

# **ZÁVĚREČNÁ STUDIJNÍ PRÁCE**

## **dokumentace**

### **Rozpoznávání SPZ pomocí OpenCV a ukládání do databáze**

Dominik Stuchlý



**Obor:** 18-20-M/01 INFORMAČNÍ TECHNOLOGIE  
se zaměřením na počítačové sítě a programování

**Třída:** IT4

**Školní rok:** 2017/2018

## **Poděkování**

*Chtěl bych velice poděkovat vedoucímu mé práce Ing. Petru Grussmanovi za navrnutí téhle problematiky, kterou jsem v práci řešil, a která mě velice zaujala. Také bych chtěl poděkovat za dobré vedení, postup a nové nápady, jak v projektu pokračovat a jak vyřešit problémy, které jsem sám nedokázal zvládnout. A také bych rád poděkoval svým spolužákům, se kterými jsem pracoval na podobném zadání, za poznatky a nápady, se kterými se se mnou podělili.*

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě      31. 12. 2017

---

*podpis autora práce*

## **ANOTACE**

Projekt se zabývá prací s knihovnou OpenCV a její konfigurací, aplikací a použitím v praxi. Úkolem je získat z video streamu data, jako jsou registrační značky aut, jaké zemi patří tato značka a s jakou přesností byla získána. Pomocí aplikace IP Webcam na mobilním telefonu bylo streamováno video jedoucích aut a následně na počítači spojeno s knihovnou OpenALPR, která uloží obrázky aut a získá z nich data. Knihovna také odešle data na určitou adresu. Tato data jsou pomocí aplikace napsané v Pythonu zpracována a uložena do databázové aplikace.

## **Klíčová slova**

OpenCV, rozpoznávání SPZ, databáze, auta, python, OpenALPR, videokamera

# OBSAH

<b>ÚVOD .....</b>	<b>5</b>
<b>1 SEZNÁMENÍ SE S KNIHOVNAMI A REGISTRAČNÍMI ZNAČKAMI.....</b>	<b>6</b>
1.1 KNIHOVNA OPENCV .....	6
1.2 OPENALPR (2.2.0) .....	7
1.3 REGISTRAČNÍ ZNAČKY .....	8
<b>2 VYUŽITÉ TECHNOLOGIE .....</b>	<b>9</b>
2.1 OPENCV 2.4 .....	9
2.2 IP WEBCAM 1.13.....	9
2.3 PYTHON 3 .....	9
2.4 SQLALCHEMY .....	9
2.5 LENOVO S60-A.....	9
<b>3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY .....</b>	<b>10</b>
3.1 INSTALACE OPENCV NA LINUX.....	10
3.2 INSTALACE OPENALPR .....	11
3.3 TEST FUNKČNOSTI KNIHOVNY OPENALPR .....	12
3.4 VYTVOŘENÍ VIDEO STREAMU .....	13
3.5 KONFIGURACE OPENALPR.....	14
3.6 DB APLIKACE .....	16
<b>ZÁVĚR .....</b>	<b>20</b>
<b>SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ .....</b>	<b>21</b>

## ÚVOD

OpenCV zajišťuje velkou škálu možností jak pracovat v reálném čase s obrázky či videi pořízenými za denních podmínek a získat z nich informace. Představa, že by člověk stál na dálnici a rychle četl registrační značky jedoucích vozidel a měl si je všechny zapamatovat, je absurdní, proto se mi líbila možnost tuhle problematiku zautomatizovat, získané značky uložit do databáze a provést s nimi různé statistické operace. Podobné mobilní systémy používají například policisté v USA; mohou snímat poznávací značky podezřelých vozidel a ty porovnávat ve svých databázích.

Mým cílem bylo tedy správně nainstalovat a nakonfigurovat knihovnu OpenALPR a zjistit, jak nejlépe zpracovat obrázky či video automobilů a z nich získat registrační značku do psané podoby. Poté získaná data zpracovat ve vlastní aplikaci, uložit je do databáze a zobrazit data se kterými by šlo provádět různé operace, například přiřazení k určitému datu, zjištění, zda a kolikrát nějaké auto s danou poznávací značkou projelo určitým místem.

Streamování videa jsem vyřešil stáhnutím aplikace IP webcam na můj chytrý telefon. IP webcam živě vysílá video na určitou IP adresu, na kterou jsem se poté mohl na svém počítači připojit a video sledovat.

