Carplate Engine Documentation

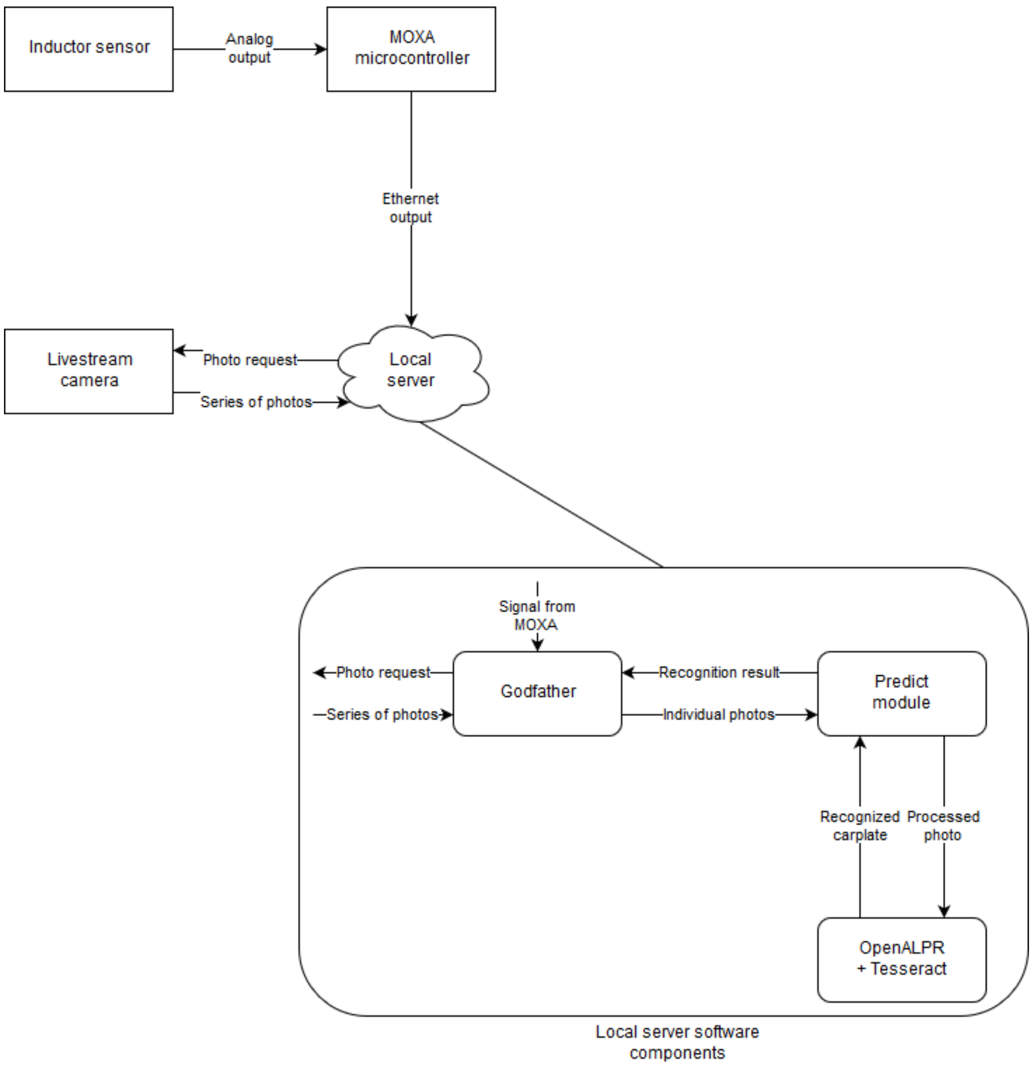
Principle of operation

The engine's software components include the HTTP image server (the Godfather) and the main optical recognition module (the Predict module), which uses OpenALPR + Tesseract OCR as a backend. Upon receiving the sensor signal, Godfather sends a request to a livestream camera, asking it to take a series of 9 photos with an interval of 100 milliseconds. The photos are then passed back to Godfather, which utilizes the Predict module to individually recognize every photo and averages out the result to increase recognition accuracy up to 99.9% (see **The math behind recognition accuracy**).

Architecture

Hardware components include an inductor sensor, which produces a signal when a car passes by it, a MOXA microcontroller, transforming an analog signal from the sensor into Ethernet output, a local server holding software components, and a livestream camera.

The following diagram describes the software architecture as well as provides an explanation of hardware relations:



Parallel safety

The advantage of using the Carplate Engine is the ability to use it in parallel with an already existing recognition system. The engine is designed in a way that no data is modified or pushed to common devices, such as cameras, thus eliminating every possibility of interference with other systems. The engine is isolated, and does not require any external backend to operate properly.

The math behind recognition accuracy

Here we will demonstrate the way of calculating expected accuracy of Predict module based on a number of photos in the series and experimental accuracy acquired through previous tests.

Test show that average accuracy of carplate recognition using Predict module reaches 74%. We will use that as a starting point to calculate average expected accuracy based on a series of carplate photos.

For our calculations we will be using the statistical principle known as **Bernoulli trial**. It states the following:

$P(k) = \binom{n}{k} p^k q^{n-k}$, where *P(k)* is the probability of exactly *k* successes in a series of *n* experiments, *p* is the success probability of a single experiment, *q* is the failure probability calculated as $q = 1 - p$.

To apply Bernoulli trial to the case where we need to know the probability of at least *k* successes in a series of *n* experiments, we sum the results $i = n - k$ times substituting *k* for *i* in range from *k* to *n*. Thus we get the expected prediction accuracy.

The following Python script can be used to automate calculations:

```
import math


def c(j, k):
      return math.factorial(j)/(math.factorial(j - k) * math.factorial(k))


p = 0.86
q = 1 - p

for f in range(3, 10):
      n = f
      m = round(n / 2)

      prob = 0

      for i in range(m, n + 1):
            prob += c(n, i) * math.pow(p, i) * math.pow(q, n - i)

      print(str(n) + ' : ' + str(prob))
```

The results are as follows:

Number of photos	Expected accuracy
3	94.6688%
4	99.01764799999999%
5	99.12943296%
6	99.54530822399998%
7	99.65661367807999%
8	99.79209896968959%
9	99.95684767501014%

As we can see, with a series of 9 photos, we already can achieve a reliable accuracy of **99.96%**.