FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Měření rychlosti vozidel z kamerového záznamu
Student:	Jan Krkoška
Vedoucí:	Mgr. Tomáš Karella
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2021/2022

Pokyny pro vypracování

Cílem této práce je vytvořit software, který bude odhadovat rychlost jedoucích vozidel. Student analyzuje metody zpracování obrazu pro odhad rychlosti a otestujete je na reálných datech. Student poté navrhne a implementuje program, který bude zobrazovat vstupní video s odhadovanou rychlostí vozidel. Součástí práce je dokumentace a otestování vzniklého kódu. Dále bude práce obsahovat i praktický test přesnosti měření.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 26. ledna 2021 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Měření rychlosti aut za pomoci statické kamery

Jan Krkoška

Department of Softwarové inženýrství
Vedoucí práce: Mgr. Tomáš Karella

13. května 2021

Poděkování

Děkuji vedoucímu práce Mgr. Tomáši Karellovi za věnovaný čas a trpělivost, kterou se mnou měl při tvorbě práce. Dále pak organizátorům AI City Challenge za dodané záznamy k testování aplikace. Též děkuji svému příteli Bc. Vojtěchu Kouřilovi za konzultace a korekturu textů práce. A v neposlední řadě rodině a přátelům za podporu a pomoc při tvorbě testovácích záznamů.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

Praha dne 13. května 2021 . . . . . .

Odkaz na tuto práci

Krkoška, Jan. *Měření rychlosti aut za pomoci statické kamery.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Abstrakt

Tato práce se zabývá návrhem a implementací aplikace měřící rychlost vozidel pomocí kamerového záznamu. Aplikace by měla být schopna vzít libovolný kamerový záznam silnice a dokázat na něm změřit rychlost projíždějících vozidel. Součástí práce je i testování na realných datech, zhodnocení výsledků a návrh dalšího zlepšení

Klíčová slova Detekce objektu, Měření rychlosti, Statická kamera, OpenCV, Yolo, Optický proud, Python

Abstract

This work is about the design and implementation of an application for measuring the speed of an vehicle by using a stationary camera. The application should be able to take any road record and measure speed of passing vehicles. Part of work is also practice test, evaluation of results and improvements proposal.

Keywords Object detection, Speed measuring, Static camera, OpenCV, Yolo, Optical Flow, Python

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Žijeme v době, kdy internet začíná být nejdůležitější součástí běžného života. Je na něj napojeno nejrůznější množství přístrojů od vlaků až po ledničku. Jedním z těchto strojů jsou i kamery. Již dnes je podél mnoha silnic rozmístěno velké množství kamer, sloužících pro kontrolu situace na vozovce. Jedním z jejich hlavních účelů je měření rychlosti.

V současné době existují dvě metody měření rychlosti - úsekové a okamžité. Úsekové využívá záběrů dvou kamer umístěných na 2 odlišných místech a měří průměrnou rychlost auta v daném úseku cesty. Využívá se především pro dálnice a rychlostní silnice. Okamžité využívá záběrů jedné kamery a indukční smyčky na silnice nebo mikrovln a Dopplerova efektu. Do silnici jsou vloženy dvě indukční smyčky, které obě reagují na přejetí auta. Z časové prodlevy je pak vypočítaná rychlost auta. V případě překročení rychlosti pak kamera auto i řidiče nasnímá, aby mu mohla býti vyměřena pokuta

Použití tohoto měření je především na křižovatkách a přechodech pro chodce. Toto měření rychlosti by se však dalo řešit jen za pomocí kamery, bez nutnosti použití další techniky. Ušetřili bychom tím nejen náklady na položení tohoto zařízení, ale i na jeho údržbu. Navíc by se toto měření mohlo využít i u jiných kamer než policejních. Projekty tohoto druhu již existují, většina z nich však předpokládá specificky nastavenou kameru, zpravidla přímo nad silnici. Tyto projekty se tedy nedají aplikovat na většinu kamer, které již silnici snímají.

Cílem této práce je analyzovat současné metody detekce rychlosti a následně navrhnout a implementovat program, který dokáže odhadnout rychlost vozidel ze vstupního video záznamu. Součástí práce je i testování a analýza výsledků vytvořeného programu.

Cíle

Hlavním cílem mojí práce je vytvoření aplikace schopné z kamerového záběru určit rychlost projíždějícího vozidla. Abych toho dosáhl, je třeba si práci rozdělit na několik podúkolů. Nejprve musím analyzovat existující metody detekce obrazu a zvolit, která je z nich nejvhodnější. Dále vytvořit program, který implementuje zvolenou metodu, pomocí které budu detektovat a následně sledovat projíždějící vozidla. Dále bude zapotřebí implementovat metodu, která dokáže na zvoleném úseku silnice vypočítat průměrnou délku jednoho pixelu. Nakonec budu muset vytvořit způsob výpočtu samotné rychlosti vozidla s ohledem na získaná data.

Kapitola 1

Současný stav

1.1 Metody měření rychlosti

V současné době se využívá několika metod měření rychlosti. Nejvíce se liší v použité technologii, ale občas i ve způsobu použití. Aby mohla být vyměřena pokuta za překočení rychlosti, musí být překročení prokazatelné. Prokazování se provádí zpravidla záznamem z policejního radaru. Radary však nejsou absolutně přesné a je tedy nutné počítat s odchylkou v měření. *V ČR jsou radary schvalovány s jednotnou odchylkou ± 3 km/h při zjištěné rychlosti do 100 km/h, respektive ± 3% při zjištěné rychlosti nad 100 km/h.*[10]

1.1.1 Mikrovlné měření

Mikrovlnné měření je nejběžnější metodou používanou na uzení ČR. K měření využívá princip Dopplerova jevu. Zařízení do měřené lokace vysílá mikrovlnné vlny. Ve chvíli, kdy do lokace vjede vozidlo, tyto vlny se od něj odrazí zpět k zařízení. Podle změny frekvence vln lze pak určit rychlost projíždějícího vozidla

Výhoda tohoto měření je, že není náročné na pozici. Využívá se na informačních tabulích, ručních radarech i policejních automobilech. Nevýhodou je nemožnost přesného zaměření na konkrétní vozidlo. Zařízení vždy vysílá vlny do určité lokace a reaguje pak na jakékoliv její narušení. Naměřená rychlost tedy může být jak vozidlo, tak i letící pták. Z tohoto důvodu tyto zařízení vyžadují vždy dohled osoby. Jedinou výjimkou jsou informační cedule, které slouží jen jako upozornění pro řidiče. Další nevýhodou je, že zařízení není schopné si poradit s více cíli najednou a je tedy nevhodné na silně vytížené silnice.

Někteří řidiči se navíc začali proti těmto radarům bránit a využívat antiradarů. To jsou přístroje, které zachytávají vlnění a včas varují řidiče před kontrolou. Vlny totiž mohou putovat až do vzdálenosti několika stovek metrů,

avšak efektivní měření lze začít až na 60 metrech. Policie většinou měří na vzdálenost okolo 35 metrů[3].

Obrázek 1.1: Mikrovlný radar [1]

1.1.2 Laserové měření

Laserové měření funguje podobně jako mikrovlnné měření, avšak místo vln využívá světelných paprsků. Ty se stejně jako mikrovlny odráží od měřeného vozidla zpět k zařízení. Ze znalosti rychlosti světla a časového rozdílu je tak možné zjistit vzdálenost měřeného vozidla. Pro samotné měření rychlosti je toto měření provedeno několikrát v pravidelné frekvenci. Z rozdílu vzdáleností vozidla mezi jednotlivými měřeními následně zařízení spočítá jeho rychlost.

Oproti mikrovlnnému měření má výhodu toho, že je mnohem lépe zaměřitelné a funguje na vzdálenost až 200 metrů. Další výhodou je, že nelze předem detekovat a přizpůsobit rychlost. V době, kdy je paprsek detekován tak už probíhá samotné měření. Přesto ale jde narušit pomocí rušiček. Ty při detekci paprsku začnou nazpět vysílat paprsek vlastní a měření tím naruší. Užívání těchto rušiček je v ČR zakázáno.[3]

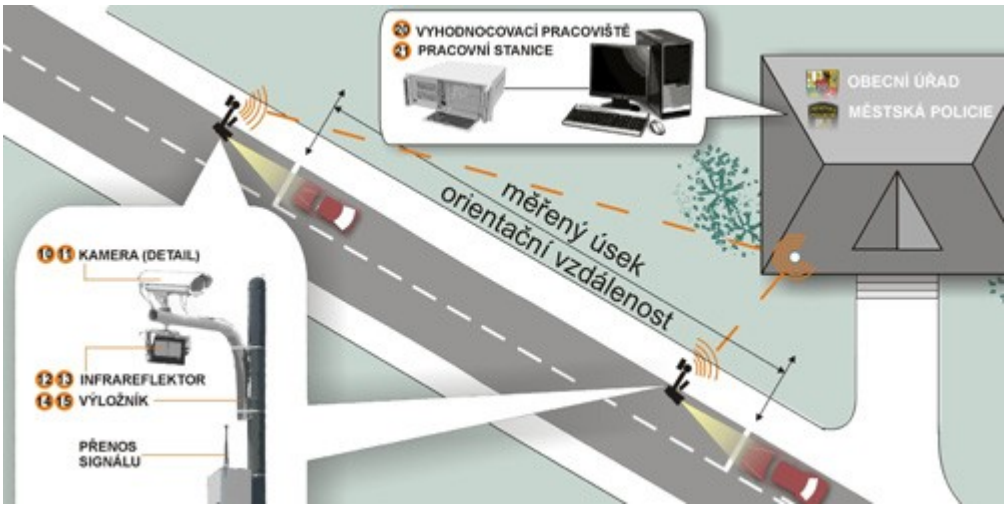
1.1.3 Úsekové měření

Úsekové měření, měří rychlost auta v rámci daného úseku silnice. Nezáleží na tom zda auto někdy během jízdy překročilo povolenou rychlost, ale na tom zda průměrná rychlost auta v daném úseku byla vyšší než maximální povolená rychlost.

Auto je na začátku úseku kamerou zaznamenáno společně s časem. Na konci úseku je opět zaznamenano a následně je z časového rozdílu a znalostí délky úseku, vypočítána průměrná rychlost řidiče. Tyto úseky mohou být

rozděleny do menších částí a průměrnou rychlost měřit několikrát za sebou. V takovém případě dostane však řidič pokutu jen jednou.

Výhoda této metody je ta, že se nedá podvést tím, že řidič před měřící kamerou na chvíli zpomalí, aby se následně zase rozjel. Nevýhoda je ta, že je použitelná pouze na usecích silnice ze kterých není možné sjet. Navíc má naměřená rychlost větší odchylku než ostatní měření protože nelze přesně změřit měřený úsek a počítá se ze zaokrouhlenou vzdáleností. V praxi to tedy jsou jen dálnice či městské obchvaty.

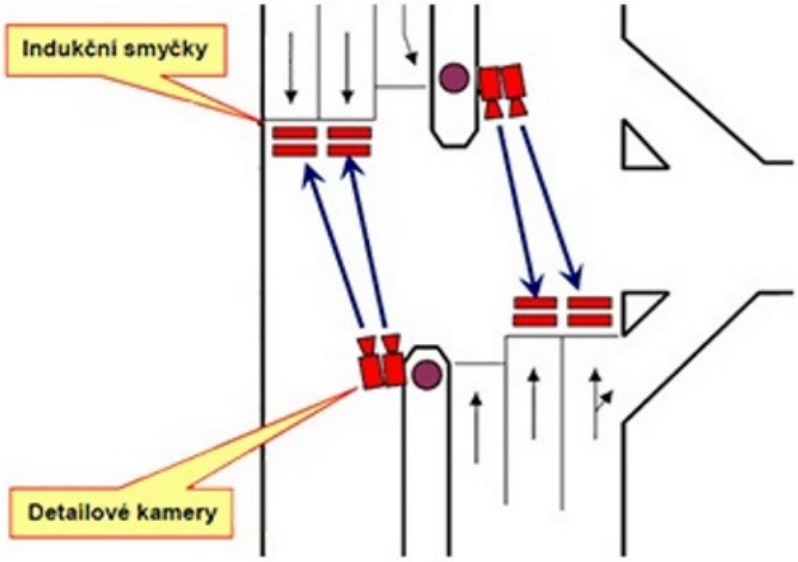


Obrázek 1.2: Schéma úsekoveho měření [2]

1.1.4 Stacionární měření

Toto měření využívá indukčních smyček zavedených ve vozovce. Na stanovené místo se umístí 2 indukční smyčky se známou vzdáleností. Zpravidla se jedná o pár desítek centimetrů. Tyto smyčky fungují jako tlaková čidla. Když přes ně přejede auto, tak z časové rozdílu v reakci smyček a jejich vzdálenosti je spočtena rychlost auta. Smyčky jsou napojeny na fotoaparát u silnice. V případě překročení rychlost je auto zaznamenáno a následně vyměřena pokuta.