Nejvhodnější knihovna k použití byla OpenALPR, která je určena konkrétně pro rozpoznávání registračních značek automobilů. V konfiguračních souborech lze snadno připojit s video stream a nastavit, kam budou získaná data posílána. Aplikace vytvořená v jazyce Python data zpracuje, uloží je do databáze a zobrazí je.

# 1 SEZNÁMENÍ SE S KNIHOVNAMI A REGISTRAČNÍMI ZNAČKAMI

## 1.1 Knihovna OpenCV

OpenCV (Open Source Computer Vision Library) je open source softwarová knihovna určená pro práci s počítačovým viděním. V roce 1999 ji začala vyvíjet společnost Intel. Nyní je knihovna pod licencí BSD, což dělá používání knihovny jednodušší v podnikání a její jakékoliv smysluplné upravování a vylepšování je vítáno.

Knihovna má více než 2500 optimalizovaných algoritmů, které zahrnují komplexní sadu klasických a nejmodernějších počítačových vidění. Tyto algoritmy jsou používány k detekci a rozpoznání obličejů, identifikaci objektů, klasifikaci lidského jednání ve videích, sledování pohyblivých objektů, extrahování 3D modelů objektů, hledání podobných obrázků z databáze obrázků, odstranění červených očí ze snímků pořízených s bleskem atd.

OpenCV má přes 47 tisíc uživatelů v komunitách a odhadovaný počet stažení přesahuje 14 miliónů. Knihovnu široce využívají společnosti jako je Google, Yahoo, Microsoft, Intel, Sony, Honda a Toyota, ale také v jiných firmách, výzkumných organizacích a vládních orgánech. Využití nalezneme třeba v robotice, bezpečnostních systémech, stereovizi, lékařských diagnostikách a ve spoustě dalších odvětvích. Konkrétními případy jsou například scelování obrazů v Google Street View, sledování důlních zařízení v Číně, pomoc robotům při navigaci manipulaci s předměty, odhalování topících se lidí v plaveckých bazénech, kontrola značek na produktech v továrnách a další.

## 1.2 OpenALPR (2.2.0)

OpenALPR (Automatic License Plate Recognition library) je open source knihovna přímo určená pro rozpoznávání registračních značek vozidel. Napsaná je v C++ s vazbami na C #, Javu, Nodes.js a Python. Knihovna analyzuje obrázky a video stream a z nich identifikuje registrační značky. Výstupem je jejich textové zobrazení.

Software může být využit mnoha různými způsoby. Například:

1. Rozpozná registrační značky z kamerových streamů. Výsledky jsou přehledné, vyhledatelné a můžou spustit upozornění. Uložiště dat může být v cloudu nebo můžou být ukládány v lokální síti.
2. Rozpozná registrační značky z kamerových streamů a výsledky pošle do vlastní aplikace.
3. Zpracuje video soubor a uloží jeho značku do SQLite databáze.
4. Analyzuje statické obrázky z příkazového řádku.
5. Vloží rozpoznané poznávací značky do vaší aplikace přímo v kódu (C, C++, C#, VB.NET, Java, Python, Node.js)

### 1.3 Registrační značky

Registrační značka (nebo také SPZ – státní poznávací značka) je ze zákona povinný prvek motorových vozidel. Obsahuje alfanumerické označení, toto označení je unikátní pro každé vozidlo. Značky různých zemí se můžou velice lišit, velikostí, barevně nebo i pořadím čísel a písmen.

Standardní značky v ČR:

- cz #@#####
- cz #@ @####

(#=číslo 0-9, @ =písmena bez diakritiky a s výjimkou G, O, Q, W)



obr. 1 česká SPZ

Jen na ukázkou, jak mohou vypadat značky v různých zemích.

Švýcarsko:



obr. 2 švýcarská SPZ

USA Louisiana:



obr. 3 USA SPZ

Izrael:



obr. 4 izraelská SPZ

Rusko:



obr. 5 ruská SPZ



## **2 VYUŽITÉ TECHNOLOGIE**

### **2.1 OpenCV 2.4**

OpenCV (Open Source Computer Vision Library) je open source multiplatformní knihovna. Určená je především pro počítačové vidění a zpracování obrazu v reálném čase. Knihovna má rozhraní v C, C++, Pythonu, Javě a je podporována pro Windows, Linux, iOS a Android.

### **2.2 IP Webcam 1.13**

IP Webcam je freeware aplikace pro Android, která změní telefon na síťovou kameru s mnoha možnostmi zobrazení. Může zobrazit fotoaparát na libovolné platformě pomocí přehrávače VLC nebo webového prohlížeče tím, že streamuje video uvnitř WiFi sítě.

