOA	BGD	Bangladesh - Noakhali	BD-47
254	254531 - 254560	BGD	PAB	BGD	Bangladesh - Pabna	BD-49
254	254561 - 254590	BGD	PCG	BGD	Bangladesh - Panchagarh	BD-52
254	254591 - 254620	BGD	PKH	BGD	Bangladesh - Patuakhali	BD-51
254	254621 - 254650	BGD	PJP	BGD	Bangladesh - Pirojpur	BD-50



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
254	254651 - 254680	BGD	PJB	BGD	Bangladesh - Rajbari	BD-53
254	254681 - 254710	BGD	RJS	BGD	Bangladesh - Rajshahi	BD-54
254	254711 - 254740	BGD	RGP	BGD	Bangladesh - Rangpur	BD-55
254	254741 - 254770	BGD	SKH	BGD	Bangladesh - Satkhira	BD-58
254	254771 - 254800	BGD	SHP	BGD	Bangladesh - Shariatpur	BD-62
254	254801 - 254830	BGD	SRP	BGD	Bangladesh - Sherpur	BD-57
254	254831 - 254860	BGD	SJG	BGD	Bangladesh - Sunamganj	BD-61
254	254861 - 254890	BGD	TAN	BGD	Bangladesh - Tangail	BD-63
254	254891 - 254920	BGD	TRG	BGD	Bangladesh - Thakurgaon	BD-64
254	254921 - 254950	BGD	UNK	BGD	Bangladesh - Unknown BGD	BD-UNK
254	254951 - 254999	BGD	OTH	BGD	Bangladesh - OtherCdGovUn	BD-OTH
256	256000 - 256069	THA	Krabi	THA	Thailand - Krabi	TH-81
256	256070 - 256139	THA	KAN	THA	Thailand - Kanchanaburi	TH-71
256	256140 - 256189	THA	KHO	THA	Thailand - Khon Kaen	TH-40
256	256190 - 256259	THA	CCH	THA	Thailand - Chachoengsao	TH-24
256	256260 - 256349	THA	CBU	THA	Thailand - Chon Buri	TH-20
256	256350 - 256419	THA	CHR	THA	Thailand - Chiang Rai	TH-57
256	256420 - 256489	THA	CHM	THA	Thailand - Chiang Mai	TH-50

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
256	256490 - 256559	THA	TNG	THA	Thailand - Trang	TH-92
256	256560 - 256629	THA	NPA	THA	Thailand - Nakhon Pathom	TH-73
256	256630 - 256699	THA	NRA	THA	Thailand - Nakhon Ratchasima	TH-30
256	256700 - 256769	THA	NST	THA	Thailand - Nakhon Si Thammarat	TH-80
256	256770 - 256839	THA	NSA	THA	Thailand - Nakhon Sawan	TH-60
256	256840 - 256909	THA	NON	THA	Thailand - Nonthaburi	TH-12
256-257	256910 - 257009	THA	PAT	THA	Thailand - Pattani	TH-94
257	257010 - 257079	THA	PNA	THA	Thailand - Phra Nakhon Si Ayutthaya	TH-14
257	257080 - 257149	THA	PAR	THA	Thailand - Phetchaburi	TH-76
257	257150 - 257219	THA	PHU	THA	Thailand - Phuket	TH-83
257	257220 - 257309	THA	YAL	THA	Thailand - Yala	TH-95
257	257310 - 257379	THA	RAT	THA	Thailand - Ratchaburi	TH-70
257	257380 - 257449	THA	SAT	THA	Thailand - Satun	TH-91
257	257450 - 257519	THA	SSM	THA	Thailand - Samut Songkhram	TH-75
257	257520 - 257589	THA	SBU	THA	Thailand - Saraburi	TH-19
257	257590 - 257709	THA	SUB	THA	Thailand - Suphan Buri	TH-72
257	257710 - 257809	THA	SUT	THA	Thailand - Surat Thani	TH-84
257	257810 - 257879	THA	SUR	THA	Thailand - Surin	TH-32
257	257880 - 257969	THA	BAT	THA	Thailand - Batong	
257-258	257970 - 258269	THA	Bangkok	THA	Thailand - Bangkok	TH-10
258	258270 - 258399	THA	SOK	THA	Thailand - Songkhla	TH-90
258	258400 - 258849	THA	UNK	THA	Thailand - Unknown THA	
258	258850 - 258919	THA	CNA	THA	Thailand - Chai Nat	TH-18
258-259	258920 - 259019	THA	CHU	THA	Thailand - Chumphon	TH-86
259	259020 - 259109	THA	NAR	THA	Thailand - Narathiwat	TH-96
259	259110 - 259179	THA	BRA	THA	Thailand - Buri Ram	TH-31
259	259180 - 259249	THA	PKK	THA	Thailand - Prachuap Khiri Khan	TH-77
259	259250 - 259319	THA	PBU	THA	Thailand - Prachin Buri	TH-25
259	259320 - 259389	THA	RET	THA	Thailand - Roi Et	TH-45
259	259390 - 259459	THA	RAN	THA	Thailand - Ranong	TH-85
259	259460 - 259529	THA	LOB	THA	Thailand - Lop Buri	TH-16
259	259530 - 259599	THA	SNA	THA	Thailand - Sakon Nakhon	TH-47
259	259600 - 259669	THA	SPR	THA	Thailand - Samut Prakan	TH-11
259	259670 - 259739	THA	SSN	THA	Thailand - Samut Sakhon	TH-74
259	259740 - 259809	THA	SAK	THA	Thailand - Sa Kaeo	TH-27
259	259810 - 259879	THA	UDT	THA	Thailand - Udon Thani	TH-41
259	259880 - 259949	THA	GOV	THA	Thailand - Government THA	



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
259-260	259950 - 260009	THA	KAL	THA	Thailand - Kalasin	TH-46
260	260010 - 260059	THA	KAM	THA	Thailand - Kamphaeng Phet	TH-62
260	260060 - 260109	THA	CNT	THA	Thailand - Chanthaburi	TH-22
260	260110 - 260169	THA	CIY	THA	Thailand - Chaiyaphum	TH-36
260	260170 - 260219	THA	TRA	THA	Thailand - Trat	TH-23
260	260220 - 260269	THA	TAK	THA	Thailand - Tak	TH-63
260	260270 - 260319	THA	NNA	THA	Thailand - Nakhon Nayok	TH-26
260	260320 - 260369	THA	NPH	THA	Thailand - Nakhon Phanom	TH-48
260	260370 - 260419	THA	NAN	THA	Thailand - Nan	TH-55
260	260420 - 260469	THA	PTH	THA	Thailand - Pathum Thani	TH-13
260	260470 - 260519	THA	PYA	THA	Thailand - Phayao	TH-56
260	260520 - 260569	THA	PNG	THA	Thailand - Phangnga	TH-82
260	260570 - 260619	THA	PTT	THA	Thailand - Phatthalung	TH-93
260	260620 - 260669	THA	PCH	THA	Thailand - Phichit	TH-66
260	260670 - 260729	THA	PTS	THA	Thailand - Phitsanulok	TH-65

