UNIVERZITA PARDUBICE

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Disertační práce

Doktorský studijní program: P2612 Elektrotechnika a Informatika
obor Informační, komunikační a řídicí technologie

Název tématu disertační práce:

Odhad hloubkové mapy za využití neuronových sítí

Autor: Ing. Beran Ladislav

Školitel: doc. Ing. Petr Doležel, Ph.D.

Školitel specialista: Ing. Luboš Rejfek, Ph.D.

Pardubice, 2022

PROHLÁŠENÍ AUTORA

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 30. 1. 2022

Ing. Ladislav Beran

PODĚKOVÁNÍ

Rád bych poděkoval mému školiteli doc. Ing. Petru Doleželovi, Ph.D. za cenné rady a podporu při tvorbě této práce. Dále bych rád poděkoval mé manželce a rodičům za trpělivost, shovívavost a podporu během studia, bez které bych toto studium nedokončil. Mé poděkování taktéž patří kolegům Luboši Rejfkovi a Pavlu Chmelařovi, se kterými jsem během studia úzce spolupracoval a kteří byli vždy nápomocni při řešení problémů.

ANOTACE

Tato práce se zabývá aktuálním tématem umělých neuronových sítí a jejich využití v odhadu hloubkové mapy na základě vstupního barevného snímku získaného z monokulární kamery. Práce je rozdělena do několika částí. V první části práce jsou popsány teoretické základy umělých neuronových sítí. Následující část se zabývá analýzou aktuálně používaných architektur neuronových sítí a výběrem vhodné struktury pro sémantickou segmentaci obrazových dat z kamerového snímače. V práci jsou taktéž rozebrány základní principy práce s mračnem bodů, jako je registrace, transformace apod. V experimentální části práce jsou popsány senzory robotické platformy jako je hloubková kamera či V-SLAM kamera, včetně postupu snímání a předzpracování množiny dat pro učení neuronové sítě. V závěru práce je proveden rozbor výsledků všech testovaných neuronových sítí, ze kterých byly na základě zvolených kritérií vybrány nejlepší, u kterých byla provedena detailní analýza.

KLÍČOVÁ SLOVA

Umělé neuronové sítě, UNS, odhad hloubkového snímku, zpracování obrazu neuronovými sítěmi, hloubková mapa, RGB-D, mračno bodů,

TITLE

The depth map estimation using neural networks

ANNOTATION

This thesis describes neural networks and implementation of these neural networks for depth scene estimation. Input for the neural network is image obtained from monocular camera. The first part of the thesis contains introduction to the neural networks. The second part describes appropriate architectures usable for image segmentation. The third part of this thesis describes methods usable for point cloud registration, transformation etc. Sensors fitted on robotics platform, like RGB-D or V-SLAM camera, are describes in experimental part of this thesis. Methods used for data pre-processing usable for neural networks are describes in this part of thesis too. The last chapter of the thesis describes methods used for analyze estimated depth images obtained from neural network. The selected neural networks were analyzed to obtain statistical parameters of tested networks.

KEYWORDS

Neural networks, depth scene, point cloud, image processing using neural networks, RGB-D

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM ZKRATEK A ZNAČEK

- BPG (*Backpropagation Gradient Descent Algorithm*) - Algoritmus zpětného šíření chyby používaný pro učení neuronových sítí
- CUDA – Architektura umožňující spustit na grafické kartě programy psané v jazyce C/C++
- DCM(*Direct Cosine Matrix*) – Matice směrových kosinů, sloužící pro korekci souřadnic, zejména u inerciálního navigačního systému
- DOF (*Degrees of Freedom*) – Stupně volnosti v prostoru
- GD (*Gradient Descent*) – Iterativní optimalizační algoritmus prvního řádu pro nalezení minima diferencovatelní funkce
- HFOV/VFOV (*Horizontal/Vertical Field Of View*) – Horizontální/vertikální zorné pole
- ICP/sICP (*Iterative Closest Point*) – Metoda registrace mračna bodů využívaná pro rekonstrukci 2D a 3D povrchů
- IDC (*Iterative Dual Correspondence*) – Metoda registrace mračna bodů využívaná pro rekonstrukci 2D a 3D povrchů
- LiDAR (*Light Detection and Ranging*) – Systém pro měření vzdálenosti pomocí výpočtu doby šíření pulsně modulovaného laserového paprsku od měřeného objektu
- MAE (*Mean Absolute Error*) – Střední absolutní chyba
- MSE (*Mean Square Error*) – Střední kvadratická chyba
- NDT/sNDT (*Normal Distribution Transform*) – Metoda registrace mračna bodů využívaná pro rekonstrukci 2D a 3D povrchů
- SLAM (*Simultaneous Localization And Mapping*) – Simultánní lokalizace a mapování prostoru, včetně přidávání a obnovy nové informace do již stávající mapy prostoru
- pIC – metoda registrace mračna bodů využívaná pro rekonstrukci 2D a 3D povrchů
- ROS (*Robot Operating System*) – operační systém zaměřený pro robotické aplikace
- RosBag – Formát pro ukládání proudu dat používaný zejména v operačním sytému ROS.
- SGD (*Stochastic Gradient Descent*) – Iterační metoda pro optimalizaci funkce
- UNS – Umělá neuronová síť

1 Úvod a cíl práce

V dnešní době dochází k masivnímu rozvoji aplikace umělých neuronových sítí ve všech oblastech lidského života. Lidstvo produkuje stále větší množství informací, které brzy nebude v lidských silách zpracovávat. Možným řešením je aplikace jednoúčelových umělých neuronových sítí, které jsou schopny zpracovávat mnohonásobně více dat než samotný člověk. Jako příklad lze uvést rozpoznávání hudby, kdy je na základě vyhodnocení určen hudební žánr, interpret a jiné. Není v silách jednotlivce či ani větší skupiny lidí si zapamatovat a s určitou pravděpodobností určit, o jaký se jedná žánr, či kdo je autorem dané skladby ze sady všech vytvořených skladeb na světě. Neuronové sítě s dostatečným výkonem jsou tyto úkoly schopny celkem spolehlivě plnit (Ghosal, a další, 2018).

Podobným způsobem je možné vybrat téměř každou lidskou činnost, kde již našly neuronové sítě uplatnění či ho v nejbližší době najdou. Dalším příkladem může být mnohem rozsáhlejší systém, který využívá více typů neuronových sítí, jako je aktuálně v provozu u společnosti Google se zaměřením rozpoznávání obrazu s názvem Google Lens. Tato aplikace široké škály umělých neuronových sítí slouží jak ke klasifikaci obsahu předloženého snímku, tak k vyhledávání podobných snímků či vyhledání informací o daném objektu. Tato aplikace je tak rozsáhlá a propojená, že stačí uživateli mobilního telefonu pouze namířit na hledaný objekt a neuronová síť předloží velké množství relevantních informací o tomto objektu (Nguyen, 2021).

Dalším ukázkovým příkladem je použití neuronových sítí u autonomních vozidel. Největšího pokroku dosáhla pravděpodobně společnost Tesla. Jejich autonomní vozidla jsou schopna řešit různé dopravní situace a dle okolí se přizpůsobit situaci. Vývoj autonomního vozidla ještě rozhodně není u konce, avšak do budoucna má rozhodně potenciál (Hart, 2020).

Technologie umělých neuronových sítí se začaly ve velkém aplikovat v posledních přibližně 10 letech a do budoucna je výrazně očekáváno jejich masivní rozšíření. Bohužel toto masivní rozšiřování umělých neuronových sítí, zejména do sekce bezpečnosti, obrany a autonomního řízení, přináší extrémní riziko zneužití. Již jsou ve vývoji autonomní zbraňové systémy, které pro vyhodnocení a cílený útok používají neuronové sítě. Autonomní bojová vozidla budou schopna vyhodnotit okolní situaci v řádu několika desítek milisekund a v závislosti na výsledku bude veden další postup útoku. Lidský

protivník, ač velice kvalitně trénovaný, ve stejném čase není ani schopen zareagovat. Aktuální úroveň těchto bojových vozidel je ve stádiu masivního rozvoje ve všech oblastech nezbytných k vedení boje (Fossaceca, a další, 2018) (Bistron, a další, 2021). Aktuálně se v problematice autonomních civilních i bojových vozidel mimo jiné řeší spolehlivé zpracování okolního prostředí za využití různých typů senzorů a kamer.

Cílem této disertační práce bylo provést studium vhodných architektur neuronových sítí pro odhad hloubkové mapy ze vstupního barevného snímku. Základním předpokladem této práce bylo použití monokulární barevné kamery pro získání vstupního snímku na pohybující se robotické platformě. Získaný barevný snímek byl předložen na vstup neuronové sítě, která byla schopna provést odhad výsledné hloubkové mapy bez jakýchkoliv jiných senzorů. Vytvořená hloubková mapa následně může být aplikována jako vstup do navigačního systému nazývaného *SLAM*, což v překladu znamená současná lokalizace a mapování robotické platformy. Výsledkem je možnost autonomního mapování a pohybu robotické platformy v neznámém prostředí a zejména v místech, kde není možné použít jiné přesné navigační systémy, jako je například satelitní navigace. V závěru práce bylo provedeno ověření kvality odhadu výsledné hloubkové mapy v závislosti na zvolené architektuře neuronové sítě, včetně vyhodnocení rychlosti trénování neuronové sítě a rychlosti zpracování výsledného snímku. Výstupem z práce je mračno bodů, vygenerované na základě vstupního barevného snímku, které bude možné použít v dalších metodách mapování a lokalizace (SLAM) robotické platformy.

Text práce je členěn do celkem 10 stěžejních kapitol. Kapitola 2 je rozdělena na dvě hlavní části. První část kapitoly popisuje historii neuronových sítí od prvních pokusů až po současnost. Druhá část kapitoly se zabývá základními stavebními prvky neuronových sítí.

Kapitola 3 detailně popisuje základní vrstvy používané v neuronových sítích se zaměřením zejména na vrstvy používané v konvolučních neuronových sítích. Kapitola taktéž obsahuje popis metod učení neuronových sítí, zejména s důrazem na pravděpodobně nejpoužívanější algoritmus zpětného šíření chyby. Následující část kapitoly se zabývá popisem optimalizačních algoritmů.

Kapitola 4 se zabývá detailním popisem modelů segmentačních konvolučních neuronových sítí. V kapitole je dále popsáno celkem 10 odlišných architektur neuronových sítí. Vybrané architektury byly v práci detailně zkoumány a implementovány pro použití v odhadu hloubkové mapy. Kapitola taktéž popisuje autorem navrženou novou modifikaci

neuronové sítě nazvanou *U-Net3,* která byla inspirována základním modelem neuronové sítě nazývané *U-Net*.

Výstupem práce je vygenerované mračno bodů a práce by nebyla celistvá bez kapitoly 5, která obsahuje základní popis práce s mračnem bodů. Kapitola popisuje postup podvzorkování mračna bodů, popis deskriptorů používaných v mračnu bodů či algoritmy registrace mračna bodů, jako je například *ICP*, *NDT* či *RANSAC*, které jsou používány při tvorbě mapy okolního prostoru. Správně zpracovaná a registrovaná mračna bodů jsou nezbytná pro systém lokalizace a mapování nazývaný SLAM.

Následující část práce, kapitola 6, popisuje platformu použitou pro snímání okolního prostředí. Platforma obsahuje více typů senzorů jako je hloubková kamera Intel RealSense D435i, použitá pro záznam barevných a hloubkových snímků, dále *V-SLAM* kamera Intel RealSense T265, která slouží pro záznam polohy pohyblivé platformy.

V kapitole 7 je rozebrán postup předzpracování dat získaných z hloubkové kamery, jako je zarovnání snímků z barevné a hloubkové kamery. Dále snímky z hloubkové kamery obsahují oblasti, kde není definovaná vzdálenost a je nezbytné tyto oblasti před použitím snímků v učení neuronových sítí doplnit. V neposlední řadě je nutné snímky z hloubkové kamery normalizovat.

Kapitola 8 je zaměřena na popis postupu trénování neuronových sítí. V kapitole je rozebrána tvorba datové sady, nastavené parametry samotného trénování a princip augmentace dat pro zvýšení robustnosti trénování neuronových sítí.

Rozbor úprav testovaných architektur neuronových sítí je popsán v kapitole 9. V kapitole je dále popsána autorem navržená architektura neuronové sítě *U-Net3.*

Kapitola 10 rozebírá hodnocení neuronových sítí na základě stanovených kritérií. U neuronových sítí, které byly vyhodnoceny jako nejlepší, bylo provedeno detailní statistické zpracování výsledků se zaměřením zejména na odchylku střední hodnoty a rozptyl v definované oblasti od hloubkové mapy získaný pomocí hloubkové kamery.

Poslední kapitola se dělí na dvě hlavní části. První část kapitoly se zabývá postupem vytvoření mračna bodů z hloubkové mapy. Druhá část kapitoly je zaměřena na rozbor kvality výsledků z vybraných neuronových sítí s ohledem na rozdíly odhadu vzdálenosti

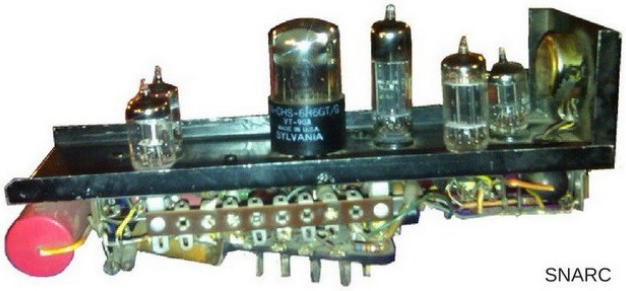
mezi vzdáleností změřenou hloubkovou kamerou a vzdáleností, která byla odhadnuta neuronovou sítí.

2 Neuronové sítě

Problematika umělých neuronových sítí (UNS) nachází velké uplatnění v širokém spektru vědních oborů, zejména však ve vědních oborech, kde je velice těžké matematicky popsat daný problém a bylo by extrémně náročné či nemožné jej řešit standardními metodami. Neuronová síť je schopna při vhodně zvolené struktuře sítě a jejímu dostatečnému naučení odhadovat výsledky s velkou přesností. Od dob vzniku vědního oboru prošly neuronové sítě radikálním rozvojem až do dnešní doby, kdy neuronové sítě jsou používané takřka v každém vědním oboru. Za masivním rozvojem stojí výrazné zvýšení výpočetního výkonu počítačů, zejména při přenosu výpočtů na výkonné grafické karty či při využití cloudových výpočtů.

2.1 Historie

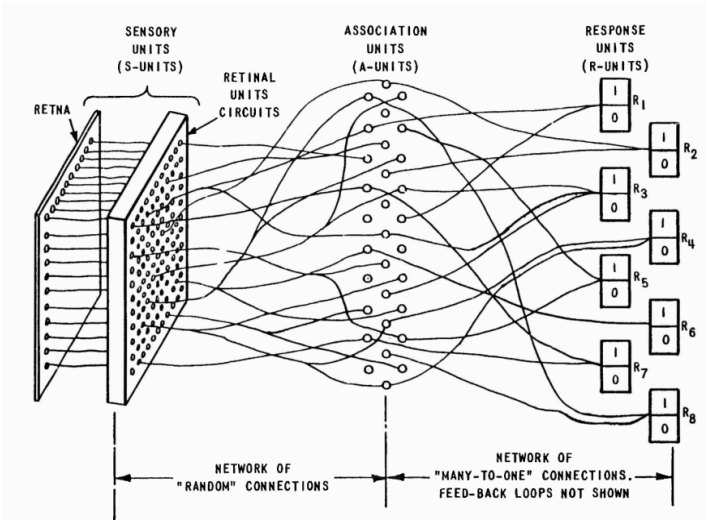
Počátky neuronových sítí sahají do roku 1943, kdy byl poprvé navržen matematický model neuronu Warrenem S. McCullochem a Waltrem Pittsem. V rámci jejich výzkumu byly převážně použity bipolární číselné hodnoty parametru, tj. hodnoty z množiny $\{-1,0,1\}$. Oba společně publikovali, že správně naučená neuronová síť je schopna vypočítat jakoukoliv spočitatelnou aritmetickou či logickou funkci. Přestože v rámci svého výzkumu ovšem nepočítali s bezprostředním praktickým využitím jejich modelu, měl jejich výzkum významný vliv na další badatele. V roce 1949 byla Donaldem Hebbem vydána kniha „*Organization of Behavior*“ (Hebb, 1949), ve které bylo navrženo učící pravidlo pro mezi neuronové rozhraní, které bylo inspirováno myšlenkou, že podmíněné reflexy, pozorovatelné u všech živočichů, jsou funkčními vlastnostmi jednotlivých neuronů. Výzkumy Mccullocha, Pittse a Hebba ovlivnily ostatní vědce, avšak technika nebyla tak rozsáhlá, aby ve 40. a 50 letech přinesla zásadní rozvoj v oblasti neuronových sítí. Jedním z příkladů, který lze zmínit, bylo sestrojení prvního neuropočítače *Snarc (Stochastic Neural Analog Reinforcement Calculator)* v roce 1951, u jehož zrodu stál Marvin Lee Minsky. Neuropočítač *Snarc* byl sice z technického pohledu celkem úspěšný, dokonce byl schopen automaticky adaptovat váhy vstupů u jednotlivých



Obrázek 1: Ukázka jednoho ze 40 neuronů prvního neuropočítače SNARC Zdroj: (SNA21)

neuronů, ovšem nikdy nebyl využit k řešení praktického problému. Jeho architektura nicméně byla využita k řešení dalších konstrukcí neuropočítačů. Zajímavostí je, že celý neuropočítač byl tvořen z celkem 3000 elektronek a celkově obsahoval 40 neuronů. Ukázka jednoho z neuronů je zobrazena na Obrázek 1.

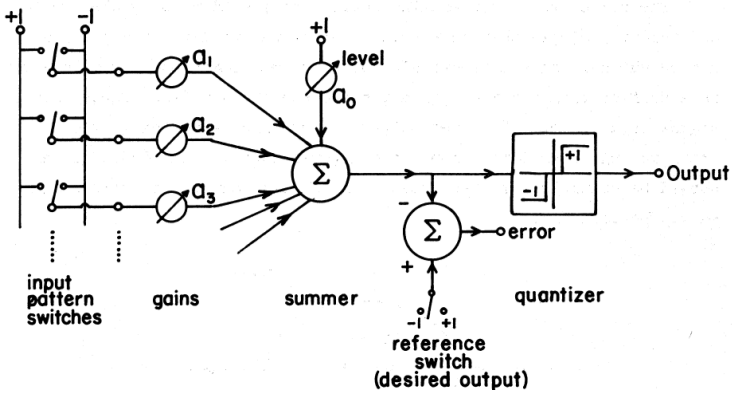
O několik let později, v roce 1957, Frank Rosenblatt vyvinul tzv. p*erceptron*, který byl zobecněním McCullovova-Pittsova modelu neuronu pro obor reálných parametrů. Taktéž pro tento model navrhl učící algoritmus a matematicky prokázal, že pro daná tréninková data je možné v konečném počtu iterací nalézt odpovídající vektor vah nezávisle na počátečním nastavení vektoru vah. Rosenblatt ze svého výzkumu vytvořil publikaci o neurovýpočtech nazvanou „*Principles of Neurodynamics"* a v průběhu roku 1958 společně s Charlesem Wightmannem sestrojili první úspěšný neuropočítač, který nazvali *Mark I Perceptron.* Neuropočítač byl navržen pro rozpoznání znaků. Celý princip spočíval v tom, že znak byl promítán na světelnou tabuli, ze které byl snímán za využití 400 fotosenzorů, organizovaných do matic. Intenzita jednotlivých obrazových bodů byla použita jako vstup do sítě perceptronů, která měla klasifikovat o jaký promítaný znak se jedná. Ukázka celého zařízení je uvedena na Obrázek 3.



Obrázek 2: Ukázka zapojení Mark I Perceptron (Rossenblatt, a další, 1960)

Obrázek 3: Ukázka MARK I Perceptron (Nat21)

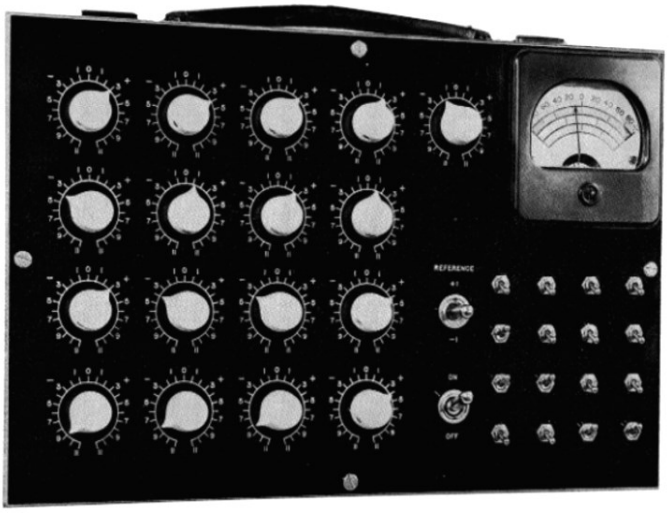
V navazujících výzkumech Bernard Widrow se svými studenty navrhnul další typ neuronového výpočetního prvku nazvaný ADALINE, což je zkratka *ADaptive LInear NEuron*. Navržený neuron je zobrazen na Obrázek 4 a celkový vzhled vytvořeného zařízení je zobrazen na Obrázek 5. Tento ADALINE neuron obsahuje výkonné učící pravidlo, které se dodnes nezměnilo (Widrow, 1960). Výsledná odezva zařízení byla nastavována manuálně sepnutím hodnoty *1* nebo *-1* do jednotlivých synapsí neuronu. Dále se proměnným rezistorem nastavoval zisk jednotlivých synapsí, resp. váhy vstupů. Výstup neuronu byl porovnáván s požadovanou hodnotou.



Obrázek 4: Návrh ADALINE neuronu v reálném zařízení (Widrow, 1960)

Výsledky vývoje z 50. a 60. lech shrnul Nils Nilsson ve své knize *Learning Machines*, vydané v roce 1965 (Nilsson, 1965). Období 50. a 60. let znamenalo ovšem velký rozmach vývoje neuronových sítí, ovšem taktéž se potýkal s několika velkými problémy.

Prvním problémem byl fakt, že většina výzkumníků přistupovala k neuronovým sítím jen za využití experimentů, avšak opomíjela analytický přístup. Dalším problémem byla velká horlivost některých výzkumníků, kteří hlásili, že během několika let bude vytvořen umělý mozek. Tyto skutečnosti spoustu výzkumníků naopak odradily od dalšího zkoumání neurovýpočtů. Posledním problémem byla kampaň proti neuronovým sítím vedená Marvinem Minskym a Seymourem Papertem, kteří se snažili zdiskreditovat výzkum neuronových sítí za cílem převést finance investované do výzkumu neuronových sítí na jiné zaměření. V průběhu své kampaně vytvořili publikaci *Perceptrons*, kterou vydali v roce 1969 (Minsky, a další, 1969). V publikaci autoři popisují známý problém XOR, tj. důkaz, že jeden perceptron není schopen vypočítat jednoduchou logickou funkci, tzv. logickou disjunkci (XOR). Funkci lze sestavit za využití dvou vrstev se třemi perceptrony, ovšem v době psaní publikace nebyl znám učící algoritmus pro vícevrstvé perceptrony. Autoři vyvodili, že není možné vzhledem ke komplikovanosti funkce, algoritmus učení vytvořit. Bohužel autoři byli úspěšní a výzkum neuronových sítí přestal být dotován, jelikož byl považován za neperspektivní. Výzkum neuronových sítí sice dále probíhal, ale veskrze izolovaně a převážně mimo území USA, kde kniha *Perceptrons* neměla takový vliv.