### **2.3 Python 3**

Python je vysokoúrovňový skriptovací jazyk. Nabízí dynamickou kontrolu datových typů a také podporuje různá programovací paradigmaty, včetně objektově orientovaného, imperativního, procedurálního nebo funkcionálního. Je vyvíjen jako open source. Zdarma nabízí instalační balíky pro Unix, Windows, iOS.

### **2.4 SQLAlchemy**

SQLAlchemy je Python SQL balíček nástrojů a mapovač vztahů mezi objekty. Poskytuje vývojářům aplikací plný výkon a flexibilitu SQL.

### **2.5 Lenovo S60-a**

Můj chytrý telefon Lenovo S60 jsem použil, abych na něho mohl nainstalovat aplikaci IP Webcam. Pomocí 13 Mpix fotoaparátu jsem nahrával video, které jsem následně přes WiFi streamoval.

## 3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

### 3.1 Instalace OpenCV na Linux

Požadované balíčky:

- GCC 4.4.x nebo novější
- CMake 2.6 nebo vyšší
- Git
- GTK+2.x nebo vyšší, včetně záhlaví (libgtk2.0-dev)
- pkg-config
- Python 2.6 nebo novější a Numpy 1.5 nebo novější s vývojářskými balíčky (python-dev, python-numpy)
- ffmpeg nebo libav vývojové balíčky: libavcodec-dev, libavformat-dev, libswscale-dev
- [volitelně] libtbb2 libtbb-dev
- [volitelně] libdc1394 2.x
- [volitelně] libjpeg-dev, libpng-dev, libtiff-dev, libjasper-dev, libdc1394-22-dev

Balíčky lze nainstalovat pomocí terminálu a následných příkazů nebo pomocí programu Synaptic Manager:

```
[compiler] sudo apt-get install build-essential
[required] sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev
libavformat-dev libswscale-dev
[optional] sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-
dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev
```

## 3.2 Instalace OpenALPR

```
//Stáhnutí a instalace deamona:

sudo apt-get update && sudo apt-get install -y openalpr openalpr-daemon
openalpr-utils libopenalpr-dev

//Když používáme deamona je nutné nainstalovat beanstalkd:

sudo apt-get install beanstalkd

//Získání nejnovější verze kódu z GitHubu:

git clone https://github.com/openalpr/openalpr.git

//Vytvoření složky pro kompilaci:

cd openalpr/src
mkdir build
cd build

//Vytvoření prostředí kompilace:

cmake -DCMAKE_INSTALL_PREFIX:PATH=/usr -DCMAKE_INSTALL_SYSCONFDIR:PATH=/etc
..

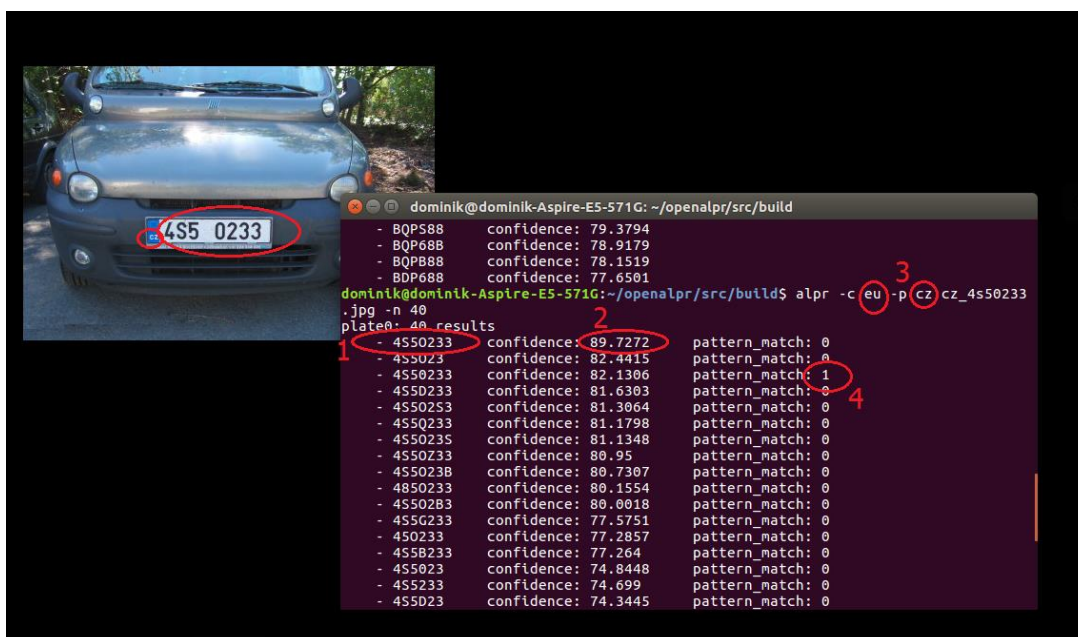
//Kompilace knihovny:

make

//Instalace binárních souborů do místního systému:

sudo make install
```