Největší výhodou tohoto měření je, že se nedá narušit a je schopné operovat na relativně malé ploše. Je tedy vhodné na křižovatky či přechody pro chodce. Nevýhodou je, že vyžaduje zásah do silnice, což komplikuje jejich nasazení, stejně jako případné opravy silnice. Toto měření se v ČR moc nepoužívá a je běžnější spíše v zahraničí.



Obrázek 1.3: Schéma stacinárního měření [3]

1.2 Podobné práce

Problémem měření rychlosti vozidel z kamerového záznamu se zabývalo už mnoho lidí a jedná se o jedno z nejběžnějších využití algoritmů pro analýzu obrazu. V této části zmíním několik prací řešící podobný problém.

1.2.1 Dvou kamerový systém měření

V roce 2016 přišla výzkumná skupina pod vedením Davida F. Llorca z univerzity v Alcale ve Španělsku s návrhem měřícího systému za pomocí dvou kamer [11]. Dvě kamery byly zaměřeny na velmi krátký usek silnice. Obě tyto kamery měřily ze stejného místa, avšak jedna byla zaměřena na úsek silnice o pár metrů dál. Při projetí auta obě tyto kamery zaregistrovaly poznávací značku a čas průjezdu. Ze znalosti časového rozdílu a délky úseku pak spočetly rychlost. Jedná se tedy o stejnou metodu, jaká se využívá při úsekových měření avšak aplikovanou na velmi krátkou vzdálenost. Jejím účelem bylo eliminovat odchylku v měření způsobenou velkou vzdáleností mezi body u klasického úsekového měření.

1.2.2 Kalibrace dopravní kamery pro měření rychlost

Skupina výzkumníků z Vysokéhu učení technického v Brně pod vedením Jakuba Sochora se problémem měření rychlosti pomocí zábývá již několik let

[12]. V této práci vytváří z projíždějících aut 3D modely, které následně porovnávají s obrazem, aby získali směr natočení vozidla. Z těchto znalostí provádí kalibraci kamery, aby získaly data o silnici a její přesné délky. Díky těmto informacím pak dovedou měřit rychlost projíždějících aut s odchylkou 1.1 km/h.

1.2.3 Měření rychlost za pomocí detekce světel

Skupina vyzkumníků z Universitas Indonesia přišla s metodou měření rychlosti vozidel pomocí předních světel [13]. Ve snímcích nalézá světla, která k sobě páruje. Světla jsou vždy umístěna na stranách vozidla. Přestože vozidla mohou mít jinou délku, jejich šířka je však víceméně stejná. Díky znalosti reálné vzdálenosti světel od sebe a změny jejich vzdálosti v obraze je možné spočíst ujetou vzdálenost mezi jednotlivými snímky. Tím jsme schopni určit, jakou rychlostí vozidlo jede. Detekce světel na autě je však poměrně složitá a tedy tato metoda je vhodná pro čistě noční použití.

Kapitola 2

Existující technologie

Než přistoupím k samotné aplikaci, je třeba vysvětlit několik algoritmů, které mohu využík k řešení mého problému a pojmů, které s těmito algoritmy souvisí. Jedná se především o umělou inteligenci a její využití v detekci objektů v obraze. V této kapitole se tedy zaměřím na pojmy a algoritmy, které jsou pro mojí práci důležité. Začnu od obecného vysvětlení umělé inteligence, postupně přejdu ke strojovému učení a neuronovým sítím. Nakonec vysvětlím způsob fungování detekce objektů a dvě metody - R-CNN a YOLO.

2.1 Umělá inteligence

Umělá inteligence je vědní obor zabývající se stroji vykazující "inteligenci"[14]. Termín inteligence však nejde přesně označit a o jeho definici se dodnes vedou spory. Obecně lze říct, že stroj je inteligentní, pokud nějakým způsobem napodobuje lidskou činnost. Snahou umělé inteligence je automatizace procesů, které vykonává lidská mysl.

2.1.1 Strojové učení

Strojové učení je výpočetní metoda využívající zkušenosti pro zlepšení svého výkonu nebo vytváření přesných předpovědí[15]. Tento termín se velmi často používá s pojmem umělá inteligence a občas je i zaměňován. Nejedná se však o totéž. Strojové učení se nyní používá prakticky ve všech odvětvích lidské činnosti. Od internetových vyhledávačů, přes bankovní systémy až po zdravotnictví. Způsoby učení je několik druhů. Nazývají se učení s učitelem, učení bez učitele a zpětnovazební.

Metoda učení s učitelem předpokládá, že máme množinu vstupů a k nim i požadované výstupy. Tedy že algoritmus dostane konkrétní případy správných řešení. Algoritmus dle těchto dat začne vytvářet mapovací funkcí tak, aby pro dané vstupy dosáhl stejných výsledků. Poté lze tento algoritmus použít pro již neoznačené vstupy.

U metody učení bez učitele předem neznáme požadovaný výsledek. V tomto procesu se stroj snaží najít strukturu v neoznačených datech. Do celého procesu se nijak nevstupuje a je založen pouze na informacích ze vstupních dat[16].

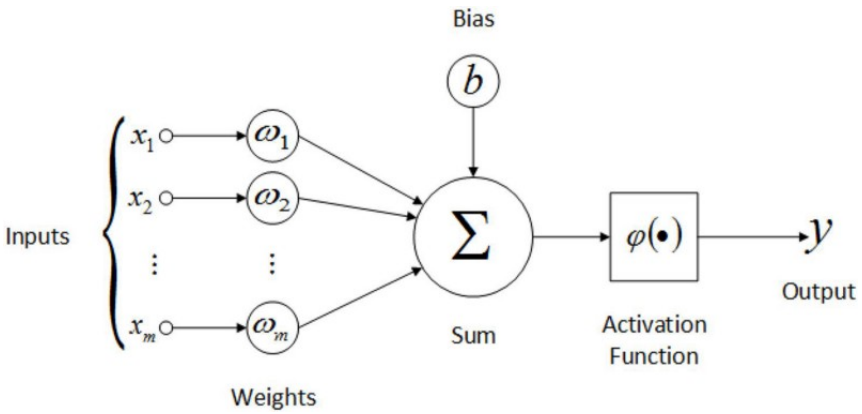
Zpětnovazební učení je učení na základně zpětné vazby. Tedy učení kdy je stroji dávána zpětná vazba o tom jak byl úspěšný. Pomocí této zpětné vazby se stroj snaží upravit svůj postup tak aby dosahl co možná největšího úspěchu.

2.1.1.1 Neuron

Protože se umělá inteligence snaží napodobit lidskou činnost, není překvapivé že se snaží napodobit i lidský mozek. Umělý neuron je výsledkem této snahy. V biologii se jedná o základní složku lidského mozku. V umělé inteligenci je neuron matematickou funkcí. Ta na začátku přijme informaci z vnějších vstupů či z jiného neuronu. *Poté, co neuron přijme vstupy, tak jejich hodnoty vynásobí váhami. Následně tyto součiny sečte a pokud je výsledek větší než stanovený práh, tak je transformuje předem danou přenosovou funkcí a pošle na výstup.* [17] Matematický model neuronu se tedy dá zapsat jako

$$f(x)\begin{cases}1 & \text{if} \sum_{i=0}^{n} \omega_i * x_i + b \geq 0 \\ 0 & \text{if} \sum_{i=0}^{n} \omega_i * x_i + b < 0\end{cases}$$

x jsou jednotlivé vstupy, ω je váha vstupu a b je práh. Graficky je tato funkce ukázaná v obrázku 2.1

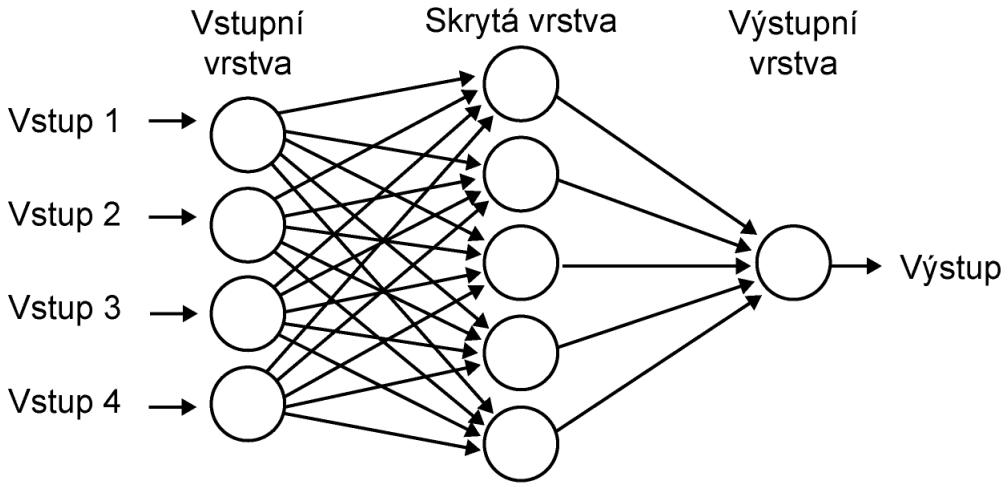


Obrázek 2.1: Grafické znázornění neuronu[4]

2.1.1.2 Neuronové sítě

Propojením výstupů neuronů s dalšími vstupy vzniká neuronová síť. Velmi často jsou v této síti neurony uspořádány do několika vrstev. První vrstva se skládá z neuronů do kterých vstupují data pouze z vnějších senzorů. Poslední

vrstva se nazývá výstupní vrstvou. Výstupy jejich neuronů již putují pouze na výstup ze sítě. Zbylé vrstvy se nazývají skryté. Jak může vypadat takové propojení je vidět na obrázku 2.2. Neuronová síť má schopnost učení. Činí tak pomocí uprav vnitřních vah a prahů v jednotlivých neuronech.



Obrázek 2.2: Ukázka propojení neuronů v neuronové síti [5]

2.1.1.3 Konvoluce a Konvoluční neuronová síť

Konvoluční neuronová síť je druhem neuronové sítě. Vyznačují využíváním konvoluční vrstvy. Díky tomu jsou schopny zpravovávat velké vstupy za použití menšího množství paramaterů. V konvoluční vrstvě, jak už název napovídá, probíhá operace konvoluce.

Konvoluce je matematická operace mezi dvěma funkcemi $x_1(t)$ a $x_2(t)$ téhož argumentu definovaný v případě spojitých funkcí integrálem

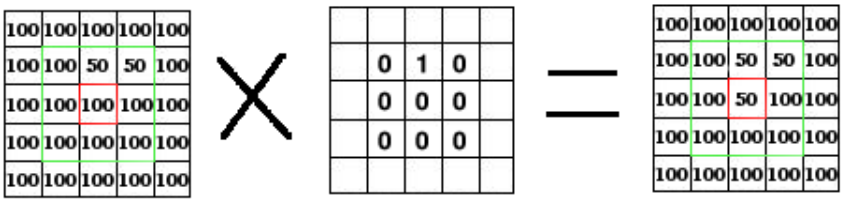
$$x(t) = x_1(t) * x_2(t) = \int_{-\infty}^{\infty} x_1(\tau)x_2(t-\tau)d\tau,$$

kde funkce $x_2(t)$ se často nazývá konvoluční jádro. [18]

Často se využívá v počítačové grafice pro zpracování dvou rozměrného diskrétního obrazu. Obraz je v počítači reprezentován jako dvourozměrná matice pixelů. Konvoluční jádro v tomto připadě je též dvourozměrnou maticí zpravidla o velikosti 3×3. Též je potřeba určit velikost kroku. Jádro postupně přikládáme na matici obrazu. Poté pronásobíme na sobě ležící prvky a jejich součet zapíšeme do výstupní matice. Jádro následně posuneme o daný krok výpočet opakujeme. Jeden tento výpočet, pro jádro o velikosti 3×3 se dá zapsat jako

$$y_{i,j} = \sum_{k=1}^{3} \sum_{l=1}^{3} x(i+k, j+l) * f(i,j)$$

i a j představují pozici na které se nacházíme $y_{i,j}$ je výsledný pixel, $x(i,j)$ je vstupní obraz a $f(i,j)$ je konvoluční jádro. Praktická ukázka výpočtu je na obrázku 2.3. Výhodou konvoluce je její schopnost zvýraznit specifické znaky matice a zároveň zachovat informaci o jejich umístění. Problém v konvoluci může být ztráta informace na okraji obrazu. Tomu se předchází tzv. zero-paddingem. Okraj celého obrazu je před konvolucí rozšířen o nuly. Velikost rozšíření záleží na velikosti jádra. Jedná se o velmi jednoduchou, ale účinou metodu.