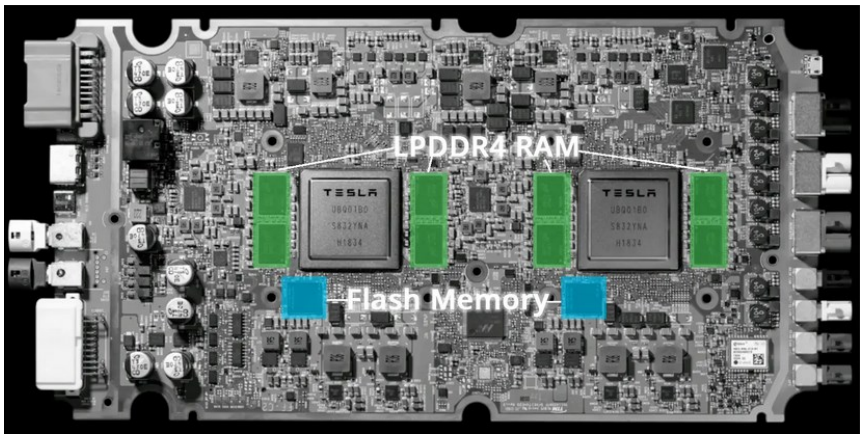


Obrázek 5: Zařízení využívající ADALINE s manuálním nastavováním vah synapsí (Widrow, 1960)

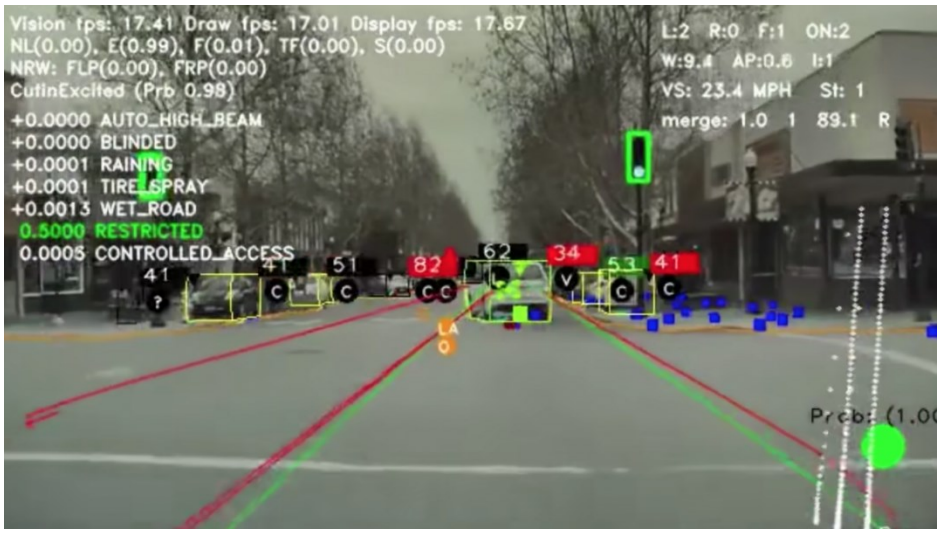
Zásadní rozvoj nastal až v 80. letech, kdy se výzkumníci znovu začali neurovýpočty zabývat a znovu začali podávat žádosti o granty. V roce 1983 agentura DARPA (*Defense Advanced Research Projects Agency*) začala podporovat výzkum neuronových sítí, později se přidalo více společností. Jedním z nejvýznamnějších vlivů měl světově uznávaný fyzik John Hopfield, který vytvořil publikace (Hopfield, 1982) a (Hopfield, 1984) ve kterých popisuje souvislosti fyzikálních modelů magnetických materiálů s modely neuronových sítí. Svými přednáškami upoutal pozornost stovek vědců po celém světě. V roce 1986 badatelé ze skupiny *Parallel Distributed Processing Group* publikovali algoritmus

zpětného šíření chyby (*BackPropagation algorithm*) v publikaci *Learning Representations by back-propagating errors* (Rumelhart, a další, 1986). Tento algoritmus je nejpoužívanějším algoritmem učení neuronových sítí dodnes. V souvislosti s vydanou publikací začal obrovský rozvoj v oblasti neuronových sítí, vznikla široká škála odborných časopisů, například *Neural Computing* či *Neural Network World* a taktéž vznikla spousta studijních oborů se zaměřením a neurovýpočty.

V dnešní době dochází k enormnímu rozvoji neuronových sítí ve všech oblastech lidské činnosti. Jednou z oblastí, kterou je vhodné aktuálně zmínit, je oblast navigace a autopilota automobilu společnosti Tesla. Lze konstatovat, že společnost Tesla dosáhla pravděpodobně největšího pokroku v oblasti zpracování obrazu a autonomního řízení. Jejich autopilot obsahuje 48 neuronových sítí, které byly učeny přes 70 000 GPU hodin. Těchto 48 neuronových sítí zpracovává surová data z kamer za účelem vytvoření segmentační mapy (mapy, chodník, krajnice, statická infrastruktura apod.), detekce statických i pohyblivých objektů (ostatní vozidla, chodci, budovy, značky apod.) a odhadu monokulární hloubky. Celý autopilot se učí na snímaných datech z flotily čítající téměř 1 milionu vozidel. Pro takto veliké množství dat bylo nutné vytvořit obří výpočetní centrum s celkem 5760 grafickými kartami Nvidia A100 s výpočetním výkonem až 80 PFLOPS (dat21). Navržené sítě vzájemně spolupracují a vytváří celkově 1000 predikcí v každý časový interval (Art21), (Shantanu, a další, 2016). Na Obrázek 6 je zobrazena výpočetní jednotka autopilota vozu Tesla. Dle specifikací je zařízení schopno zpracovat datový tok ekvivalentní 1260 snímkům ve full HD kvalitě za vteřinu. Ukázka výstupu z autopilota je zobrazena na Obrázek 7.



Obrázek 6: Ukázka výpočetní jednotky autopilota ve voze Tesla s procesorem HW3 (Cle21)

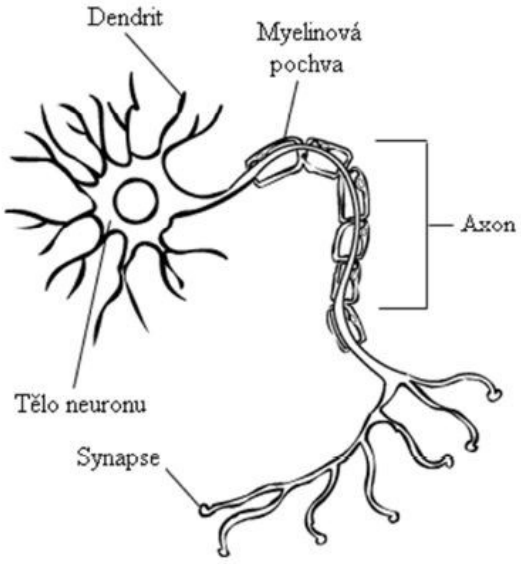


Obrázek 7: Ukázka detekce silnice, vozidel a chodců z autopilota vozidla Tesla (Hart, 2020)

Vývoj neuronových sítí trval přibližně půl století, avšak největší boom zažil v posledních přibližně 15 letech s masivním rozvojem počítačové techniky. Aktuální trend je další masivní rozvoj neuronových sítí ve všech oblastech a bude zajímavé sledovat, jaký pokrok přinese dalších 15 let vývoje.

2.2 Biologický neuron

Neuronové sítě nacházení inspiraci v biologických strukturách neuronů v živých organismech. Biologický neuron (Obrázek 8) se skládá z těla neuronu, nazývaného soma, které je velké několik mikrometrů. Z těla neuronu vybíhají tisíce dendritů, které tvoří vstup neuronu. Z těla neuronu vychází pouze jedno vlákno, nazývané axon. Axon slouží jako výstup z neuronu a jeho konec se rozděluje do mnoha výběžků nazývaných synapse. Synapse jsou složité biologické útvary zajišťující spojení neuronu s dalšími neurony, čímž se podílejí na procesu učení. Pro představu, lidský mozek je tvořen přibližně 10^{11} neurony, přičemž každý axon je propojen s dalšími desítkami tisíc dalších neuronů. Mozek v průměru u muže váží 1350 gramů, u žen přibližně 1200 gramů. Mozek je vzhledem ke své váze extrémním spotřebitelem kyslíku, ačkoliv



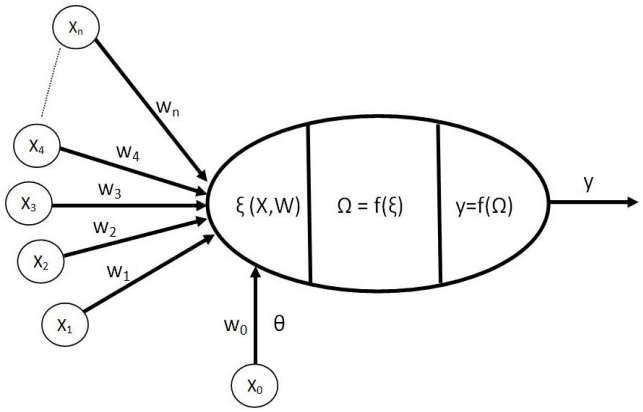
Obrázek 8: Biologický neuron (Doležel, 2011)

hmotnost mozku tvoří přibližně 2% z celkové hmotnosti člověka, tak mozek spotřebuje přibližně čtvrtinu vdechovaného kyslíku.

Excitace neuronu probíhá na základě elektrochemických reakcí způsobených přečerpávání kladných iontů na vnější stranu axonu a záporných iontů na vnitřní stranu axonu. Povrch axonu je pokryt vodivou membránou, která je schopna při zvýšení elektrického potenciálu uvnitř neuronu nad prahovou úroveň prudce depolarizovat, což vede k vytvoření elektrického impulsu, který se šíří po axonu do dalších neuronů. Příchozí impuls z axonu je přiveden na dendrit následujícího neuronu, přičemž dojde k uvolnění molekuly chemické látky (mediátorů či transmiterů), které vyvolají další potenciálovou vlnu. Tato vlna je následně v synapsích dále excitována (zesílena) či inhibována (utlumena) a následně je přivedena do samotného neuronu.

2.3 Umělý neuron

Umělý neuron je inspirován biologickým neuronem a prakticky kopíruje jeho základní části. Struktura umělého neuronu je zobrazena na Obrázek 9. Struktura neuronu obsahuje definované množství vstupů, z nichž se ve většině literatury vstup označuje vektorem ***X***, váhy jednotlivých dendritů jsou označeny vektorem vah ***W***, výstup z neuronu je označen proměnnou ***y***. Ze struktury modelu neuronu je patrné, že vážené vstupy $x_0, x_1 \dots x_n$, souhrnně označené vektorem ***X,*** a jejich příslušné váhy $w_0, w_1 \dots w_n$jsou transformovány pomocí agregační funkce $\xi(X, W)$ na jedinou skalární hodnotu y_ξ, která je za využití aktivační funkce Ω a výstupní funkce na hodnotu výstupu y z neuronu.



Obrázek 9: McCullochův-Pittsův model neuronu (Zdroj: Vlastní)

2.3.1.1 Vstupy neuronu

Vstupní vektor X může představovat kvantitativní či kvalitativní formu. Kvalitativně vyjádřená forma obvykle představuje booleovské hodnoty ve smyslu Ano/Ne. Každý vstup představuje vlastnost, která v systému je či není obsažena, například rychlost letadla překročila povolenou hranici. Kvantitativní forma vektoru reprezentuje hodnotu,

například reálné číslo. Tato forma představuje skutečnou hodnotu měřené veličiny, jako je například rychlost letadla, směr větru apod.

Vstupy neuronu jsou pro jednoduchost zapisovány ve formě vektoru (2.1).

$$X = [x_1, x_2, \dots, x_N] \tag{2.1}$$

Každý vstup $\boldsymbol{X}_i$ do neuronu modelující dendrity neuronu je normován na hodnotu v intervalu ‹0,1› pro unipolární vstup, případně ‹-1,1› pro bipolární vstup, přičemž vstup X_0, občas označován θ, je použit jako prahová hodnota neuronu. Hodnota jednotlivých vstupů je násobena váhou jednotlivého vstupu $w_{i..}$

2.3.2 Váhy neuronu

Váhy neuronu jsou úzce spjaty s propojením neuronů v neuronové síti. Každý vstup do neuronu je ohodnocen vlastní váhou, kterou lze interpretovat jako citlivost neuronu na konkrétní vstup. Tyto váhy simulují excitační či inhibiční funkci synapsí. Váhy neuronu jsou obvykle reprezentovány reálným číslem. Matematická závislost mezi váhou neuronu a vstupem neuronu je nazývána konfluencí vstupního signálu s jeho vahou (2.2).

$$z_i = x_i(k) \oplus w_i(k) \tag{2.2}$$

V případě základního modelu neuronu lze operaci konfluence nahradit obyčejným součinem, z čehož vychází rovnice (2.3).

$$z_i = x_i(k) \cdot w_i(k) \tag{2.3}$$

2.3.3 Práh neuronu

Práh θ v biologickém neuronu lze brát jako bariéru, kterou musí signál překonat, aby mohl signál dále postupovat soustavou. Hodnotu prahu lze tedy brát jako míru, do které je neuron aktivní či neaktivní. Je-li součet vektorů násobených příslušnými vahami menší než hodnota prahu, je neuron neaktivní a jeho výstupní hodnota odpovídá pasivnímu stavu neuronu. Při překročení prahové hodnoty dojde k aktivaci neuronu a výstupní hodnota z agregační funkce ξ je aplikována na aktivační funkci Ω. Obvykle se jako práh neuronu používá první vstup do neuronu $x_0 = 1$, který je vážen vstupní váhou w_0.

2.3.3.1 Agregační funkce

Agregační funkce slouží k transformaci vstupního vektoru $\boldsymbol{X}$ na skalární signál ξ(k), který je aplikován na vstup aktivační funkce Ω neuronu. Agregační funkce je popsána rovnicí (2.4)

$$\xi(k) = G_{i=1}^{n} z_i(k) \quad (2.4)$$

Obecný postup agregace lze v zjednodušeném modelu neuronu nahradit obyčejným součtem, z čehož vychází rovnice (2.5)

$$\xi(k) = \sum_{i=1}^{n} x_i(k) \cdot w_i(k) + \theta \quad (2.5)$$

Prahovou hodnotu θ lze považovat za další vstup se vstupní hodnotou 1, respektive -1, z čehož plyne, že je možné prahovou hodnotu θ nahradit vstupem x_0 násobený váhou w_0, jak je uvedeno v rovnici (2.6), z čehož vychází výsledná podoba agregační funkce ξ (2.7).

$$\theta = x_0 \cdot w_0 \quad (2.6)$$

$$\xi(k) = \sum_{i=1}^{n} x_i(k) \cdot w_i(k) + x_0 \cdot w_0 = \sum_{i=0}^{n} x_i(k) \cdot w_i(k) \quad (2.7)$$

2.3.4 Aktivační funkce

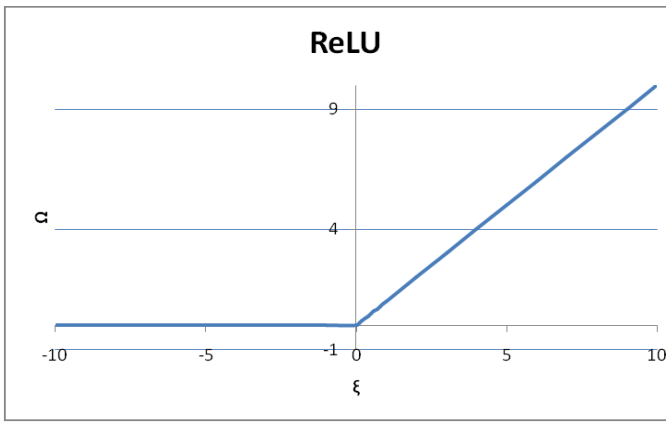
Úkolem aktivační funkce Ω je transformace výstupu z agregační funkce na výstupní hodnotu neuronu. Konkrétní tvar přenosové funkce může být jakýkoliv a existuje velké množství aktivačních funkcí. Důvodem pro zavedení aktivační funkce je nutnost transformace vnitřního potenciálu generovaného agregační funkcí k vygenerování výstupu. Obecně existuje velké množství hladkých i ostrých variant aktivační funkce – např. ostrý hyberbolický tangens vs. hyperbolický tangens. Obecně se dá říct, že ostré funkce při porovnání s hladkou verzí jsou výrazně rychlejší na výpočet ale za cenu menší přesnosti, především ale nemají v některých bodech derivaci, což může způsobovat problémy (Lisa Lab, 2015).

Podmínky pro aktivační funkce:

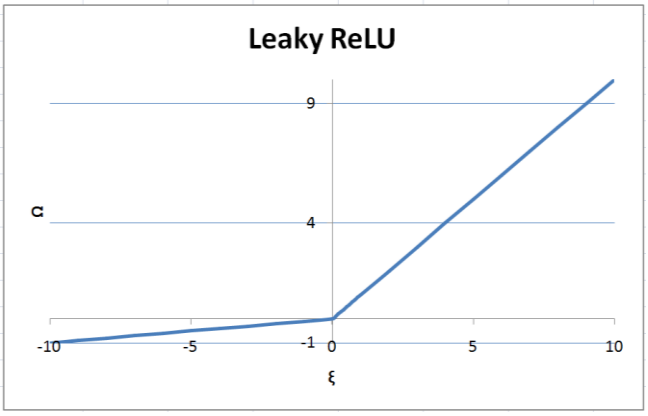
- **Nelinearita** – Když by aktivační funkce byla lineární, na výstupu neuronu by byla opět lineární funkce vstupů. Z vlastností složených lineárních funkcí vyplývá, že pokud jsou všechny funkce lineární, výsledek bude taktéž lineární. U neuronových sítí je ovšem nezbytné, aby po transformaci byl výstup lineárně separovatelný, čehož je možné dosáhnout pouze použitím nelineárních funkcí.
- **Diferencovatelnost** – tato vlastnost vychází z Backpropagation algoritmu. Pokud aktivační funkce nebude diferencovatelná, nelze backpropagation algoritmus použít.

2.3.4.1 ReLU

Aktivační funkce ReLu (Rectified Linear Unit) transformuje záporné hodnoty z agregační funkce ξ na hodnotu 0. Velkou výhodou této aktivační funkce je jednoduchá derivace a velmi malá výpočetní náročnost. Funkce není shora ohraničená a díky tomu je síť schopna odhalit míru chyby. Průběh aktivační funkce je dle rovnice (2.7). Problémem u tohoto typu aktivační funkce je tzv. (Dying ReLU problem) – v případě, že vstupní data jsou záporná či jejich hodnota je blízká nule, gradient funkce se blíží k nulové hodnotě, což způsobí, že není možné provést zpětné šíření chyby (metoda BackPropagation) a síť není možné naučit. Problém s "*Dying RELU problem*" řeší modifikovaná verze *ReLU* funkce, např. "*Leaky ReLU*". Leaky ReLU funkce na rozdíl od ReLU má v záporné části grafu koeficient α, který mění směrnici přímky. Kladná část *Leaky ReLU* je shodná s *ReLU*.



Obrázek 10: Průběh aktivační funkce ReLU (Zdroj: Vlastní)



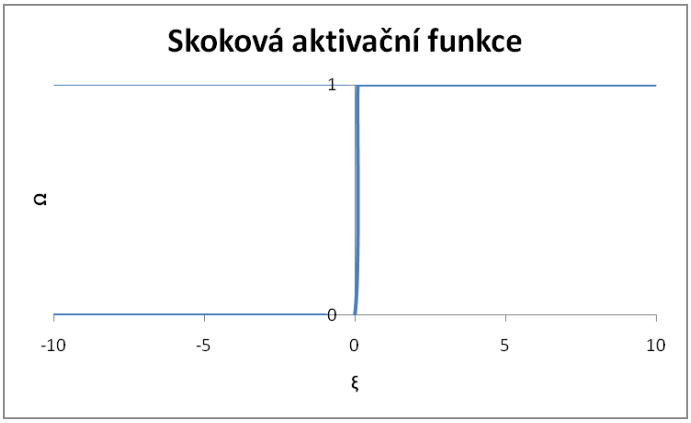
Obrázek 11: Průběh aktivační funkce Leaky ReLU (Zdroj: Vlastní)

ReLU:
$$\Omega = f(\xi) = \begin{cases} 0 \text{ pro } \xi \leq 0 \\ \Omega = \xi \text{ pro } \xi > 0 \end{cases} \quad (2.7)$$

Leaky ReLU:
$$\Omega = f(\xi) = \begin{cases} -\alpha \cdot \xi \text{ pro } \xi \leq 0 \\ \Omega = \xi \text{ pro } \xi > 0 \end{cases} \quad (2.8)$$

2.3.4.2 Skoková funkce

Skoková funkce, občas nazývaná binární aktivační funkce je nejjednodušší aktivační funkce. Neuron je buď aktivní či není. Derivace skokové funkce je rovna jedné, kromě bodu $x = 0$, kde není derivace definovaná, což extrémně zjednodušuje výpočet, ovšem je vhodná pouze do binárních systémů. Průběh aktivační funkce je popsán rovnicí (2.9)

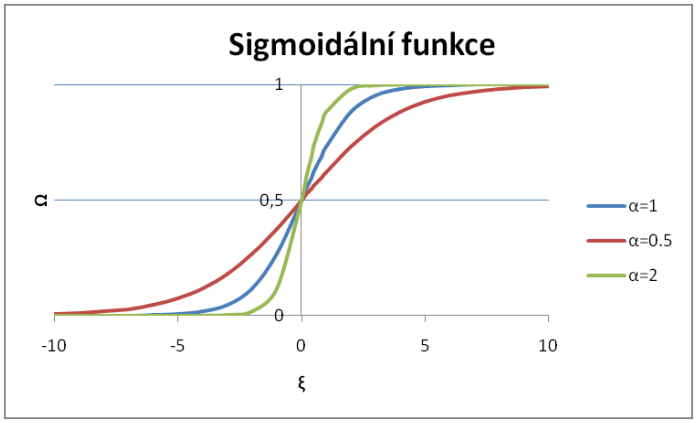


Obrázek 12: Průběh skokové aktivační funkce (Zdroj: Vlastní)

$$\Omega = f(\xi) = \begin{cases} 0 \text{ pro } \xi \leq 0 \\ 1 \text{ pro } \xi > 1 \end{cases} \qquad (2.9)$$

2.3.4.3 Sigmoidální funkce

Sigmoidální aktivační funkce nabývá hodnot v intervalu ‹0,1›. Průběh funkce je dán rovnicí (2.10). K sigmoidě se občas přidává i parametr α, který určuje míru zakřivení sigmoidy. Sigmoida je lichá funkce, která díky svému tvaru velmi dobře modeluje výstup z neuronu. Nevýhodou sigmoidální funkce je lineární oblast v okolí nulového bodu což může zpomalovat učení neuronové sítě a taktéž může způsobit maskování gradientu v hlubších sítích (Le Cunn, a další, 1998).