### 3.3 Test funkčnosti knihovny OpenALPR



obr. 6 Test funkčnosti

Příkaz pro získání čísla registrační značky ze statického obrázku:

```
~/openalpr/src/build$ alpr -c eu -p cz cz_4s50233.jpg -n 40
```

Dostaneme 40 výsledků, jak je na konci příkazu zadáno. Na obrázku vidíme (1), že se výsledné číslo “4S50233” poznávací značky shoduje s jistotou 89,7 % (2) s obrázkem auta. V příkaze jsem také nastavil (3), aby knihovna značku porovnávala s druhem značek Evropské unie a konkrétně s českou poznávací značkou a vidíme, že se shoda našla (4).

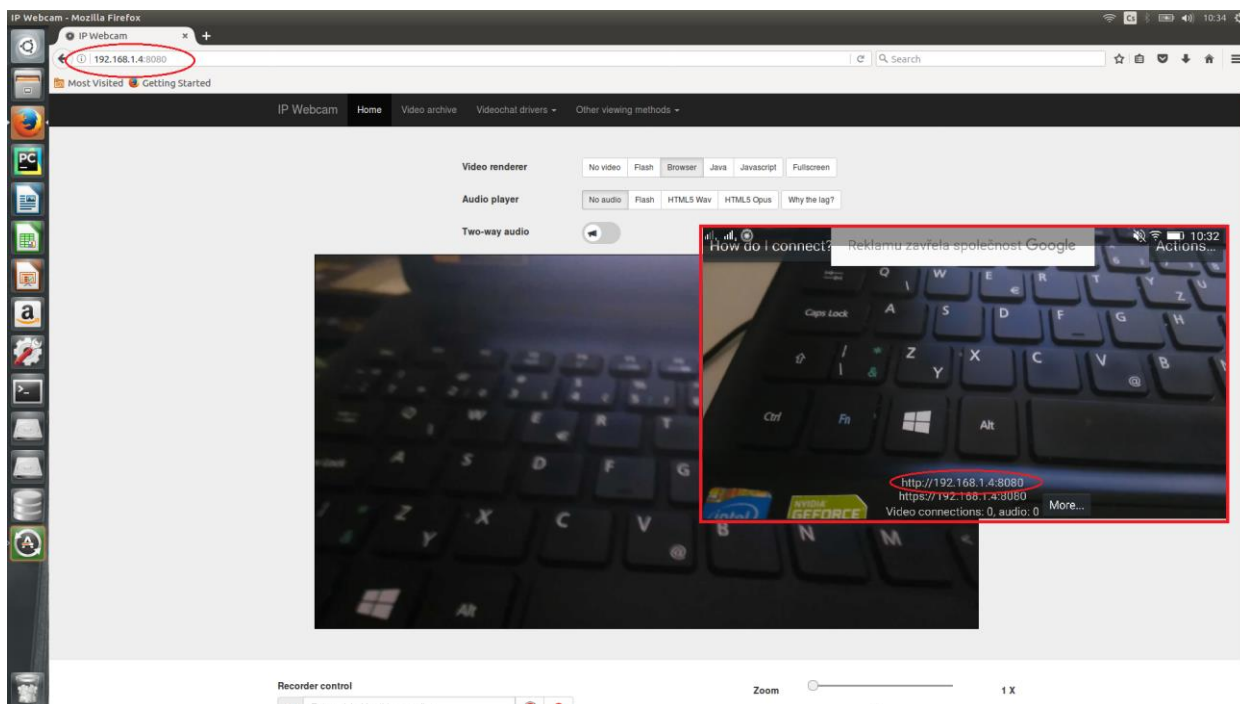
### 3.4 Vytvoření video streamu

Na můj mobil Lenovo S60 jsem nainstaloval z Obchod Play aplikaci IP Webcam, která zajišťuje stream videa.

