Gymnázium, Praha 6, Arabská 14

Turistická aplikace, vedoucí práce Mgr. Jan Lána

# Turistická aplikace s počítačovým viděním

Ročníková práce

Denis Mašek 2.E

Duben 2022

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 29. dubna 2022

# Anotace

Úkolem mé ročníkové práce bylo naprogramovat aplikaci pro operační systém Android, která dokáže zkontrolovat, jestli uživatelé na daném turistickém místě doopravdy byli.

# Zadání

Úkolem mé ročníkové práce bude naprogramovat aplikaci pro operační systém Android, která dovede uživatele na hezká místa v Praze a pomocí počítačového vidění (z anglického computer vision) dokáže ověřit, jestli uživatel na daném místě doopravdy byl.

# Obsah

# 1. Úvod

Tato práce se zabývá nejen samotnou tvorbou mého programu, ale také historií a problematikou počítačového vidění jako takového. Jeho využití sahá již mnoho desítek let dozadu a v této práci je shrnutí těch nejdůležitějších okamžiků v jeho vývoji od jeho vzniku až po vznik mé aplikace.

## 1.1. Cíl práce

Cílem této práce bylo naprogramovat aplikaci, která by dokázala lidem ukázat hezká místa v jejich okolí a navnadit je na to, aby se zvedli od počítačů s šli si něco zahrát ven, jako to dělá například Geocaching nebo Pokémon GO.

Aby to lidi opravdu bavilo a nechodili jenom po památkách jako normálně, tak jsem do programu přidal algoritmus, který dokáže z fotografie poznat, jestli člověk na dané místo opravdu došel a vyfotil si ho.

Tím jsem se snažil udělat svou aplikaci svým způsobem kompetitivní, ale toto soutěžení ještě není úplně implementováno, takže teď se s ostatními uživateli srovnávat nemůžete. Určitě je ale v plánu tuto funkci přidat později, než půjde aplikace na Obchod Play.

# 2. Historie

I když se zdá, že je počítačové vidění velmi novou technologií, tak jeho počátky a první využití sahají mnoho desítek let zpět. Historie počítačového vidění je už kupodivu velmi rozsáhlá a od jeho vzniku už byl vyvinut nespočet nových algoritmů a softwarů, které ho využívají, a ještě více takových aplikací, jako je třeba ta má.

## 2.1. Počátky

Většina toho, co víme o zrakovém vnímání lidí, pochází z padesátých a šedesátých let minulého století z neurofyziologických testů na kočkách. Studováním toho, jak neurony reagují na různé stimulace, vědci zjistili, že lidské vidění je takzvaně hierarchální, což znamená, že lidský mozek nejdříve rozeznává snadné věci jako hrany a rohy, potom z nich tvoří základní tvary a z nich už složitější modely. Vyzbrojeni těmito znalostmi se velké množství výzkumníků a vědců pustilo do práce a byli z počátku velmi optimističtí, dokonce si mysleli, že během jenom pár desítek let budeme mít počítače stejně chytré jako lidi. Výsledkem tohoto optimismu bylo obrovské množství peněz v řádu milionů dolarů, které dostali tito výzkumníci k dispozici, toto prvotní nadšení ale moc dlouho nevydrželo. Laboratoře podávaly slabé výsledky, které zdaleka nedosahovali slíbených kvalit, a tak přívod financí rychle vyschl a tomuto prvotnímu nadšení byl konec. (1)

## 2.2. Příchod internetu

S příchodem internetu v osmdesátých letech minulého století měli najednou výzkumníci přístup k mnohem více datům, než bylo předtím možné, a tak se začalo počítačovému vidění a neuronovým sítím opět dařit. Instituty a společnosti, které na tomto pracovali, se opět dostali k hezké sumě peněz, a tak začalo vznikat mnoho úspěšných projektů. Například přišly první rozpoznávače textu, které pomáhaly v převodu dokumentů psaných na strojích do digitální podoby, a dokonce i první neuronové sítě pro rozpoznávání obrázků. (1)

## 2.3. Databáze ImageNet

ImageNet je přímo obrovská (jedna z největších na světě) databáze označených obrázků užívaná pro trénování právě neuronových sítí, kterou založila Fei-Fei Li, která teď momentálně pracuje jako profesorka na Stanfordské univerzitě. ImageNet je jeden ze světově největších crowdsourcing projektů, který využíval primárně platformu Amazon Mechanical Turk a k popisu obrázků používal převážně podobný projekt zvaný WordNet (databáze slov a jejich vztahů – synonyma, homonyma, antonyma…). V tuto chvíli obsahuje databáze ImageNet 14,197,122 fotografií rozdělených celkově do 21,841 podsetů podle jejich druhu. (2)