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
260	260730 - 260779	THA	PAN	THA	Thailand - Phetchabun	TH-67
260	260780 - 260829	THA	PRA	THA	Thailand - Phrae	TH-54
260	260830 - 260879	THA	MSA	THA	Thailand - Maha Sarakham	TH-44
260	260880 - 260929	THA	MUK	THA	Thailand - Mukdahan	TH-49
260	260930 - 260979	THA	MHS	THA	Thailand - Mae Hong Son	TH-58
260-261	260980 - 261029	THA	YAS	THA	Thailand - Yasothon	TH-35
261	261030 - 261079	THA	RAY	THA	Thailand - Rayong	TH-21
261	261080 - 261129	THA	LPA	THA	Thailand - Lampang	TH-52
261	261130 - 261179	THA	LPH	THA	Thailand - Lamphun	TH-51
261	261180 - 261239	THA	LOE	THA	Thailand - Loei	TH-42
261	261240 - 261299	THA	SSK	THA	Thailand - Si Sa Ket	TH-33
261	261300 - 261349	THA	SIB	THA	Thailand - Sing Buri	TH-17
261	261350 - 261399	THA	SUK	THA	Thailand - Sukhothai	TH-64
261	261400 - 261449	THA	NKH	THA	Thailand - Nong Khai	TH-43
261	261450 - 261499	THA	NBL	THA	Thailand - Nong Bua Lam Phu	TH-39
261	261500 - 261549	THA	ATH	THA	Thailand - Ang Thong	TH-15
261	261550 - 261599	THA	UTT	THA	Thailand - Uttaradit	TH-53
261	261600 - 261649	THA	UTH	THA	Thailand - Uthai Thani	TH-61
261	261650 - 261699	THA	UBR	THA	Thailand - Ubon Ratchathani	TH-34
261	261700 - 261749	THA	AMC	THA	Thailand - Amnat Charoen	TH-37
261	261750 - 261799	THA	BUK	THA	Thailand - Bueng Kan	TH-38
262	262000 - 262999	MYS		MYS	Malaysia	MYS
263	263000 - 263999	SGP		SGP	Singapore	SGP
264	264000 - 264999	MMR		MMR	Myanmar	MMR
265	265000 - 265049	LAO	GOV	LAO	Laos - Government LAO	
265	265050 - 265099	LAO	UNK	LAO	Laos - Unknown LAO	
265	265100 - 265149	LAO	ATT	LAO	Laos - Attapeu	LA-AT
265	265150 - 265199	LAO	BOK	LAO	Laos - Bokeo	LA-BK
265	265200 - 265249	LAO	BOL	LAO	Laos - Bolikhamsai	LA-BL
265	265250 - 265299	LAO	CHA	LAO	Laos - Champasak	LA-CH
265	265300 - 265349	LAO	HOU	LAO	Laos - Houaphanh	LA-HO
265	265350 - 265399	LAO	KHA	LAO	Laos - Khammouane	LA-KH

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
265	265400 - 265449	LAO	LNA	LAO	Laos - Luang Namtha	LA-LM
265	265450 - 265499	LAO	LPR	LAO	Laos - Luang Prabang	LA-LP
265	265500 - 265549	LAO	ODU	LAO	Laos - Oudomxay	LA-OU
265	265550 - 265599	LAO	PHO	LAO	Laos - Phongsali	LA-PH
265	265600 - 265649	LAO	SAL	LAO	Laos - Salavan	LA-SL
265	265650 - 265699	LAO	SAV	LAO	Laos - Savannakhet	LA-SV
265	265700 - 265749	LAO	VIE	LAO	Laos - Vientiane Province	LA-VI
265	265750 - 265799	LAO	UVI	LAO	Laos - Urban Vientiane	LA-VT
265	265800 - 265849	LAO	SAI	LAO	Laos - Sainyabuli	LA-XA
265	265850 - 265899	LAO	SEK	LAO	Laos - Sekong	LA-XE
265	265900 - 265949	LAO	XIA	LAO	Laos - Xiangkhouang	LA-XI
265	265950 - 265999	LAO	XAI	LAO	Laos - Xaisomboun	LA-XS
266	266000 - 266024	KHM	BAN	KHM	Cambodia - Banteay Meanchey	KH-1
266	266025 - 266049	KHM	BAT	KHM	Cambodia - Battambang	KH-2
266	266050 - 266074	KHM	KCM	KHM	Cambodia - Kampong Cham	KH-3
266	266075 - 266099	KHM	KCG	KHM	Cambodia - Kampong Chhnang	KH-4
266	266100 - 266124	KHM	KSP	KHM	Cambodia - Kampong Speu	KH-5
266	266125 - 266149	KHM	KTH	KHM	Cambodia - Kampong Thom	KH-6
266	266150 - 266174	KHM	KAM	KHM	Cambodia - Kampot	KH-7

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
266	266175 - 266199	KHM	KAN	KHM	Cambodia - Kandal	KH-8
266	266200 - 266224	KHM	KOH	KHM	Cambodia - Koh Kong	KH-9
266	266225 - 266249	KHM	KEP	KHM	Cambodia - Kep	KH-23
266	266250 - 266274	KHM	KRA	KHM	Cambodia - Kratie	KH-10
266	266275 - 266299	KHM	MON	KHM	Cambodia - Mondulhiri	KH-11
266	266300 - 266324	KHM	ODM	KHM	Cambodia - Oddar Meanchey	KH-22
266	266325 - 266349	KHM	PAI	KHM	Cambodia - Pailin	KH-24
266	266350 - 266374	KHM	PHN	KHM	Cambodia - Phnom Penh	KH-12
266	266375 - 266399	KHM	SIH	KHM	Cambodia - Sihanoukville	KH-18
266	266400 - 266424	KHM	PVI	KHM	Cambodia - Preah Vihear	KH-13
266	266425 - 266449	KHM	PUR	KHM	Cambodia - Pursat	KH-15
266	266450 - 266474	KHM	PVE	KHM	Cambodia - Prey Veng	KH-14
266	266475 - 266499	KHM	RAT	KHM	Cambodia - Ratanakiri	KH-16
266	266500 - 266524	KHM	SIE	KHM	Cambodia - Siem Reap	KH-17
266	266525 - 266549	KHM	STU	KHM	Cambodia - Stung Treng	KH-19
266	266550 - 266574	KHM	SVA	KHM	Cambodia - Svay Rieng	KH-20
266	266575 - 266599	KHM	TAK	KHM	Cambodia - Takeo	KH-21
266	266600 - 266649	KHM	GOV	KHM	Cambodia - Government	
266	266650 - 266699	KHM	UNK	KHM	Cambodia - Unknown	
266	266700 - 266999	KHM	TBO	KHM	Tbong Khmum	KH-22
267	267000 - 267999	VNM		VNM	Vietnam	VNM
270	270000 - 270999	IDN		IDN	Indonesia	IDN
271	271000 - 271999	PNG		PNG	Papua New Guinea	PNG
272	272000 - 272999	TLS		TLS	Timor Leste	TLS
273	273000 - 273999	BRN		BRN	Brunei	BRN
290	290000 - 290999	LKA		LKA	Sri Lanka	LKA
291	291000 - 291999	PHL		PHL	Philippines	PHL
292	292000 - 292999	JPN		JPN	Japan	JPN
293	293000 - 293999	MDV		MDV	Maldives	MDV
294	294000 - 294999	CXR		CXR	Christmas Island	CXR
295	295000 - 295999	CCK		CCK	Keeling Islands	CCK