Obrázek 2.3: Ukázka konvoluce [5]

2.1.2 Počítačové vidění

Počítačové vidění je označení pro obecné systémy, které pracují na základě informací získaných ze zpracování obrazu[19]. Cílem počítačového vidění je analyzovat vstupní obrazová data a najít v nich informace co mají představovat. Využívá se například při detekci změn v obraze u bezpečnostních kamer či indexování obrázků a videí pro účely databáze.

2.2 Metody detekce objektů

Metody detekce objektů jsou technologie zabývající se rozpoznáním objektů na digitálních fotografiích a videích. Často se v těchto metodách využívá neuronových sítí. Celý proces se skládá ze dvou částí, lokalizace objektu a klasifikace obrazu. Lokalizace objektu vyhledává v obraze jednotlivé objekty a snaží se určit jejich ohraničení. Klasifikace obrazu se snaží identifikovat objekt v obraze a stanovuje pravděpodobnostní jistotu odhadu. Obě tyto části mají své problémy, které musí řešit.

Lokalizace objektu je časově velmi nestabilní, protože předem neznáme velikosti objektů ani jejich počet. Navíc kvalitní obrazy mají několik set tisíc

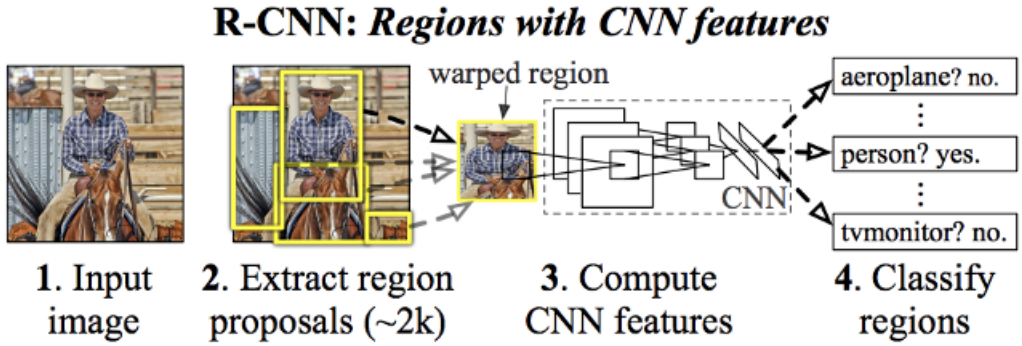
pixelů a přesto se může jednat o jeden objekt. Některé objekty navíc mohou splývat do sebe podobnou barvou. Je tedy třeba zajistit stabilní způsob lokalizace, který však nebude zacházet do přílišných detailů.

Klasifikace obrazu určuje jaký objekt se nachází v obraze. Klasifikace se provádí vždy nad obrazem jako celkem. Pro jeden obraz tedy dostaneme jednu klasifikaci, i kdyby se na něm vyskytovaly objekty dva. Dříve se toto provádělo pomoci porovnávání podobností s již známými snímky. Dnes se využívá předem naučených neuronových sítí. Klasifikace naráží na problém - aby mohl být určitý typ objektu identifikován, musí být na něj program naučen. Z tohoto důvodu se vytváří sítě určené vždy k danému účelu.

Detekce objektů využívá principů obou těchto metod. Obě jsou samy o sobě časově poměrně náročné, což při využití obou metod dělá detekci objektů velmi pomalou. Při použití naivního řešení může detekce objektu v jednom obraze trvat jednotky až desítky minut. Bylo tedy vytvořeno několik metod pro její zrychlení. Tyto metody se liší především v přístupu k lokalizaci objektu. Všechny dosahují dostačující rychlosti pro praktické použití i dobré přesnosti.

2.2.1 R-CNN

První metodou je R-CNN(Regional - Convolution Neural Network), která přišla s řešením, kdy je z obrazu před klasifikací vytaženo přibližně 2000 lokalit s dobrou šancí na obsažení objektu[20]. Klasifikace pak probíhá nad jednotlivými kandidáty pomocí neuronové sítě. Díky tomu, že klasifikaci provádíme nad menšími obrazy místo nad velkým celkem, se nám její rychlost výrazně zvyšuje. Tato metoda byla pozdější ještě vylepšena metodou Fast R-CNN, kde



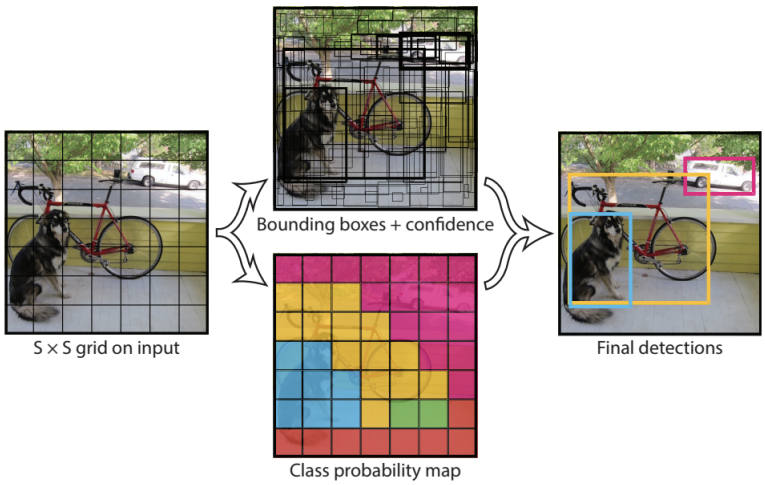
Obrázek 2.4: Ukázka funkce R-CNN[6]

se konvoluce obrazu provádí nad celým obrazem místo nad zvolenými regiony. Další vylepšení přišlo s metodou Faster R-CNN, kde se i odhady regionů dělají pomocí neuronové sítě.

Tato metoda se neustále rozvíjí a pravidelně vychází nové verze. V současnosti je tato metoda velmi přesná i dostatečně rychlá pro běžné použití.

2.2.2 YOLO

Metoda YOLO (You Only Look Once)[21] přistupuje k detekci odlišně. Zatímco R-CNN se snaží rozdělit obraz na regiony a nad nimi provádět klasifikaci, YOLO provádí obě tyto části současně. Obraz je rozdělen na mřížku o velikosti $S * S$. V každé buňce této mřížky se pak provádí lokalizace a klasifikace objektu pomoci neuronové sítě s předpokladem, že se střed objektu nachází v dané buňce. Tento proces se provádí nad každou buňkou několikrát - zpravidla pětkrát. Tímto postupem dostaneme velké množství klasifikovaných lokací společně s jejich pravděpodobnostními jistotami. Pokud pravděpodobnostní jistota přesáhne určitou mez, je objekt detekován.



Obrázek 2.5: Ukázka detekce objektu metodou YOLO[7]

Výhodou této metody je mnohem vyšší rychlost než R-CNN. V roce 2018 byla jeho rychlost testována na Univerzitě Jinmei. Použitý procesor byl Inter Core i7-7770, 3.60GHZ*8, paměť 7.7GB , Grafická karta byla GTX 1080. Operační systém byl Ubuntu16.04, 64-bit. Při tomto testování bylo dosaženo rychlosti zpracování jednoho snímku 0.03 sekund, což je 33 snímků za vteřinu [22]. Nejběžnější rychlost snímkování videa, která se využívá v například v televizi je 30 snímků za sekundu. Při použití dostatečně silného hardwaru YOLO algoritmus tedy dokáže zpracovávat snímky v reálném čase. Nevýhodou však je, že nedosahuje takové přesnosti a nedokáže si poradit s malými objekty, jako je hejno ptáků. Je tedy mnohem vhodnější v situaci, kde je mnohem důležitější objekty nacházet a nezáleží tolik na přesnosti.

2.3 Metody odhadování pohybu

Přestože metody detekce objektů nám poskytují dobrý způsob k nalézání objektu ve snímcích, nejedná se o jediný způsob, jak k němu přistupovat. Metody odhadování pohybu, jak už název napovídá, hledají objekty pomocí jejich pohybu. Tyto metody porovnávají po sobě následující snímky a snaží se v nich najít pohybující se objekty. Z principu věci se tedy jedná o metody určené striktně pro video záznamy. Na rozdíl od metod detekce objektů se však už nezabývají klasifikací samotného objektu. Všechny metody odhadování pohybu též vyžadují statickou kameru. Jejich výhodou oproti metodám detekce objektů je mnohem vyšší rychlost, způsobená především odstraněním klasifikace. Dále některé tyto metody dokáží objekty nejenom nalézt, ale též i jejich pohyb sledovat a dokonce i předvídat. Jsou tedy mnohem vhodnější do situací, kdy je důležitější objekty nalézat, ale není důležité o jaký objekt se jedná.

2.3.1 Diferenční metoda

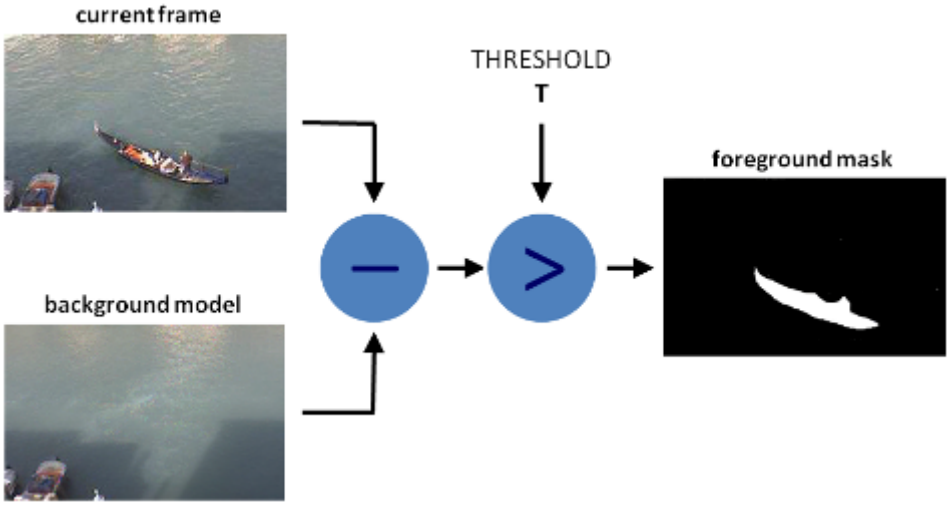
Diferenční metoda je v podstatě naivním řešením odhadování pohybu. Funguje na principu porovnávání dvou po sobě jdoucích snímků. Jakoukoliv dostatečně velkou změnu v pixelech mezi snímky detekuje jako pohyb.

Tato metoda je velmi jednoduchá na implementaci a nenáročná na výpočet. Má však několik závažných problémů. Pro správnou funkčnost totiž vyžaduje prakticky statické prostředí, tedy takové, kde je velmi nízká pravděpodobnost jakékoliv změny obrazu. Jako pohyb totiž označí libovolnou změnu v obraze, a to včetně lehkého vánku či změny osvětlení. Objekt, který má být detekován, též musí mít dobrý kontrast aby nesplýval s okolím. Též je zapotřebí, aby záznam měl vysokou frekvenci snímků, aby se eliminovaly problémy změny osvětlení. Z těchto důvodů má tato metoda jen velmi omezené použití v praxi.