Odkaz na aplikaci:

<https://play.google.com/store/apps/details?id=com.pas.webcam&hl=c>

Mobil a počítač musí být připojené na stejné síti, protože IP webcam vytvoří server v lokální síti. Po zapnutí aplikace a stisknutí “Start server“ se spustí video stream na níže uvedené adrese. Na tu adresu se poté můžeme na počítači pomocí webového prohlížeče připojit a video si zobrazit. Kameru lze ovládat jak na mobilu, tak také na počítači.



obr. 7 video stream

### 3.5 Konfigurace OpenALPR

Nejpodstatnější bylo nastavování knihovny v souboru `alprd.conf`, který se nachází v adresáři `/etc/openalpr`:

1. Odkaz na video stream, který bude OpenALPR zpracovávat a získávat z něho poznávací značky aut.
2. Počet možných variací značek, který program nahlásí.
3. Povolení pro to, aby program ukládal obrázky, na kterých rozpozná registrační značku auta z videa. Cestu kam obrázky ukládat si můžeme vybrat, nebo nechat tu v předvolbách `/var/lib/openalpr/plateimages/`.
4. Velice důležité je také nastavit, kam získaná data bude OpenALPR posílat. Já data posílal do lokální sítě na `http://localhost:5000/api` a odtud pak v aplikaci napsané v Pythonu data zpracovával.

```
[daemon]

; country determines the training dataset used for recognizing plates. Valid values are: us,
eu
country = us

; text name identifier for this location
site_id = projekt

; Declare each stream on a separate line
; each unique stream should be defined as stream = [url]
stream = http://192.168.1.4:8080/video
;stream = http://127.0.0.1/example_second_stream.jpeg
;stream = webcam

topn is the number of possible plate character variations to report
topn = 10

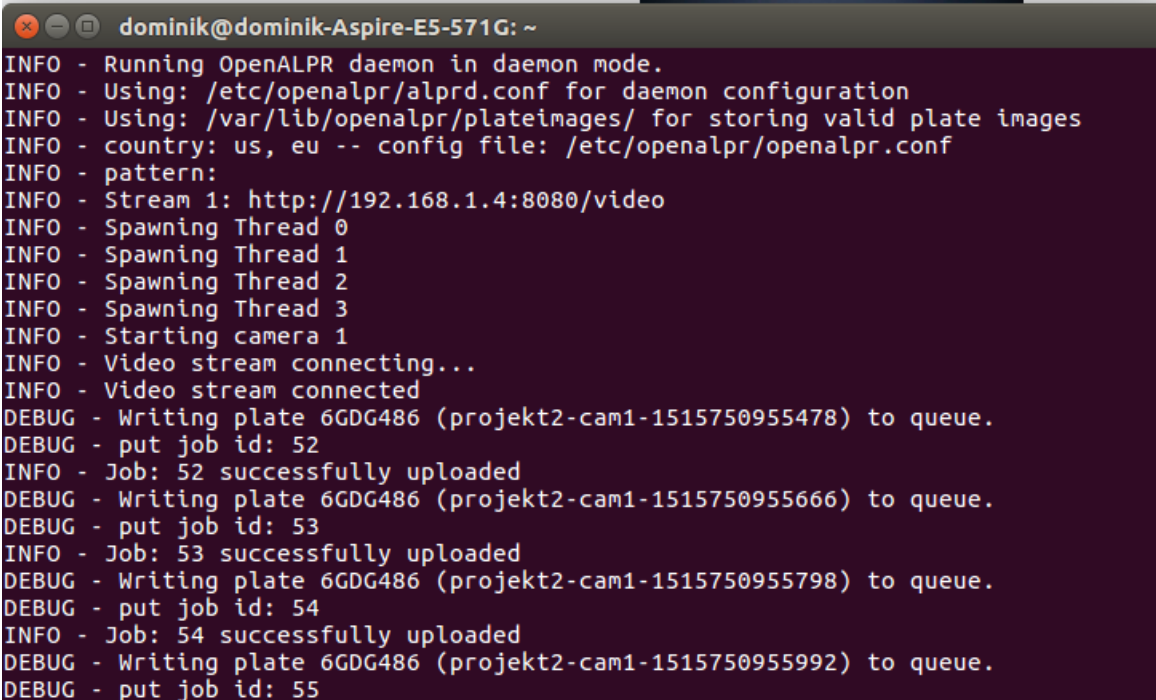
; Determines whether images that contain plates should be stored to disk
store_plates = 1
store_plates_location = /var/lib/openalpr/plateimages/

; upload address is the destination to POST to
upload_data = 1
upload_address = http://localhost:5000/api
```