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
Australia – New Zealand						
300	300000 – 300999	AUS	UNK		Australia - Unknown AUS	
301	301000 – 301499	AUS	FIS		Australia - Federal-Interstate	
301	301500 – 301999	AUS	GOV		Australia - Government AUS	
302	302000 – 302999	AUS	ACT	AU-ACT	Australia - Australian Capital Territory	AU-ACT
303	303000 – 303999	AUS	NT	AU-NT	Australia - Northern Territory	AU-NT
304	304000 – 304999	AUS	NSW	AU-NSW	Australia - New South Wales	AU-NSW
305	305000 – 305999	AUS	QLD	AU-QLD	Australia - Queensland	AU-QLD
306	306000 – 306999	AUS	SA	AU-SA	Australia - South Australia	AU-SA
307	307000 – 307999	AUS	TAS	AU-TAS	Australia - Tasmania	AU-TAS
308	308000 – 308999	AUS	VIC	AU-VIC	Australia - Victoria	AU-VIC
309	309000 – 309999	AUS	WA	AU-WA	Australia - Western Australia	AU-WA
310-311	310000 – 311999	NZL		NZL	New Zealand	NZL
Africa						
401	401000 – 401999	MAR		MAR	Morocco	MAR
402	402000 – 402999	DZA		DZA	Algeria	DZA
403	403000 – 403899	TUN	TUN	TUN	Tunisia	
403	403900 – 403999	TUN	INT	TUN	International Tunisia	TUN-INT
404	404000 – 404999	LBY		LBY	Libya	LBY
405	405000 – 405999	EGY		EGY	Egypt	EGY
406	406000 – 406999	SDN		SDN	Sudan	SDN



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
407	407000 - 407999	ETH		ETH	Ethiopia	ETH
408	408000 - 408999	AGO		AGO	Angola	AGO
409	409000 - 409999	BEN		BEN	Benin	BEN
410	410000 - 410999	GNB		GNB	Guinea-Bissau	GNB
411	411000 - 411999	BFA		BFA	Burkina-Faso	BFA
412	412000 - 412999	BDI		BDI	Burundi	BDI
413	413000 - 413999	COM		COM	Comoros	COM
414	414000 - 414999	TCD		TCD	Chad	TCD
415	415000 - 415999	DJI		DJI	Djibouti	DJI
416	416000 - 416999	GNQ		GNQ	Equatorial Guinea	GNQ
417	417000 - 417999	ERI		ERI	Eritrea	ERI
418	418000 - 418999	GAB		GAB	Gabon	GAB
419	419000 - 419999	GMB		GMB	Gambia	GMB
420	420000 - 420999	GHA		GHA	Ghana	GHA
421	421000 - 421999	GIN		GIN	Guinea	GIN
422	422000 - 422999	CMR		CMR	Cameroon	CMR
423	423000 - 423999	KEN		KEN	Kenya	KEN
424	424000 - 424999	COG		COG	Congo-Brazzaville	COG
425	425000 - 425999	COD		COD	Dem. Rep of Congo	COD
426	426000 - 426999	CAF		CAF	Central African Republic	CAF
427	427000 - 427999	LBR		LBR	Liberia	LBR
428	428000 - 428999	MDG		MDG	Madagascar	MDG
429	429000 - 429999	MWI		MWI	Malawi	MWI
430	430000 - 430999	MLI		MLI	Mali	MLI
431	431000 - 431999	MRT		MRT	Mauritania	MRT
432	432000 - 432999	MUS		MUS	Mauritius	MUS
433	433000 - 433999	NER		NER	Niger	NER
434	434000 - 434999	NGA		NGA	Nigeria	NGA
435	435000 - 435999	RWA		RWA	Ruanda	RWA
436	436000 - 436999	STP		STP	Sao Tome Principe	STP
437	437000 - 437999	SYC		SYC	Seychelles	SYC
438	438000 - 438999	SLE		SLE	Sierra Leone	SLE
439	439000 - 439999	SEN		SEN	Senegal	SEN
440	440000 - 440999	SOM		SOM	Somalia	SOM
441	441000 - 441199	TZA		TZA	Tanzania	TZA
441	441200 - 441399	TZA	ZZB	TZA	Tanzania - Zanzibar	TZA
442	442000 - 442999	TGO		TGO	Togo	TGO
443	443000 - 443999	UGA		UGA	Uganda	UGA
444	444000 - 444999	ZMB		ZMB	Zambia	ZMB
445	445000 - 445999	SSD		SSD	South Sudan	SSD
446	446000 - 446999	CPV		CPV	Cape Verde	CPV
447	447000 - 447999	CIV		CIV	Ivory Coast	CIV
448	448000 - 448999	ESH		ESH	Western Sahara	ESH
480	480000 - 480999	ZAF		ZAF	South Africa	ZAF
481	481000 - 481999	NAM		NAM	Namibia	NAM
482	482000 - 482999	BWA		BWA	Botswana	BWA
483	483000 - 483999	ZWE		ZWE	Zimbabwe	ZWE
484	484000 - 484999	MOZ		MOZ	Mozambique	MOZ
485	485000 - 485999	SWZ		SWZ	Eswatini	SWZ
486	486000 - 486999	LSO		LSO	Lesotho	LSO