2.3.2 Odstranění pozadí

Metoda odstranění pozadí funguje na podobném principu jako diferenční metoda, tedy porovnávání dvou snímků. Na rozdíl od ní ale snímek neporovnává s předchozím, ale s modelem pozadí. Tento model pozadí si metoda sama vytváří v průběhu záznamu. Pixely, které jsou delší dobu neměnné se ukládají a průměrují s předchozími. Čím déle tedy záznam běží, tím více máme informací o jeho pozadí. Pomocí znalostí vzhledu pozadí tak velmi snadno určíme, které pixely do něj nepatří a jsou tedy součástí pohybujícího se objektu. Objekt dokážeme nalézt dokonce i když se dočasně zastaví. Problémem může být, pokud se objekt zastaví na příliš dlouhou dobu. Metoda jej totiž začne brát jakožto součást pozadí a začne jej průměrovat se svým vytvořeným modelem pozadí. Tento problém však nemá řešení. Není totiž možné určit, který objekt se skutečně má pozadím stát a který ne. Příkladem takového objektu je třeba

zaparkované auto. V některých případech se však jedná jen o dočasný stav, což je např. stání na semaforu.



Obrázek 2.6: Grafické znázornění metody odstranění pozadí[8]

2.3.3 Optický proud

Optický proud je metoda popisující pohyb objektů mezi jednotlivými snímky. Spočívá v tom, že každý pixel na snímku je popsán vektorem. Tento vektor určuje směr a průměrnou rychlost pohybu daného pixelu. Díky těmto vektorům jsme pak schopni snadno sledovat pohyby objektů mezi jednotlivými snímky a dokonce je i předvídat.

Tato metoda se dá dále dělit na dva způsoby sledování - husté a řidké. V hustém sledování sledujeme všechny pixely v obraze. Pomocí směru pohybu jsme pak schopni určit, které pixely patří kterým objektům a vytvářet velmi přesné obrazy. V řidkém sledování sledujeme vždy jen určité význačné pixely. Tato metoda je vhodnější, pokud chceme sledovat, kde se objekt nachází, ale nepotřebujeme znát jeho kompletní přesné umístění.

Detekci pixelů mezi snímky jde řešit několika způsoby. Nejběžnější je metoda Lucas-Kanade. Tato metoda předpokládá víceméně konstatní pohyb pixelů a jejich detekci řeší pomocí kritéria nejbližších čtverců. Kritérium nejbližších čtverců zkoumá okolí sledovaného pixelu. Z navazujícího snímku vyhledá pixel, jehož okolí je nejpodobnější okolí pixelu z předchozího snímku.

Výhoda této metody je její rychlost. Protože známe rychlost a směr pohybu pixelu, můžeme předvídat jeho budoucí umístěni. Nemusíme tedy kontrolovat celý snímek, ale stačí nám část, kde předpokládáme jeho výskyt. Tímto způsobem jsme tedy schopni velmi rychle nacházet velké množství bodů v obraze. Nevýhodou této metody je, že si není schopna poradit s náhlými změnami

v rychlosti či směru pohybu objektu. Dalším problémem může být skrytí pohybujícího se objektu za překážky v popředí.

Obrázek 2.7: Ukázka práce optického proudu[9]

Kapitola 3

Návrh aplikace

V této sekci se budu zabývat návrhem samotné aplikace. Budu se věnovat tomu, jak by měla aplikace fungovat a jaké problémy je třeba řešit. Stejně tak navrhnu řešení těchto problémů a zdůvodním výběr řešení.

3.1 Požadavky

3.1.1 Funkční požadavky

F1 Měření rychlosti vozidel
Aplikace bude schopna měřit rychlost projíždějících vozidel na vloženém záznamu.

F2 Vrácení upraveného záznamu
Aplikace po dokončení měření vytvoří nový záznam. Tento záznam bude kopii původního záznamu s označenými vozy a jejich rychlostmi

F3 Výpis záznamu
Po dokončení měření bude vytvořen textový výpis všech identifikovaných vozidel a jejich rychlostí.

3.1.2 Nefunkční požadavky

N1 Aplikovatelnost na uživatelské záznamy
Aplikace bude fungovat pro záznamy vložené uživatelem

N2 Podpora více video formátů
Schopnost pracovat s video formáty avi a mp4.

N3 Robustnost
Odolnost vůči špatným vstupům či nedostatečným informacím

N4 Uživatelská přívětivost
Design uživatelského rozhraňı by měl být intuitivní a přehledný pro snadnou orientaci v aplikaci.

N5 Rychlost
Zpracování videa by mělo proběhnout v rozumné době.

3.2 Způsob užití

Aplikace bude určena k použití především na dálnice a silnice 1. třídy. Bude předpokládat že měřený úsek silnice neobsahuje žádnou zatáčku ani křižovatku.

Při spuštění aplikace bude zapotřebí nejprve vložit video záznam. Následně uživatel zadá parametry meřeného úseku. Při nespokojenosti bude možné tento úsek upravit. Když bude uživatel spokojen, spustí program měření. Ten po dokončení měření uloží do složky upravený video záznam a výpis detekovaných aut a jejich rychlstí. Graficky je toto použití na obrázku 3.1

3.3 Grafické rozhraní

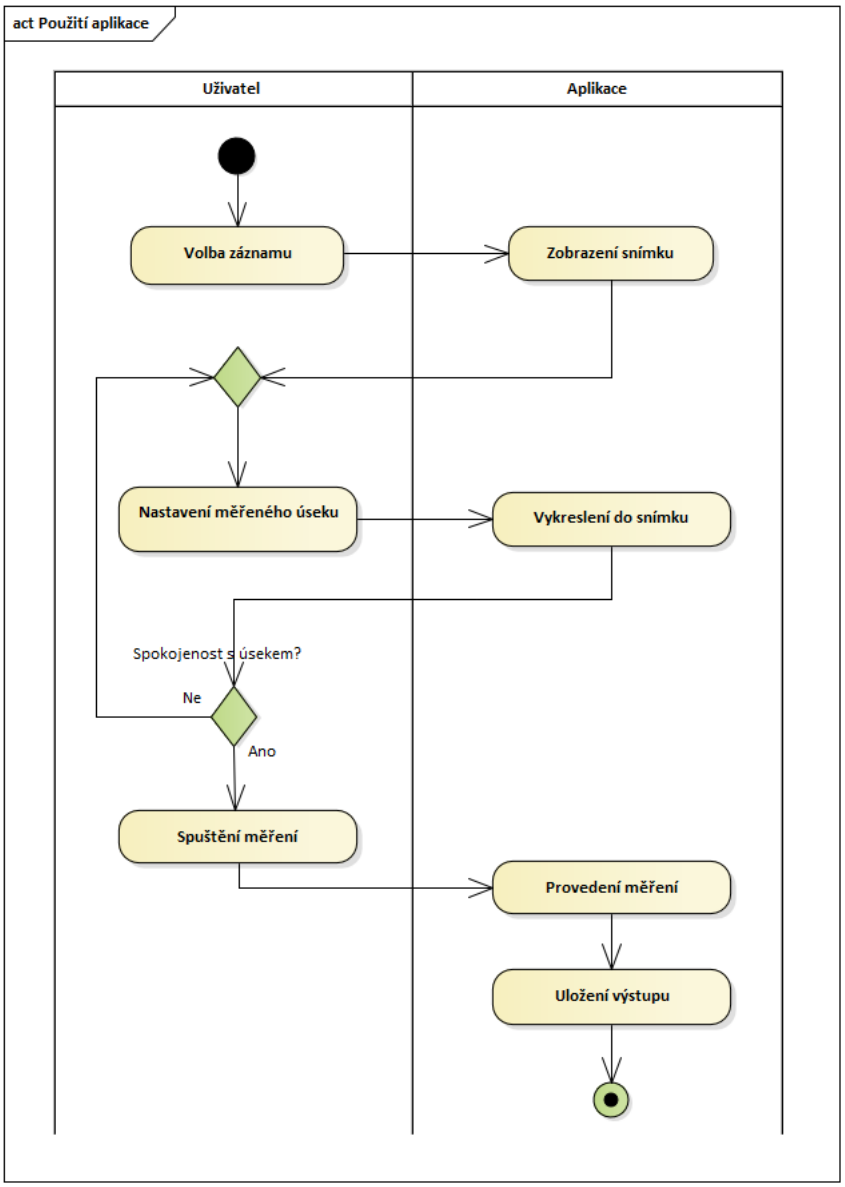
Grafické rozhraní bude sloužit k interakci uživatele s programem. Skrze něj uživatel vloží do programu video záznam a parametry měřeného úseku. Po vybrání záznamu, rozhraní zobrazí jeho první snímek. Tento snímek bude sloužit jednak pro potvrzení, že byl vložen správný záznam a také pro snazší nastavení parametrů měřeného úseku. Parametry jsou začátek a konec měřeného úseku a vzdálenost mezi nimi. Jejich vykreslení bude možné dvěma způsoby. Hlavní metodou bude vykreslení pomocí myši do snímku. Drobné úpravy pak bude možné přímo zapsat do textového pole.

3.4 Proces řešení

Na začátku procesu bude nahrán záznam do programu. Ten ho zpracuje a pro jednotlivá vozidla nacházející se v záznamu určí rychlost, jakou projela. Celý tento proces se bude dělit do několika podúloh. První úlohou je detekování vozidla. Dále je potřeba toto vozidlo identifikovat a sledovat. Nakonec je třeba změřit jeho rychlost a zobrazit ji. Tyto procesy je třeba opakovat pro každý snímek. Celý tento proces je graficky popsán na obrázku 3.3

3.4.1 Detekce vozidla

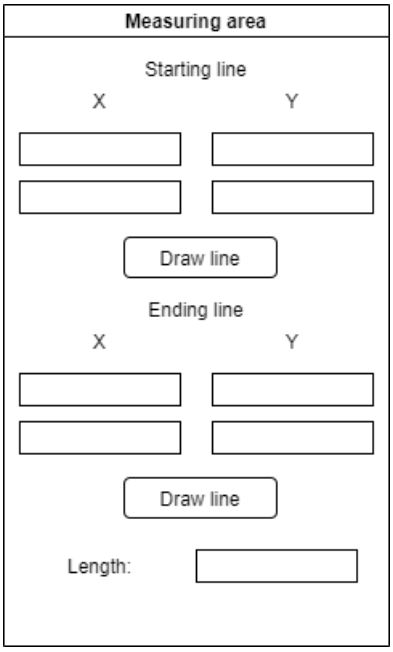
Detekce vozidla je proces, ve kterém se snažím nalézt vozidlo v záznamu. Tedy v každém snímku videa nalézt zda a kde se nachází vozidlo. Vozidlo může být



Obrázek 3.1: Případ použití aplikace

osobní auto, motorka či nákladní vůz. Pro účely měření není zapotřebí mezi nimi rozlišovat.

Pro detekci mohu použít jednu ze dvou metod - detekci objektu a odhadování pohybu. Odhadování pohybu je mnohem rychlejší než detekce objektu a hodí se při živém přenosu obrazu. Naráží však na problém splývání vozidel. Když se na silnici pohybuje více vozidel jedoucí velmi blízko sebe, či dokonce



Obrázek 3.2: Návrh tabulky s nastavením měřeného úseku

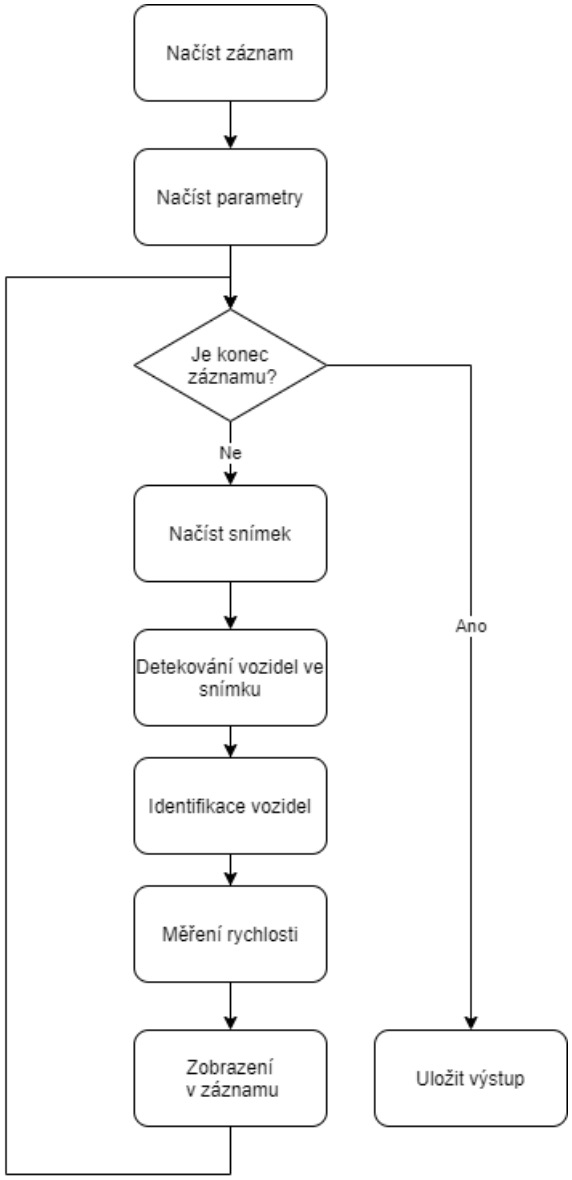
se částečně v obrazu překrývají, odhadování pohybu tyto vozidla označí za jedno vozidlo. To je problém především v dopravní špičce, kdy je na silnici velké množství vozidel s minimálními rozestupy. Pro identifikaci vozidla jsem tedy dal přednost metodám detekce objektu.