obr. 8 konfigurační soubor

Pomocí tohoto příkazu můžeme sledovat protokoly OpenALPR. Jestli všechno probíhá v pořádku, spojení s kamerou, získání dat, uložení obrázku a nahrání dat na `http://localhost:5000/api`.

```
tail -f /var/log/alpr.log
```



```
dominik@dominik-Aspire-E5-571G: ~
INFO - Running OpenALPR daemon in daemon mode.
INFO - Using: /etc/openalpr/alprd.conf for daemon configuration
INFO - Using: /var/lib/openalpr/plateimages/ for storing valid plate images
INFO - country: us, eu -- config file: /etc/openalpr/openalpr.conf
INFO - pattern:
INFO - Stream 1: http://192.168.1.4:8080/video
INFO - Spawning Thread 0
INFO - Spawning Thread 1
INFO - Spawning Thread 2
INFO - Spawning Thread 3
INFO - Starting camera 1
INFO - Video stream connecting...
INFO - Video stream connected
DEBUG - Writing plate 6GDG486 (projekt2-cam1-1515750955478) to queue.
DEBUG - put job id: 52
INFO - Job: 52 successfully uploaded
DEBUG - Writing plate 6GDG486 (projekt2-cam1-1515750955666) to queue.
DEBUG - put job id: 53
INFO - Job: 53 successfully uploaded
DEBUG - Writing plate 6GDG486 (projekt2-cam1-1515750955798) to queue.
DEBUG - put job id: 54
INFO - Job: 54 successfully uploaded
DEBUG - Writing plate 6GDG486 (projekt2-cam1-1515750955992) to queue.
DEBUG - put job id: 55
```

*obr. 9 sledování protokolů*

Na prvním řádku vidíme, že OpenALPR je bez problémů spuštěné. Na druhém řádku je napsané, že daemon právě používá `alprd.conf` konfigurační soubor a jeho nastavení. Na třetím řádku je uvedena cesta do složky (`/var/lib/openalpr/plateimages/`), kde se ukládají obrázky poznávacích značek vytvořené knihovnou OpenALPR jakmile kamera nějakou SPZ zachytí. Ve čtvrtém řádku vidíme, že je nastaveno, aby knihovna rozpoznávala značky Spojených Států amerických a Evropské unie.

### 3.6 DB aplikace

Zde můžeme vidět kód, který zajišťuje zpracování výsledků, které nám OpenALPR posílá. Data uloží do databáze.

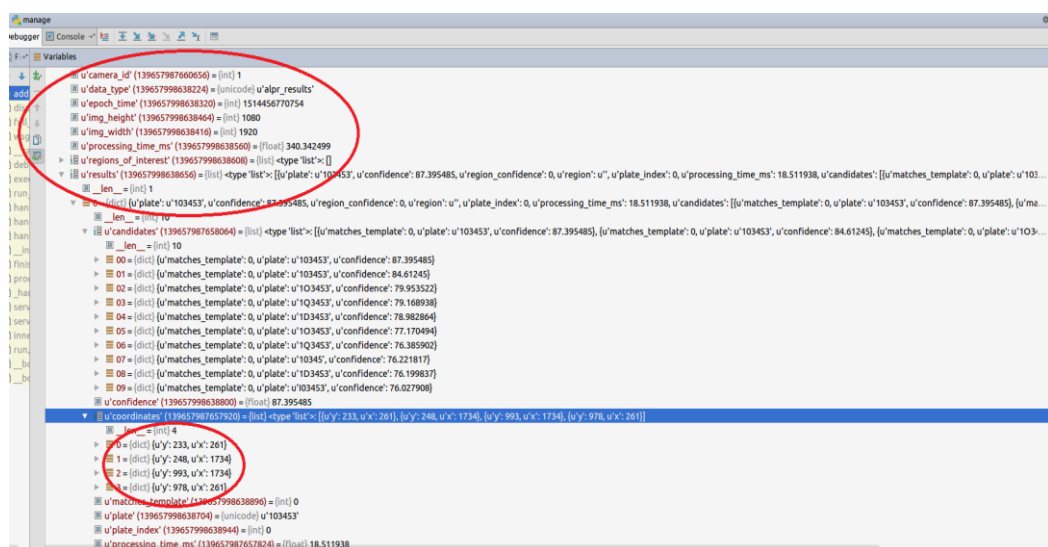
```

22 @main_blueprint.route('/results')
23 def results():
24     times = Time.query.all()
25     times_times = [time.time for time in times]
26     grouped_times = [list(g) for k, g in itertools.groupby(times, key=lambda d: d.time.date())]
27     results = {}
28     for datetime_ in grouped_times:
29         date = datetime_[0].time.strftime('%Y/%m/%d')
30         results[date] = datetime_
31     print(results)
32     return render_template('main/results.html', times=results)
33
34 @main_blueprint.route('/api', methods=['GET', 'POST'])
35 def add_message():
36     content = request.get_json(silent=True)
37     #print(content)
38     for prvek in content['results']:
39         time = Time(content['epoch_time'], content['uuid'])
40
41         for candidate in prvek['candidates']:
42             auto = Auto(**candidate)
43
44             time.cars.append(auto)
45             db.session.add(auto)
46
47             db.session.add(time)
48             db.session.commit()
49
50     return "Ok"
51