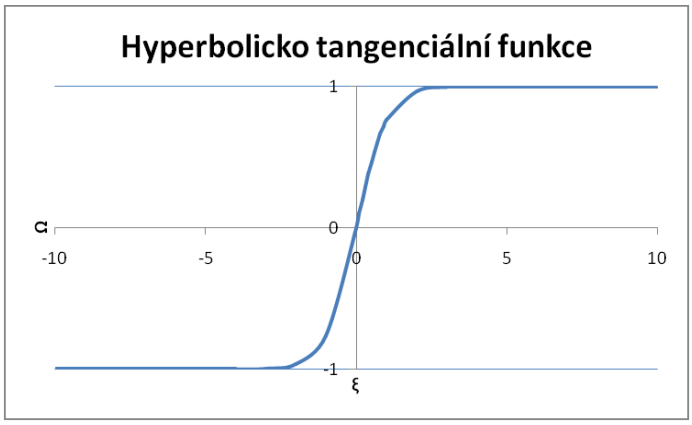


Obrázek 13: Průběh sigmoidální funkce (Zdroj: Vlastní)

$$\Omega = f(\xi) = \frac{1}{1+e^{-\alpha\xi}} \qquad (2.10)$$

2.3.4.4 Hyperbolicko-tangenciální funkce

Hyperbolický tangens má podobné parametry jako sigmoidální funkce. U většiny klasifikačních úloh síť s hyperbolicko-tangenciální funkcí konverguje rychleji než sigmoida (Özkan, a další, 2003). Problémem u hyperbolicko-tangenciální funkce je saturace v případě větší odchylky, než je lineární oblast. Při učení neuronové sítě může dojít k zamaskování gradientu či neuronová síť může stagnovat v lokálním minimu. Další výhodou je centrování středu aktivační funkce (Zero centered), což umožní snadnější modelování vstupu u dat, které mají vstupní hodnoty hodně kladné či záporné.

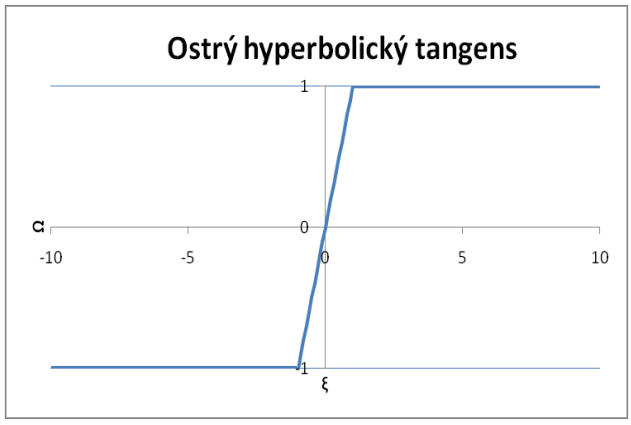


Obrázek 14: Průběh Hyperbolicko tangenciální funkce (Zdroj: Vlastní)

$$\Omega = f(\xi) = \tanh(\xi) \qquad (2.11)$$

2.3.4.5 Ostrý hyperbolický tangens

Ostrý hyperbolický tangens je dán rovnicí (2.12). Ostrý hyperbolický tangens je občas nazýván saturační funkcí. Funkce je založena na hyperbolickém tangentu s tím rozdílem, že není vyhlazená. Aktivační funkce je oproti hyperbolickému tangentu méně výpočetně náročná, ale i přes to dosahuje srovnatelných výsledků jako s hladkou verzí. Další výhodou je centrování středu aktivační funkce (*Zero-centered*), což umožní snadnější modelování vstupu u dat, které mají vstupní hodnoty hodně kladné či záporné.



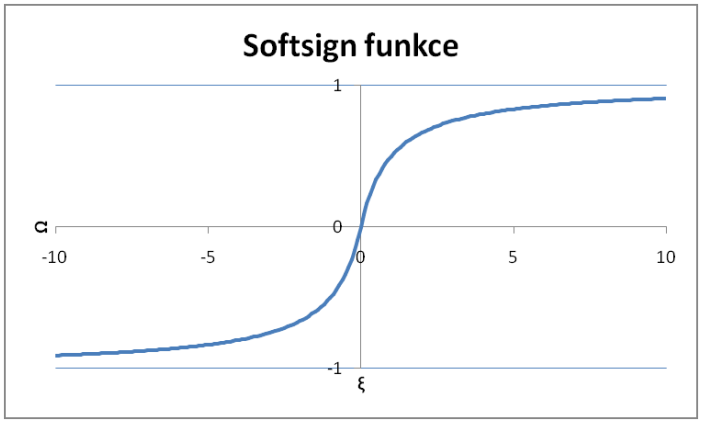
Obrázek 15: Ostrý hyperbolický tangens (Zdroj: Vlastní)

$$\Omega = f(\xi) = \begin{cases} -1 \text{ pro } \xi \leq -1 \\ \Omega = \xi \text{ pro } \xi \in (-1,1) \\ 1 \text{ pro } \xi \geq -1 \end{cases} \qquad (2.12)$$

2.3.4.6 Softsign funkce

Softsign funkce je podobná hyperbolickému tangentu, ale má polynomiální asymptoty, ovšem je robustnější, co se týče inicializace vah (Glorot, a další, 2010). Softsign funkce je dána rovnicí (2.13).

$$\Omega = f(\xi) = \frac{\xi}{1+|\xi|} \tag{2.13}$$

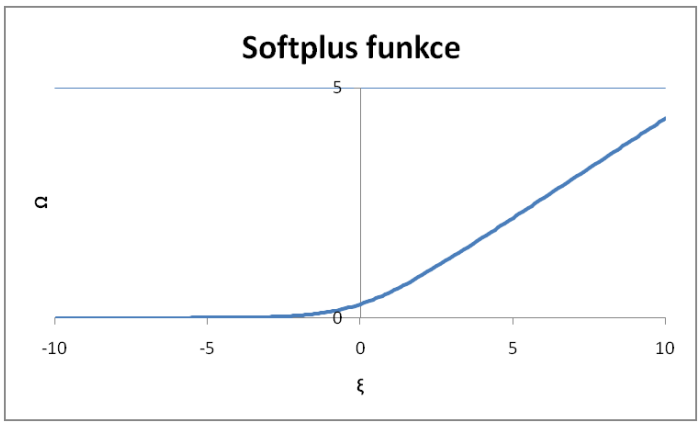


Obrázek 16: Softsign funkce (Zdroj: Vlastní)

2.3.4.7 Softplus funkce

Softplus funkce je ve skutečnosti vyhlazená ReLu funkci, díky čemuž je bez problémů diferencovatelná. Softplus funkce je dána rovnicí (2.14).

$$\Omega = f(\xi) = \log(1 + e^{\xi}) \tag{2.14}$$



Obrázek 17: Softplus funkce (Zdroj: Vlastní)

2.3.5 Výstupní funkce

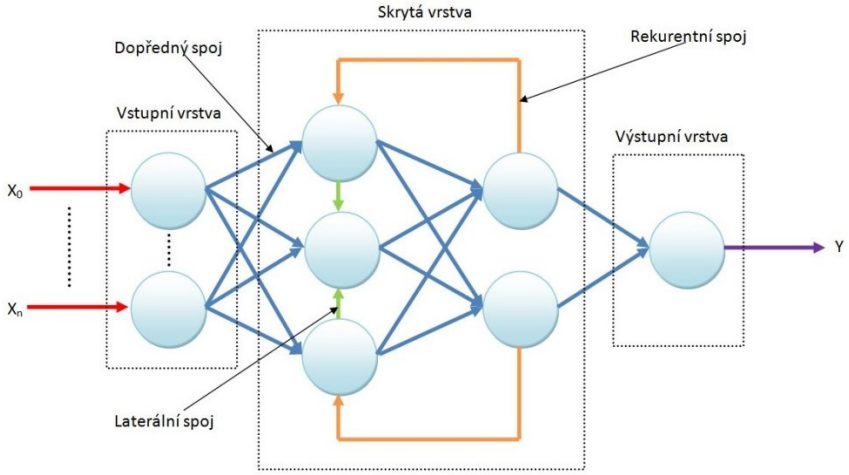
Výstupní funkce $y = f(\Omega)$ neuronu slouží k dotvoření konečné hodnoty výstupu z neuronu. Existují celkem dva typy výstupní funkce. Identická výstupní funkce kopíruje výstupní hodnotu z aktivační funkce. V případě neidentické výstupní funkce je mezi aktivační a výstupní funkcí nejčastěji použita lineární funkční závislost.

3 Umělá neuronová síť

V předchozí části práce byla popsána struktura obecného neuronu včetně matematických vztahů mezi vstupy neuronu a transformovaným výstupem. Vzájemným propojením těchto neuronů, dojde k vytvoření umělé neuronové sítě. Neuronová síť díky své struktuře umožňuje paralelní a distribuované zpracování dat. Díky schopnosti učit se, představují adaptivní systémy a jsou schopny učit se i v případě, když člověk nezná či není schopen stanovit skutečné matematické závislosti.

Topologie neuronové sítě, občas označovaná architektura neuronové sítě popisuje propojení jednotlivých neuronů. Shluky těchto vzájemně propojených neuronů jsou označovány jako vrstvy. Podle umístění vrstev je možné tyto vrstvy rozdělit do skupin:

1. *Vstupní vrstva* – vrstva zajišťující vstup signálu z vnější soustavy do neuronové sítě
2. *Vnitřní (skrytá) vrstva* – množství vnitřních, občas nazývaných skrytých vrstev může být principiálně více
3. *Výstupní vrstva* – vrstva zajišťující předání signálu z neuronové sítě do vnější soustavy



Obrázek 18: Vrstevnatá neuronová síť (Zdroj: Vlastní)

Architektura dané neuronové sítě definuje množství neuronů v jednotlivých vrstvách neuronové sítě včetně propojení těchto vrstev. V rámci architektury sítě se používají následující typy propojení:

1. *Laterální spojení* – spojení přímo mezi neurony stejné vrstvy
2. *Dopředné spojení* – spojení mezi neurony sousedních vrstev

3. *Rekurentní spojení* – speciálním případem spojení, který vede signál proti dopřednému směru toku signálu uvnitř neuronové sítě. V případě, že neuronová síť obsahuje rekurentní spojení neuronů, tak jsou tyto druhy neuronových sítí nazývány jako rekurentní sítě

3.1 Základní vrstvy umělé neuronové sítě

V problematice architektury umělých neuronových sítí existuje velké množství různých vrstev, které jsou typické pro danou aplikaci. Tato kapitola popisuje nejčastěji používané vrstvy v oblasti sémantické segmentace.

3.1.1 Plně propojená vrstva (Dense)

Plně propojená vrstva, občas nazývaná anglickým názvem *Dense* je vrstva, která vytváří spojení mezi každým vstupním a každým výstupním prvkem. Toto spojení zajišťuje distribuci informace na výstup ze vstupu. Výhodou je, že je možné na vstupu do vrstvy mít jiné rozměry matice než na výstupu. Nevýhodou této vrstvy je, že na vstupu očekává vektor, takže je nezbytné vstupní matici převést na vektor. Další nevýhodou je velký počet parametrů, který je pro vstupní matici, resp. vstupní vektor o délce x a výstupní vektor o délce y. Výsledný počet parametrů lze vypočítat dle rovnice (3.1)

$$Param = \mathrm{y} \cdot (\mathrm{x} + 1) \tag{3.1}$$

Kde:

- x je délka vstupního vektoru
- y je délka výstupního vektoru
- $Param$ je počet parametrů nezbytných pro realizaci vrstvy

3.1.2 Serializační vrstva (Flaten)

Serializační vrstva, občas nazývaná *Flatten* vrstva, se používá pro vytvoření vektoru z N-dimenzionálního pole. Vrstva se využívá zejména ve spojení s plně propojenou vrstvou, která na svém vstupu potřebuje právě vektor. Pro příklad, pokud máme na vstupu do neuronové sítě snímek o rozměru $H \times W \times C$, kde H je výška, W je šířka a C je barevná hloubka obrazu (C=1 pro černobílý snímek, C=3 pro RGB), výsledný vzniklý vektor bude mít délku $H \cdot W \cdot C$, tj. V případě snímku o rozměrech $128 \times 128 \times 3$ vznikne vektor o rozměru 49152×1.

3.1.3 Normalizační vrstva (Batch Normalization)

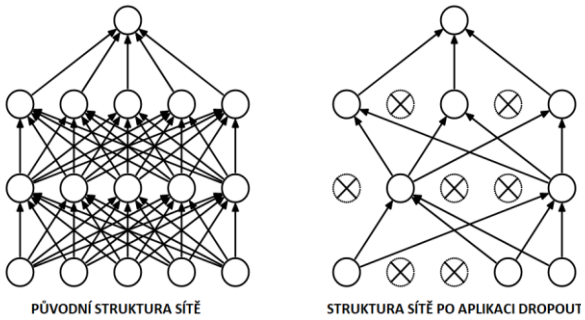
Normalizační vrstva, v anglické literatuře označovaná jako Batch Normalization, se používá pro urychlení tréninku a zároveň snížení citlivosti sítě na počáteční podmínky, zejména při použití mezi nelineárními vrstvami typu *ReLU*. Vrstva využívá tzv. *Mini Batch*, kde dochází k pře mapování jednotlivých vstupních vah s určitou střední hodnotou a směrodatnou odchylkou na hodnoty výstupu s $N(\mu = 0, \sigma = 1)$, což způsobí jednodušší a rychlejší učení. Důvodem je fakt, že během učení, je možné, aby některý neuron měl silnou odezvu, která by zakryla slabší odezvy ostatních neuronů. Nevýhodou normalizace je ovšem určitá závislost mezi jednotlivými daty.

$$
\begin{aligned}
&Vstup: \ \beta = \{x_{1\ldots m}\}, Parametry\ k\ určení: \mu_\beta, \sigma_\beta \\
&Výstup: \left\{y_i = \ BN_{\mu_\beta, \sigma_\beta}(x_i)\right\} \\
&KROK\ 1: \mu_{Batch} = \frac{1}{m}\sum_{i=1}^{m} x_i \\
&KROK\ 2: \sigma^2_{Batch} = \frac{1}{m}\sum_{i=1}^{m}(x_i - \ \mu_{Batch})^2 \\
&KROK\ 3: \widehat{x_i} = \frac{x_i - \mu_{Batch}}{\sqrt{\sigma^2_{Batch} + \epsilon}} \\
&KROK\ 4: y_i \ \equiv BN_{u_{Batch}, \sigma_{Batch}}(x_i)
\end{aligned}
\tag{3.2}
$$

Algoritmus normalizace funguje tak, že nejdříve jsou ze vstupních dat (i jednotlivých kanálů) vypočítány základní statistické parametry μ, σ^2 vstupního datového souboru (Krok 1 a 2). Posléze dochází k aplikaci těchto vypočtených statistických parametrů na vstupní data, čímž dojde k přepočtení vstupních parametrů, a tudíž k normalizaci (Krok 3). Posledním krokem je škálování a posun na základě vypočtených hodnot (Krok 4).

3.1.4 Výběrová vrstva (Drop-Out)

Výběrová vrstva, v anglické literatuře označovaná vrstva *DropOut*, je jednou z velice používaných technik regularizace v neuronových sítích. Výběrová vrstva je přesným opakem plně propojené vrstvy (*Dense)*. Metoda vychází z myšlenky, náhodné deaktivace neuronů, kdy jejich výstupní hodnota je nastavena na hodnotu nula v průběhu učení.



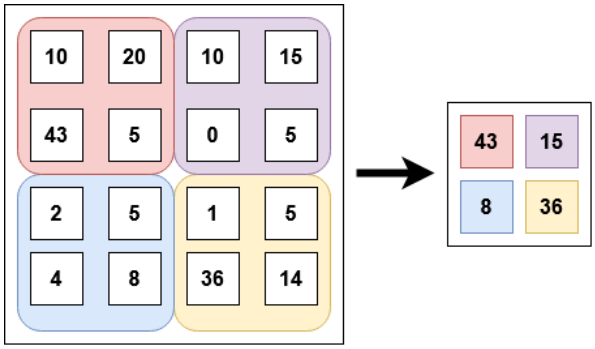
Obrázek 19: Ukázka aplikace Dropout. (Zdroj: (Srivastava, a další, 2014), přeloženo)

Výsledkem je robustnější neuronová síť, kde díky náhodnému vypínání neuronů je neuronová síť „nucena“ šířit chybu jinou cestou. Dále je neuronová síť méně náchylná na přetrénování (Srivastava, a další, 2014).

Na Obrázek 19 je zobrazeno porovnání struktur sítě při použití výběrové vrstvy. Levá část zobrazuje původní propojení neuronů. Pravá část zobrazuje tu samou síť při použití výběrové vrstvy. Neurony označené křížkem jsou deaktivované, respektive jejich výstup v následující vrstvě není aplikován.

3.1.5 Podvzorkovací vrstva s výběrem maxima (Max-Pool)

MaxPool vrstva slouží ke snížení počtu vstupů, čímž snižuje i náročnost učení. Vrstva má definovanou velikost jádra, horizontální krok h a vertikální krok v. Princip *MaxPool* vrstvy je takový, že z oblasti ohraničené jádrem je vybrána maximální hodnota a ta je zapsána na příslušnou pozici nové matice. Následuje krok, kdy jádro je na vstupní matici posouváno s horizontálním a vertikálním krokem. Rozměry výstupní matice lze definovat rovnicí (3.3). Vrstva se chová jako filtr typu horní propust, kde zvýrazňuje vysokofrekvenční složku ze vstupního vzoru. Tato vlastnost může být občas na škodu, zejména v případě, kdy ve vstupním vzoru existuje např. silný impulsní šum, který vrstva zvýrazní. V rámci učení dochází k výběru nejsilnější odezvy od skupiny neuronů. Nevýhodou je ovšem fakt, v lokálním místě může existovat skupina neuronů, která generuje podobné výsledky, přičemž z těchto silných odezev je vybrána pouze nejsilnější odezva a ostatní budou maskovány. Pro eliminaci maskování je potřeba vhodně zvolit velikost jádra a taktéž horizontální a vertikální krok.



Obrázek 20: Ukázka funkce MaxPool vrstvy (Zdroj: Vlastní)

$$Size = \frac{m}{v} \times \frac{n}{h} \tag{3.3}$$

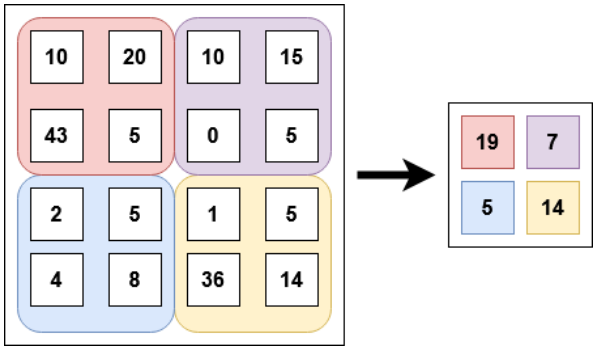
Kde:

- m je vertikální rozměr vstupní matice
- n je horizontální rozměr vstupní matice
- h je horizontální krok posuvu jádra
- v je vertikální krok posuvu jádra
- $Size$ je výsledný rozměr matice po aplikace MaxPool

Na Obrázek 20 je zobrazena ukázka nejčastější konfigurace *MaxPool* vrstvy – velikost jádra 2x2, horizontální i vertikální posuv 2. Z Obrázek 20 i rovnice (3.3) je patrné, že při výše uvedené konfiguraci je výstupní matice poloviční.

3.1.6 Podvzorkovací vrstva s výběrem průměru (Average pool)

AveragePool vrstva slouží stejně jako *MaxPool* vrstva ke snížení počtu vstupů. Parametry vrstvy jsou téměř totožné jako *MaxPool*. Na rozdíl od *MaxPool* vrstvy provádí průměrování výstupů ze sekce neuronů. Výhodou je, že skupina neuronů generujících silnou odezvu bude generovat i silnou odezvu v průměru. *AveragePool* vrstva se chová jako filtr typu dolní propust, tudíž velice dobře eliminuje bodové rušení, například impulsním šumem. Nevýhodou je to, že odstraňuje vysokofrekvenční složku, což vede například k rozmazání hran, které jsou většinou důležité pro další zpracování.

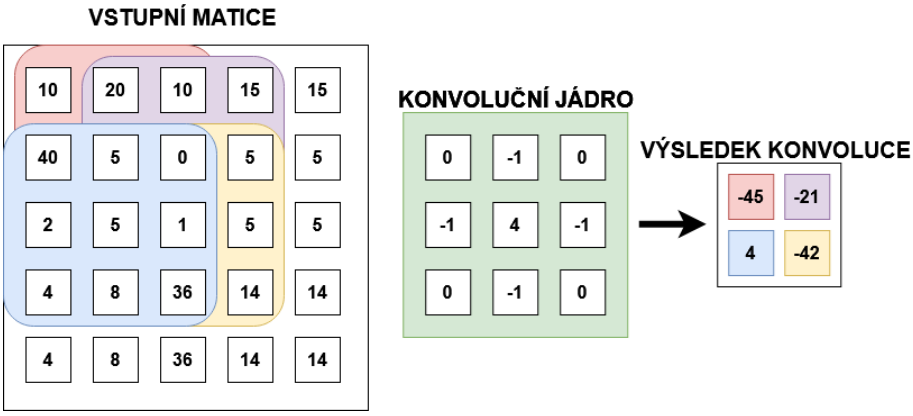


Obrázek 21: Ukázka funkce AveragePool vrstvy (Zdroj: Vlastní)

3.1.7 Konvoluční vrstva

Konvoluční neuronové sítě jsou nejpoužívanější neuronové sítě pro zpracování obrazových dat. Jak již z názvu vyplývá, jsou založeny na konvoluci obrazu s definovaným konvolučním jádrem.

Konvolučních jader ve zpracování obrazu existuje nepřeberné množství podle způsobu užití. Diskrétní podoba konvoluční funkce je definována rovnicí (3.4). Je třeba definovat, zda výsledná matice $g(x, y)$ bude mít po aplikaci konvolučního jádra shodné rozměry, či dojde k rozšíření velikosti matice o velikost konvolučního jádra. V prvním případě, kdy požadujeme, aby matice $g(x, y)$ měla shodné rozměry, je nutné zdůraznit fakt, že okrajové pixely nebudou správně násobeny konvolučním jádrem → vznikne nehomogenní okraj. Vhodnější metodou je použití rozšíření vstupní matice o rozměry konvolučního jádra, čímž je zajištěno, že konvoluční jádro je správně aplikováno i na okrajové pixely. Hodnota rozšířených pixelů je v naprosté většině případů duplikována na základě hodnoty nejbližšího pixelu. Výslednou matici po konvoluci je ovšem nutné posléze oříznout. Na Obrázek 22 je zobrazen princip konvoluce vstupního vzoru s konvolučním jádrem.



Obrázek 22: Ukázka principu 2D konvoluce (Zdroj: Vlastní)

$$g(x,y) = f(x,y) * h(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(i,j) \cdot h(x-i, y-j) \quad (3.4)$$

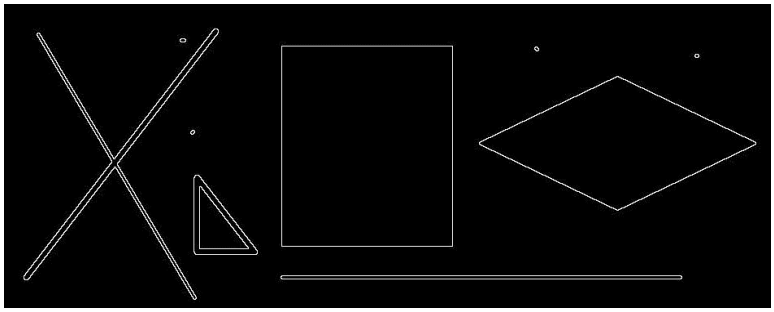
Kde:

- $g(x,y)$ je výsledná matice po aplikaci konvolučního jádra,
- $f(x,y)$ je vstupní matice,
- $h(x,y)$ je použité konvoluční jádro

3.1.7.1 Konvoluční jádra

Konvoluční jádro se volí dle požadovaného výsledku. V obrazovém zpracování se nejčastěji používá konvoluční jádro pro detekci hran či bodu, zaostření či rozostření.