Z těchto metod jsem zvolil metodu YOLO. Důvod této volby je především rychlost. Jak bylo řečeno v kapitole 2.2.2, YOLO dosahuje vyšší rychlosti detekce než R-CNN. Trpí oproti němu sice větší nepřesností, pro moje potřeby je však dostačující. Pokud budou všechna vozidla v obraze detekována, tak detekce splnila svůj účel, i kdyby část vozidla chyběla. Z těchto důvodů dávám přednost metodě YOLO před R-CNN.

3.4.2 Identifikace vozidla

Identifikace vozidla je proces, při kterém dojde k jednoznačnému určení vozidla. Slouží k tomu, abych mohl mezi sebou odlišit jednotlivá vozidla a zároveň mohl sledovat jejich pohyb mezi jednotlivými snímky.

Možností jak k tomu přistoupit je vícero. Jednou z nich je porovnávání vzdálenosti mezi podobnými objekty(vozidly) v rámci dvou navazujících snímků. Pokud jejich vzdálenost nepřesáhne povolenou hodnotu, identifikuji tyto objekty jako jeden objekt. Snímky videa mají většinou rychlost okolo 30 snímků za sekundu. Vzdálenost, kterou vozidlo urazí mezi snímky, nemůže být při této rychlosti snímkování příliš veliká. Narážím zde ale na problém pozice kamery.



Obrázek 3.3: Posloupnost procesů v programu

Vozidla, která jsou od kamery dále, se budou mezi snímky pohybovat jen minimálně, zatímco vozidla blízko kamery se po obraze budou pohybovat velmi rychle, i kdyby obě vozidla měla stejnou rychlost. Nastavení povolené změny vzdálenosti je tedy poměrně složité. Pokud bude povolená změna vzdálenosti příliš malá, vozidla blízko kamery nebudu schopen sledovat. Pokud bude naopak příliš velká, riskuji při velkém množství vozidel jejich záměnu. Z tohoto

důvodu jsem přistoupil k metodě optického proudu.

Tato metoda místo vozidla sleduje pixel na vozidle a ke každému přistupuje individuálně. Dokáže si tedy poradit s vícero pixely, které se po obraze pohybují každý jinou rychlostí. Lze předpokládat, že se vozidla nebudou po silnici pohybovat chaoticky a nebudou náhodně měnit rychlost. Není tedy důvod k obavě, že by se pixel optickému proudu ztratil. Rozdíly v pohybu vozidla v obraze se sice s postupujícími snímky budou měnit, ale tato změna bude víceméně konstatní. To vše činí optický proud ideálním řešením této situace.

3.4.3 Měření rychlosti

Měření rychlosti bude pracovat na stejném principu jako úsekové měření. Základem tohoto měření měření času projíždějících vozidel na vybraném úseku se známou vzdáleností. Výpočet proběhne standardní cestou, tedy vzdálenost vydělená časem. Parametry o měřeném úseku, budou zadané uživatelem před spuštěním programu. Mezi tyto parametry patří pozice měřeného úseku a jeho délka

Abych mohl změřit rychlost vozidla, potřebuji sledovat vybraný bod(pixel) na daném vozidle. Reálně tak měření času neprobíhá nad celým vozidlem, ale pouze nad jeho částí. S tímto pomůže výše zmíněný optický proud, který sleduje vybraný pixel. Důležité je, aby pixel byl dostatečně odlišitelný od svého okolí a mezi snímky neměnil umístění na vozidle. Ideální pozice pro tento pixel je například poznávací značka vozidla.

Kapitola 4

Implementace

4.1 Python

Program je naprogramován v jazyce Python. Důvod k výběru tohoto jazyka je ten, že obsahuje velké množství veřejně dostupných knihoven určených ke strojovému učení. Většina z nich je navíc neustále aktualizována a obsahují zpravidla nejnovější technologie.

4.2 Knihovny

Pro vytvoření aplikace byly využity tyto knihovny:

- Numpy - Obsahuje funkce urychlující maticový výpočet. Protože všechny obrázky jsou repezentovány maticí, je tato knihovna prakticky nutností pro práci s nimi. Zároveň je potřeba pro správnou funčnost knihovny OpevCV
- OpenCV - Knihovna funcí zaměřující se na počítačové vidění. Obsahuje metody YOLO detekce, optický proud a správu videa.
- Shapely - Knihovna pracuje se geometrickými tvary. V programu je použita pro snažší práci s měřených úsekem.
- Tkinter - Základní knihovna v pythonu. Obsahuje grafické prvky tvořící uživatelské rozhraní.
- Pillow - Knihovna určená pro práci s obrázky. V aplikaci je využita pro práci se snímkem v náhledu. Obsahuje funkce přímo určené pro knihovnu Tkinter.

4.3 První spuštění

Před prvním spuštěním aplikace, je třeba nejprve mít správnou verzi Pythonu. Verze použitá při programování aplikace je 3.8. Při použití starší verze je možné, že nejpůjde spustit. Následně je třeba nainstalovat potřebné knihovny. Jejich seznam je v přiloženém textovém souboru requirements.txt. Tyto knihovny je možné stáhnout všechny najednou přikazem "pip install -r requirements.txt". Poté je třeba stáhnout externí soubory. Tyto soubory je třeba vložit do složky config ve výchozím adresáři. Samotná aplikace se spouští z výchozího adresáře příkazem "python run.py"

4.4 Uživatelské rozhraní

Uživatelské rozhraní se skládá ze dvou částí - okno náhledu a tabulka parametrů měřeného úseku.

4.4.1 Náhled

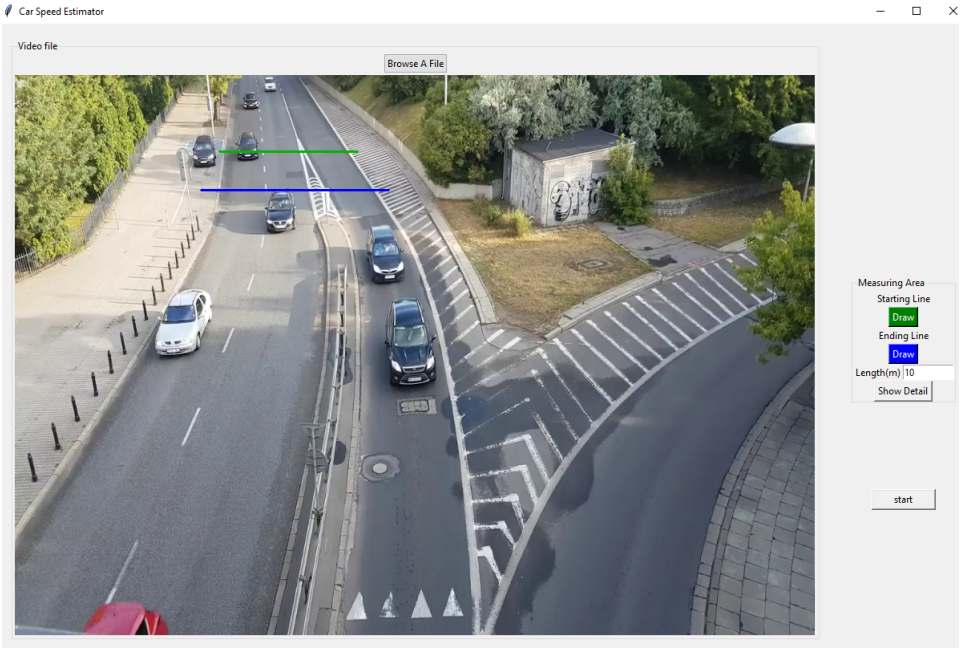
V části s náhledem je na počátku prázdný obraz a dialog výběru souborů, který umožňuje prohledávání složek pro vyhledání video záznamu. Povolené souborové formáty jsou mp4 a avi. Po vybrání je uživateli zobrazen první snímek. Ten je zmenšen na velikost 700×1000 pixelů, aby se vešel do okna aplikace. V případě menší velikosti zůstává snímek nezměněn.

4.4.2 Parametry měřeného úseku

Tato část rozhraní slouží k nastavení měřeného úseku. Hranice tohoto úseku jsou určeny uživatelem, pomocí dvou linii - startovní(zelená) a koncovou(modrá). Barevné odlišení je především pro snažší odlišení těchto linii, pro účely programu jsou zcela zaměnitelné.Tabulka s parametry má dva mody zobrazení - základní a rozšířené.

Základní zobrazení ukazuje pouze omezené množství informací. Základní zobrazení poskytuje uživateli možnost grafického vyznačení hranice úseku. Stisknutím tlačítka "Draw"se aktivuje režim kreslení vybrané linie. Následným zakliknutím dvou bodů ve snímku se mezi těmito body vykreslí daná linie a deaktivuje se režim kreslení. Tyto linie lze kdykoliv překreslit opakováním této akce. Pro správnou funci programu je třeba, aby vozidlo při přejezdu těchto linii, je přejelo v celé šíři vozidla. Následně je třeba zadat do příslušného textového pole skutečnou délku úseku v metrech. Po zadání všech parametrů lze tlačítkem "start"spustit zpracování video záznamu.

Rozšířené zobrazení je vhodné pro pokročilé uživatele a rozbrazí se tlačítkem "Show detail". Přidává do tabulky textová pole s informacemi o pozici počáteční a koncové linie. Do těchto polí lze ručně zadat požadovanou pozici koncových bodů obou linií. Stisknutím tlačítka "Show lines"se tyto linie automaticky



Obrázek 4.1: Ukázka vzhledu aplikace

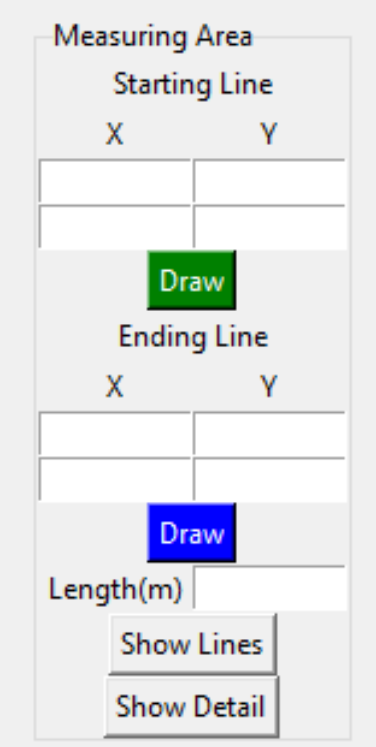
překreslí do náhledu dle zadaných hodnot. Vzhled obou zobrazení je vidět na obrázku 4.1.

4.5 Měření rychlosti

Na začátku procesu je načten video záznam a parametry měřeného úseku zadané uživatelem. Následně je spuštěn proces měření rychlosti. Ten se skládá ze tří částí, jak bylo popsáno v kapitole 3.4.

4.5.1 Detekce vozidel

Detekce probíhá pomocí metody YOLO. Pro správnou funkci této části je třeba externích konfiguračních souborů. Program je nastaven pro sledování výhradně objektů označených jako ”car”, ”truck”a ”motorbike”. Ostatní detekované objekty jsou ignorovány. Hranice pravděpodobnostní jistoty je nastavena na 60%. Pro omezení vícenásobné identifikace jsou ignorovány všechny detekce do vzdálenosti 10 pixelů od okraje snímku. Výsledkem je seznam detekovaných objektů a jejich pozic.