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
USA-CANADA (North Americas)						
500	500000 - 500999	USA	GOV		United States of America - Government USA	USA-GOV
501	501000 - 501999	USA	DC	US-DC	United States of America - Columbia	US-DC
502	502000 - 502999	USA	AK	US-AK	United States of America - Alaska	US-AK
503	503000 - 503999	USA	WA	US-WA	United States of America - Washington	US-WA
504	504000 - 504999	USA	OR	US-OR	United States of America - Oregon	US-OR
505	505000 - 505999	USA	CA	US-CA	United States of America - California	US-CA
506	506000 - 506999	USA	ID	US-ID	United States of America - Idaho	US-ID
507	507000 - 507999	USA	NV	US-NV	United States of America - Nevada	US-NV
508	508000 - 508999	USA	MT	US-MT	United States of America - Montana	US-MT
509	509000 - 509999	USA	WY	US-WY	United States of America - Wyoming	US-WY
510	510000 - 510999	USA	UT	US-UT	United States of America - Utah	US-UT
511	511000 - 511999	USA	AZ	US-AZ	United States of America - Arizona	US-AZ
512	512000 - 512999	USA	ND	US-ND	United States of America - North Dakota	US-ND
513	513000 - 513999	USA	SD	US-SD	United States of America - South Dakota	US-SD
514	514000 - 514999	USA	NE	US-NE	United States of America - Nebraska	US-NE
515	515000 - 515999	USA	CO	US-CO	United States of America - Colorado	US-CO
516	516000 - 516999	USA	NM	US-NM	United States of America - New Mexico	US-NM
517	517000 - 517999	USA	KS	US-KS	United States of America - Kansas	US-KS
518	518000 - 518999	USA	OK	US-OK	United States of America - Oklahoma	US-OK
519	519000 - 519999	USA	TX	US-TX	United States of America - Texas	US-TX
520	520000 - 520999	USA	AR	US-AR	United States of America - Arkansas	US-AR
521	521000 - 521999	USA	MN	US-MR	United States of America - Minnesota	US-MN
522	522000 - 522999	USA	WI	US-WI	United States of America - Wisconsin	US-WI
523	523000 - 523999	USA	IA	US-IA	United States of America - Iowa	US-IA
524	524000 - 524999	USA	IL	US-IL	United States of America - Illinois	US-IL
525	525000 - 525999	USA	MO	US-MO	United States of America - Missouri	US-MO
526	526000 - 526999	USA	MI	US-MI	United States of America - Michigan	US-MI
527	527000 - 527999	USA	IN	US-IN	United States of America - Indiana	US-IN
528	528000 - 528999	USA	OH	US-OH	United States of America - Ohio	US-OH
529	529000 - 529999	USA	KY	US-KY	United States of America - Kentucky	US-KY
530	530000 - 530999	USA	AL	US-AL	United States of America - Alabama	US-AL
531	531000 - 531999	USA	TN	US-TN	United States of America - Tennessee	US-TN
532	532000 - 532999	USA	LA	US-LA	United States of America - Louisiana	US-LA
533	533000 - 533999	USA	MS	US-MS	United States of America - Mississippi	US-MS
534	534000 - 534999	USA	ME	US-ME	United States of America - Maine	US-ME
535	535000 - 535999	USA	VT	US-VT	United States of America - Vermont	US-VT
536	536000 - 536999	USA	NH	US-NH	United States of America - New Hampshire	US-NH
537	537000 - 537999	USA	CT	US-CT	United States of America - Connecticut	US-CT



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
538	538000 - 538999	USA	MA	US-MA	United States of America - Massachusetts	US-MA
539	539000 - 539999	USA	RI	US-RI	United States of America - Rhode Island	US-RI
540	540000 - 540999	USA	NY	US-NY	United States of America - New York	US-NY
541	541000 - 541999	USA	NJ	US-NJ	United States of America - New Jersey	US-NJ
542	542000 - 542999	USA	DE	US-DE	United States of America - Delaware	US-DE
543	543000 - 543999	USA	PA	US-PA	United States of America - Pennsylvania	US-PA
544	544000 - 544999	USA	MD	US-MD	United States of America - Maryland	US-MD
545	545000 - 545999	USA	VA	US-VA	United States of America - Virginia	US-VA
546	546000 - 546999	USA	WV	US-WV	United States of America - West Virginia	US-WV
547	547000 - 547999	USA	NC	US-NC	United States of America - North Carolina	US-NC
548	548000 - 548999	USA	SC	US-SC	United States of America - South Carolina	US-SC
549	549000 - 549999	USA	GA	US-GA	United States of America - Georgia	US-GA
550	550000 - 550999	USA	FL	US-FL	United States of America - Florida	US-FL

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
551	551000 - 551999	USA	HI	US-HI	United States of America - Hawaii	US-HI
552	552000 - 552999	USA	PRI	US-PRI	United States of America - Puerto Rico	US-PR
553	553000 - 553999	USA	GU	US-GU	United States of America - Guam	US-GU
554	554000 - 554999	USA	AS	US-AS	United States of America - American Samoa	US-AS
555	555000 - 555999	USA	VI	US-VI	United States of America - Virgin Islands	US-VI
556	556000 - 556999	USA	TX	US-TX	United States of America - Texas	US-TX
557	557000 - 559999	USA	VA	US-VA	United States of America - Virginia	US-VA
560	560000 - 560999	CAN	FED		Canada - Federal	CAN-FED
561	561000 - 561999	CAN	BC	CA-BC	Canada - British Columbia	CA-BC
562	562000 - 562999	CAN	AB	CA-AB	Canada - Alberta	CA-AB
563	563000 - 563999	CAN	SK	CA-SK	Canada - Saskatchewan	CA-SK
564	564000 - 564999	CAN	MB	CA-MB	Canada - Manitoba	CA-MB
565	565000 - 565999	CAN	ON	CA-ON	Canada - Ontario	CA-ON
566	566000 - 566999	CAN	QC	CA-QC	Canada - Quebec	CA-QC
567	567000 - 567999	CAN	NS	CA-NS	Canada - Nova Scotia	CA-NS
568	568000 - 568999	CAN	NB	CA-NB	Canada - New Brunswick	CA-NB
569	569000 - 569999	CAN	NL	CA-NL	Canada - Newfoundland Labrador	CA-NL
570	570000 - 570999	CAN	NWT	CA-NT	Canada - North-West Territories	CA-NT
571	571000 - 571999	CAN	NU	CA-NU	Canada - Nunavut	CA-NU
572	572000 - 572999	CAN	PEI	CA-PE	Canada - Prince Edouard Island	CA-PE
573	573000 - 573999	CAN	YU	CA-YU	Canada - Yukon	CA-YT
577-580	577000 - 580999	USA	VA	US-VA	United States of America - Virginia	US-VA
581	581000 - 581999	USA	TX	US-TX	United States of America - Texas	US-TX
582	582000 - 583999	USA	FL	US-FL	United States of America - Florida	US-FL
584	584000 - 584999	USA	IL	US-IL	United States of America - Illinois	US-IL
585	585000 - 585999	USA	NY	US-NY	United States of America - New York	US-NY
586	586000 - 586999	USA	MNP	US-MP	United States of America - Northern Mariana Islands	US-MP
587	587000 - 588999	USA	AZ	US-AZ	United States of America - Arizona	US-AZ
589	589000 - 589999	USA	NC	US-NC	United States of America - North Carolina	US-NC
590	590000 - 590999	USA	OK	US-OK	United States of America - Oklahoma	US-OK