Základním požadavkem na konvoluční jádro ve zpracování obrazu je jeho neutralita, tj. součet vah v konvolučním jádře je ideálně roven hodnotě nula. V případě konvolučního jádra použitého jako filtr šumu, se nejčastěji používá součet vah v konvolučním jádře roven jedné. Neutralita jádra způsobuje, že nedochází ke změně střední hodnoty výsledné matice. Druhým nedílným požadavkem je lichý počet koeficientů, čímž je zaručena jednoznačnost hodnoty centrálního prvku matice. V případě použití sudého počtu koeficientů, dochází k nejednoznačnosti v centrálním prvku matice. Pro představu, jak konvoluční jádra vypadají, jsou v práci uvedeny základní typy konvolučních jader. Pro ukázku funkce těchto konvolučních jader je použit vstupní snímek, který je zobrazen na Obrázek 23.



Obrázek 23: Vstupní data pro ukázku konvolučních jader (Zdroj: Vlastní)

Neuronové sítě používají podobná konvoluční jádra, ovšem s tou odlišností, že hodnoty v konvolučním jádře jsou postupně evaluovány na hodnotu, která nejlépe vyhovuje požadovaným výsledkům.

3.1.7.1.1 Konvoluční jádro pro detekci bodu

Bod v obraze je definován jako skoková změna intenzity jasu jednoho či skupiny pixelů ve všech směrech oproti okolí. Pro detekci bodu se používají například následující konvoluční jádra, jejichž podoba je uvedena v rovnici (3.5) a (3.6).

$$g(x,y) = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{3.5}$$

$$g(x,y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{3.6}$$

Obě konvoluční jádra v obraze zvýrazňují skokové změny intenzity. Konvoluční jádra tohoto typu se chovají v obraze jako filtr typu horní propust, čímž dochází k detekci bodu či zaostření hran.

3.1.7.1.2 Konvoluční jádro pro detekci hrany

Hrana v obraze je definována jako skoková změna intenzity jasu jednoho či skupiny pixelů ve v definovaném směru oproti okolí. Detektory hran lze rozdělit na detektory založené na několika principech.

1) Aproximace maxima první derivace (Roberts, Prewitt, Sobel, Canny)
2) Hledání průchodu druhých derivací nulou (Marr–Hildreth)
3) Lokální aproximace obrazové funkce modelem – výpočet derivace analyticky z parametru modelů

Pro detekci hran se používají například následující konvoluční jádra, jejichž podoba je uvedena v rovnici (3.7) až (3.9).

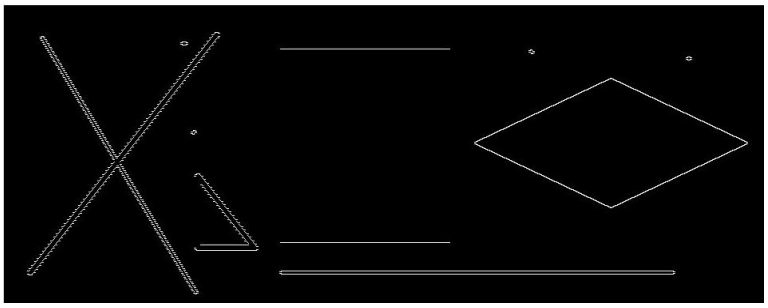
$$g(x,y) = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \tag{3.7}$$

$$g(x,y) = \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix} \tag{3.8}$$

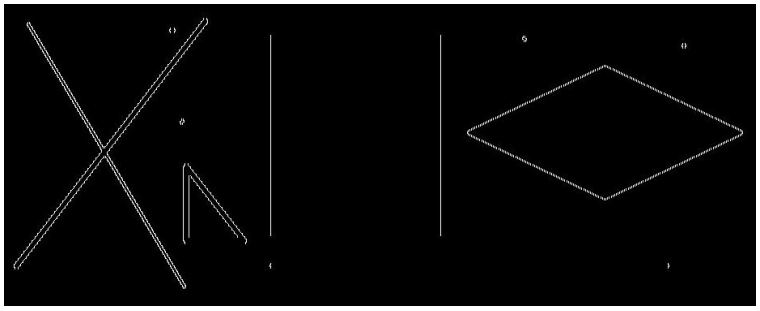
$$g(x,y) = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \tag{3.9}$$

Konvoluční jádro popsané maticí (3.7), v obraze zvýrazňuje horizontální hrany, konvoluční jádro (3.8) zvýrazňuje vertikální hrany a konvoluční jádro (3.9) zvýrazňuje hrany s úhlem 45°.

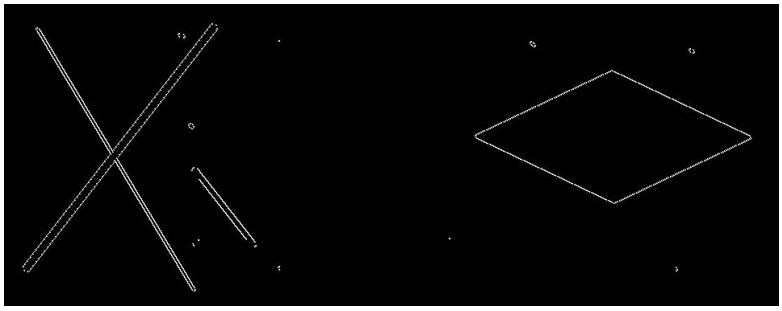
Na Obrázek 24 je zobrazeno jádro (3.7), které zvýrazňuje horizontální čáry. Vertikální čáry jsou naopak úplně potlačeny. Druhé jádro (3.8), jehož výsledek je zobrazen na Obrázek 25 naopak úplně potlačí horizontální čáry, zatímco vertikální jsou zvýrazněny. Na Obrázek 26 je vidět aplikace konvolučního jádra (3.9), kde dochází k eliminaci horizontálních i vertikálních čar, naopak čáry pod úhlem jsou zvýrazněny.



Obrázek 24: Výsledek po detekci horizontálních čar za využití jádra (3.7) (Zdroj: Vlastní)



Obrázek 25: Výsledek po detekci horizontálních čar za využití jádra (3.8) (Zdroj: Vlastní)



Obrázek 26: Výsledek po detekci čar pod úhlem -45° (3.9) (Zdroj: Vlastní)

Ve zpracování obrazu se velice často používají jádra *Prewitt* či *Sobel.* Tyto jádra mají podobu uvedenou rovnicemi (3.10), či (3.11). Jádra jsou vyvážená, tudíž nedochází ke změně střední hodnoty. Ukázka rozdílu použití jader je zobrazena na Obrázek 22.

Prewitt

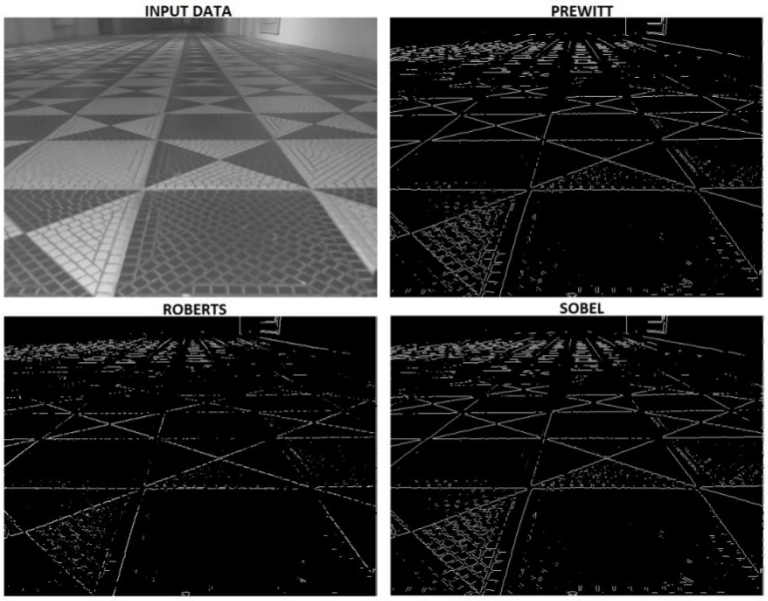
$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \tag{3.10}$$

a) b) c) d)

Sobel

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix} \tag{3.11}$$

a) b) c) d)



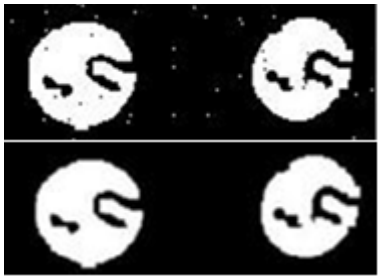
Obrázek 27: Porovnání výstupů z jednotlivých hranových detektorů (Zdroj: Vlastní)

3.1.7.1.3 Konvoluční jádro pro odstranění šumu

Šum v obraze, ať už je typu *salt & pepper* či *Gaussovský šum*, je definován jako impulsní šum s vysokofrekvenční složkou. Pro odstranění šumu lze velice jednoduše použít konvoluční jádra s funkcí průměrování, které velice dobře tento impulsní šum odstraní. Nevýhodou je ovšem fakt, že dojde k rozostření užitečného signálu, například hran.

$$g(x,y) = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{3.12}$$

Rovnice (3.12) popisuje základní konvoluční jádro pro operaci průměrování, resp. operaci rozostření.



Obrázek 28: Ukázka šumu typu salt&pepper (nahoře) a jeho eliminace (dole) za využití konvolučního jádra (3.12) (Zdroj: Vlastní)

Alternativou je použití upraveného konvolučního jádra, jako je uvedeno v rovnici (3.13), která popisuje Gaussovský filtr. Tento filtr nepočítá prostý aritmetický průměr, ale aritmetický průměr vážený Gaussovou funkcí. Výsledná odezva je taková, že je zesílen vliv pixelu ve středu konvolučního jádra.

$$g(x,y) = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \tag{3.13}$$

3.1.8 Separovatelná konvoluční vrstva

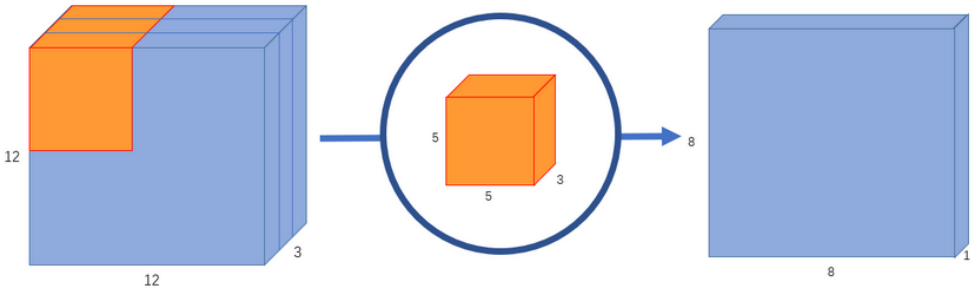
Separovatelné konvoluční vrstvy jsou alternativou ke klasickým konvolučním vrstvám, avšak jsou navrženy tak, aby byly výrazně efektivnější z hlediska výpočetního času. Základní princip spočívá ve vytvoření mezikroku ve výpočtu a základních vlastností matic, kdy určitou matici lze rozložit na dva vektory.

$$\begin{bmatrix}3 & 6 & 9\\ 4 & 8 & 12\\ 5 & 10 & 15\end{bmatrix} = \begin{bmatrix}3\\ 4\\ 5\end{bmatrix} \times \begin{bmatrix}1 & 2 & 3\end{bmatrix} \tag{3.14}$$

3.1.8.1 Porovnání výpočetní náročnosti

Pro ukázku úspory výpočetního času je nutné stanovit si základní počet aritmetických operací pro „klasickou“ konvoluci.

V případě vstupní matici o rozměru $A \times A \times C$, například o rozměru matice $12 \times 12 \times 3$ na kterou je aplikováno konvoluční jádro o rozměru $d \times d \times N$ například rozměr $5 \times 5 \times 3$. Výstupní matici z konvoluce bude mít obecný rozměr $K \times K \times C$, při aplikaci operace konvoluce u výše uvedených matic, vznikne výstupní matice o rozměru $8 \times 8 \times 1$. Dochází zde ke zmenšení rozměru výstupní matice. Pro výpočet je uvažován pohyb konvolučního jádra o 1 (*stride = 1*) a nedochází k doplňování hodnot (*no padding*).



Obrázek 29: Ukázka aplikace konvoluce bez využití paddingu (Zdroj: (Wang, 2018))

Výpočetní náročnost jedné aplikace konvolučního jádra lze stanovit rovnicí (3.15).

$$O = d^2 \times C = 5^2 \cdot 3 = 75 \qquad (3.15)$$

Při aplikaci konvolučního jádra na celý snímek je možné stanovit výpočetní náročnost rovnicí (3.16). Z rovnice vyplývá, že výpočetní náročnost výrazně stoupá s dimenzí vstupní matice. Ve výpočtech byla uvažována hloubka konvolučního jádra $N = 1$.

$$O = (A - d + 1)^2 \times d^2 \times \mathrm{C} \times \mathrm{N} = 8^2 \cdot 5^2 \cdot 3 \cdot 1 = 4800 \qquad (3.16)$$

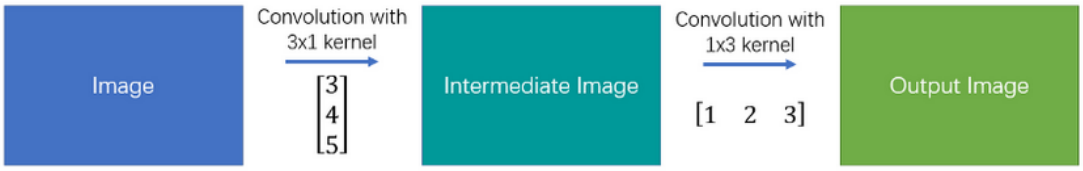
Obecně jsou konvoluce aplikovány na mnohem větší rozměry matic. V případě reálnější skutečnosti (zejména i pro porovnání výpočetní náročnosti s pozdějšími výpočty) kdy se uvažuje, že hloubka konvolučního jádra bude například $N = 256$, vychází celkový počet aritmetických operací dle rovnice (3.17).

$$O = (A - d + 1)^2 \times d^2 \times \mathrm{C} \times \mathrm{N} = 8^2 \cdot 5^2 \cdot 3 \cdot 256 = 1\,228\,800 \qquad (3.17)$$

Kde:

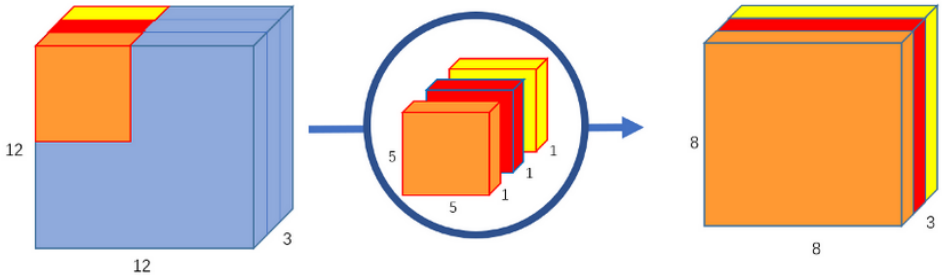
- O je výsledný počet operací
- K je rozměr výstupní matice
- d je rozměr konvolučního jádra
- C je počet barevných kanálů
- N je obecná hloubka konvolučního jádra

Pro další výpočty bude brána poslední konfigurace, kdy ze vstupní matice $12 \times 12 \times 3$ vznikne při konvoluci výstupní matice o rozměru $8 \times 8 \times 256$. Rovnicí (3.17) došlo ke stanovení výpočetní náročnosti „klasické“ konvoluční vrstvy. Pro snížení výpočetních nároků na výpočet konvoluce byla vytvořena metoda hluboké konvoluce (*Depthwise convolution*) a metoda bodové konvoluce (*Pointwise convolution*).



Obrázek 30: Ukázka aplikace separovatelné konvoluce (Zdroj: (Wang, 2018))

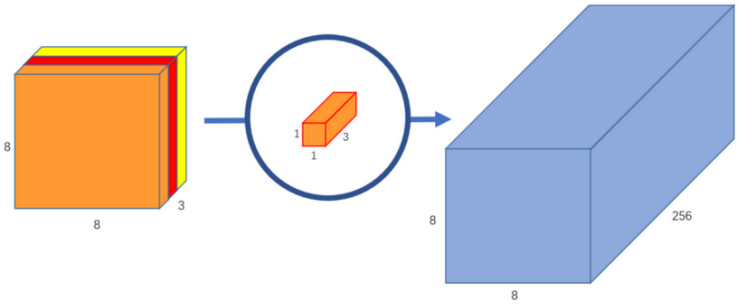
Metoda hloubkové konvoluce využívá konvoluční jádro nikoliv o rozměru $d \times d \times N$ jako v „klasické“ konvoluci ale využívá se rozměr $d \times d \times 1$. Dá se tedy říct, že v hloubkové konvoluci nedochází ke konvoluci s N kanály v matici ale jen N-krát s jedním kanálem jako je zobrazeno na Obrázek 31. Výsledného rozměru se docílí sloučením paralelně zpracovávaných kanálů do jedné matice. Výsledná matice je vlastně mezi výpočet separovatelné konvoluce. Pro získání výsledků je nutné použít druhou část algoritmu, tj. bodovou konvoluci.



Obrázek 31: Aplikace hloubkové konvoluce (Zdroj: (Wang, 2018))

Výpočetní náročnost jedné aplikace konvolučního jádra v hloubkové konvoluci lze stanovit rovnicí (3.18).

$$O = (A - d + 1)^2 \times d^2 \times \mathrm{C} = 8^2 \cdot 5^2 \cdot 3 = 4800 \qquad (3.18)$$



Obrázek 32: Aplikace bodové konvoluce (Zdroj: (Wang, 2018))

Bodová konvoluce následuje hloubkovou konvoluci. Při bodové konvoluci je vytvořeno 256 konvolučních jader N, každé o rozměru $1 \times 1 \times 3$. Počet aritmetických operací při bodové konvoluci je možné určit rovnicí (3.19).

$$O = A^2 \times \mathrm{d}^2 \times \mathrm{C} \times \mathrm{N} = 8^2 \cdot 1^2 \cdot 3 \cdot 256 = 49152 \qquad (3.19)$$

Pro zjištění celkového počtu aritmetických operací, je nutné sečíst výsledky z rovnic (3.18) a (3.19). Výsledná hodnota je 52 952 operací. Porovnáním výsledků

s „klasickou“ konvolucí, kde výsledný počet operací byl roven 1 228 800 operací, vychází, že počet aritmetických operací při použití separovatelné konvoluce je přibližně 23× menší.

3.2 Učení neuronové sítě

3.2.1 Algoritmus zpětného šíření chyby

Algoritmus zpětného šíření chyby, tj. backpropagation algoritmus, je nejrozšířenější metodou pro učení dopředných neuronových sítí. Používá se při učení vícevrstvých neuronových sítí při učení s učitelem, tj. v případě kdy pro učení je vytvořena sada vstupů a požadovaných odezev sítě na daný vstup. Metoda zpětného šíření chyby je speciální případ metody gradientního sestupu.

3.2.1.1 Historie algoritmu zpětného šíření chyby

Metoda je založena na principu nalezení lokálního minima diferencovatelné funkce, přičemž využívaná metoda, nazývaná *Gradient Descent* byla poprvé zmíněna již v roce 1847 Luisem Augustinem Cauchy v publikaci (Cauchy, 1847), ovšem konvergenční vlastnosti byly studovány až téměř o sto let později Haskallem Currym v publikaci (Curry, 1944). Základy metody byly navrženy v kontextu s teorií řízení odvozeny z principu dynamického programování, které popsal Richard Bellman v roce 1958 ve své publikaci (Bellman, 1958). Z Belmannovy práce vycházel Henry J. Kelley v publikaci (Kelley, 1960), ve které autor popsal analytický vývoj metody nazvané *Method of steepest descent* pro optimalizaci správné sekvence leteckých tras nutných k obsloužení s minimálním množstvím kroků, v případě letecké trasy s minimálním počtem nalétaných kilometrů. Podobnou publikaci se zaměření na optimalizaci nejmenšího množství mezikroků vydal v roce 1961 Arthur E. Bryson v publikaci (Bryson, 1961).

V roce 1970 v rámci diplomové práce Seppo Linnainmaa publikoval moderní variantu zpětného šíření chyby (Linnainmaa, 1970), (Linnainmaa, 1976), která je efektivní i v řídkých sítích (Schmidhuber, 2014). Metoda využívá automatického derivování diskrétních sítí vnořených diferencovatelných funkcí.

3.2.1.2 Princip algoritmu

Algoritmus zpětného šíření chyby je založen na faktu, že neuronové síti předkládáme určitý vstup a zároveň máme požadovanou odezvu. Předložením určitého vzoru na vstup neuronové sítě je získána odezva sítě, která je porovnána s požadovaným výstupem. Výstupem je tzv. chybová funkce, která udává jak hodně se síť „spletla“ při svém odhadu. Na základě gradientního sestupu a chybové funkce se vypočítá, o jakou

hodnotu se má konkrétní váha u konkrétního neuronu změnit. Cílem celého učení je najít globální minimum – najít takové nastavení všech koeficientů sítě, aby byla chyba minimální. Základním problémem je ovšem fakt, že z principu není globální minimum známé. Při učení může síť „uvíznout“ v lokálním minimu, které může být zásadně odlišné od globálního minima. Výsledkem uváznutí je špatné či žádné naučení sítě. Při tvorbě této práce k tomuto faktu častokrát došlo. Výsledná odezva sítě na jakýkoliv vstupní snímek byla monolitická šedá plocha. Pro naučení neuronové sítě je nutné celé učení několikrát opakovat a z výsledků vybrat řešení s nejmenší chybovou funkcí.