Obrázek 4.2: Rozhraní parametrů měřeného úseku

4.5.2 Identifikace vozidla

Seznam detekovaných objektů a jejich pozic je předán trackeru, což je podprogram určený k identifikaci vozidla. Tuto identifikaci provádí pomocí sledování a porovnávání bodů na vozidle. Pokud se ve snímku objeví nové vozidlo, tracker nalezne 3 body, které začne sledovat. Tyto body jsou střed vozidla, střed největší jednolité plochy na vozidle a střecha vozidla. Střed jednolité plochy je zpravidla čelní sklo či kapota vozidla. Tyto tři body tracker sleduje pomocí metody optického proudu. Pokud alespoň dva body leží v oblasti, kterou označila detekce, je vozidlo identifikováno. Tímto způsobem lze snížit výskyt chyb vznikajících nepřesností při detekci vozidel metodou YOLO.

4.5.3 Odhad rychlosti

Odhad rychlostí stojí na jednoduchém měření času, za jakou dobu dokáže vozidlo přejet měřený úsek. Toto měření neprobíhá nad celým vozidlem, ale pouze nad sledovanými body.

Měřený úsek je vytvořen jakožto čtyřúhelník z bodů, které zadal uživatel jako vstupní parametry. Když libovolný sledovaný bod vozidla vstoupí do tohoto čtyřúhelníku, začne běžet čítač snímků. Každý bod má svůj vlastní

čítač. V okamžiku, kdy všechny body opustí měřený úsek, je pro každý bod spočtena jeho rychlost dle délky úseku a času průchodu. Nasledně jsou tyto rychlosti zprůměrovány do jednoho výsledku.

Čas průchodu je spočten pomocí čítače a ze znalosti snímkové frekvence daného videa. Tato frekvence je zjištěna na samotném počátku měření. Snímková frekvence má také vliv na přesnost měření - čím je frekvence vyšší, tím je měření přesnější.

4.6 Výstup

Po ukončení měření je vytvořena upravená kopie video záznamu ve formátu avi. V této kopii jsou označená vozidla i s jejich rychlostmi. Dalším výstupem je soubor v csv formátu. V tomto souboru lze nalézt všechna nalezená vozidla i s rychlostmi které byli naměřeny. V případě, že se rychlost změřit nepovedla, je automaticky nastavena na rychlost 0 km/h.

4.7 Externí soubory

Toto jsou soubory, které je nutné před spuštěním programu stáhnout. Všechny tyto soubory je třeba vložit do složky config.

4.7.1 yolov3.cfg

Jedná se o konfigurační soubor pro Yolo detekci. V programu jsem využil základní soubor nastavení. Tento soubor je možné stáhnout na webových stránkách vývojové skupiny, která přišla s metodou Yolo.

4.7.2 yolov3.weights a coco.names

Toto jsou váhy pro neuronovou síť použitou v metodě Yolo. Soubor coco.names je pak soubor jmen pro jednotlivé objekty na které jsou váhy nastaveny. Tento soubor je též volně dostupný na stránkách vývojové skupiny[7].

Kapitola 5

Testování

V této kapitole se budu zabývat testování aplikace na praktických datech. Testování bylo prováděno v průběhu celé tvorby programu. Pro testování bylo využito jednak vlastních materiálů a jednak materiálů zaslaných organizací AI City Challenge[23]. Vlastní videa a výsledky všech testů lze najít na mém dropboxu[24].

5.1 Test detekce vozidel

V tomto testu jsem zkoušel zda je program schopen detekovat všechna vozidla projíždějící měřeným úsekem. Test byl prováděn na několika video záznamech s různou hustotou vozidel a různými přírodními podmínkami. Protože tento test probíhal společně s testem rychlosti, byla délka záznamu omezena na jednu minutu. Všechny záznamy a jejich nastavení lze najít na výše zmíněném odkaze.

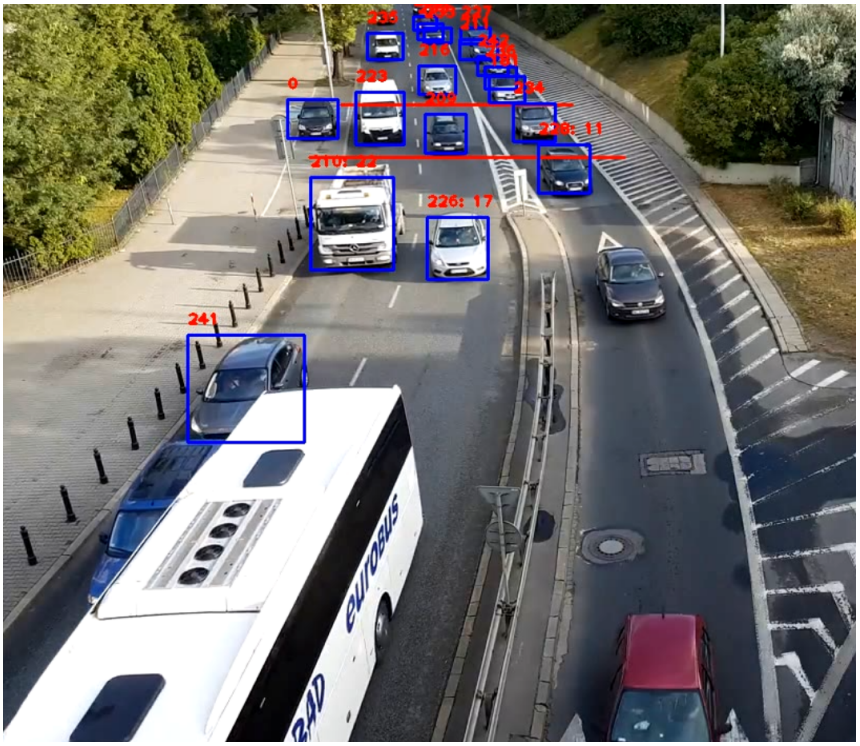
- Test 1 - Zasněžená nefrekventovaná křižovatka
- Test 2 - Pohled shora na silnici s vytvořením kolony
- Test 3 - Hustý déšť na dálničním výjezdu
- Test 4 - Frekventovaná křižovatka
- Test 5 - Pohled přímo proti směru jízdy

Výsledky testu jsou v tabulce 5.1

Během těchto testů byla aplikace schopna detekovat téměř všechna vozidla na silnici. Ukázalo se, že není neobvyklé, když vozidlo dočasně není detekováno. Ukázku takové ztráty lze vidět na obrázku 5.1. Aplikace je však na tyto situace připravena a je většinou schopna vozidlo identifikovat i po této ztrátě.

Číslo testu	Skutečný počet	Počet identifikací
1	11	10
2	61	63
3	4	4
4	58	59
5	27	30

Tabulka 5.1: Výsledky testu měření počtu vozidel



Obrázek 5.1: Druhý test detekce vozidel

Příčinou většinou bývá nedostatečné vyostření vozidla ve snímku. Nastává zpravidla v situaci, když vozidlo jede příliš velkou rychlostí. Aplikace tedy v současném stavu vyžaduje poměrně kvalitní záběry pro správnou funkčnost. Tento problém by mohl býti odstraněn použitím lepší konfigurace vah a prahů v neuronové síti. Dalším možným řešením je použití detekce pouze na nové objekty a jejich sledování nechat na některé metodě dostupné v knihovně OpenCV.

Též občas dochází k vícenásobné detekci. Tedy situaci, kdy je jedno vozidlo identifikováno vícekrát. Tato situace nastává především u nákladní vozů. Aplikace má totiž tendenci označit jak kamión, tak i přívěs za dvě rozdílná

vozidla. Tento problém by mohl býti odstraněn, stejně jako i ztráta, použitím lepší konfigurace neuronové sítě.

5.2 Test rychlosti

Tento test měřil, jakou dobu bude trvat zpracování jedné minuty jednoho videa. Detekce probíhala jen pomocí procesoru Intel Core i5-7400 @ 3.00GHz, velikost RAM 8 GB, bez využití grafické karty. Výsledky testů jsou v tabulkce 5.2

- Test 1 - Rozlišení záznamu 1280x720. Rychlost snímkování 30 fps. Formát avi. Velikost 69.7 MB
- Test 2 - Rozlišení záznamu 1280x720. Rychlost snímkování 10 fps. Formát avi. Velikost 58.3 MB
- Test 3 - Rozlišení záznamu 1280x720. Rychlost snímkování 25 fps. Formát avi. Velikost 136 MB
- Test 4 - Rozlišení záznamu 2560x1920. Rychlost snímkování 10 fps. Formát avi. Velikost 220 MB

Číslo testu	Doba zpracování	Čas na jeden snímek
1	14 minut 31 sekund	0,48 sekund
2	4 minuty 43 sekund	0,47 sekund
3	10 minut 52 sekund	0,43 sekund
4	5 minut 44 sekund	0,57 sekund

Tabulka 5.2: Výsledky testu měření rychlosti

Z výsledků těchto měření je vidět, že aplikace zpracovává video rychlostí přibližně dvou snímků za sekundu. To není pro detekci objektů špatný čas. Rychlost by šla zlepšit zapojením grafické karty. Aby to bylo možné musela by aplikace vyžadovat speciální ovladače, které nejsou dostupné pro všechny grafické karty.

5.3 Testování přesnosti měření

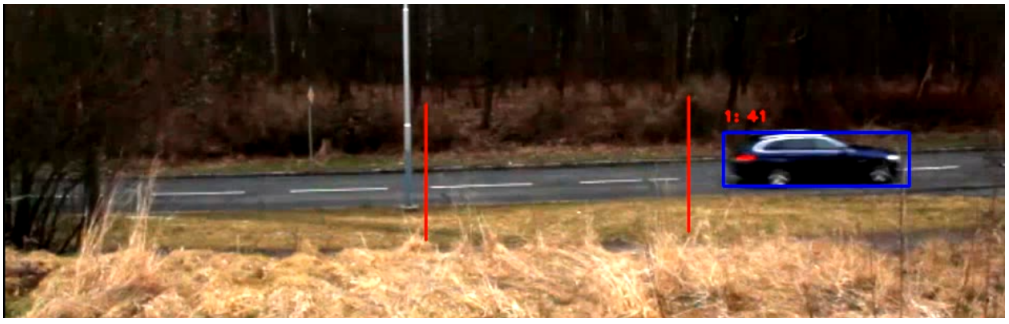
Toto testování se zaměřilo na testování přesnosti naměřené rychlosti. Pro měření bylo využito mého osobního auta. Skutečná rychlost byla získána z tachometru a pokud to bylo možné, potvrzena z radaru na rychlostní ceduli. Snímkovací frekvence nahrávky byla 25 fps s rozlišením 1280x720. Jako místo testování byla vybrána nefrekventovaná silnice za areálem Vysoké školy báňské. Vzdálenost měřeného úseku byla ručně změřena a činí 8 metrů. Auto projíždělo přes měřený úsek v obou směrech.