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
South America						
602	602000 - 602999	GTM		GTM	Guatemala	GTM
603	603000 - 603999	BLZ		BLZ	Belize	BLZ
604	604000 - 604999	SLV		SLV	El Salvador	SLV
605	605000 - 605999	NIC		NIC	Nicaragua	NIC
606	606000 - 606999	HND		HND	Honduras	HND
607	607000 - 607999	CRI		CRI	Costa Rica	CRI
608	608000 - 608999	PAN		PAN	Panama	PAN
609	609000 - 609099	CUW		CUW	Curacao	CUW
609	609100 - 609199	ABW		ABW	Aruba	ABW
609	609200 - 609299	SXM		SXM	Sint Maarten	SXM
610	610000 - 610999	MEX	COM	MEX	Mexico - Mexico common	MEX-COM
611	611000 - 611999	MEX	DF	MEX	Mexico - Distrito Federal	MX-DFE
612	612000 - 612999	MEX	FED	MEX	Mexico - Mexico (federal)	MEX-FED
613	613000 - 613999	MEX	AGS	MEX	Mexico - Aguascalientes	MX-AGU
614	614000 - 614999	MEX	BC	MEX	Mexico - Baja California	MX-BCN
615	615000 - 615999	MEX	BCS	MEX	Mexico - Baja California Sur	MX-BCS
616	616000 - 616999	MEX	CAMP	MEX	Mexico - Campeche	MX-CAM
617	617000 - 617999	MEX	CHIS	MEX	Mexico - Chiapas	MX-CHP
618	618000 - 618999	MEX	CHIH	MEX	Mexico - Chihuahua	MX-CHH
619	619000 - 619999	MEX	COAH	MEX	Mexico - Coahuila	MX-COA
620	620000 - 620999	MEX	COL	MEX	Mexico - Colima	MX-COL
621	621000 - 621999	MEX	DGO	MEX	Mexico - Durango	MX-DUR

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
622	622000 - 622999	MEX	GTO	MEX	Mexico - Guanajuato	MX-GUA
623	623000 - 623999	MEX	GRO	MEX	Mexico - Guerrero	MX-GRO
624	624000 - 624999	MEX	HGO	MEX	Mexico - Hidalgo	MX-HID
625	625000 - 625999	MEX	JAL	MEX	Mexico - Jalisco	MX-JAL
626	626000 - 626999	MEX	MEX	MEX	Mexico - Mexico (Etat)	MX-MEX
627	627000 - 627999	MEX	MICH	MEX	Mexico - Michoacan	MX-MIC
628	628000 - 628999	MEX	MOR	MEX	Mexico - Morelos	MX-MOR
629	629000 - 629999	MEX	NAY	MEX	Mexico - Nayarit	MX-NAY
630	630000 - 630999	MEX	NL	MEX	Mexico - Nuevo Leon	MX-NLE
631	631000 - 631999	MEX	OAX	MEX	Mexico - Oaxaca	MX-OAX
632	632000 - 632999	MEX	PUE	MEX	Mexico - Puebla	MX-PUE
633	633000 - 633999	MEX	QRO	MEX	Mexico - Queretaro	MX-QUE
634	634000 - 634999	MEX	QR	MEX	Mexico - Quintana Roo	MX-ROO
635	635000 - 635999	MEX	SLP	MEX	Mexico - San Luis Potosi	MX-SLP
636	636000 - 636999	MEX	SIN	MEX	Mexico - Sinaloa	MX-SIN
637	637000 - 637999	MEX	SON	MEX	Mexico - Sonora	MX-SON
638	638000 - 638999	MEX	TAB	MEX	Mexico - Tabasco	MX-TAB
639	639000 - 639999	MEX	TAMPS	MEX	Mexico - Tamaulipas	MX-TAM
640	640000 - 640999	MEX	TLAX	MEX	Mexico - Tlaxcala	MX-TLA
641	641000 - 641999	MEX	VER	MEX	Mexico - Veracruz	MX-VER
642	642000 - 642999	MEX	YUC	MEX	Mexico - Yucatan	MX-YUC
643	643000 - 643999	MEX	ZAC	MEX	Mexico - Zacatecas	MX-ZAC
644	644000 - 644999	MEX	BCF	MEX	Mexico - Baja California Fronterizo	MEX-BCF
645	645000 - 645999	MEX	CHIH	MEX	Mexico - Chihuahua Fronterizo	MEX-CHI
646	646000 - 646999	MEX	SONF	MEX	Mexico - Sonora Fronterizo	MEX-SON
647	647000 - 647999	MEX	TAMPSF	MEX	Mexico - Tamaulipas Fronterizo	MEX-TAM
651	651000 - 651999	COL		COL	Colombia	COL
652	652000 - 652999	VEN		VEN	Venezuela	VEN
653	653000 - 653999	GUY		GUY	Guyana	GUY
654	654000 - 654999	SUR		SUR	Suriname	SUR
656	656000 - 656999	PER		PER	Peru	PER



Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
657	657000 - 657999	BRA		BRA	Brazil	BRA
658	658000 - 658999	ECU		ECU	Ecuador	ECU
659	659000 - 659999	BOL		BOL	Bolivia	BOL
660	660000 - 660999	PRY		PRY	Paraguay	PRY
661	661000 - 661999	CHL		CHL	Chile	CHL
662	662000 - 662999	ARG		ARG	Argentina	ARG
663	663000 - 663999	URY		URY	Uruguay	URY
665	665000 - 665999	KNA		KNA	St Kitts and Nevis	KNA
666	666000 - 666999	CUB		CUB	Cuba	CUB
667	667000 - 667999	AIA		AIA	Anguilla	AIA
668	668000 - 668999	ATG		ATG	Antigua and Barbuda	ATG
670	670000 - 670999	BHS		BHS	Bahamas	BHS
671	671000 - 671999	BRB		BRB	Barbados	BRB
672	672000 - 672999	BMU		BMU	Bermuda	BMU
673	673000 - 673999	VGB		VGB	British Virgin Islands	VGB
674	674000 - 674999	LCA		LCA	St Lucia	LCA
675	675000 - 675999	DOM		DOM	Dominican Republic	DOM
676	676000 - 676999	DMA		DMA	Dominica	DMA
677	677000 - 677999	VCT		VCT	St Vincent and Grenadines	VCT
678	678000 - 678999	FLK		FLK	Falkland Islands	FLK
680	680000 - 680999	TCA		TCA	Turks and Caicos	TCA