Kvalita výsledků neuronové sítě je nejčastěji určena střední kvadratickou chybou (3.20) či celkovou energií sítě (3.21).

$$MSE = \frac{1}{\mathrm{N}}\sum_{i=1}^{N}(t_i - r_i)^2 \tag{3.20}$$

$$E = \frac{1}{2}\sum_{i=1}^{n}(t_i - r_i)^2 \tag{3.21}$$

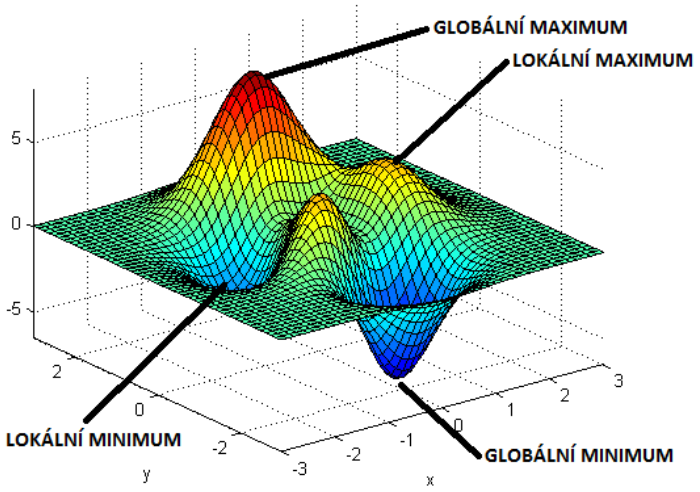
Kde:

- MSE je výsledná chybová funkce; Mean Square Error
- E je výsledná chybová funkce – Celková energie sítě
- N je počet vzorků předložených síti
- t_i je požadovaná odezva sítě na í-tý vzorek
- r_i je skutečná odezva sítě na í-tý vzorek

Cílem učení neuronové sítě je minimalizovat zvolenou chybovou funkci (3.20) či (3.21). Metoda využívá tzv. řetězového pravidla tzv. c*hain-rule*, pro postupné šíření chyby gradientu u jednotlivých vrstev, ve směru od výstupní vrstvy, přes vnitřní skryté vrstvy po vrstvu vstupní. Postupně tedy dochází k úpravě konkrétních vah, dokud nedojde ideálně k nalezení globálního minima chybové funkce.

Algoritmus obsahuje parametr signalizující rychlost učení, tzv. *Learning-rate* v intervalu $\langle 0,1 \rangle$. Rychlost učení je závislá na konkrétním nastavení, ovšem vyšší hodnoty rychlosti učení způsobí rychlejší učení díky větší modifikaci jednotlivých vah, ovšem může se stát, že učení uvázne v lokálním minimu, ze kterého nebude schopen algoritmus se dostat.

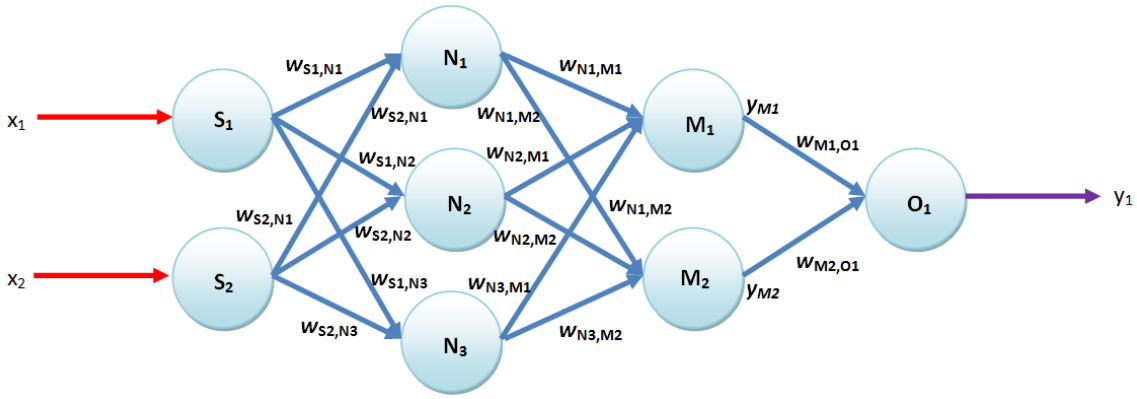
Jednou z podmínek funkce algoritmu je diferencovatelnost aktivační funkce u všech použitých neuronů, tj. musí existovat konečná derivace aktivační funkce. Algoritmus využívá metody postupného sestupu, tzv. *Gradient Descent*, kde ve směru největšího gradientu roste nejvíce i chyba. Algoritmus tedy postupuje proti směru tohoto gradientu.



Obrázek 33: Ukázka různých maxim a minim (Zdroj: (Gandhi, 2018), upraveno)

3.2.1.3 Postup algoritmu

Pro ukázku práce se použije dopředná neuronovou síť obsahující V vstupů, O výstupů a $S_1, \dots, S_M$ neuronů ve vnitřních vrstvách. Ilustrace zapojení sítě je zobrazeno na Obrázek 18.



Obrázek 34: Dopředná neuronová síť (Zdroj: Vlastní)

Pro jednoduchou ukázku výpočtu se bude počítat, že síť bude obsahovat dva vstupní neurony, z nichž každý bude mít pouze jeden vstup, tj. síť bude mít celkem dva vstupy x_1, x_2, dále budeme mít dvě vnitřní vrstvy. První vnitřní vrstva S_1 se třemi neurony N_1, N_2, N_3, druhá vnitřní vrstva S_2 se dvěma neurony M_1, M_2 a jeden výstupní neuron O_1, který má pouze jeden výstup y_1. Mezi i-tým neuronem k-té vrstvy a j-tým neuronem k+1 vrstvy je vytvořena synapse s vahou w^{k+1}. V kooperaci s ilustrací, váha synapse mezi neuronem M_1 a O_1 je označena $w_{M1,O1}$, výstupní hodnota z neuronu M_1 je označena y_{M1}.

Agregační funkce použitá u neuronu je vypočítána dle rovnice (2.5) v kapitole 2.3.3.1. Pro možnost aplikace algoritmu zpětného šíření chyby je nutné použít aktivační funkci, která je diferencovatelná. Obecně je možné použít jakoukoliv diferencovatelnou aktivační funkci, ale pro jednoduchost a ukázku byla použita aktivační funkce φ^{k+1} jednotkový skok, jejíž průběh je zobrazen na Obrázek 12, a její derivace je rovna jedné.

$$y^{k+1}{}_j = \varphi^k\left(\sum_{i=1}^{S_M} w^{k+1}{}_{j,i} \cdot y_i^k + w^{k+1}{}_{\mathrm{j}}\right) = \varphi^k\left(y_a^{k+1}{}_j\right) \qquad (3.22)$$

Kde:

- $y^{k+1}{}_j$ je výstupní potenciál j-tého neuronu z k-té vrstvy
- φ je použitá aktivační funkce
- S_M je počet neuronů
- $w^{k+1}{}_{j,i}$ je váha spojení synapse y_i^k
- $w^k{}_j$ je prahová hodnota neuronu j-tého neuronu z k-té vrstvy

Dále je možné rovnici (3.22) převést na maticovou podobu, uvedenou rovnicemi (3.23) až (3.32)

$$\boldsymbol{y}^{\mathrm{M}} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_V \end{bmatrix} \qquad (3.23)$$

$$\boldsymbol{y^{k+1}} = \boldsymbol{\varphi^{k+1}}\left(\boldsymbol{W^{k+1}} \cdot y^k + \boldsymbol{w}^{k+1}\right) \qquad (3.24)$$

Kde:

$$\mathrm{k} = 0,1,\cdots \mathrm{M}-1$$

- $\boldsymbol{\varphi^{k+1}}$ je sloupcový vektor použitých aktivační funkcí
- $\boldsymbol{W}^{k+1}$ je matice obsahující váhy jednotlivých synapsí

$$\boldsymbol{W}^{k+1} = \begin{bmatrix} w^{k+1}{}_{1,1} & \cdots & w^{k+1}{}_{1,S_k} \\ \vdots & \cdots & \vdots \\ w^{k+1}{}_{S_{k+1},1} & \cdots & w^{k+1}{}_{S_k,S_{k+1}} \end{bmatrix}$$

- $\boldsymbol{w}^{\mathrm{k+1}}$ je sloupcový vektor prahových hodnot neuronu

$$\boldsymbol{w}^{k+1} = \begin{bmatrix} w_1^{k+1} \\ \vdots \\ w_{S_{k+1}}^{k+1} \end{bmatrix}$$

Pokud je na vstup sítě přiveden vektor vstupů $\boldsymbol{y}^0$, cyklickou aplikací vznikne odezva sítě, označená vektorem $\boldsymbol{y}^M$. Cílem metody je nastavit váhy jednotlivých synapsí $w_{j,i}^M$ tak, aby chybová funkce E (3.26), (například celková energie sítě či střední kvadratická

chyba) mezi skutečnou odezvou sítě $\boldsymbol{y}^M$ a požadovanou odezvou sítě $\tilde{\boldsymbol{y}}$ byla ideálně nulová (3.25).

$$\boldsymbol{e} = \tilde{\boldsymbol{y}} - \boldsymbol{y}^M \tag{3.25}$$

$$\mathrm{E} = \frac{1}{2}\sum_{i=1}^{n}\left(\tilde{y}_i - \mathrm{y_i^M}\right)^2 \tag{3.26}$$

Algoritmus zpětného šíření chyby slouží k postupné aktualizaci jednotlivých vah tím, že nová váha je vypočítána dle rovnice (3.27 a 3.28)

$$\breve{w}_{j,i}^k = w_{j,i}^k - \alpha \cdot \frac{\partial \mathrm{E}}{\partial w_{j,i}^k} \tag{3.27}$$

$$\breve{w}_j^k = w_j^k - \alpha \cdot \frac{\partial \mathrm{E}}{\partial w_j^k} \tag{3.28}$$

Kde:

- $\breve{w}_{j,i}^k$ je aktualizovaná hodnota jednotlivých vah na základě chybové funkce E
- α je koeficient rychlosti učení
- $\frac{\partial \mathrm{E}}{\partial w_{j,i}^k}$ je parciální derivace chybové funkce

Pro výpočet aktualizované hodnoty jednotlivých vah $\breve{w}_{j,i}^k$ je nutné vypočítat parciální derivaci $\frac{\partial \mathrm{E}}{\partial w_{j,i}^k}$. Taktéž je nutné stanovit lokální gradient j-tého neuronu, označený $\delta^k{}_j$.

$$\delta^k{}_j = \frac{\partial E}{\partial y_a^k{}_j} \tag{3.29}$$

Aplikací na parciální derivace $\frac{\partial \mathrm{E}}{\partial w_{j,i}^k}$hodnotu $y_a^k{}_j$ jak na čitatele, tak jmenovatele, hodnota se nezmění, ale je možné stanovit lokální gradient j-tého neuronu $\delta^k{}_j$. Z rovnic (3.24), (3.26) a 3.29) je možné stanovit rovnice (3.30) a (3.31).

$$\frac{\partial \mathrm{E}}{\partial w_{j,i}^k} = \frac{\partial E}{y_a^k{}_j} \cdot \frac{y_a^k{}_j}{\partial w_{j,i}^k} = \delta^k{}_j \cdot y^{k-1}{}_j \tag{3.30}$$

$$\frac{\partial \mathrm{E}}{\partial w^k{}_j} = \frac{\partial E}{y_a^k{}_j} \cdot \frac{y_a^k{}_j}{\partial w^k{}_j} = \delta^k{}_j \tag{3.31}$$

Odborná literatura, například (Haykin, 1999) popisuje, že pro lokální gradient platí následující rekurentní vztah:

$$\boldsymbol{\delta}^k = \boldsymbol{\Phi}^{\mathrm{k}}\left(\mathbf{y}_{\mathrm{a}}^{\mathrm{k}}\right) \cdot \left(\mathbf{W}^{\mathrm{k+1}}\right)^T \cdot \boldsymbol{\delta}^{k+1}$$

Kde:

- $\boldsymbol{\delta}^k$ je sloupcový vektor lokálních gradientů neuronů

$$\boldsymbol{\delta}^k = \begin{bmatrix} \boldsymbol{\delta}^k{}_1 \\ \vdots \\ \boldsymbol{\delta}^k{}_{Sk} \end{bmatrix}$$

- $\boldsymbol{\Phi}^{\mathrm{k}}(\mathbf{y}_{\mathrm{a}}^{\mathrm{k}})$ je matice obsahující derivace aktivačních funkcí $\dot{\boldsymbol{\varphi}}^{\boldsymbol{k}}(\boldsymbol{y}_{\boldsymbol{a}\ \boldsymbol{j}}^{\boldsymbol{k}})$

$$\boldsymbol{\Phi}^{\mathrm{k}}(\mathbf{y}_{\mathrm{a}}^{\mathrm{k}}) = \begin{bmatrix} \dot{\boldsymbol{\varphi}}^{\boldsymbol{k}}(\boldsymbol{y}_{\boldsymbol{a}\ \boldsymbol{1}}^{\boldsymbol{k}}) & 0 & \cdots & 0 \\ 0 & \dot{\boldsymbol{\varphi}}^{\boldsymbol{k}}(\boldsymbol{y}_{\boldsymbol{a}\ \boldsymbol{2}}^{\boldsymbol{k}}) & \cdots & 0 \\ \vdots & \cdots & \ddots & \vdots \\ 0 & 0 & \cdots & \dot{\boldsymbol{\varphi}}^{\boldsymbol{k}}(\boldsymbol{y}_{\boldsymbol{a}\ \boldsymbol{Sk}}^{\boldsymbol{k}}) \end{bmatrix}$$

$$\dot{\boldsymbol{\varphi}}^{\boldsymbol{k}}(\boldsymbol{y}_{\boldsymbol{a}\ \boldsymbol{j}}^{\boldsymbol{k}}) = \frac{\boldsymbol{\partial \varphi}^{\boldsymbol{k}}}{\boldsymbol{\partial y}_{\boldsymbol{a}\ \boldsymbol{j}}^{\boldsymbol{k}}}$$

- $\boldsymbol{W}^{k+1}$ je matice obsahující váhy jednotlivých synapsí

$$\boldsymbol{W}^{k+1} = \begin{bmatrix} w^{k+1}{}_{1,1} & \cdots & w^{k+1}{}_{1,S_k} \\ \vdots & \cdots & \vdots \\ w^{k+1}{}_{S_{k+1},1} & \cdots & w^{k+1}{}_{S_k,S_{k+1}} \end{bmatrix}$$

Pro výpočet algoritmu zpětného šíření chyby byly stanoveny všechny důležité rovnice. V algoritmu nejprve dojde k výpočtu odezvy sítě na stanovené vstupní parametry. Jako výchozí hodnotu učení je nutné definovat inicializační hodnotu vah jednotlivých synapsí. Odborná literatura (Doležel, 2011), (Haykin, 1999) doporučuje použít počáteční hodnotu vah s $N(0, 0.1)$. Po aplikaci počátečních podmínek je vypočítána odezva sítě pomocí rovnice (3.23). Vypočítaná odezva sítě $\boldsymbol{y}^{\boldsymbol{M}}$ se porovná s očekávanou hodnotou $\tilde{\boldsymbol{y}}$ a stanoví se hodnota lokálního gradientu neuronů ve výstupní vrstvě $\boldsymbol{\delta}^{\boldsymbol{M}}$ (3.32).

$$\boldsymbol{\delta}^{\boldsymbol{M}} = -\boldsymbol{\Phi}^{\boldsymbol{M}}(\mathbf{y}_{\mathrm{a}}^{\mathrm{M}}) \cdot (\tilde{\boldsymbol{y}} - \boldsymbol{y}^{M}) \qquad (3.32)$$

Kde:

$\boldsymbol{\delta}^k$ je sloupcový vektor lokálních gradientů u výstupních neuronů

$\boldsymbol{\Phi}^{\boldsymbol{M}}(\mathbf{y}_{\mathrm{a}}^{\mathrm{M}})$ je matice obsahující derivace aktivačních funkcí výstupních neuronů

$\tilde{\boldsymbol{y}}$ je požadovaný vektor výstupu z neuronové sítě

$\boldsymbol{y}^{\boldsymbol{M}}$ je vypočítaný vektor výstupu z neuronové sítě

Po získání hodnot lokálních gradientů výstupní vrstvy je možné rekurzivně vypočítat příslušné gradienty ve vnitřních vrstvách. Po získání gradientů je možné

za využití rovnic vypočítat aktualizace vah synapsí (3.27), (3.30) a jejich prahů (3.28), (3.31).

Popsaný algoritmus je celkem pomalý a velice často konverguje v lokálním minimu, proto vznikla celá řada modifikací, například Levenbergova–Marquardtova modifikace BPG algoritmu, která do algoritmu implementuje Newtonovu optimalizační metodu. Celá metoda je v detailu popsána například v publikacích (Doležel, 2011), (Gavin, 2020), (Lera, a další, 2002)

3.2.2 Dávkové učení

V případě učení neuronových sítí se používají dva přístupy. První přístup se dá považovat za streamované učení, občas označované jako online učení. Toto učení je založeno na faktu, že neuronová síť nemá k dispozici data, která dorazí v kroku *N+1*. Výsledné trénování sítě je prováděno vždy na jednom aktuálním vzorku dat, posléze se provede aktualizace parametrů za každý tréninkový vzor. Je potřeba podotknout, že gradienty tréninkového vzoru nesledují globální gradient a ve výsledku při učení dochází ke „kličkování“ gradientu. Tento styl učení je ve srovnání s druhým přístupem výrazně pomalejší.

Druhý přístup učení je dávkové učení, v zahraniční literatuře nejčastěji označované *Batch*. Dávkové učení vypočítá krok aktualizace parametrů až po vypočítání gradientu na *n* tréninkových vzorech. Tyto gradienty jednotlivých tréninkových vzorů se typicky zprůměrují a výsledný gradient je výrazně blíže ke globálnímu gradientu. Potenciálním problémem může být v tréninkových vzorech tzv. seskupení dat (například v případě učení druhů zvířat, několik snímků jednoho druhu zvířete za sebou). Pro dávkové učení je nezbytné provést dostatečné promíchání tréninkových vzorů, což způsobí, že gradient z každé dávky bude maximálně sledovat globální gradient. V případě datasetů s menším počtem tříd se doporučuje použít velikost dávky tréninkových dat stejné jako je počet tříd. V jiných případech, pokud je to výpočetně možné, je vhodné použít dávku o velikosti 10 až 100 prvků (Hinton, 2012). Další nespornou výhodou dávkového učení je velice dobrá možnost paralelizace. V neuronových sítích se pro výpočet používají maticové operace, které jsou současně velice dobře implementovány na grafických kartách, což výrazně zkracuje dobu učení.

3.3 Optimalizační algoritmy

Optimalizační algoritmy popisují metodu, jak zapracovat vypočítaný gradient do modelu. Nejčastějším problémem je nezbytnost velkých trénovacích sad pro dosažení náležité generalizace. Bohužel velká trénovací sada výrazně prodlužuje dobu učení a je výrazně výpočetně náročnější. Učení probíhá formou minimalizace chybové funkce, což je v případě učení neuronové sítě s učitelem suma rozdílu predikovaného výsledku od skutečného. Optimalizačních algoritmů existuje velké množství a v této kapitole je uveden přehled nejpoužívanějších jednoduchých algoritmů.

$$E(\xi) = \frac{1}{N}\sum_{i=1}^{N}\Psi(x^i, y^i, \xi) \tag{3.33}$$

Kde

- Ψ je zvolená chybová funkce
- N je celkový počet vzorků
- x^i je požadovaná hodnota výstupu sítě pro i-tou epochu
- y^i je skutečná hodnota výstupu sítě pro i–tou epochu
- ξ je vektor parametrů.

3.3.1.1 Stochastický gradientní sestup

Stochastický gradientní sestup (*Stochastic Gradient Descent*) je pravděpodobně nejrozšířenějším optimalizačním algoritmem pro učení neuronové sítě. Stochastický gradientní sestup je rozšířením metody gradientního sestupu (*Gradient Descent*).

Vyjde-li se ze základní definované rovnice (3.33) lze odvodit základní rovnici pro SGD.

$$\nabla E(\xi) = \frac{1}{N}\sum_{i=1}^{N}\nabla_{\xi}\Psi(x^i, y^i, \xi) \tag{3.34}$$

V případě použití nejčastější ztrátové funkce (3.34), je možné ilustrovat metodu SGD.

$$\Psi(y^i, \hat{y}^i) = \frac{1}{2}(y^i - \hat{y}^i)^2 \tag{3.35}$$

Kde $\hat{y}^i$ je výstup sítě pro i-tou iteraci. Kombinací rovnic (3.34) a (3.35) je možné vypočítat gradient funkce:

$$\nabla E(\xi) = \frac{1}{2N}\sum_{i=1}^{N}\nabla_{\xi}\frac{1}{2}(y^i - \hat{y}^i)^2 \tag{3.36}$$

V rovnici (3.36) je použit vektor parametrů ξ, který se skládá z vah w a vychýlení b, je nutné dle těchto dvou parametrů provést parciální derivace $\frac{\partial E}{\partial w}$ a $\frac{\partial E}{\partial b}$, konkrétněji změna chybové funkce při změně váhy w_{mn}^{l}mezi neurony v l-té vrstvě sítě spojující neuron m a n.

Při použití operací derivace a maticového násobení lze dojít k výsledným rovnicím pro změnu chybové funkce v závislosti na libovolné váze, jako je uvedeno v (Groman, 2019)

$$\frac{\partial E}{\partial w^{l}} = \mathrm{y}^{\mathrm{l}-1}\,\delta^{\mathrm{l}} \tag{3.37}$$

Dle rovnice (3.37) je možné usoudit, že pokud bude výstup z předchozí vrstvy $\mathrm{y}^{\mathrm{l}-1}$ velice malý $\mathrm{y}^{\mathrm{l}-1} \approx 0$, poté bude gradient $\frac{\partial E}{\partial w^{l}}$ taktéž velice malý, což způsobí pomalé učení, neboť změna vah v jednotlivých vrstvách je velice pomalá.

3.3.1.2 ADAM

Metoda adaptivního odhadu momentu je jednou z nejčastěji používaných optimalizačních metod. Metoda vypočítává adaptivní rychlost učení pro jednotlivé parametry. Metoda ukládá exponenciálně rozkládající se průměr umocněných gradientů u_t.

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{3.38}$$

$$u_t = \beta_2 u_{t-1} + (1-\beta_2) g_t^2 \tag{3.39}$$

Kde u_t a m_t jsou odhady střední hodnoty a rozptylu gradientů. Metoda ADAM tyto dva odhady při inicializaci nastaví na nulové vektory. Autoři metody zjistili, že při začátku učení a v případě malých hodnot β_1 a β_2 jsou spočítané odhady nestranné. Z výše uvedených důvodů jsou do metody zavedeny korekce uvedené v rovnici (3.40) a (3.41)

$$\widehat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{3.40}$$

$$\hat{u}_t = \frac{u_t}{1-\beta_2^t} \tag{3.41}$$

Tyto korekce jsou následně použity pro definici update pravidla ADAM algoritmu. Autoři metody doporučují nastavit počáteční hodnoty parametrů $\beta_1 = 0.9$, $\beta_2 = 0.999$ a $\varepsilon = 10^{-8}$. (Kingma, a další, 2015), (Groman, 2019),

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{u}_t}+\varepsilon}\widehat{m}_t \tag{3.42}$$

3.3.1.3 Adaptivní gradient

Optimizér adaptivní gradient (*AdaGrad*) je navržen tak, aby docházelo k adaptivní změně hodnoty rychlosti učení v závislosti na výskytu daných parametrů. V případě menšího výskytu parametrů jsou prováděné aktualizace vetší a u parametrů s větším výskytem jsou prováděné aktualizace menší. Výhodou adaptivního gradientu je že není nutné sledovat a nastavovat rychlost učení v čase. Nejčastější rychlostí učení se v případě adaptivního gradientu používá hodnota 0,01. Nevýhodou algoritmu je ovšem fakt, že v určitých případech dojde ke snížení rychlosti učení na velice malou hodnotu a může dojít k uvíznutí v lokálním minimu. Při každé epoše učení si optimalizační algoritmus uchovává předchozí hodnoty změny rychlosti učení (tzv. mezipaměť gradientu c_g, uvedený rovnicí (3.44)), které jsou uvedeny jako součet druhých mocnin v děliteli pro výpočet rychlosti učení (Duchi, a další, 2011), (Bengio, a další, 2016).

$$\theta_i = \theta_i - \varepsilon \frac{\sum_{b \epsilon B} \left[\frac{\delta\, cost(\theta)}{\delta \theta}\right]_b}{\sqrt{c_g}} \tag{3.43}$$

$$c_g = c_g + \left(\sum_{b \in B} \left[\frac{\delta\, cost(\theta)}{\delta \theta}\right]_b\right)^2 \tag{3.44}$$

3.4 Chybová funkce

Pro zjištění, jak dobře je schopna neuronová síť modelovat data se využívá funkce, která reprezentuje míru chyby sítě. V odborné literatuře se taktéž nazývá *cost funkce*. Součástí chybové funkce je funkce ztrátová, občas nazývaná jako *loss funkce*.