```

obr. 10 zpracování výsledků

Tady si můžeme zobrazit všechna data, která OpenALPR pošle. Kód registrační značky, identifikační číslo kamery, řekne nám, že to jsou výsledky přicházející z od alpr, jak dlouho proces trval, parametry obrázku (výška, šířka), nebo také přesné souřadnice na obrázku, ve kterých se poznávací značka nachází



obr. 11 poslaná data



Vytvoření tabulky time a auto. Jakmile aplikace zde uloží data získaná z alpr, vytvoří se záznam v kolik hodin tak proběhlo, přiřadí se k danému času záznamy z tabulky auto a také přiřadí obrázek dané SPZ. V tabulce auto jsou uloženy ostatní výsledky z OpenALPR, jako je textová podoba poznávací značky, přesnost a další.

```

46 class Time(db.Model):
47     tablename = 'time'
48     id = db.Column(db.Integer, primary_key=True, autoincrement=True)
49     time = db.Column(db.DateTime, nullable=False, default=datetime.datetime.now())
50     cars = db.relationship('Auto', backref='time', lazy=True)
51     image = db.Column(db.LargeBinary, nullable=False)
52
53     def __init__(self, epoch_time, image):
54         self.time = datetime.datetime.fromtimestamp(epoch_time/1000)
55
56         with open('/var/lib/openalpr/plateimages/{}.jpg'.format(image), 'rb') as f:
57             self.image = base64.b64encode(f.read())
58
59     @property
60     def image_source(self):
61         return self.image.decode('utf-8')
62
63 class Auto(db.Model):
64     tablename = 'auto'
65     id = db.Column(db.Integer, primary_key=True)
66     time_id = db.Column(db.Integer, db.ForeignKey('time.id'))
67     plate = db.Column(db.String, nullable=False, index=False)
68     confidence = db.Column(db.Float, nullable=False, index=True)
69     processing_time_ms = db.Column(db.Float)
70     region = db.Column(db.String)
71     # Use custom constructor
72     # pylint: disable=W0231
73     def __init__(self, **kwargs):
74         for k, v in kwargs.items():
75             if hasattr(self, k):
76                 setattr(self, k, v)
77
78
79

```

obr. 12 vytvoření tabulek

Zobrazení tabulky auto pomocí SQLite Manageru:

id	time_id	plate	confidence	processing_time_ms	region
1	107	T548	74.337242	84.356489	
2	107	8T15418	86.586632	92.525551	
3	107	8T15418	86.586632	92.525551	
4	111	5N0F222	86.784805	85.880882	
5	111	5N0F222	86.784805	85.880882	
6	111	5N0F222	86.784805	85.880882	
7	112	KZ66ZYT	91.727798	87.680321	
8	112	KZ66ZYT	91.727798	87.680321	
9	112	KZ56ZYT	84.333504		