## 2.4. AlexNet

První opravdu velký pokrok na poli počítačového vidění a neuronových sítí přišel až v roce 2012 na soutěži ILSVRC (ImageNet Large Scale Visual Recognition Challenge). ILSVRC je každoroční soutěž pořádaná právě projektem ImageNet, ve které mají výzkumné týmy možnost nechat své algoritmy na daném datasetu otestovat a soutěžit potom o algoritmus s nejlepší přesností.

V letech lety 2010 a 2011 se procento chyb u vítězných algoritmů pohybovalo okolo 26 procent, ale v roce 2012 přišel výzkumný tým z Torontské univerzity s jejich hlubokou neuronovou sítí nazvanou AlexNet, která nastavila budoucím algoritmům laťku opravdu vysoko. Dosáhla procenta chyb pouze 16,4 procent a hluboké neuronové sítě se tak staly standardem pro většinu algoritmů v budoucích soutěžích, které pak dosáhly výsledku pouze pár procent chyb. (1)

## 2.5. Software pro počítačové vidění

Už kolem roku 2000 začaly vznikat první softwary pro programátory a výzkumníky, které pomáhaly s programováním aplikací, které používají počítačové vidění. Tyto softwary obsahují velké množství různých algoritmů, které každý fungují trochu jinak a z nich si můžete vybrat vždy ten správný na to, na co ho právě potřebujete.

OpenCV

OpenCV je jeden z prvních, ale zároveň velmi dobrý software pro počítačové vidění. Jeho beta verze byla vydána Intelem už v červnu roku 2000 a první verze 1.0 byla vydána v roce 2006. OpenCV má algoritmy snad na všechno, na co si dokážete vzpomenout, ať už se jedná o detekci obličejů, vyhledávání objektů, rozpoznávání pohybů, stereo vidění nebo třeba i rozšířenou realitu. OpenCV má knihovny pro většinu známých programovacích jazyků, ať už se jedná o Python, C++ nebo třeba právě Javu. OpenCV také funguje na velkém množství operačních systémů, ať už je řeč o systémech pro počítače jako Windows, Linux nebo Mac OS, nebo o systémech pro přenosná zařízení jako třeba právě Android. OpenCV je velmi obsáhlá a pokročilá knihovna, kterou není úplně snadné používat, ale naštěstí je na internetu vysoký počet různých návodů na její využití. (3)

TensorFlow

TensorFlow je novější a ještě lepší software pro počítačové vidění a jiné neuronové sítě. Byl vydána devátého listopadu 2015 teamem Google Brain a od té doby se jedná o velmi dobrý nástroj pro tvoření deep learning (učení do hloubky) projektů. Jednou z hlavních výhod TensorFlow je impresivní rychlost a velmi efektivní komunikace s procesory a grafickými kartami, na kterých pracuje, a to hlavně kvůli podpoře technologií jako jsou Nvidia CUDA a brzy snad i AMD GPUFORT. TensorFlow má oficiální knihovny pro programovací jazyky jako C, C++, Java, Swift atd, ale dostupné jsou i knihovny třetích stran pro jazyky jako C#, Scala, Rust… (3)

PyTorch

PyTorch je také velmi nový software pro počítačové vidění a neuronové sítě, který je velmi podobný TensorFlow jak v rychlosti, tak v efektivitě. Hlavní rozdíl mezi nimi je, že PyTorch je mnohem snazší na používání, proto jej také využívá většina (i velmi důležitých) projektů, jako třeba autopilot v autech značky Tesla. PyTorch byla vydána laboratoří Facebooku pro výzkum umělé inteligence (FAIR – Facebook AI Research laboratory) v září roku 2016 a má knihovny pro programovací jazyky Python a C++. (4)

# 3. Můj program

Tato kapitola se zabývá vývojem mé aplikace a problémy, na které jsem během vývoje narazil. Také jsou zde předvedeny programy a knihovny, které jsem k vývoji využil.