Existuje velké množství používaných funkcí, proto jsou v této kapitole zmíněny nejčastěji využívané. Ztrátová funkce vyjadřuje míru chyby, obecně se vyhodnocuje míra chyby jednoho testovaného vzoru či celé učené dávky. Pro určení chybové funkce dávkového učení se využívá průměr všech ztrátových (*loss*) funkcí jednotlivých testovaných vzorů. Chybová funkce celé sítě je tedy průměr ztrátových funkcí celého datasetu, což lze zapsat rovnicí (3.45).

$$cost(\theta) = \frac{1}{n} \sum_1^n loss(\theta, (x_n, y_n) \tag{3.45}$$

Ztrátovou funkci je možné dále regulovat, což obecně zabraňuje přeučení neuronové sítě, protože nutí výslednou funkci být méně lineární (Nielsen, 2015), (Lisa Lab, 2015). Ukázky základních ztrátových funkcí a jejich matematického vyjádření jsou uvedeny níže.

- Negativní logaritmická pravděpodobnost (*Negative log-likehood*)

$$loss(\theta,(x,y)) = -\log(\mathrm{P}(\mathrm{Y}=\mathrm{y}|\mathrm{f}_\theta(x))) \tag{3.46}$$

- Kvadratická ztrátová funkce (*Squared loss*)

$$loss(\theta,(x,y)) = \|\mathrm{f}_\theta(x)-y\|^2 \tag{3.47}$$

- Absolutní ztrátová funkce (*Absolute loss*)

$$loss(\theta,(x,y)) = |\mathrm{f}_\theta(x)-y| \tag{3.48}$$

- Křížová entropie (*Cross-Entropy*)

$$loss(\theta,(x,y)) = -\mathrm{y}\cdot\log(\mathrm{f}_\theta(x)) - (1-y)\cdot\log(1-\mathrm{f}_\theta(x)) \tag{3.49}$$

- Střední kvadratická chyba (*MSE*)

$$loss(\theta,(x,y)) = \frac{1}{2}\|\mathrm{f}_\theta(x)-y\|^2 \tag{3.50}$$

- Odmocnina střední kvadratické chyby (*RMSE*)

$$loss(\theta,(x,y)) = \sqrt{\frac{1}{2}\|\mathrm{f}_\theta(x)-y\|^2} \tag{3.51}$$

4 Modely konvolučních neuronových sítí

V průběhu vývoje modelů segmentačních neuronových sítí se ustálily základní návrhové struktury, které je možné s různou mírou úspěchu použít pro učení. V rámci práce byly tyto základní struktury upraveny pro potřeby tohoto výzkumu. Jednalo se zejména o přizpůsobení výstupních struktur sítě. Základní modely sítí jsou vytvořeny pro klasifikaci jednotlivých obrázků z datového setu. Pro příklad je možné uvést dataset nazývaný MNIST (LeCun, a další, 2021) , konkrétně klasifikaci čísel. Při učení jsou síti předkládány obrázky jednotlivých číslic a výsledná kategorie, do které daný vzor spadá.

V případě zaměření této práce to je výrazně složitější. Práce je zaměřena na odhad hodnoty jednotlivých pixelů, kde hodnota pixelu je reprezentována 8 bitovým číslem. Konkrétní hodnota pixelu ve výsledném snímku reprezentuje normovanou vzdálenost[1] mezi kamerou a objektem. Dalo by se tedy říct, že neuronová síť musí při svém učení pro každý jednotlivý pixel vytvořit 256 kategorií, do které každý pixel bude zařazen. Výstupní vrstvy modelů byly přizpůsobeny tak, aby výstupní matice měla shodné rozměry jako matice vstupní. Z výše uvedených důvodů nebylo možné použít celé vytvořené modely, ale pouze jejich základní struktury, které byly více či méně upraveny pro potřeby učení. V některých případech byly základní struktury upraveny tak, že docházelo ke kombinaci struktur z více různých modelů. Nejčastěji bylo použito předávání příznaků mezi enkodérem a dekodérem, což je typické zejména pro síť *U-Net,* ovšem bylo to použito i u sítí, u kterých předávání příznaků typické není.

V této kapitole budou popsány zejména základní struktury jednotlivých modelů s tím, že budou vyznačeny úpravy, které byly na modelu provedeny.

4.1 Metody segmentace

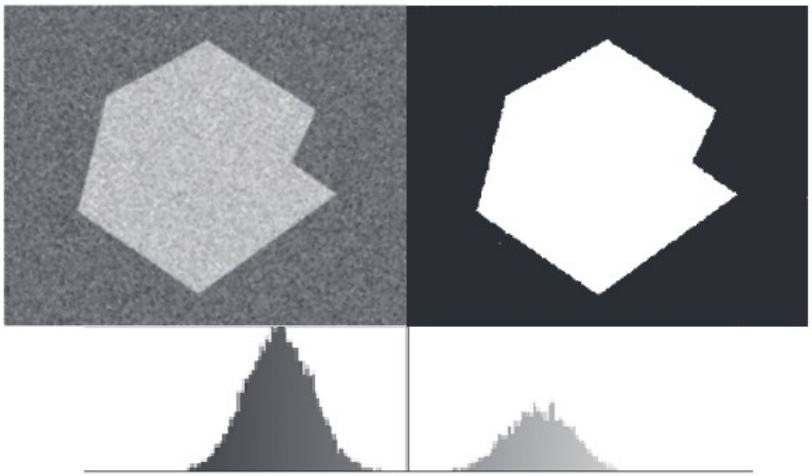
Sémantická segmentace obrazu slouží pro rozdělení vstupního snímku na dvě a více oblastí. Sítě jsou v tomto případě zaměřené na odhad různých oblastí v obraze pro jejich segmentaci, jako je například detekce silnice, chodců, okolí či automobilů ve vstupním snímku. Segmentační neuronové sítě používají skupinu metod pro klasifikaci jednotlivých

[1] Pro získání skutečné vzdálenosti je nutné normovanou vzdálenost převést. Při učení byly získané snímky z hloubkové kamery přepočítány. Minimální vzdálenost byla 0 mm, čemuž odpovídá hodnota pixelu = 0. Maximální vzdálenosti 10 m a dále byla přiřazena hodnota pixelu 255. Hodnoty mezi minimální a maximální hodnotou byly lineárně rozděleny.

pixelů. V této kapitole jsou uvedeny pouze základní informace o metodách a bližší informace je možné získat například v publikaci (Doughferty, 2009).

4.1.1 Prahování

Nejjednodušší metoda prahování, v anglické literatuře označovaná *Thresholding* je založena na hodnocení jasu každého pixelu v obraze. Metoda pracuje na principu hledání prahu v histogramu, pro který platí, že hodnoty menší, než práh jsou prohlášeny za pozadí, hodnoty větší, než práh jsou prohlášeny za popředí. Velice často ovšem vstupní snímek obsahuje oblasti s menším a vyšším jasem, tudíž není histogram homogenní a není možné použít globální práh a je nutné použít adaptivní prahování. Na Obrázek 35 je zobrazen vstupní snímek s impulsním šumem (vlevo nahoře) a jeho histogram (vlevo dole). Metoda provede nastavení správného prahu na základě dat z histogramu, čímž dojde k eliminaci pozadí a šumu (vpravo nahoře). Výsledkem je upravený histogram (vpravo dole).



Obrázek 35: Ukázka sémantické metody prahování. Zdroj: (Doughferty, 2009), Upraveno.

4.1.2 Regionální metody

Skupina metod v anglické literatuře označovaná *Region-Based methods,* jsou metody založené na zjišťování podobností pixelů v definované oblasti, jako je například hledání podobné barvy, podobného jasu či podobné textury. Dle určitého algoritmu jsou ve vstupním obraze rozmístěny inicializační pixely (semínka), a metoda postupně hledá podobné hodnoty pixelu v okolí inicializačního pixelu. Metoda iterativně opakuje detekci hodnot okolních pixelů, tj. detekovaný segment tzv. expanduje, než budou všechny pixely zařazeny do skupin. Při použití metod expandujícího segmentu ovšem není zaručeno, že při různých počátečních podmínkách (například jiná poloha inicializačního pixelu v obraze)

bude výsledný obraz identický. Výhodou ovšem je fakt, že metoda je odolná vůči šumu, který má impulsní charakteristiku.

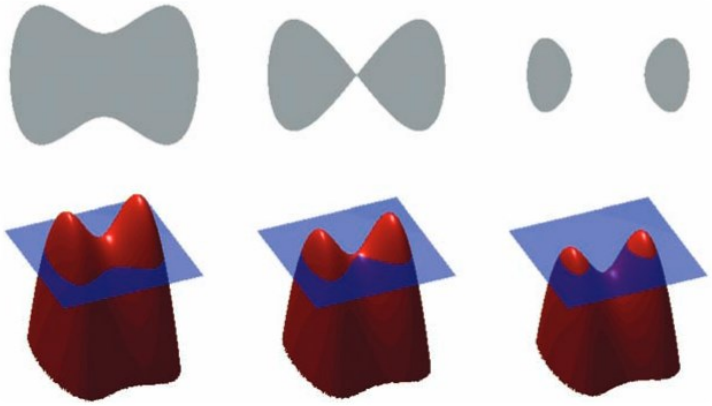
Obrázek 36: Regionální metoda: Vstupní snímek (vlevo), výsledek při použití lokálního adaptivního prahování (vpravo). (Zdroj: (Doughferty, 2009), Upraveno)

4.1.3 Metody založené na hranici

Metody v anglické literatuře označovaná „*Boundary-Based methods*“, jsou založeny na hledání náhlých nespojitostí v hodnotách pixelů. Skoková změna barvy či jasu obvykle signalizuje hranu, čehož tyto metody využívají. Problémem této metody může být například to, že používané detektory (například Canny, viz kapitola 3.1.7.1.2) mohou vytvářet přerušované nebo i falešné hranice.

4.1.4 Segmentace rozvodím

Metoda v anglické literatuře označovaná „*Watershed Segmentation methods*“, je metoda založená na postupném vyplňování obrazu „vodou“. Jas definován jako překážka, tj. čím vyšší jas, tím je vyšší překážka. Skupina pixelů s podobnou intenzitou je poté považována za hráz. Postupným vyplněním je vytvořena souvislá kontura, která je oddělena souvislou hranicí vyšších jasů jednotlivých pixelů. Princip metody je zobrazen na Obrázek 37.

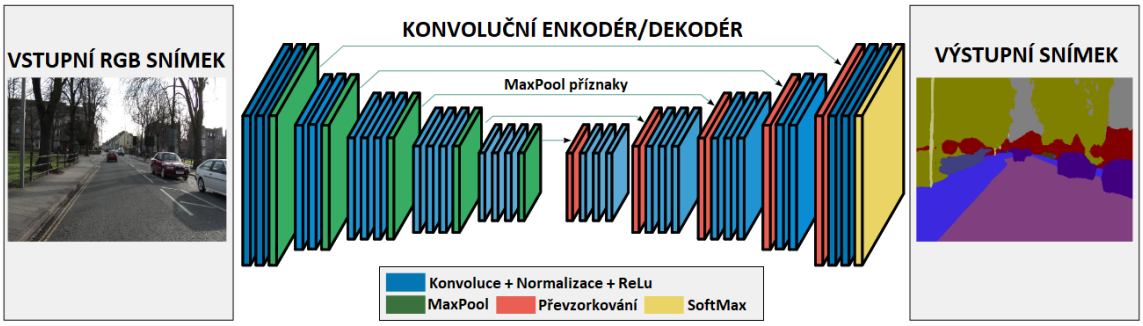


Obrázek 37: Ukázka segmentace rozvodím. Zdroj: (Doughferty, 2009)

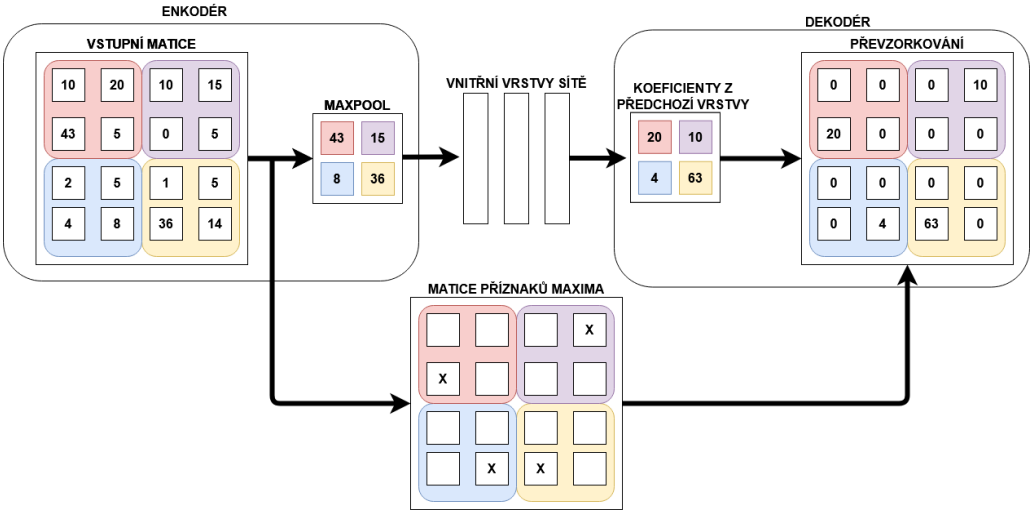
4.2 Vybrané modely neuronových sítí

4.2.1 SegNet

Neuronová síť *SegNet* (Badrinarayanan, a další, 2017) je založena na klasifikační síti *VGG16* a segmentuje obrázek pixel po pixelu. Základní struktura sítě je uvedena na Obrázek 38. Síť se skládá z kodéru a dekodéru. Blok kodéru se skládá ze 13 konvolučních vrstev, které jsou shodné s prvními 13 vrstvami mateřské *VGG16* sítě. Výhodou je, lze použít před trénovanou mateřskou *VGG16* síť a tím urychlit a optimalizovat výsledné učení. Pro snížení počtu parametrů jsou v síti *SegNet* odstraněny plně propojené vrstvy, které jsou například použity v neuronových sítích typu *FCN*. Model sítě *SegNet* je založen na využívání indexů získané z *MaxPooling* vrstvy, které jsou následně využívány při převzorkování v dekodéru. Princip předávání příznaků je zobrazen na Obrázek 39.



Obrázek 38: Struktura sítě SegNet (Zdroj: (Badrinarayanan, a další, 2017), Přeloženo)



Obrázek 39: Princip předání příznaků mezi ekodérem a dekodérem v síti SegNet (Zdroj: Vlastní)

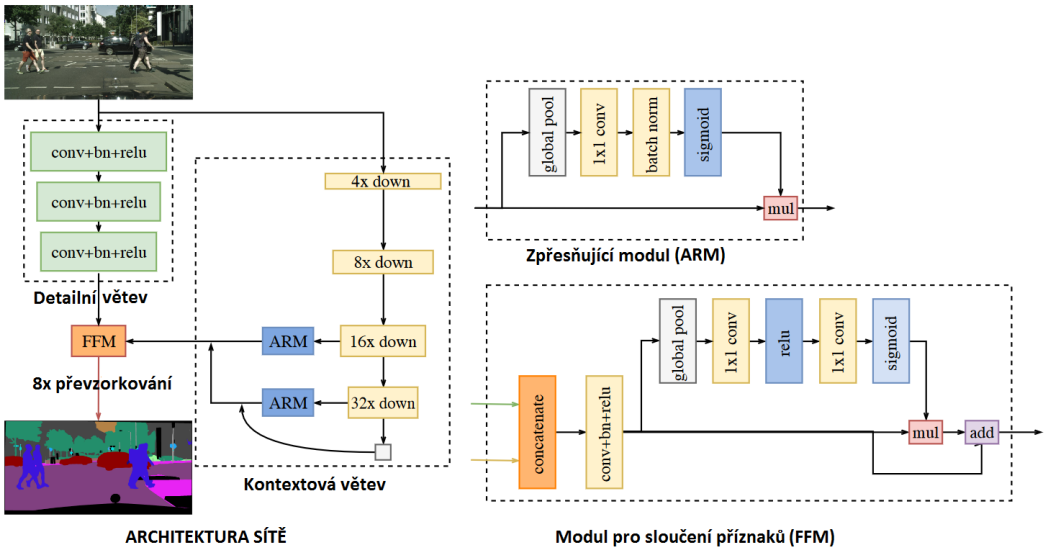
Podrobnější informace o neuronové síti *SegNet* lze nalézt v publikaci (Badrinarayanan, a další, 2017), (Chen, a další, 2019), (Sharma, a další, 2020) či (Ascar, a další, 2020).

4.2.2 BiSeNet

Síť BiSeNet (*Bilateral Segmentation Network*) je síť navržená pro lepší přesnost odhadu detailů a rychlejší zpracování výsledků (Yu, a další, 2018), (Yu, a další, 2021). Ve většině případů sémantické segmentace, jsou pro vyšší rychlost zpracování a menší složitost sítě obětovány detaily a dochází pouze k detekci velkých objektů, což ovšem vede k výraznému snížení přesnosti. Model sítě *BiSeNet* je navržen tak, že dochází k oddělenému zpracování sémantických i kontextových částí obrazu. Model je navržen tak, aby došlo ke zvýšení kvality sémantické segmentace bez zvýšení nároků na výpočetní výkon. V modelu je taktéž navržena řízená agregační vrstva, pomocí které je provedeno lepší vzájemné propojení. (Yu, a další, 2018)

Model sítě je založen na celkem dvou paralelních větvích. První větev, nazývaná detailní, je tvořena malým počtem konvolučních vrstev se širokými kanály. Touto větví jsou zachovány nízkoúrovňové detaily v obraze, které jsou posléze předány do modulu sloučení příznaků, který je ve výstupních vrstvách neuronové sítě. Větev se skládá ze tří konvolučních vrstev, které provádějí podvzorkování vstupního snímku. Výsledkem jsou dimenze obrazu, které jsou rovny $\frac{1}{8}$ dimenze vstupního snímku.

Druhá větev, nazývaná sémantická případně kontextová, je založena na extrakci sémantického kontextu v obraze. Sémantická větev obsahuje velké množství konvolučních vrstev, provádějící sémantickou segmentaci a podvzorkování. Návrh sémantické segmentace je proveden tak, aby výpočet byl jednoduchý díky snížení množství zpracovávaných kanálů.



Obrázek 40: Základní struktura sítě BiSeNet (Zdroj: (Yu, a další, 2018), Upraveno a přeloženo)

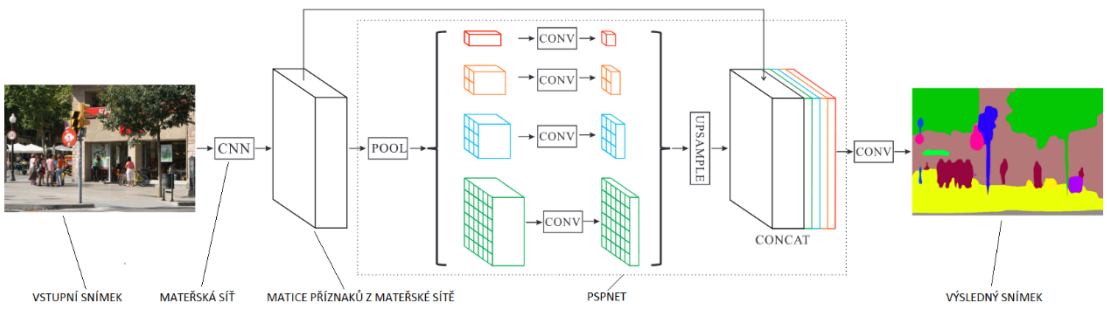
Na Obrázek 40 je zobrazena architektura sítě *BiSeNet*, kde jsou patrné základní komponenty sítě. Jak je vidět, síť obsahuje dvě větve – kontextovou větev a detailní větev.

Kontextová větev slouží k extrakci sémantických příznaků ze vstupního obrazu a lze použít velké množství základních architektur, jako je například *SegNet*, *ResNet*, *Xception* apod. Kontextová větev, obsahuje velké množství podvzorkování. Z architektury sítě je patrné, že její součástí je modul extrakce detailů (ARM), který obsahuje globální průměr, sloužící k zachování globálních detailů, následuje konvoluční vrstva, normalizace a aktivační funkce.

Výsledky ze sémantické a detailní větve mají rozdílné dimenze a nelze je jednoduše sečíst. Aby bylo možné tyto příznaky sloučit, je v modelu sítě přidán modul sloučení příznaků. Výstupem je sloučený snímek ze sémantické větve modelu s detaily z detailní větve modelu. Pro zachování dimenzí vstupního a výstupního snímku je posléze provedeno převzorkování na stejné rozměry.

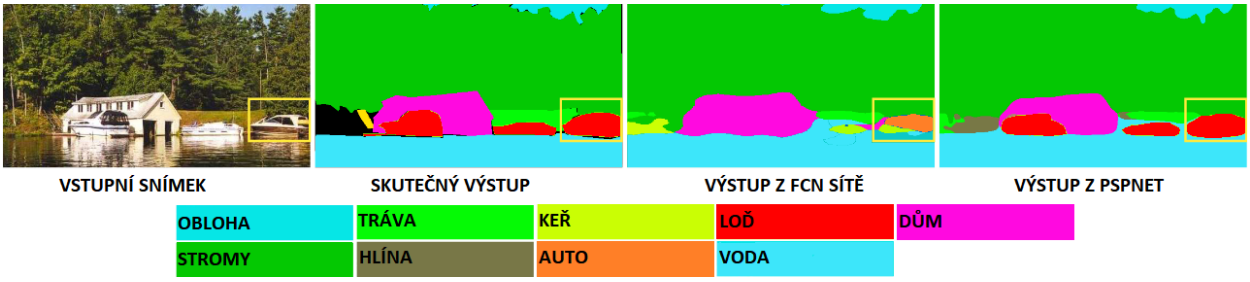
4.2.3 PSPNet

Síť *PSPNet* (Pyramidal Scene Parsing Network) (Zhao, a další, 2017) se řadí mezi sémantické segmentační sítě. Síť je založena na architektuře sítě *ResNet*, případně je možné použít *VGG-Net, Xception* a jiné základní struktury, nazývané jako mateřská síť. Mateřská síť slouží pro získání příznaků z obrazu, které jsou následně zpracovány v navazující struktuře sítě PSPNet. Tyto příznaky síti slouží pro rozpoznání hran a dalších důležitých příznaků v obraze. Tyto vzory jsou následně zpracovány v navázaných vrstvách. Tyto získané příznaky jsou posléze pyramidálně zpracovány v následujících strukturách. V principu nejsou používány jen příznaky z výstupních vrstev mateřské sítě, ale velice často jsou použity i výstupy z různých částí mateřské sítě.