- Test 1 - Jízda zleva do prava. Rychlost 40 km/h
- Test 2 - Jízda zleva do prava. Rychlost 50 km/h
- Test 3 - Jízda zleva do prava. Rychlost 60 km/h
- Test 4 - Jízda zprava do leva. Rychlost 50 km/h
- Test 5 - Jízda zprava do leva. Rychlost 60 km/h

Výsledky měření jsou v tabulce 5.3

Číslo testu	Skutečná rychlost	Naměřená rychlost	Odchylka
1	40 km/h	41 km/h	1 km/h
2	50 km/h	51 km/h	1 km/h
3	60 km/h	neúspěch	neúspěch
4	50 km/h	48 km/h	2 km/h
5	60 km/h	neúspěch	neúspěch

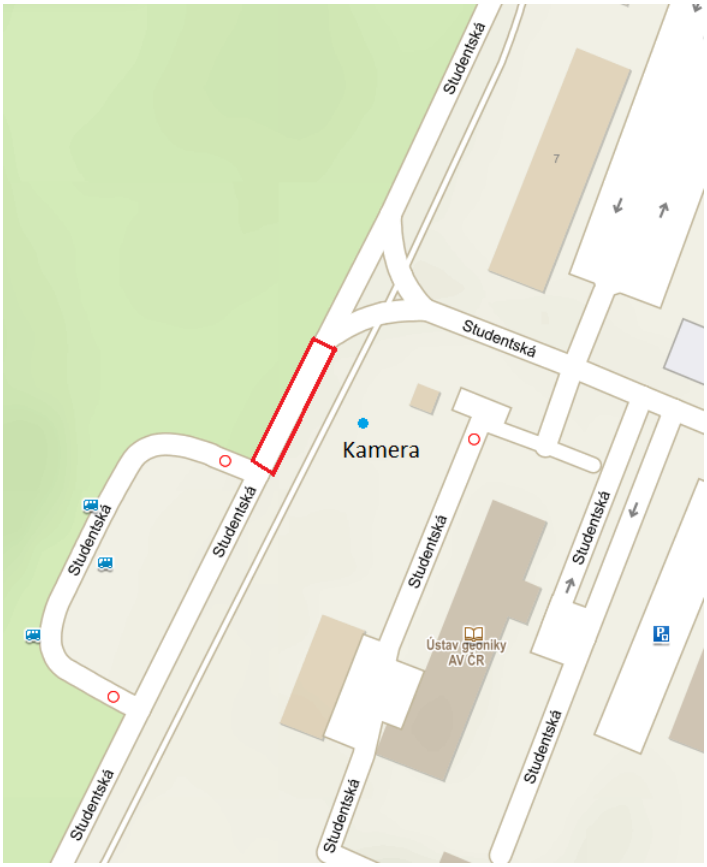
Tabulka 5.3: Výsledky testu měření rychlosti



Obrázek 5.3: Test 1 - skutečná rychlost: 40km/h

Neúspěch měření u třetího testu bylo zapříčiněno lampou stojící na kraji meřeného úseku a zakrývající část silnice. Při průjezdu vozidla totiž zakrývá jeho část. Toto způsobí problém trackeru, kterému dočasně zmizí sledovaný pixel. Tracker ve snaze ho najít určí nový podobný pixel na vozidle, který začne sledovat. U tohoto pixelu však může nastat stejný problém na dalším snímku. Toto se může opakovat tak dlouho, dokud je část vozidla skrytá za lampou. Protože meřený úsek začíná krátce za lampou, začínám měřit rychlost ještě v okamžiku kdy je část vozidla skrytá za lampou. Tím vznikl tento neuspěch.

Neuspěch u testu 5 byl způsoben špatným zaostřením auta, které se tak občasně ztrácelo z detekce.



Obrázek 5.2: Úsek silnice využitý pro měření

Přestože 2 jízdy z 5 byly neuspěšné, na výpisu je vidět, že měření rychlosti proběhlo i v těchto případech. Pokud bych vzal i tyto naměřené hodnoty, výsledek by vypadal tak jako na tabulce 5.4

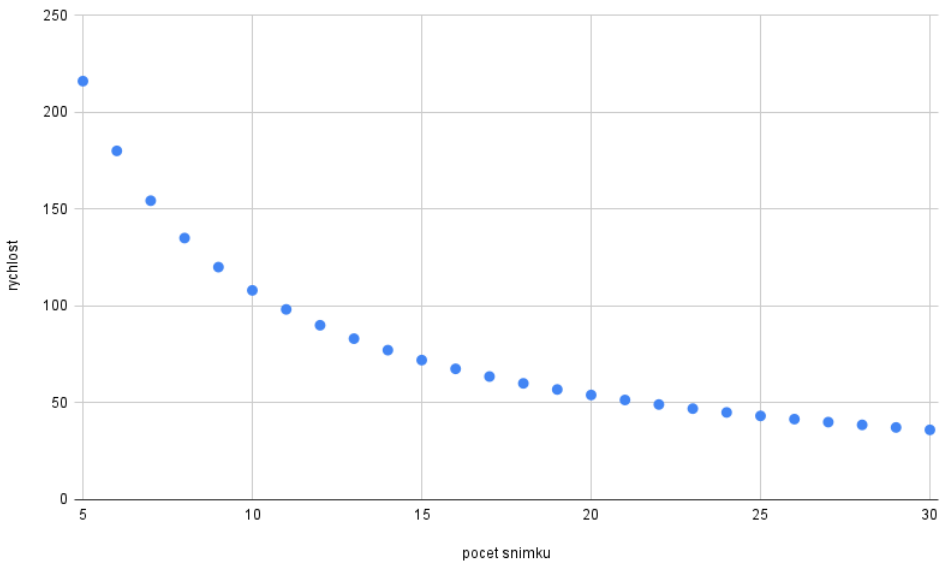
Číslo testu	Skutečná rychlost	Naměřená rychlost	Odchylka
1	40 km/h	41 km/h	1 km/h
2	50 km/h	51 km/h	1 km/h
3	60 km/h	67 km/h	7 km/h
4	50 km/h	48 km/h	2 km/h
5	60 km/h	60 km/h	0 km/h

Tabulka 5.4: Upravené výsledky testu měření rychlosti

Z těchto výsledků vychází, že maximální naměřená ochylka je 7 km/h. Zbytek měření se však vešlo do 2km/h což jsou velmi dobré výsledky a ani největší odchylka není příliš veliká.

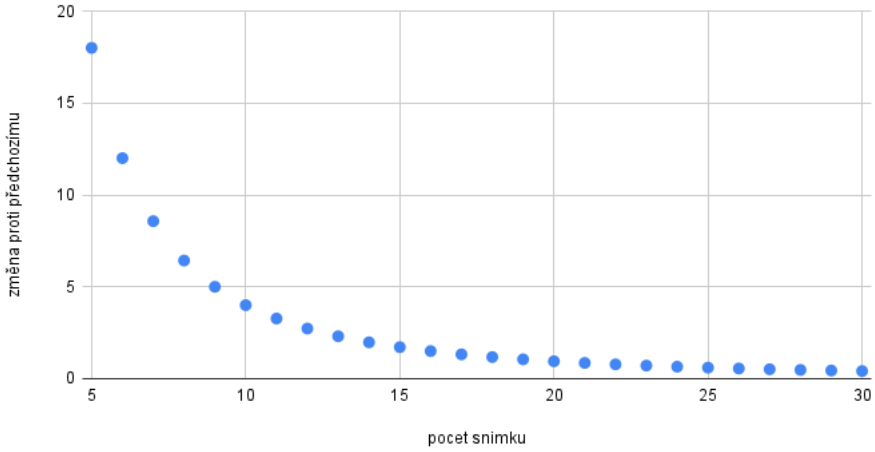
5.3.1 Problém krátké vzdálenosti

Během testování jsem objevil problém při výpočtu rychlosti, a tím je závislost přesnosti měření na vstupních parametrech. Výpočet rychlosti probíhá takto: vzdálenost/(počet snímků/fps videa). Vzdálenost je pevně daná velikost měřeného úseku a fps záleží na vloženém videu. Tedy jedinou veličinou vstupující do této funkce je počet snímků ve kterých se vozidlo nacházelo v měřeném úseku. Pro ukázku jsem vytvořil graf 5.4 ve kterém jsou ukázány vypočtené rychlosti v závislosti na počtu snímků. Počítal jsem, že délka měřeného úseku je 10 metrů a frekvence snímků je 30 fps. Pro přehlednost jsem též omezil rozsah grafu od 5 do 30 snímků.



Obrázek 5.4: Graf výsledné rychlosti dle počtu snímků

Z grafu jde vidět, že rychlosti začínají velmi vysoko avšak velmi rychle klesají. Toto klesání postupně zpomaluje. Vzhledem k tomu že počet snímků je diskrétní veličina, graf není spojitý. Rozdíly mezi dvěma po sobě jdoucími výsledky jsou při nízkém počtu snímků vysoké. Tyto rozdíly jsou vidět v grafu 5.5



Obrázek 5.5: Graf rozdílů vypočtené rychlosti mezi snímky

Aby aplikace mohla dávat rozumné výsledky, je třeba aby tyto rozdíly nebyly příliš velké. Pro dobrý odhad by tyto rozdíly neměly být větší než 5 km/h. Z grafu 5.5 lze vidět, že těchto rozdílů se dosahuje za doby, kdy vozidlo stráví v měřeném úseku alespoň 16 snímků. Pokud by v měřeném úseku bylo vozidlo zachyceno na méně snímcích, riskuji velkou odchylku měření. V případě jiné délky úseku či jiné snímkovací frekvence se požadovaný minimální počet snímků změní. Delší úsek toto minimum zvyšuje, menší snímkovací frekvence naopak snižuje. S tímto je třeba počítat během vyměřování úseku. Zároveň z tohoto lze vyvodit, že aby mohla moje aplikace přesně měřit vozidla ve vysokých rychlostech, je třeba vyměřit velkou vzdálenost nebo použít vysoko-frekvenční kameru. Velkou vzdálenost však kamera ne vždy dostatečně pokrývá a vysoko-frekvenční kamera se běžně nepoužívá.

Závěr

Cílem práce bylo provést analýzu existujících metod zpracování obrazu. Následně vytvořit návrh a implementaci softwaru schopného odhadovat rychlost projíždějících aut z kamerového záběru. Nakonec provést testování přesnosti měření tohoto softwaru na reálných datech.

V prvních dvou kapitolách jsem provedl rešerši současného stavu měření rychlosti na českých silnicích. Též jsem provedl analýzu projektů řešících stejný nebo podobný problém jako moje práce a následně analýzu existujících metod zpracování obrazu. Z této analýzy vyšlo, že nejvhodnější metodou pro vyhledání vozidel bude metoda YOLO a to především díky své rychlosti zpracování obrazu. Pro sledování vozidel byla zvolena metoda optického proudu. Tato metoda má dobré vlastnosti pro sledování plynulého pohybu, což můžeme od vozidel na silnici očekávat.

V další kapitole jsem vytvořil návrh aplikace, která by měla zpracovávat záznam a určit rychlost projíždějících vozidel. Následně byla dle tohoto návrhu aplikace implementována. Tato aplikace vrací na výstup upravený záznam včetně textového výpisu rychlostí projíždějících vozidel ve formátu csv. Aplikace je snadno ovladatelná a schopná přijmout uživatelem vytvořené záznamy. Tyto záznamy mohou být ve formátu mp4 nebo avi. Aplikace je též odolná proti špatným vstupům a pracuje v rozumné době. Byly tedy splněny všechny požadavky na aplikaci.

V závěru byl proveden test přesnosti měření. Při tomto testu byla aplikace schopna odhadnout rychlost projíždějícího vozidla s maximální odchylkou 7 km/h. Tato odchylka je zcela dostačující pro odhad rychlosti vozidla.

Jedním ze způsobů jak zlepšit aplikaci by bylo využití lepšího nastavení vah neuronové sítě. Váhy použité v moji aplikaci jsou nastaveny na 83 různých objektů, včetně zvířat a běžných předmětů. Použití vah specializovaných čistě na vozidla by zvýšilo přesnost detekce a možná i rychlost.

Další možností je využít jiného způsobu sledování. Použití detekce by šlo omezit pouze na vyhledávání nových objektů. Po jejich nalezení předat informace o něm trackeru a ten by vozidlo sledoval pomocí podobnosti mezi

snímky.

Též by šlo využít pro sledování metodu hustého optického proudu. Tato metoda místo pár bodů na vozidle sleduje všechny body. To by mohlo vyřešit problémy při částečném zakrytí vozidla a zpřesnit výpočet rychlosti. Bylo by to však na úkor rychlosti zpracování.

Přestože aplikace dává poměrně přesné odhady rychlosti, pro účely použití Policii ČR je však příliš nepřesná. Policie totiž vyžaduje maximální odchylku měření 3 km/h, což moje aplikace přesahuje. Je však použitelná pro méně náročné měření rychlosti, například sledování hustoty provozu.

Literatura

[1] Neltronics. How does a Speed camera or radar gun work? https://www.neltronics.com.au/how-does-a-speed-camera-or-radar-gun-work/, dostupné: 2021-3-25.

[2] Antiradary. Úsekové měření - další forma skrytého zdanění. https://www.antiradary.net/usekove-mereni/, dostupné: 2021-3-3.

[3] Bednář, M. Jak se měří rychlost na českých silnicích? Na toto všechno si dejte pozor. https://www.autoforum.cz/zivot-ridice/jak-se-meri-rychlost-na-ceskych-silnicich-na-toto-vsechno-si-dejte-pozor/, dostupné: 2021-3-3.

[4] Ahire, J. B. Artificial neuron. https://cdn-images-1.medium.com/max/800/1*WRG_Re8vGVuHDYigtq2IBA.jpeg, dostupné: 2021-4-9.

[5] Institut biostatistiky a analýzy LF MU. Umělá inteligence. https://portal.matematickabiologie.cz/res/image/Umela%20inteligence/obr-4-4-usporadani-neuronu-do-vrstev.png, dostupné: 2021-4-10.

[6] Gandhi, R. R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms. https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e/, dostupné: 2021-2-27.