Country Short Code	Range (CountryCode)	Country (short)	State (short)	Location	Country – State (long)	Country – State (CountryName)
681	681000 - 681999	TTO		TTO	Trinidad and Tobago	TTO
682	682000 - 682999	GRD		GRD	Grenada	GRD
685	685000 - 685999	HTI		HTI	Haiti	HTI
686	686000 - 686999	JAM		JAM	Jamaica	JAM
687	687000 - 687999	CYM		CYM	Cayman	CYM
693	693000 - 693999	MSR		MSR	Montserrat	MSR
Pacific						
700	700000 - 700999	MHL		MHL	Marshall Islands	MHL
701	701000 - 701999	COK		COK	Cook Islands	COK
703	703000 - 703999	FJI		FJI	Fiji	FJI
705	705000 - 705999	SLB		SLB	Solomon Islands	SLB
706	706000 - 706999	KIR		KIR	Kiribati	KIR
707	707000 - 707999	WSM		WSM	Samoa	WSM
708	708000 - 708999	FSM		FSM	Micronesia	FSM
709	709000 - 709999	NRU		NRU	Nauru	NRU
710	710000 - 710999	TUV		TUV	Tuvalu	TUV
711	711000 - 711999	NIU		NIU	Niue	NIU
712	712000 - 712999	NFK		NFK	Norfolk Island	NFK
713	713000 - 713999	PLW		PLW	Palau	PLW
714	714000 - 714999	PCN		PCN	Pitcairn Island	PCN
OCR						
995	995500 - 995599	OCR	ILU	-	OCR – ILU	-
995	995600 - 995699	OCR	ISOT	-	OCR – ISO Type	-
995	995700 - 995799	OCR	AMLOC	-	OCR – American Locomotive	-
995	995800 - 995899	OCR	EURLOC	-	OCR – European Locomotive	-
995-996	995900 - 996099	OCR	UIC	-	OCR – UIC	-
996	996100 - 996199	OCR	RUS	-	OCR – Russian wagon	-
996	996200 - 996299	OCR	BRA	-	OCR – Brazilian wagon	-
996	996300 - 996399	OCR	USDOT	-	OCR – USDOT	-
996	996400 - 996499	OCR	CHS	-	OCR – CHASSIS	-
996	996500 - 996599	OCR	AAR	-	OCR – Association of American Railroads	-
996	996600 - 996699	OCR	ACCRUS A	-	OCR – ACCR_USA	-
996	996700 - 996799	OCR	ISO	-	OCR – ISO	-
996	996800 - 996899	OCR	AAR	-	OCR – Association of American Railroads	-
ADR						
997	997000 - 997099	ADR	LADR	ADR	ADR - EURADR	ADR-EUR
997	997100 - 997199	ADR	UADR	ADR	ADR - USAADR	ADR-USA
997	997200 - 997299	ADR	SADR	ADR	ADR - SAMADR	ADR-SAM
997	997300 - 997399	ADR	CADR	ADR	ADR - CASADR	ADR-CAS
997	997400 - 997499	ADR	AADR	ADR	ADR - AUSADR	ADR-AUS
997	997500 - 997599	ADR	IADR	ADR	ADR - IMOADR	ADR-IMO
998	998000 - 998999	UN		UN	UN	UN
999	999980 - 999989	ADR	ADR	ADR	ADR - A PLATES	ADR-A
999	999990 - 999999	ADR	EADR	ADR	ADR - EMPTYADR	ADR-EADR

[Back to top](#)

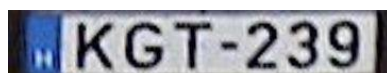
RETRIEVING COUNTRY NAMES FROM RETURNED PLATE TYPE VALUES FOR LEGACY ENGINES

(FOR ENGINE VERSION 7.2.7.X OR BELOW)

I. This section only applies to region-specific engines that are version 7.2.7.x or below, excluding USA, Canada, and Australia

(e.g. *cmanpr-7.2.7.116-latin*)

The type value (T) is a decimal between 100 and 99999. If the type result is divided by 100 the result is the code of the country (C). The remainder defines the subtype (S) within the country. Example: Latin engine



cmNP::type: 111

$C = 111 / 100 = 1$ defines HUN (Hungary) as country.

The remainder ($S = 111 \bmod 100$) = 11, refers to the standard, single-row Hungarian EU-plate.

II. This section only applies to region-specific engines of USA, Canada, and Australia that are version 7.2.7.x or below

(e.g. *cmanpr-7.2.7.117-usa*)

The type value (T) is a decimal, which is greater than 100000 consisting three main components.

$$T = 100000 * C + 100 * F + S$$

Where:

- C - state code within the USA; $C = T / 100000$
- F - Format - a number between 0 and 999- is built up according to the followings:
 - 0-stands for a number in the plate text
 - 1-stands for a letter in the plate text
 - N-stands for the length of the plate text

(in the formula below 2^N equals to the greatest power of 2 below or equal to F)

- B: Decimal value of the plate text's binary format created by the previous rules The

B value can be calculated according to the following formula: $F = 2^N - 2 + B$

Subtype (S): Serves to distinct plates with the same format within a state (2-digit decimal between 0 and 99)

Example: USA and Canada engine



cmNP::type: 1121701

T=1121701.

$C=T/1000000=1121701/1000000=11$ defines USA-CA (California).

F=217

Subtype (S)=1,

$F=2^N-2+B$ where N is the greatest power of two below 217, which is 128. $128=2^7$ so $N=7$

$217=2^7-2+B$

$217=126+B$

$B=91$ converted to binary: 1011011, so the format of the plate is the following:

'letter number letter letter number letter letter'

[Back to top](#)

COUNTRY AND STATE ID'S FOR LEGACY ENGINES

(For ENGINE VERSION 7.2.7.X or below)


(e.g. cmanpr-7.2.7.117-usa)