Obrázek 41: Struktura sítě PSPNet (Zdroj: (Zhao, a další, 2017), upraveno)

Architektura je navržena tak, aby brala v potaz globální kontext obrazu, nikoliv jen okolní pixely jako je tomu například u sítě FCN. Porovnání výsledků FCN a PSPNet sítě je zobrazeno [2] na Obrázek 42



Obrázek 42: Porovnání sítě FCN a PSPNet (Zdroj: (Zhao, a další, 2017), upraveno)

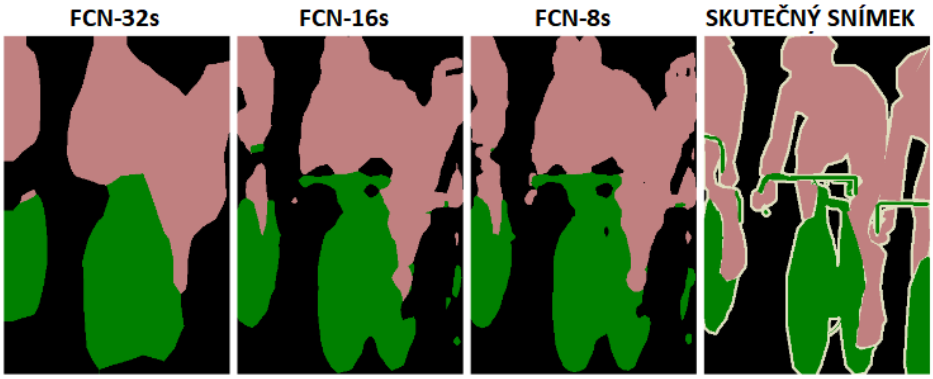
Z Obrázek 42 jsou patrné rozdíly ve výsledcích sítě *FCN* a *PSPNET*. Síť *FCN* ze segmentace nebyla schopna určit žádný z objektů označených jako loď. Místo toho dvě lodě určila jako dům či okolní trávu a jednu označila jako auto. Síť *PSPNet* bere v potaz globální okolí a tudíž objekty, které sousedí s vodou, označila jako loď.

Další podrobnosti o neuronové síti PSPNET lze nalézt například v těchto publikacích (Zhao, a další, 2017) , (Deng, a další, 2021).

4.2.4 FCN

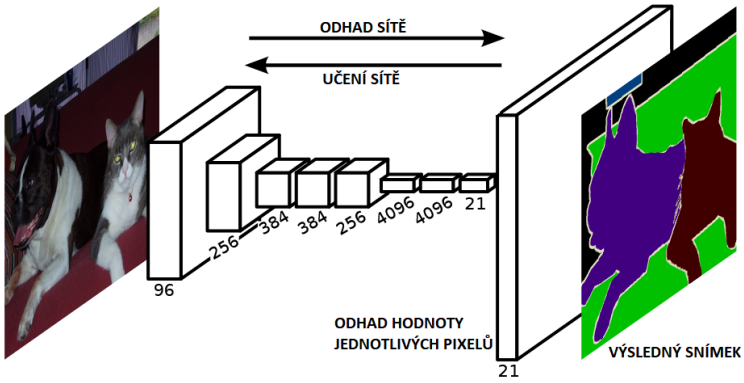
Síť *FCN* (*Fully Convolutional Network*) je plně konvoluční síť se zaměřením na sématickou segmentaci typu pixel po pixelu. Číslo udávané za sítí označuje pixelový krok v konvoluční vrstvě (tzv. *stride*). Čím je tato hodnota větší, tím je hrubší krok a menší rozlišení sítě. Ukázka výstupu variant sítě *FCN* je zobrazena na Obrázek 43. Z obrázku je patrné, že pro kvalitní segmentaci obrazu je nutné používat variantu sítě s pixelovým krokem (*stride*) maximálně 8. Hrubý odhad výsledku je v síti samozřejmě způsoben i tím, že ve výstupní části neuronové sítě dochází k převzorkování odhadu. V případě sítě FCN-8 je odhad sítě převzorkován celkem 8x, v případě *FCN-32* dokonce 32x. Pokud dojde ke srovnání, tak rozlišení sítě *FCN-32* je oproti síti *FCN-8* celkem 4x menší (Long, a další, 2016).

[2] Některé barvy objektů byly oproti originálu změněny pro lepší zobrazení výsledků.



Obrázek 43: Porovnání výsledků ze sítě typu FCN-32, FCN-16 a FCN-8 (Zdroj: (Long, a další, 2016), upraveno a přeloženo)

Na Obrázek 44 je zobrazena základní struktura sítě *FCN*. Z grafu je vidět že dochází k postupné kompresi vstupního snímku na menší matice, až dojde k vytvoření matice o rozměru $N \times N \times 4096$. Výsledný vektor je za využití plně propojené vrstvy a aktivační funkce, nejčastěji *softmax*, zredukován na celkem 21 kategorií, z nichž každá reprezentuje danou kategorii pixelu (v příkladu uvedeném na Obrázek 44). Tato matice je posléze převzorkována dle použité sítě (pro FCN-8 celkem 8x) a dojde k vytvoření matice o definovaném rozměru $(8 \cdot N) \times (8 \cdot N) \times C$, který už reprezentuje segmentovaný snímek.



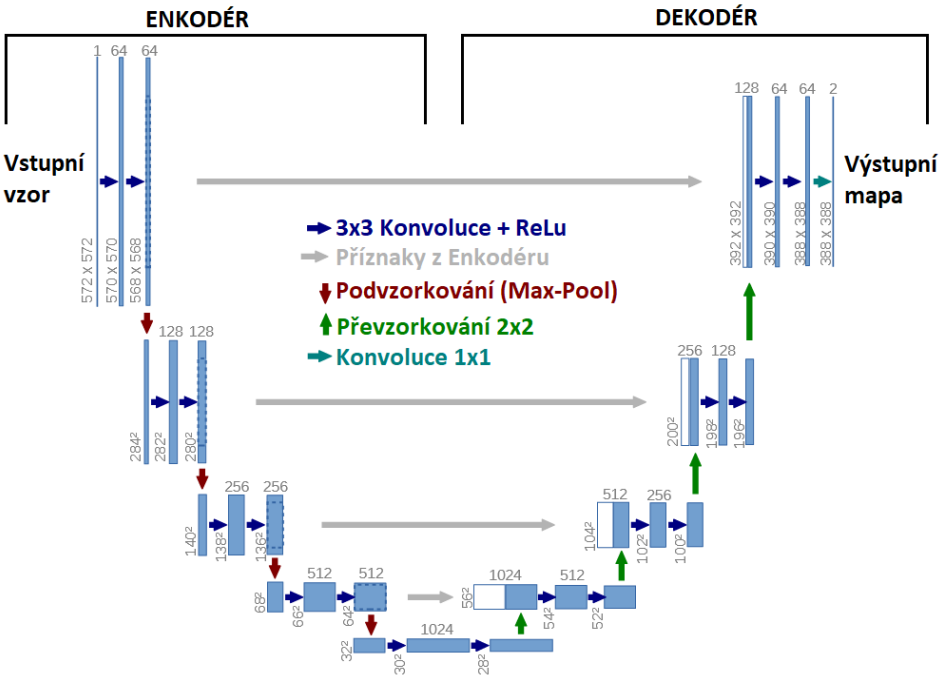
Obrázek 44: Ukázka zpracování FCN sítě (Zdroj: (Long, a další, 2016), upraveno a přeloženo)

Podrobnější informace o aplikaci neuronové sítě typu FCN lze nalézt například v těchto publikacích (Long, a další, 2016), (Fu, a další, 2018), (Liu, a další, 2018) či (Zhao, a další, 2020)

4.2.5 U-Net

Neuronová síť U-Net je pojmenována podle struktury sítě, která připomíná písmeno U. Síť U-Net je možné rozdělit na dvě části. Levá část představuje typickou konvoluční síť, ve které dochází k postupnému podvzorkování vstupního vzoru až na vektor koeficientů (nejčastěji 1x1024 koeficientů). Tato část se občas označuje jako

enkodér. Typická síť představuje pět U-Net bloků, u nichž každý obsahuje dvě konvoluční vrstvy s jádrem o velikosti 3x3 pixely, vrstvu *ReLu*, dále sdružovací vrstvu s funkcí maxima s velikostí jádra 2x2 a krokem 2, což vytváří podvzorkování vstupních dat. Druhá část sítě představuje dekodér, kde dochází k převzorkování a konvoluci. Dekodér bere v potaz matici příznaků z enkodéru. Základní odlišností od ostatních sítí a pravděpodobně její největší síla, je velké množství spojení mezi vstupními vrstvami (enkodéry) a výstupními vrstvami (dekodéry). Příznaky extrahované z enkodéru na stejné úrovni jsou předány do Dekodéru, čímž se zajistí další parametr při učení. Dalším základním prvkem sítě je vrstva normalizace umístěná před aktivační funkcí. Jako aktivační funkce je nejčastěji používána funkce *TanH, Sigmoida, Soft-Max* či *ReLu*. Ukázka typické struktury sítě U-Net je uvedena na Obrázek 45.



Obrázek 45: Struktura sítě U-Net. (Zdroj: (Ronneberger, a další, 2015), upraveno a přeloženo)

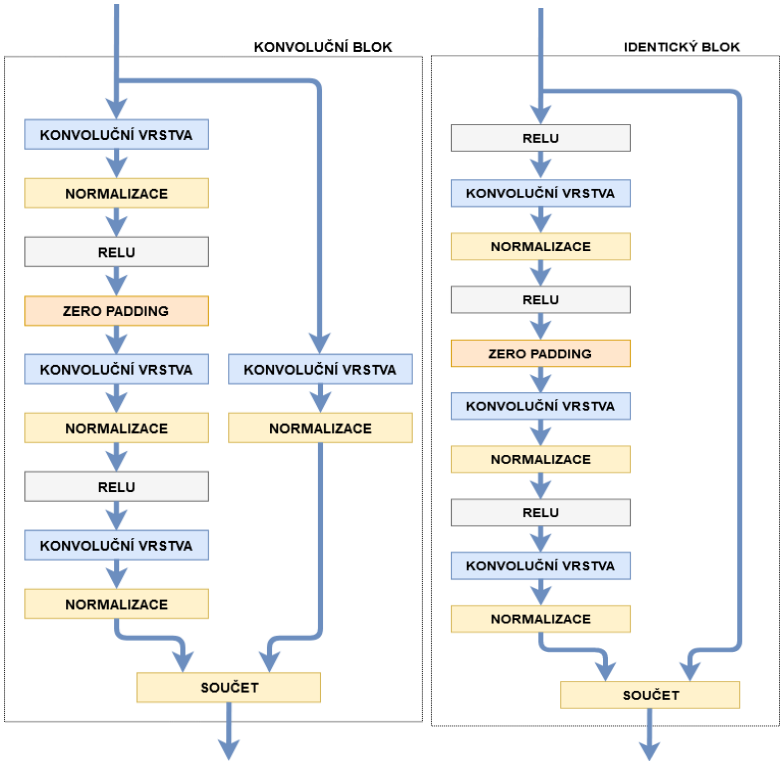
Podrobnější informace o neuronové síti typu *U-Net* je možné najít v publikaci (Ronneberger, a další, 2015) , (Zheng, a další, 2020), (Satya, a další, 2020), (Woo, a další, 2021).

4.2.6 ResNet

Název sítě *ResNet* vychází ze základní architektury sítě – Reziduální síť. Tato síť obsahuje velké množství tzv. reziduálních bloků. Základní myšlenkou *ResNet* sítě je reziduum, což je spojení vrstev o více než jednu vrstvu kupředu. V síti tedy není problém spojit vnitřní pátou vrstvu s vnitřní dvanáctou vrstvou. Výstupy z těchto sítí jsou jednoduše

sečteny. Tyto reziduální vrstvy umožňují zmírnit problém zanikajícího gradientu. V principu je možné výstupy z těchto vnitřních vrstev vyvést mimo model a použít v navazující síti.

Nejčastěji používané architektury jsou *ResNet-12*, *ResNet-18*, *ResNet-50*, *Resnet-101* a *Resnet-200*, kde číslice označuje počet residuálních bloků v modelu. Tyto bloky lze rozdělit na dva druhy.



Obrázek 46: Stavební bloky ResNet sítě: Konvoluční blok (levá část), Identický blok (pravá část) (Zdroj: Vlastní)

Konvoluční blok, který je zobrazen na Obrázek 46, levá část. Identický blok je zobrazen na Obrázek 46, pravá část. Tyto bloky jsou univerzální, přičemž počet a kombinace těchto bloků určuje typ ResNet sítě. Rozměry vstupních a výstupních matic jsou zpravidla stejné a změny rozměrů – podvzorkování či převzorkování je prováděno mimo tyto základní stavební bloky.

Základní architektura sítě ResNet je vždy stejná, liší se jen počtem sekcí v blocích. Každá sekce je tvořena sledem konvolučního a identického bloku. Síť byla pro potřeby této práce rozdělena na vstupní vrstvy sítě, dále 4 bloky a výstupní vrstvy. Blok č. 1 a č. 4 je tvořen jedním konvolučním a jedním identickým blokem a jsou stejné pro všechny varianty sítě. Varianta sítě je určena vnitřními bloky č. 2 a č. 3. Tyto bloky jsou tvořeny jedním konvolučním blokem a identickými bloky, jejichž počet je dle varianty sítě a je uveden v Tabulka 1. Výstupy z jednotlivých bloků

Tabulka 1: Počet bloků pro varianty sítě ResNet

Typ sítě	Počet bloků	
	Sekce 1	Sekce 2
ResNet-12	1	2
ResNet-18	2	3
ResNet-50	2	10
ResNet-101	3	22
ResNet-200	8	50

jsou vyvedeny mimo model a v případě potřeby je možné využít je jako základ pro další modely sítí, jako je například *DenseNet*, *RefineNet*, *PSPNet*, *SegNet*, *BiSeNet* apod. Tyto výše uvedené modely byly v rámci této práce zkoumány. Nejčastěji pro tyto modely byla jako mateřská síť použita síť *ResNet-50* či *Resnet-101*. Podrobnější informace o aplikaci sítí typu *ResNet* jsou uvedeny například v (He, a další, 2016), (Budhiman, a další, 2019) či (Erwandi, a další, 2020).

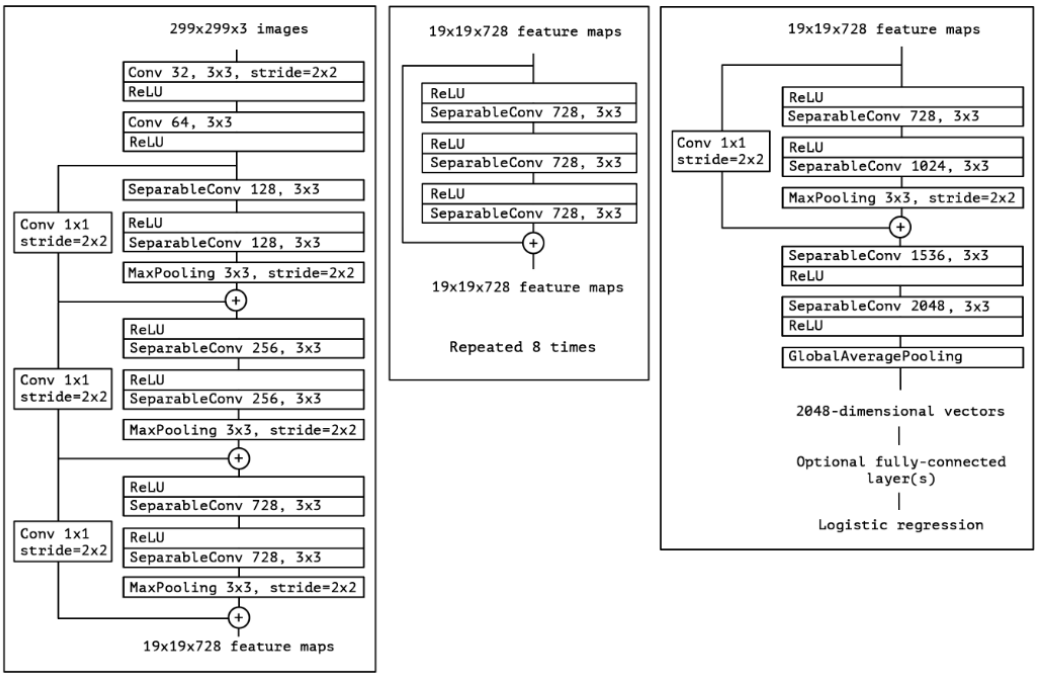
4.2.7 Xception

Síť *Xception* byla vyvinuta vývojáři ze společnosti Google (Chollet, 2017). Jedná se o hlubokou konvoluční neuronovou síť. Síť implementuje tzv. hloubkově separovatelné konvoluce (*Depthwise Separable Convolutions*), které jsou vzhledem ke klasickým konvolucím efektivnější z hlediska výpočetního času (Fabien, 2021). *Xception* síť funguje na principu separovatelné konvoluce a „zkratek“ mezi konvolučními bloky typických pro sítě *ResNet*. Při bližším porovnání je vidět výrazné podobnosti se sítí *ResNet*, základní odlišnost je jen v kombinaci separovatelné a standardní konvoluce.

Základní síť se dělí celkem na tři části. První část je vstupní konvoluční blok, který obsahuje bloky jak separovatelné konvoluce, tak standardní konvoluce v zapojení, které je typické pro sítě *ResNet*. Ukázka vstupního konvolučního bloku je na Obrázek 47, levá část. Vstupní blok se od konvolučního bloku *ResNet* sítě liší tím, že příznaky z vyšších bloků jsou postupně předávány hlouběji do sítě. Rozdíl mezi *ResNet* konvolučním blokem a *Xception* konvolučním blokem lze vypozorovat z Obrázek 46 levá část a Obrázek 47 levá část.

Vstupní blok je následován opakujícím se identickým blokem, který je velice podobný identickému bloku v sítích *ResNet*. Základní rozdíly je možné vypozorovat z Obrázek 46 pravá část a Obrázek 47 prostřední část. Tento střední identický blok se ve struktuře sítě nejčastěji osmkrát opakuje. Základní rozdíl oproti *ResNet* identickému bloku je použití separovatelné konvoluce namísto klasické konvoluce.

Síť je ukončena koncovým blokem, který se skládá z konvolučního bloku následovaného výstupními vrstvami. Tyto vrstvy slouží k přizpůsobení výstupních matic pro daný řešený problém. Ukázka propojení výstupního bloku je zobrazena na Obrázek 47 pravá část.

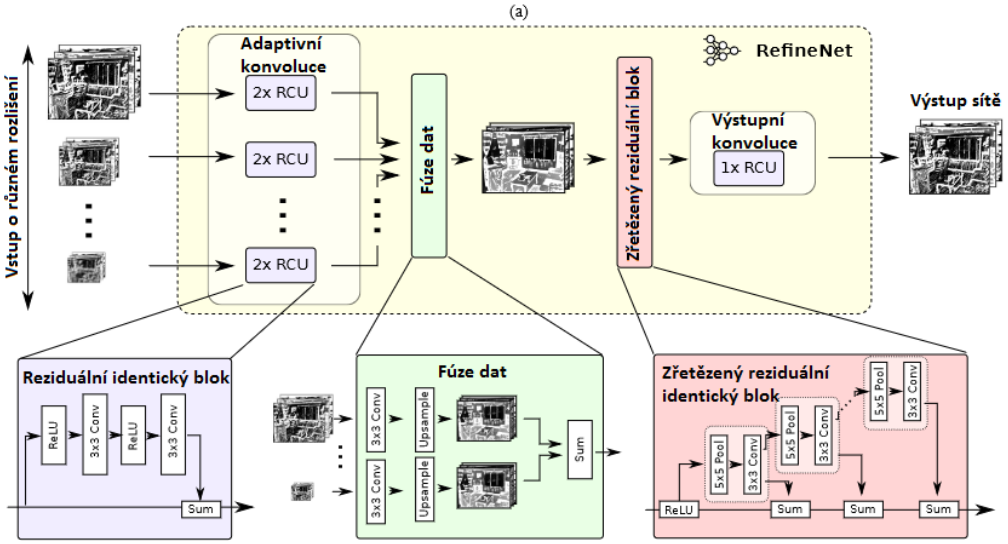


Obrázek 47: Struktura sítě Xception. Konvoluční blok (vlevo), identický blok (uprostřed), výstupní blok (vpravo). (Zdroj: (Chollet, 2017))

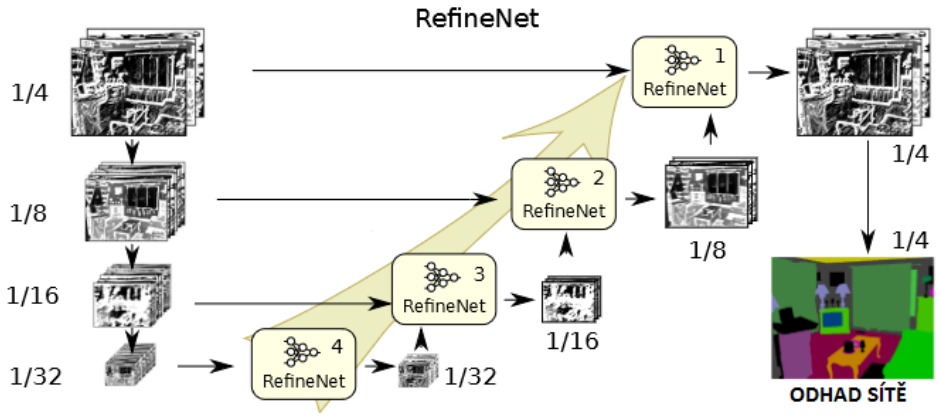
Další informace o aplikaci neuronové sítě *Xception* v různých odvětvích lze nalézt v (Chollet, 2017), (Urmee, a další, 2019), (Wu, a další, 2020) či (Rismiyati, a další, 2020).

4.2.8 RefineNet

Neuronová síť *RefineNet* byla navržena a publikována autory (Guosheng, a další, 2016) v roce 2016. Základní model sítě byl inspirován neuronovou sítí *ResNet* a implementuje základní bloky z výše uvedené sítě. Konkrétně se jedná o implementaci identického bloku, jak v základní formě, tak v upravené, zřetězené formě, jak je patrné ze struktury sítě zobrazené na Obrázek 48. Základním stavebním prvkem sítě *RefineNet* je blok uvedený na Obrázek 49. Tyto základní stavební bloky jsou využity v paralelním zpracování vstupních snímků, přičemž každá větev má jinou úroveň podvzorkování. Autory navržená neuronová síť má celkem čtyři úrovně podvzorkování – $\frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}$ původního snímku. V rámci těchto paralelních větví jsou přenášeny jiné detaily ze vstupního snímku. Všechny získané příznaky z různých úrovní podvzorkování jsou následně sloučeny a předány do hlubších vrstev neuronové sítě. Tento princip dle výzkumu autorů výrazně zvyšuje přesnost určení objektů sémantické segmentace.