[7] Redmon, J. Darknet: Open Source Neural Networks in C. http://pjreddie.com/darknet/yolo, 2013–2016.

[8] OpenCV. How to Use Background Subtraction Methods. https://docs.opencv.org/master/Background_Subtraction_Tutorial_Scheme.png, dostupné: 2021-4-9.

[9] Patait, A. An Introduction to the NVIDIA Optical Flow SDK. https://www.edge-ai-vision.com/wp-content/uploads/2019/03/Football-512x288-1024x576.png, dostupné: 2021-4-9.

[10] Dopravní právo. Překročení rychlosti. http://www.dopravni-pravo.cz/prekroceni-rychlosti/, dostupné: 2020-11-30.

[11] Llorca, D. F.; Salinas, C.; et al. Two-camera based accurate vehicle speed measurement using average speed at a fixed point. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2016, pp. 2533–2538.

[12] Sochor, J.; Juránek, R.; et al. Traffic surveillance camera calibration by 3d model bounding box alignment for accurate vehicle speed measurement. *Computer Vision and Image Understanding*, volume 161, 2017: pp. 87–98.

[13] Sina, I.; Wibisono, A.; et al. Vehicle counting and speed measurement using headlight detection. In *2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, 2013, pp. 149–154, doi:10.1109/ICACSIS.2013.6761567.

[14] Kučera, D. artificial intellingence and man umělá inteligence a člověk. *VŠPP Enterpreneurship studies*, 2018: p. 5.

[15] Mohri, M.; Rostamizadeh, A.; et al. *Foundations of machine learning*. MIT press, 2018.

[16] Lloyd, S.; Mohseni, M.; et al. Quantum algorithms for supervised and unsupervised machine learning. *arXiv preprint arXiv:1307.0411*, 2013.

[17] Durčák, P. Neuronové sítě a princip jejich fungování. https://www.napocitaci.cz/33/neuronove-site-a-princip-jejich-fungovani-uniqueidgOkE4NvrWuNY54vrLeM670eFNQh552VdDDulZX7UDBY/?query=neuronov%E9%20s%EDt%EC&serp=1, dostupné: 2021-3-5.

[18] Institut biostatistiky a analýzy LF MU. Konvoluce. https://portal.matematickabiologie.cz/index.php?pg=analyza-a-modelovani-dynamickych-biologickych-dat--signaly-a-linearni-systemy--modely-velicin-spojitych-v-case-ii--1-konvoluce, dostupné: 2021-4-12.

[19] Havle, O. Strojové vidění I: Principy a charakteristiky. *Machine vision I: Principles and characteristics, AUTOMA*, volume 1, 2008.

[20] Girshick, R.; Donahue, J.; et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition (CVPR), June 2014.

[21] Redmon, J.; Divvala, S.; et al. You Only Look Once: Unified, Real-Time Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[22] Yang, W.; Jiachun, Z. Real-time face detection based on YOLO. In *2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII)*, 2018, pp. 221–224, doi:10.1109/ICKII.2018.8569109.

[23] Naphade, M.; Tang, Z.; et al. The 2019 AI City Challenge. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019, p. 452–460.

[24] Car speed estimator. `https://www.dropbox.com/sh/j7g7h34zftju5wt/AACqRDoNw3T2Yn7l-1hI5HuPa?dl=0`, accessed: 2021-05-10.

[25] Car speed estimator. `https://github.com/opendatalabcz/car-speed-estimator`, accessed: 2021-05-10.

Příloha **A**

Návod k instalaci

Pro správný běh aplikace je třeba mít verzi Pythonu 3.8 a vyšší. Aplikaci lze stáhnout na githubu[25]. Testovací data můžete získat na mém dropboxu [24].

Po stažení aplikace je třeba naistalovat knihovny. Toto lze provést příkazem: "pip install -r requirements.txt". Při použití verze pythonu 3.9 a vyšší je možné, že použitá verze OpenCV nebude fungovat. V takovém případě stáhněte verzi potřebnou pro váš python příkazem: "pip install opencv-python"

Do složky config je třeba vložit externí soubory coco.names, yolov3.cfg a yolov3.weights. Tyto soubory lze stáhnout ze stejné lokace jako testovací data.

Příloha B

Uživatelská příručka

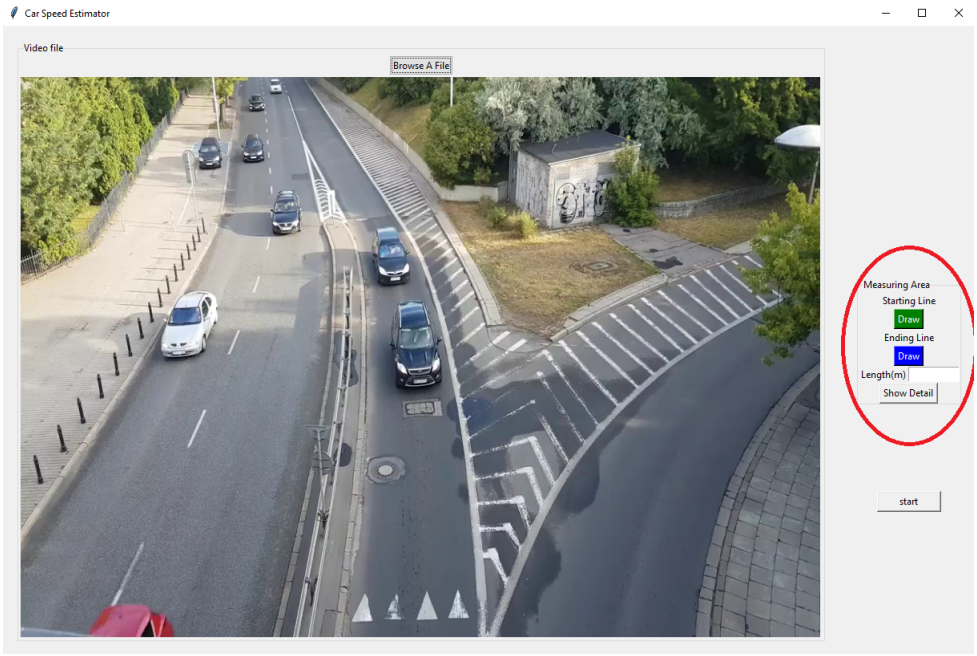
Program se spustí příkazem: ”python run.py”. Po spuštění aplikace se zobrazí uživatelské rozhraní. V levé části je místo pro vložení videa. Podporované formáty jsou avi a mp4. Po vybrání videa se zobrazí v aplikaci jeho první snímek. V pravé částí rozhraní je nastavení měřeného úseku. Tento úsek je ohraničen dvěmi liniemi. Tyto linie lze vykreslit výběrem vybrané linie a následně kliknutím na dva body v zobrazeném snímku. Mezi těmito body se následně vykreslí zvolená linie.

Druhou možností je zobrazením detailů parametrů. Toto zobrazí textová pole, do kterých lze vložit pozice zvolených pixelů. Tlačítkem ”Show lines”se provede jejich vykreslení

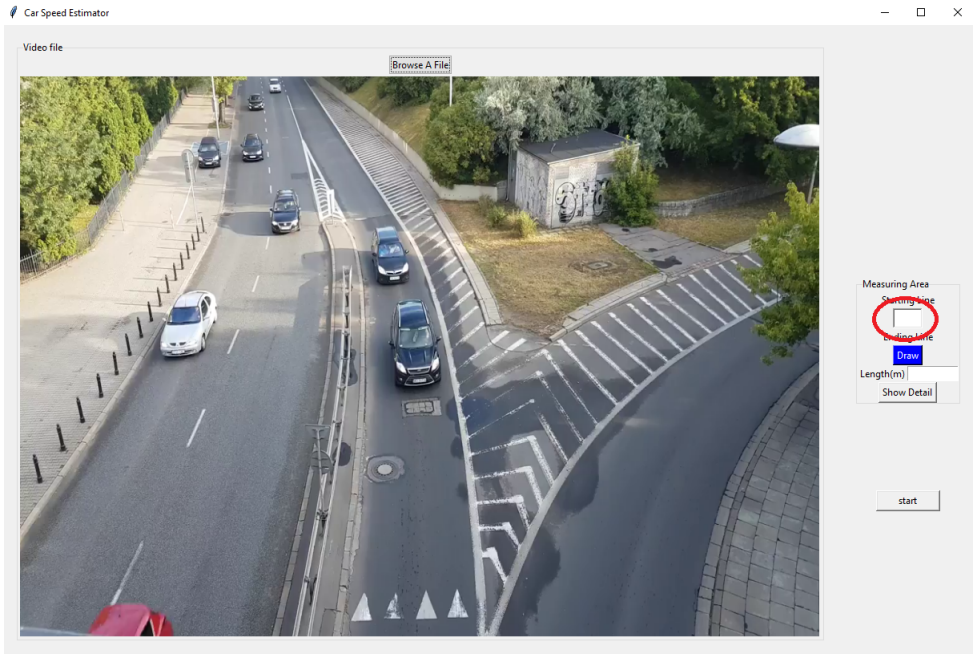
Obě linie jsou vzájemně zcela zaměnitelné a nezáleží na tom, přes kterou budou vozidla projíždět jako první.

Po zvolení úseku je třeba zadat v metrech reálnou vzdálenost mezi liniemi. Toto se zadává do kolonky ”Lenght”. Po zadání všech parametrů se program spustí tlačítkem ”Start”

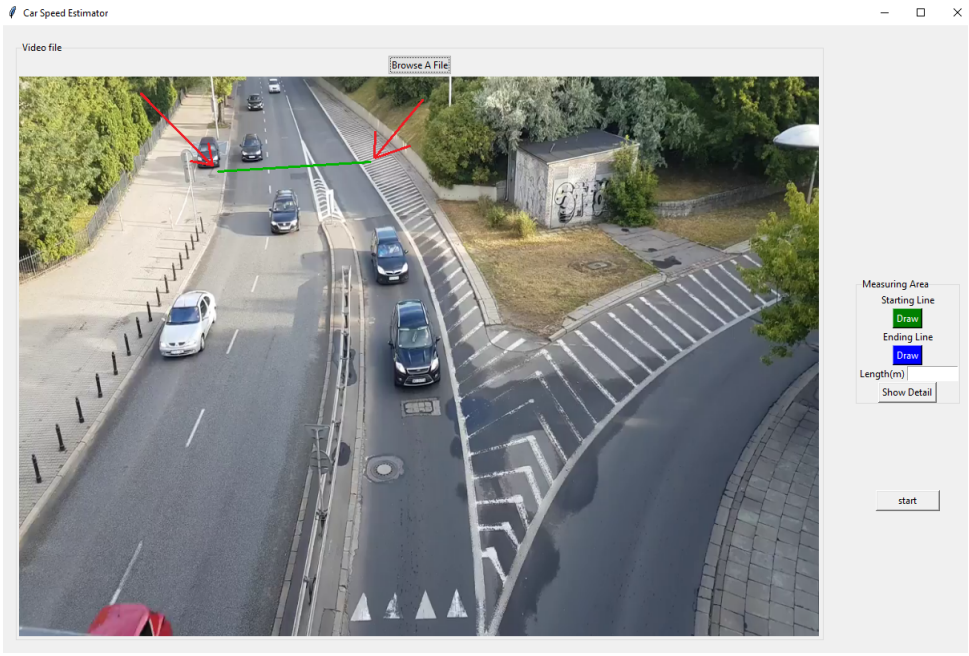
V průběhu práce bude aplikace zobrazovat snímky, na kterých momentálně pracuje. Pokud si nepřejete zpracovat celý záznam, je možné stisknutím klávesy ”escape”zpracování přerušit. Výsledné video bude uloženo do úvodní složky pod jménem ”Result”stejně jako csv soubor.



Obrázek B.1: Nastavení měřeného úseku



Obrázek B.2: Zakliknutí vybrané linie



Obrázek B.3: Zakliknutí vybraný bodů ve snímku

Příloha C

Obsah přiloženého CD

- `readme.txt` ..... soubor s popisem obsahu CD
- `requirements.txt` ..... soubor s popisem knihoven potřebných pro běh aplikace
- `src` ..... složka obsahující zdrojové kódy
- `config` ..... složka pro vložení externích souborů
- `run.py` ..... spouštěcí script aplikace
- `thesis` ..... složka s textovou částí práce
  - `thesis.pdf` ..... text práce v PDF formátu