Country IDs							
1.	HUN	56.	FIN	112.	KGZ	26.	USA-CT
2.	UKR	57.	EST	113.	FRA	27.	USA-OR
3.	RUS	58.	JOR	128.	GBR	28.	USA-NE
4.	LTU	59.	IRN	136.	IRL	29.	USA-WA
5.	CZE	60.	ARE_CD	140.	ZAF	30.	USA-AZ
6.	LVA	61.	ARE_Dubai	142.	IMN	31.	USA-WY
7.	AUT	62.	ARE_AbuDhabi	143.	JEY	32.	USA-KS
8.	DEU	63.	ARE_Ajman	144.	GGY	33.	USA-AL
9.	ROU	64.	ARE_Fujairah	145.	NZL	34.	USA-AK
10.	SVK	65.	ARE_Sharjah	147.	LBY	35.	USA-AR
11.	POL	66.	ARE_UmmalQuwain	163.	URY	36.	USA-DE
12.	NLD	67.	ARE_RasAlKhaimah	164.	GTM	37.	USA-DC
13.	FRA	68.	ARE_Unknown	166.	SLV	38.	USA-HI
14.	TWN	70.	NOR	167.	HND	39.	USA-ID
15.	BRA	71.	LBN	168.	PRY	40.	USA-LA
16.	KOR	72.	ARG	175.	PER	41.	USA-MS
17.	ITA	73.	MCO	176.	MAR	42.	USA-MT
18.	CHL	74.	LIE	180.	CHN	43.	USA-NV
19.	SVN	75.	KAZ	181.	CHN	44.	USA-NH
20.	MLT	76.	MAR	184.	VAT	45.	USA-NM
21.	ESP	77.	SYR	185.	VEN	46.	USA-ND
22.	SGP	78.	HKG	186.	ANT	47.	USA-RI
23.	BIH	79.	GIB	187.	IND	48.	USA-SC
24.	SMR	80.	CHN	188.	VNM	49.	USA-SD
25.	CHE	81.	CHN	189.	BOL	50.	USA-UT
26.	SWE	82.	KOR	208.	IRQ	51.	USA-VT
27.	BEL	83.	EGY	227.	PAK	52.	USA-WV
28.	GBR	84.	KOS	700.	USR	60.	CAN-ON
29.	DNK	85.	ISR	USA State IDs		61.	CAN-QC
30.	AND	86.	CRI			62.	CAN-BC
31.	HRV	87.	AUT_OLD	1.	USA-GOV	63.	CAN-MB
32.	SRB	88.	DEU_OLD	2.	USA-IL	64.	CAN-NS
33.	GRC	89.	FRA_NEW	3.	USA-IN	65.	CAN-AB
34.	LUX	90.	MDA	4.	USA-MI	66.	CAN-SK
35.	MYS	91.	MNE	5.	USA-WI	67.	CAN-NB
36.	IRL	92.	TKM	6.	USA-OH	73.	CAN-FED
37.	PRT	93.	MNG	7.	USA-ME	Australian State IDs	
38.	THA	94.	BWA	8.	USA-TN		
39.	MEX	95.	NAM	9.	USA-MN	1.	AUS-FIS
40.	ZAF	96.	SWZ	10.	USA-PA	2.	AUS-ACT
41.	MKD	97.	ADR	11.	USA-CA	3.	AUS-NT
42.	BGR	98.	UN	12.	USA-FL	4.	AUS-NSW
43.	ALB	99.	UNKNOWN	13.	USA-VA	5.	AUS-QLD
44.	ISL	100.	DEU_TEMP	14.	USA-NY	6.	AUS-SA

45.	NZL	101.	DEU_TEST	15.	USA-TX	7.	AUS-TAS
46.	QAT	102.	DEU_OLDTIMER	16.	USA-IA	8.	AUS-VIC
47.	BLR	103.	DEU_SEASON	17.	USA-OK	9.	AUS-WA
48.	DZA	104.	DEU_TEMP_OLD	18.	USA-CO		
49.	TUR	105.	DEU_TEST_OLD	19.	USA-KY		
50.	GEO	106.	DEU_OLDTIMER_OLD	20.	USA-MO		
51.	COL	107.	DEU_SEASON_OLD	21.	USA-NJ		
52.	SAU	108.	MOZ	22.	USA-GA		
53.	OMN	109.	LSO	23.	USA-MD		
54.	KWT	110.	ZWE	24.	USA-NC		
55.	BHR	111.	AZE	25.	USA-MA		

RESULT STRUCTURE EXAMPLES

(For engine VERSION 7.3.9.X AND ABOVE)

SAMPLE #1 - NUMBER PLATE TYPE FROM HUNGARY				
				
CMANPR RESULT				RESULT FROM
PLATE TYPE	101011			cmNP: type
	RETURNED VALUE	CONVERTED TO BGR	COLOR	
	DECIMAL	HEXADECIMAL	NAME	
CHARACTER COLOR	0	0x000000	black	cmNP::CHR color
CHARACTER BACKGROUND COLOR	16777215	0xFFFFFFFF	white	cmNP::CHR bkcolor
DEDICATED AREA COLOR	16777215	0xFFFFFFFF	white	cmNP::color (anpr.GetColor())
NUMBER PLATE TEXT RESULTS				RESULT FROM
	If unicode_in_text=0	If unicode_in_text=1		
NUMBER PLATE (ASCII VERSION)	KGT239	KGT239		cmNP::text
NUMBER PLATE (UNICODE VERSION)	KGT239			cmNP::wtext

SAMPLE #2 - NUMBER PLATE TYPE FROM CROATIA				
				
CMANPR RESULT				RESULT FROM
PLATE TYPE:	112001			cmNP: type
	RETURNED VALUE	CONVERTED TO BGR	COLOR	
	DECIMAL	HEXADECIMAL	NAME	
CHARACTER COLOR:	0	0x000000	black	cmNP::CHR color
CHARACTER BACKGROUND COLOR:	16777215	0xFFFFFFFF	white	cmNP::CHR bkcolor
DEDICATED AREA COLOR:	16777215	0xFFFFFFFF	white	cmNP:color (anpr.GetColor())
NUMBER PLATE TEXT RESULTS				RESULT FROM
	If unicode_in_text=0	If unicode_in_text=1		
NUMBER PLATE (ASCII VERSION)	V!360GC	V(017D)360GC		cmNP::text
NUMBER PLATE (UNICODE VERSION)	VŽ360GC			cmNP::wtext

SAMPLE #3 - NUMBER PLATE TYPE FROM CHINA

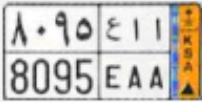


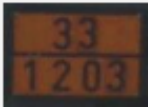
CMANPR RESULT				RESULT FROM
PLATE TYPE:	240334			cmNP: type
	RETURNED VALUE	CONVERTED TO BGR	COLOR	
	DECIMAL	HEXADECIMAL	NAME	
CHARACTER COLOR:	16777215	0xFFFFFFFF	white	cmNP::CHR color
CHARACTER BACKGROUND COLOR:	255	0x0000FF	blue	cmNP::CHR bkcolor
DEDICATED AREA COLOR:	255	0x0000FF	blue	cmNP::color (anpr.GetColor())
NUMBER PLATE TEXT RESULTS				RESULT FROM
	If unicode_in_text=0	If unicode_in_text=1		
NUMBER PLATE (ASCII VERSION)	!N719N0	(4eac)N719N0	cmNP::text	
NUMBER PLATE (UNICODE VERSION)	京N719N0			cmNP::wtext