## 3.1. Použité programy

Java

Java je univerzální multiplatformní programovací jazyk, který je objektově orientovaný a navržený tak, aby měl co nejméně implementačních závislostí. Jeho cílem je umožnit vývojářům „napsat jednou, spustit kdekoli," což znamená, že kompilovaný kód Java může běžet na všech platformách, které podporují Javu bez nutnosti kompilace. Java aplikace jsou většinou kompilovány do byte kódu, který může běžet na libovolném virtuálním stroji (Java Virtual Machine) bez ohledu na architekturu počítače. Původně byl vyvinutý firmou Sun Microsystems v roce 1995. V současnosti je Java jedním z nejpopulárnějších programovacích jazyků s nahlášenými 9 miliony vývojářů. (5)

Android Studio

Android Studio je vývojové prostředí založené na IntelliJ IDEA. Android studio bylo firmou Google oficiálně představeno 16. května 2013 na konferenci Google I/O jako náhrada za již zastaralé vývojové prostředí Eclipse. Od června 2013 je zdarma k dispozici pro uživatele na platformách Windows, Mac OS X a Linux. Android Studio využívá pro automatizaci sestavování programu nástroj Gradle, který je založen na starších alternativách jako Apache Ant nebo Apache Maven a zásadně usnadňuje vývoj programu. V Android Studiu se dá vyvíjet programy pomocí několika programovacích jazyků, hlavně tedy Java a Kotlin, kdy Kotlin je v podstatě spojením Javy a Pythonu. Kód psaný v Javě se dá dokonce pomocí jednoho tlačítka převést do kódu v Kotlinu a obráceně, pokud tedy spolupracujete na projektu s někým, kdo umí ten druhý jazyk, tak to není až takový problém. (4)

## 3.2. Použité knihovny

OpenCV

OpenCV je svobodná a otevřená multiplatformní knihovna pro manipulaci s obrazem. Je zaměřena především na počítačové vidění a zpracování obrazu v reálném čase. Původně ji vyvíjela společnost Intel.

Picasso

Picasso je velmi versatilní a snadno používatelná knihovna pro stahování obrázků z internetu. Vytvořil ji uživatel GitHubu Square.

cURL

cURL je nástroj pro stahování a přetahování souborů specifikovaných URL adresou vytvořený původně Danielem Stenbergem.

Firebase

Firebase je cloudová databáze od Googlu, kterou můžete do určitého množství měsíčně přetažených dat využívat zdarma.

## 3.3. Vzhled a GUI

V samotné aplikaci můžete za celou dobu používání vidět tři aktivity a jedno vyskakovací okno pro přihlášení do Google účtu. V první aktivitě si můžete vybrat místo, na které chcete dojít, v druhé aktivitě se můžete na mapě podívat, kde se dané místo nachází a ve třetí aktivitě ho už můžete vyfotit.

Když aplikaci spustíte, tak první, co uvidíte, je první aktivita, na které vidíte svoji lokaci, tlačítko pro reload a samotná místa, která si můžete vybrat. Když kliknete na tlačítko reloadu, tak se obnoví vaše aktuální pozice a místa se srovnají podle toho, jaké je k vám nejblíže. Na to teď momentálně využívám rozdíly v geografické lokaci a Pythagorovu větu, ale v případě více než 3 míst bych musel vymyslet jiné řešení, protože násobit a odmocňovat případně tisíce hodnot kvůli jednomu reloadu není úplně v rámci možností.

Dále když kliknete na tlačítko Sign In, tak se vám otevře dialog pro přihlášení do Google účtu. Tento dialog váš účet připojí k databázi Firebase, kam se budou v budoucnu ukládat vaše data. Bohužel mé znalosti a zkušenosti s Firebase jsou téměř nulové, proto jsem zatím dokázal implementovat pouze přihlášení a zapisování dat přijde potenciálně až v nějaké další verzi.

Nyní máte u každého místa tlačítko s jeho názvem a znak kompasu, který by měl otevřít mapu. Bohužel jsem svůj authentikační kód do Google Maps API nahrál na GitHub a musel jsem ho poté deaktivovat, což mapy rozbilo. Každopádně se jedná o další možnou opravu v budoucích verzích. Tlačítko s názvem místa je ale mnohem zajímavější, to vás totiž pošle do další aktivity, a to na vyfocení místa.