obr. 13 tabulka auto

Vytvoření HTML stránky, která na adrese 170.0.0.1:5000/results vypíše uložená data z databáze.






```

1 <header class="site-header">
2 <!-- Navigation -->
3 <nav class="navbar navbar-default navbar-fixed-top" role="navigation">
4 <div class="container">
5 <!-- Brand and toggle get grouped for better mobile display -->
6 <div class="navbar-header">
7 <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">
8 <span class="sr-only">Toggle navigation</span>
9 <span class="icon-bar"></span>
10 <span class="icon-bar"></span>
11 <span class="icon-bar"></span>
12 </button>
13 <a class="navbar-brand" href="{{ url_for('main.home') }}"></a>
14 </div>
15 <!-- Collect the nav links, forms, and other content for toggling -->
16 <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
17 <ul class="nav navbar-nav">
18 <li><a href="{{ url_for('main.about') }}"></a></li>
19 <li><a href="{{ url_for('main.results') }}">Results</a></li>
20 <li><a href="{{ url_for('user.members') }}">Members</a></li>
21 </ul>
22 </div>
23 </nav>
24 </header>
25
26 {% extends '_base.html' %}
27
28 {% block content %}
29 <h2>{{ k }}</h2>
30 <table class="table">
31 <thead>
32 <tr>
33 <th>Time</th>
34 <th>Plates</th>
35 <th>Image</th>
36 </tr>
37 </thead>
38 <tbody>
39 <tr>
40 <td>{{ time.time.strftime('%Y/%m/%d %H:%M:%S') }}</td>
41 <td>
42 <div>
43 <p>{{ car.plate }} (confidence: {{ car.confidence|round(2) }})</p>
44 </div>
45 </td>
46 <td>
47 
48 </td>
49 </tr>
50 </tbody>
51 </table>
52 </div>
53 </block>

```

obr. 14 kód zobrazovací stránky

Zde vidíme výsledky celé práce. HTML stránky results, kde se zobrazují data získaná přes video stream, knihovnu OpenALPR a uložena do databáze. Výsledky jsou řazeny podle data, kdy byly získány. U každého času jsou 3 variace registrační značky, s jakou jistotou to jsou ony a obrázek pořízený z kamery.

Results		
2018/01/12		
Time	Plates	Image
2018/01/12 11:12:17	T548 (confidence: 74.34) 8T15418 (confidence: 84.36) 8T15418 (confidence: 86.59)	
2018/01/12 11:14:54	5N0F222 (confidence: 92.53) 5N0222 (confidence: 86.78) 5N0F222 (confidence: 85.88)	
2018/01/12 11:28:13	KZ66ZYT (confidence: 91.73) K266ZYT (confidence: 87.68) KZ56ZYT (confidence: 84.33)	
2018/01/14		
Time	Plates	Image
2018/01/14 14:15:23	8T15418 (confidence: 86.86) 8T1548 (confidence: 83.94) 8T5418 (confidence: 83.44)	
2018/01/14 14:42:53	TE9O36 (confidence: 89.28) 1TE9O36 (confidence: 87.9) TE9O6 (confidence: 82.89)	

obr. 15 výsledky

## Závěr

Při práci na projektu jsem se seznámil s knihovnami OpenCV a OpenALPR. Tyto knihovny se mi podařilo stáhnout a nainstalovat. Důležitým bodem bylo propojení video streamu, vytvořeném aplikací IP webcam, s OpenALPR; zde byla klíčová práce se souborem alprd.conf. Nastavením v konfiguračním souboru jsem také zajistil, aby čísla poznávacích značek, které OpenALPR získá z videa, poslal na adresu, ze které je pak získá a zpracuje aplikace napsaná v Pythonu. Ta pak značky a ostatní informace uloží do databáze. Aplikace v Pythonu pak ještě vezme data z databáze a zobrazí je na internetové stránce v lokální síti.

OpenALPR dokáže porovnat také typ poznávací značky, z jaké je země (např. eu, us, gb), ale zatím se mi nepodařilo, aby to určil stejně jako číslo značky automaticky a nahrál je do databáze společně. Kromě toho, bych chtěl do budoucna vylepšit mou databázovou aplikaci, aby měla více funkcí, na lepším zobrazení dat na stránce a také bych chtěl zapracovat na vytvořené databázi.

V praxi by se dala aplikace použít, kdybychom chtěli mít přehled nad konkrétním místem, kde projíždějí auta, a my bychom zde umístili kameru a získávali jejich poznávací značky. V praxi již podobné aplikace fungují, zejména v bezpečnostních systémech, například rychlostní radary na silnicích, závorové systémy, pátrání po odcizených vozidlech.

## SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

- [1] *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-01-14]. Dostupné z: <https://cs.wikipedia.org/wiki/Python>
  
- [2] OpenCV. *OpenCV: Open Source Computer Vision Library* [online]. OpenCV team [cit. 2017-12-29]. Dostupné z: <https://opencv.org>
  
- [3] OpenALPR: openalpr documentation. *OpenALPR: openalpr documentation* [online]. OpenALPR Technology [cit. 2017-12-30]. Dostupné z: <http://doc.openalpr.com/index.html>
  
- [4] *GitHub: GitHub OpenALP* [online]. Commerce, MI: matthill [cit. 2017-12-30]. Dostupné z: <https://github.com/openalpr/openalpr>