SAMPLE #4 - A NUMBER PLATE TYPE FROM OMAN



CMANPR RESULT				RESULT FROM
PLATE TYPE:	215006			cmNP: type
	RETURNED VALUE	CONVERTED TO BGR	COLOR	
	DECIMAL	HEXADECIMAL	NAME	
CHARACTER COLOR:	0	0x000000	black	cmNP::CHR color
CHARACTER BACKGROUND COLOR:	65535	0x00FFFF	yellow	cmNP::CHR bkcolor
DEDICATED AREA COLOR:	65535	0x00FFFF	yellow	cmNP::color (anpr.GetColor())
NUMBER PLATE TEXT RESULTS				RESULT FROM
	If unicode_in_text=0	If unicode_in_text=1		
NUMBER PLATE (ASCII VERSION)	2219RS	2219RS	cmNP::text	
NUMBER PLATE (UNICODE VERSION)	2219RS		cmNP::wtext	

SAMPLE #5 - NUMBER PLATE TYPE FROM SAUDI ARABIA				
				
CMANPR RESULT				RESULT FROM
PLATE TYPE:	210009			cmNP::type
	RETURNED VALUE	CONVERTED TO BGR	COLOR	
	DECIMAL	HEXADECIMAL	NAME	
CHARACTER COLOR:	0	0x000000	black	cmNP::CHR color
CHARACTER BACKGROUND COLOR:	16777215	0xFFFFFFFF	white	cmNP::CHR bkcolor
DEDICATED AREA COLOR:	42495	0x0080FF	orange	cmNP::color (anpr.GetColor())
NUMBER PLATE TEXT RESULTS				RESULT FROM
	If unicode_in_text=0	If unicode_in_text=1	If unicode_in_text=2	
NUMBER PLATE (ASCII VERSION)	!!!!!!	(0668)(0660)(0669) (0665)(0639)(0627) (0627)	8095EAA	cmNP::text
NUMBER PLATE (UNICODE VERSION)	٨٠٩٥ ع ١١			cmNP::wtext

SAMPLE #6 - ADR PLATE TYPE				
				
CMANPR RESULT				RESULT FROM
PLATE TYPE:	997001			cmNP::type
	RETURNED VALUE	CONVERTED TO BGR	COLOR	
	DECIMAL	HEXADECIMAL	NAME	
CHARACTER COLOR:	0	0x000000	black	cmNP::CHR color
CHARACTER BACKGROUND COLOR:	16777215	0xFFFFFFFF	white	cmNP::CHR bkcolor
DEDICATED AREA COLOR:	16777215	0xFFFFFFFF	white	cmNP::color (anpr.GetColor())
ADR PLATE TEXT RESULTS				RESULT FROM
	If unicode_in_text=0	If unicode_in_text=1		
ADR PLATE (ASCII VERSION)	331203	331203	cmNP::text	
ADR PLATE (UNICODE VERSION)	331203			cmNP::wtext

SAMPLE #7 - ADR PLATE TYPE				
				
CMANPR RESULT				RESULT FROM
PLATE TYPE:	999999			cmNP: type
	RETURNED VALUE	CONVERTED TO BGR	COLOR	
	DECIMAL	HEXADECIMAL	NAME	
CHARACTER COLOR:				cmNP::CHR color
CHARACTER BACKGROUND COLOR:	16777215	0xFFFFFFFF	white	cmNP::CHR bkcolor
DEDICATED AREA COLOR:	16777215	0xFFFFFFFF	white	cmNP::color (anpr.GetColor())
ADR PLATE TEXT RESULTS				RESULT FROM
	If unicode_in_text=0	If unicode_in_text=1		
ADR PLATE (ASCII VERSION)				cmNP::text
ADR PLATE (UNICODE VERSION)				cmNP::wtext

[Back to top](#)

ABBREVIATIONS

ANPR

Automatic Number Plate Recognition

ALPR

Automatic License Plate Recognition

LPR

License Plate Recognition

ADR

Hazardous Material Identification Code for dangerous goods (also known as Kemler Code). It is an abbreviation derived from the name of the treaty that established it (European Agreement Concerning the International Carriage of Dangerous Goods by Road).

EADR

Empty ADR plate, which is an ADR plate without text

CARMEN®

The brand name of the ARH optical character recognition software family

CARMEN® FreeFlow

ARH license plate recognition software for free-flowing traffic applications and server applications

CARMEN® Parking Digital (can also be referred to as CPD)

ARH license plate recognition software for access control and slow-moving traffic applications

CPD

See CARMEN® Parking Digital

BGR

(Blue, Green, Red) Color code system, it displays pixel layout

RGB

(Red, Green, Blue) Color code system, it displays pixel layout

ASCII

American Standard Code for Information Interchange, character-encoding scheme

API

Application Programming Interface, software library including a set of routines, protocols, and tools for building software applications.

[Back to top](#)

DEFINITIONS

general engine

Recognition engine without country/state identification. Outputs are pure OCR results, where the type (cmNP::type) is always zero (unidentified).

region-specific engine

Recognition engine with country/state identification. Each engine is tuned to a certain geographic region. Outputs contain a country/state identification value in the cmNP::type member of the result structure. Non-zero type indicates that the country of the license plate was successfully identified during the process. Type=0 refers to an unidentified country or state.

current engine

Recognition engine currently selected to process input images. It is possible to install multiple engines concurrently on the same system/server, and you can designate any one of those for processing during runtime.

default engine

Recognition engine selected to process input images by default. It is possible to install multiple engines concurrently on the same system/server, and you can designate any one of those as default.

engine-dependent value

Property value that may be different for each engine release. Obtain the value of the given property by calling the GetProperty() function.

SYMBOLS

{0,1,2,4,5} braces mean that the property can only assume one of the discrete values listed inside
[16..25] brackets mean that the property can assume one of the integer values within the listed range

cmNP

Output structure of the cmAnpr development library

[Back to top](#)

CONTACT INFORMATION

Headquarters:

Adaptive Recognition, Hungary Inc.
Alkotás utca 41 HU
1123 Budapest Hungary
Web: adaptiverecognition.com

Service Address:

Adaptive Recognition, Hungary Inc.
Ipari Park HRSZ1113/1 HU
2074 Perbál Hungary
Web: adaptiverecognition.com/support/

Adaptive Recognition Hungary Technical Support System (ATSS) is designed to provide you the fastest and most proficient assistance, so you can quickly get back to business.

Information regarding your hardware, latest software updates and manuals are easily accessible for customers via our [Documents Site](http://www.adaptiverecognition.com/doc) (www.adaptiverecognition.com/doc) after a quick registration.

New User

If this is your first online support request, please contact your sales representative to register you in our Support System. More help [here](http://www.adaptiverecognition.com/support/) (www.adaptiverecognition.com/support/)!

Returning User

All registered ATSS customers receive a personal access link via e-mail. If you previously received a confirmation message from ATSS, it contains the embedded link that allows you to securely enter the support site.