V druhé aktivitě můžeme vidět pohled kamery telefonu, číslo nula nahoře a tlačítko pro vyfocení dole. Číslo nula je defaultní hodnota tohoto textového pole, která nám říká, že nebyla detekována žádná podobnost. Jakmile stiskneme tlačítko pro vyfocení, tak začne proces porovnávání, který může kvůli slabému výkonu mobilních telefonů trvat pár vteřin a poté se nula změní buďto na jinou číselnou hodnotu, pokud je podobnost menší než 0.02 nebo na text Ověřeno, když je hodnota rovna nebo větší 0.02.


*Obrázek 1 První aktivita*


*Obrázek 2 Druhá aktivita*

## 3.4. Fungování programu

Hlavní kus kódu, který zde ukáži, je porovnávání obrázků, protože ten je na této práci asi to nejdůležitějšího.

Kód nejdříve načte obrázky do objektů, se kterými dokáže OpenCV pracovat, a poté vytvoří instanci třídy SIFT. SIFT je takzvaný deskriptor, což znamená, že projde celý obrázek a vyhledá v něm tzv. features, kterým poté přiřadí označení, podle kterých se dají porovnávat. (6)

Když SIFT dokončí svoji práci, předá features a jejich pozice třídě FlannBasedMatcher, která porovná místa a features obou obrázků a zapíše do ArrayListu matches shodné body. (6)

```
//Kontrola podobnosti

SIFT sift = SIFT.create();
MatOfKeyPoint keypoints1 = new MatOfKeyPoint();
MatOfKeyPoint keypoints2 = new MatOfKeyPoint();
Mat descriptors1 = new Mat();
Mat descriptors2 = new Mat();
sift.detectAndCompute(img1, new Mat(), keypoints1, descriptors1);
sift.detectAndCompute(img2, new Mat(), keypoints2, descriptors2);
ArrayList<MatOfDMatch> matches = new ArrayList<>();
FlannBasedMatcher flann = new FlannBasedMatcher();
flann.knnMatch(descriptors1, descriptors2, matches, 2);
```

*Kód 1 Porovnání obrázků*

Dále program pomocí relativní vzdálenosti jednotlivých bodů vyfiltruje ty, které jsou nejspíš špatně a spočítá ze zbývajících bodů výslednou hodnotu. (6)

```
ArrayList<DMatch> good_points = new ArrayList<>();
for (Iterator<MatOfDMatch> iterator = matches.iterator(); iterator.has-
Next(); ) {
    MatOfDMatch matOfDMatch = (MatOfDMatch) iterator.next();
    if (matOfDMatch.toArray()[0].distance / matOfDMatch.to-
Array()[1].distance < 0.5) {
        good_points.add(matOfDMatch.toArray()[0]);
    }
}
MatOfDMatch good_points_mat = new MatOfDMatch();
good_points_mat.fromList(good_points);
//System.out.println(keypoints1.total());
//System.out.println(good_points.size());
double percentage;
double good100 = good_points.size() * 100;
if (keypoints1.total() > keypoints2.total()) {
    percentage = good100 / keypoints2.total();
} else {
    percentage = good100 / keypoints1.total();
}
```

*Kód 2 Spočítání podobnosti*

# 4. Zhodnocení

Zadáním mé práce bylo vytvořit aplikaci, která zvedne lidi ze židlí a ukáže jim místa, na kterých ještě třeba nebyli. Plus k tomu je samozřejmě ověřování, jestli člověk na dané místo doopravdy došel, které se nakonec ukázalo jako velmi nepraktické na telefonech Android, ale přesto alespoň nějak funguje. Sice se mi nakonec úplně nepodařilo napojení na online databázi, ale dostal jsem se velmi blízko, a možná se mi to někdy podaří, považuji tedy práci za úspěšnou.

# 5. Zdroje


1. ***Motionmetrics*. [Online] [Citace: 27. 4 2022.] https://www.motionmetrics.com/.**

**2. *Super Annotate Blog*. [Online] [Citace: 27. 4 2022.] https://blog.superannotate.com/.**

**3. *Geeks for Geeks*. [Online] [Citace: 27. 4 2022.] https://www.geeksforgeeks.org/.**

**4. *Wikipedia*. [Online] [Citace: 28. 3 2022.] https://en.wikipedia.com.**

**5. *IT Slovník*. [Online] [Citace: 28. 3 2022.] https://it-slovnik.cz/.**

**6. Pysource. *YouTube*. [Online] [Citace: 27. 4 2022.] https://www.youtube.com/.**

# 6. Seznam obrázků