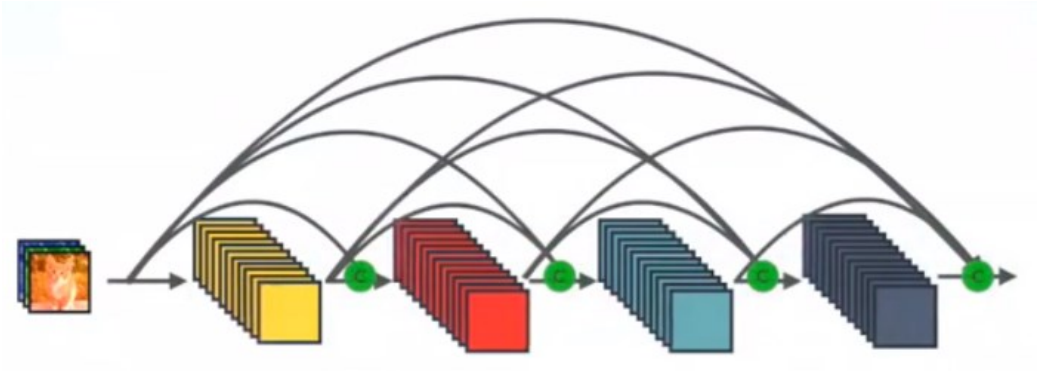
Obrázek 48: Struktura sítě RefineNet (Zdroj: (Guosheng, a další, 2016), Přeloženo a upraveno)



Obrázek 49: Ukázka zapojení základního bloku RefineNet (Zdroj: (Guosheng, a další, 2016), Přeloženo a upraveno)

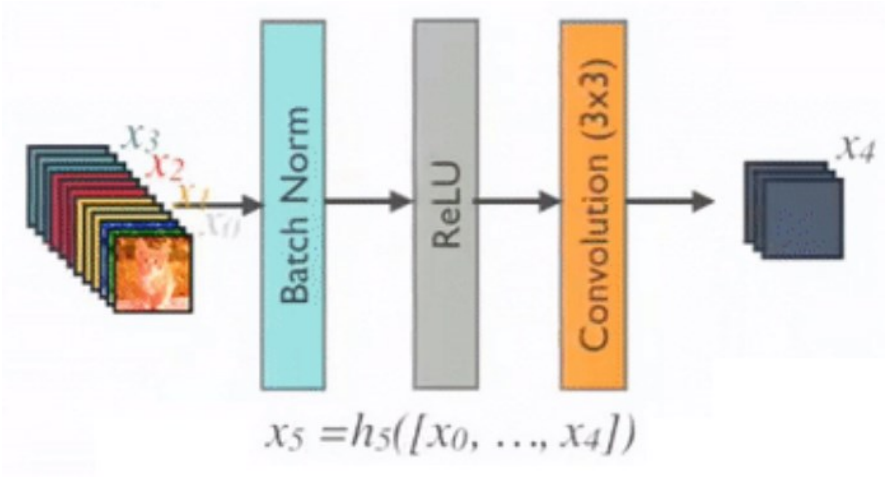
4.2.9 DenseNet

DenseNet je konvoluční neuronová síť, příbuzná síti *ResNet*. Síť využívá standardní konvoluční bloky a navíc má tzv. „*Dense*“ blok, který používá tzv. *channel-wise* spojení, které si lze představit jako přemostění jedné konvoluční vrstvy. Ve výsledku je vstup z předcházející vrstvy použít ve vrstvě následující. Tento princip byl již dříve podobně použit v neuronových sítích typu *ResNet* a byl pojmenován jako identický blok. Autoři metody (Huang, a další, 2017) navíc do sítě aplikovali další příznaky typu identického bloku. Tj. dochází zde k přenosu příznaků ze vstupu první konvoluční vrstvy nejen do následující, jak je tomu u *ResNet* sítě, ale i do hlubších vrstev. Počet těchto spojení je dán parametrem nazývaným „*growth-rate*“, případně *hloubka propojení vrstev*. Ukázka těchto spojení je zobrazena na Obrázek 50. Obecně lze konstatovat, že každá vrstva obdrží kolektivní příznaky z předcházejících vrstev.



Obrázek 50: DenseNet stavební blok (Zdroj: (Tsang, 2018), Upraveno)

Pro přenos příznaků mezi jednotlivými *Dense* bloky slouží tranzitní vrstva, která provádí pouze normalizaci, implementaci aktivační funkce (nejčastěji funkci *ReLu*) a konvoluční vrstvu. Její struktura je zobrazena na Obrázek 51. Jedná se vlastně jen o přizpůsobení dimenzí pro následující *Dense* bloky. *DenseNet* síť není většinou použita jako holá síť, ale implementuje nějakou z mateřských sítí. V případě této práce byla použita mateřská síť *Resnet-101* včetně extrakce příznaků z vnitřních vrstev mateřské sítě, které byly postupně přidávány do sítě, jako další zdroj příznaků.



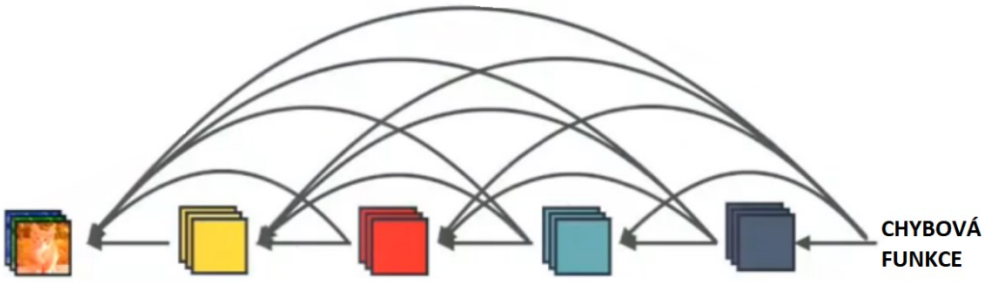
Obrázek 51: DenseNet – Transitní vrstva. (Zdroj: (Tsang, 2018), Upraveno)

Počet *Dense* bloků závisí na konkrétní složitosti sítě. V této práci byla použita hloubka propojení vrstev 5, tj. příznaky ze vstupu Dense bloku jsou předávány do následujících pěti vrstev. Počet Dense bloků závisí na struktuře a typu sítě. Nejčastější použité varianty jsou uvedeny v Tabulka 2.

Tabulka 2: Počet bloků pro varianty sítě DenseNet

Typ sítě	Počet bloků			
	Sekce 1	**Sekce 2**	**Sekce 3**	**Sekce 4**
DenseNet-121	6	12	24	16
DenseNet-169	6	12	32	32
DenseNet-201	6	12	48	16
DenseNet-300	6	12	64	48

Síť ve své konfiguraci přináší výhodu rychlejšího šíření chybové funkce skrz celou navrženou síť, což způsobuje rychlejší a kvalitnější učení sítě ve srovnání se sítí *ResNet*. Veškeré podrobnosti a grafy ohledně detailní struktury sítě je možné nalézt v publikaci autorů (Huang, a další, 2017).



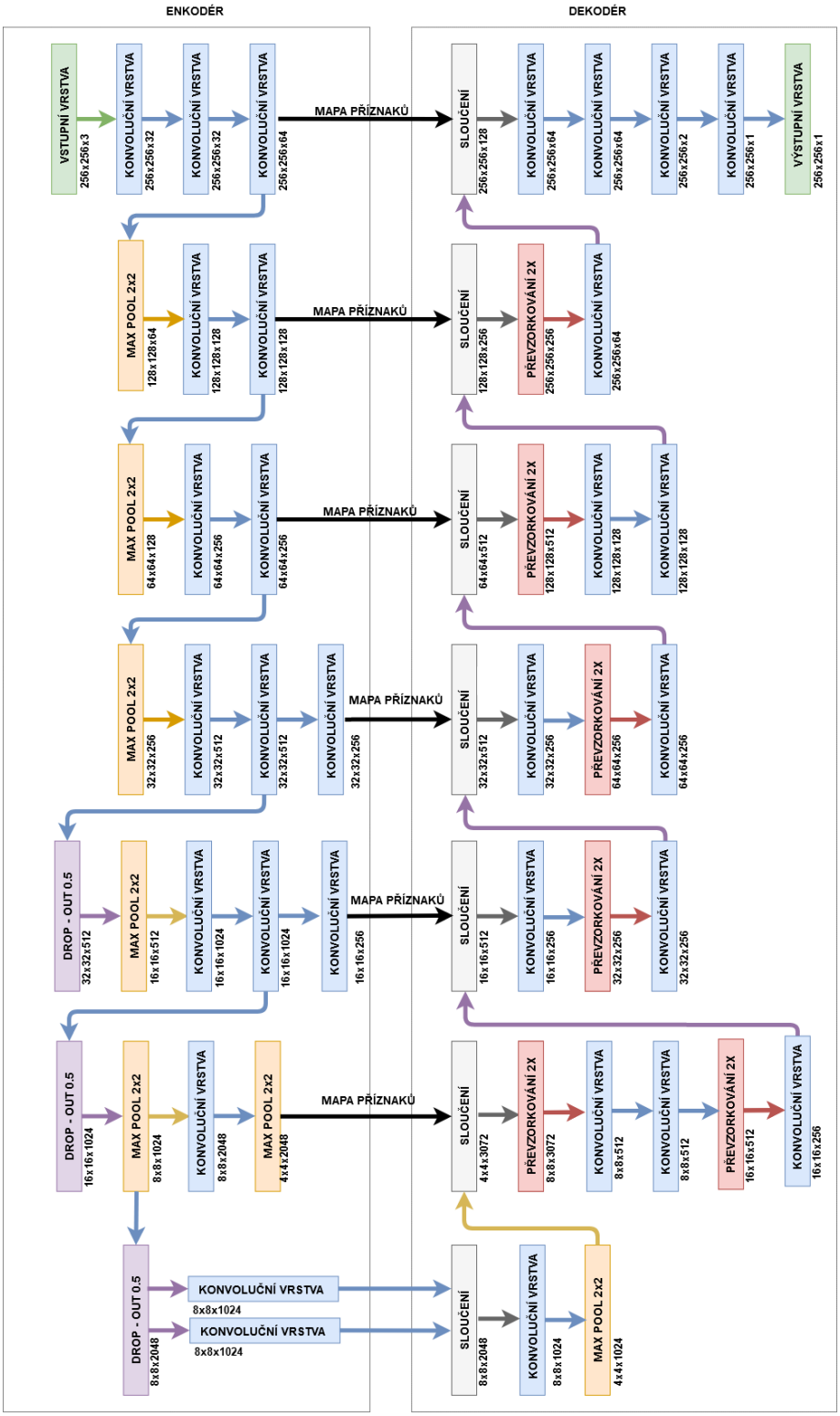
Obrázek 52: Zpětné šíření chyby v síti DenseNet (Zdroj: (Tsang, 2018), Upraveno)

4.2.10 Unet3

V rámci výzkumu byla autorem práce navržena rozšířená architektura sítě nazvaná *U-Net3*. Architektura přímo vychází z architektury *U-Net*, která vyšla při analýze aktuálních architektur sítí nejlépe. Základní tvar sítě do písmena U byl zachován, ale byly přidány konvoluční vrstvy, zejména na výstupu pro zvýšení hloubky sítě. Další výraznou modifikací je nepřímé předávání příznaků za využití jednoduché konvoluční vrstvy v mapě příznaků, podobné konvolučnímu bloku v síti *Xception* či *ResNet.*

Síť je navržena jako sedmi-úrovňová konvoluční síť, přičemž standardní architektura neuronové sítě U-Net je pouze pěti-úrovňová. Nejhlubší vrstva sítě má po sloučení ze dvou paralelních konvolučních vrstev rozměry 8×8×2048. Sedmi úrovňová architektura sítě samozřejmě přináší výrazně větší výpočetní náklady, než v případě standardní používané architektury *U-Net*.

Předávání příznaků mezi enkodérem a dekodérem je v horních třech vrstvách prováděno shodně jako u architektury U-Net, tudíž bez jakýchkoliv modifikací rovnou z konvoluční vrstvy. V hlubších vrstvách, tj. vrstvách č. 4 a č. 5, jsou příznaky předávány přes konvoluční vrstvu, která poskytuje možnost dotvoření detailů, které jsou dále předány do dekodéru.



Obrázek 53: Nově vytvořená architektura sítě U-NET3 (Zdroj: Vlastní)

V rámci celé architektury sítě v bloku enkodéru jsou prováděny operace *MaxPool* s krokem rovným dvěma, což poskytuje zmenšení rozměru aktuální matice o polovinu. Dle názoru autora práce, větší krok funkce *MaxPool* může způsobit ztrátu detailů v obraze, což je v případě odhadu hloubkové mapy nežádoucí. Samozřejmě je potřeba taktéž brát v potaz vstupní dimenze obrazu. V případě této práce byl použit vstupní snímek s rozměry

256×256×3, u kterého by velký krok byl nežádoucí, a mohlo by dojít ke ztrátě detailů potřebných pro odhad hloubkové mapy.

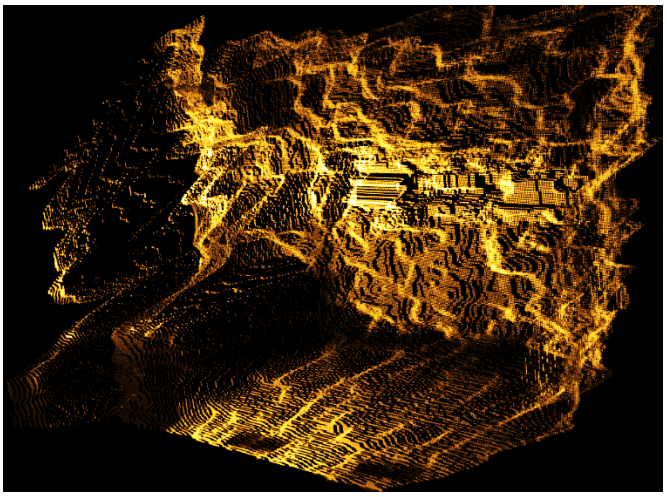
Sekce dekodéru je téměř totožná s původní architekturou *U-Net*. V dekodéru dochází k převzorkování a sloučení mapy příznaků získaných z enkodéru. Dále zde dochází k dvojnásobnému převzorkování původní matice. Takto velké převzorkování vyplývá z architektury enkodéru a taktéž ze symetrické struktury sítě *U-Net*.

Rozměr konvolučních vrstev v enkodéru i dekodéru je vždy v rozměru o násobcích 2^n a to zejména z praktických důvodů při slučování vrstev. Podobným způsobem je tento princip aplikován i u jiných architektur neuronových sítí.

V rámci celé architektury byla použita aktivační funkce *ReLU,* až na aktivační funkci ve výstupní vrstvě, kde byla použita *Sigmoida*. Důvod byl čistě funkční. Pokud byla použita odlišná aktivační funkce než *ReLu*, například *TanH, Sigmoida, či SoftSign*, došlo vždy k zablokování v lokálním minimu, které se projevovalo jako veskrze monolitická šedá plocha, u některých sítí se objevily artefakty, které ovšem nekorespondovaly s obrysy objektů. Tento problém se projevil u všech testovaných sítí. Z výše uvedených důvodů byl podobný princip (tj. aktivační funkce *Sigmoida* ve výstupní vrstvě, aktivační funkce *ReLU* v ostatních vrstvách) použit u všech testovaných architektur sítí. Důvod proč dochází ve všech testech i různých architekturách k zaseknutí v lokálním minimu, nebyl přesně odhalen, byla pouze stanovena domněnka, že je to povahou aktivační funkce, které mají lineární oblast pouze v části svého rozsahu a posléze pomalu konvergují k maximální hodnotě, přičemž aktivační funkce *ReLU* je lineární bez omezené maximální hodnoty.

5 Mračno bodů

Mračno bodů, v zahraniční literatuře nazývané PointCloud, poskytuje možnost, jak uchovat prostorový snímek získaný speciální hloubkovou kamerou, jako je například Intel D435i, která byla pro měření dat pro tuto práci použita. Ve skutečnosti kamera neposkytuje přímo mračno bodů, avšak generuje dva snímky. První snímek je standardní RGB a druhý je tzv. hloubkový snímek, poskytující informaci o třetím rozměru. Za využití speciálních metod je možné získané snímky sloučit do jednoho snímku a vznikne mračno bodů. Na Obrázek 54 je zobrazeno surové mračno bodů. Pro lepší přehlednost je obrázek vykreslen bez barvy jednotlivých pixelů.



Obrázek 54: Vytvořené mračno bodů (Zdroj: Vlastní)

Při tvorbě mračna bodů je nutné ovšem počítat s tím, že snímač RGB kamery a snímač hloubkové kamery není většinou umístěn na stejném místě, ale jsou vůči sobě posunuty. Dalším problémem je to, že RGB kamera má vlastní FOV a hloubková kamera má taktéž vlastní FOV, přičemž tyto FOV nemusí být a zpravidla ani nejsou stejné. Podrobné metody, jak provést zarovnání získaných snímků, jsou detailně popsány v kapitole 7.3.

Body získané hloubkovou kamerou jsou rozděleny do dvou kategorií. První kategorie je tzv. organizované mračno bodů. Organizované mračno bodů se svou strukturou podobá matici, kde jsou jednotlivé body rozděleny do řádků a sloupců. Tento typ je získán převážně kamerami typu *Time of Flight* (ToF) či stereokamerami. Výhodou je informace o okolních pixelech a jejich vzájemný vztah. Výsledkem je rychlejší a jednodušší vyhledávání v okolních bodech.

Druhou kategorie jsou tzv. neorganizovaná mračna bodů. V neorganizovaných mračnech bodů vzniká problém s určením nejbližšího bodu či okolí bodu p_q. Velice zde záleží na zadané hodnotě *d,* která určuje průměr abstraktní koule v 3D prostoru, ve kterém se nachází nejbližší sousední body bodu p_q. Všechny body v okolí bodu p_q musí

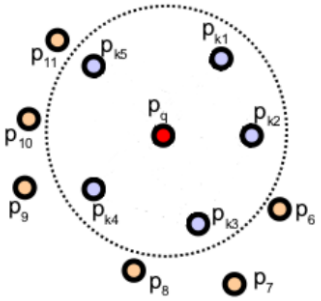
splňovat podmínku danou rovnicí (5.1). Body splňující tuto podmínku jsou na Obrázek 55 zobrazeny uvnitř kružnice.

$$\left|p_q - p_x\right| \leq \frac{d}{2} \tag{5.1}$$

Kde:

- p_q je centrální bod
- p_x je testovaný bod
- d je průměr abstraktní koule ve 3D prostoru

Pro další zpracování je většinou nezbytné provést rekonstrukci povrchu. Při rekonstrukci povrchu dochází k aproximaci skupin bodů do trojúhelníkové či mnohoúhelníkové sítě. Nejčastěji se pro aproximaci používá Delaunayho triangulace. Bližší informace lze získat například v publikaci (Delaunay, 1934) či (Maur, 2002).



Obrázek 55: Reprezentace sousedních bodů bodu p_a (Zdroj: (Rusu, a další, 2011))

5.1 Klíčové body

Kamera D435i snímá scénu s rozlišením až 1280×720. Za předpokladu zpracování jediného snímku, se jedná o 921 600 bodů, které je nutné zpracovat při každé operaci s mračnem bodů. Časová náročnost zpracování jednoho mračna bodů lze stanovit proměnnou $O(n)$, kde n je počet bodů daného mračna bodů. V případě, že dochází při zpracování jednoho bodu k porovnání s k okolními body v jeho sousedství, extrémně narůstá časová složitost výpočtu $O(k \cdot n)$. Pro snížení množství těchto redundantních bodů a se používají specializované metody, ze kterých jsou extrahovány tzv. klíčové body. Klíčové body jsou dostatečně charakteristické, aby pokryly oblasti původních bodů.

5.2 Podvzorkování

Metoda podvzorkování je jedna z nejjednodušších metod snížení redundance a taktéž výraznému snížení výpočetní náročnosti. Při podvzorkování dochází k překrytí odstraněných pixelů mřížkou z nových bodů s pravidelným vzorkováním. Mřížka je sestavena tak, že zachovává původní tvar i přesnost. Tento typ podvzorkování se občas v odborné literatuře nazývá voxelizace. Voxel je v podstatě 3D pixel. Metoda

podvzorkování pracuje tak, že se mračno bodů rozdělí do velkého množství oblastí tvaru krychle s požadovaným rozlišením. Každá tato vytvořená oblast pokrývá několik bodů a požadavkem je, aby v dané oblasti zůstal jeden jediný pixel (voxel). Po eliminaci nadbytečných bodů je vypočítán centroid oblasti, což je bod, jehož souřadnice jsou střední hodnoty všech bodů patřících do voxelu.

5.3 Lokální Deskriptory

Deskriptor je datová struktura nesoucí informaci o geometrickém okolí zkoumaného bodu. Hlavní myšlenkou je jednoznačně určit bod napříč mračnem bodů nezávisle na šumu, rotaci, translaci či transformaci. Některé deskriptory zaznamenávají i informaci o objektu, jako je například pozice kamery.

Pro určení optimálních deskriptorů je nutné určit následující kritéria:

- Odolnost vůči šumu – nízký vliv šumu, vzniklého během měření, na odhad deskriptorů
- Odolnost vůči transformaci – Konzistentní vzdálenost mezi body při použití rotačních či translačních transformací
- Konzistentnost při změně měřítka – minimální odlišnost určení deskriptoru při změně vzorkování mračna bodů, při změně měřítka musí být výsledky stejné či velice podobné

5.3.1 Odhad povrchových normál v mračnu bodů

Povrchová normála je základní a jednoduchý deskriptor. Normála je v prostoru vektor, který je kolmý na rovinu. Vektor určující směr normály je označován jako normálový vektor. Vytvoření normálového vektoru v konkrétním je v případě geometrického povrchu velice triviální – stačí vytvořit vektor kolmý na tento povrch. V případě práce s mračnem bodů a také důsledku toho, že body jsou reprezentované na reálném povrchu, existují dvě možnosti, jak vytvořit normálový vektor.

1) Tvorba rekonstruovaného povrchu ze zadaného mračna bodů za využití metod rekonstrukce povrchu. Normály jsou posléze určeny z rekonstruovaného povrchu

2) Využití aproximačních metod k odvození normál přímo z mračna bodů

V praktickém využití se nejčastěji používá metoda aproximace z důvodu, že první metoda (rekonstrukce povrchu) je časově náročná. Nejjednodušší způsob odhadu normály

je využití rovinné tečny k ploše. Plocha je posléze daná bodem x a normálovým vektorem $\vec{n}$. Algoritmus odhadu normálového vektoru pro daný bod p_i je určen rovnicemi (5.2) až (5.5)

$$d_i = (p_i - x) \cdot \vec{n} \tag{5.2}$$

$$x = \bar{\mathrm{p}} = \frac{1}{m}\sum_{i=1}^{m} p_i \tag{5.3}$$

Pokud dojde ke stanovení, že $d_i = 0$, dojde k určení řešení pro $\vec{n}$, které je dané analýzou vlastních čísel a kovarianční matice C, vyjádřené rovnicí (5.4). (Poi21)

$$\begin{gathered} C = \frac{1}{m}\sum_{i=1}^{m}(p_i - \bar{p}) \cdot (p_i - \bar{p})^T \\ C \cdot \overrightarrow{v_j} = \lambda_j \cdot \overrightarrow{v_j} \\ j \in \{0,1,2\} \end{gathered} \tag{5.4}$$

Kde:

- d_i je vzdálenost bodu p_i od plochy
- $\vec{n}$ je normálový vektor
- m je počet bodů v okolí bodu p_i
- $\bar{\mathrm{p}}$ je 3D centroid nejbližších sousedů testovaného bodu p_i
- $C \in \mathrm{R}^{3x3}$ je kovarianční matice
- λ_j je j-té vlastní číslo
- $\overrightarrow{v_j}$ je j-tý vlastní vektor

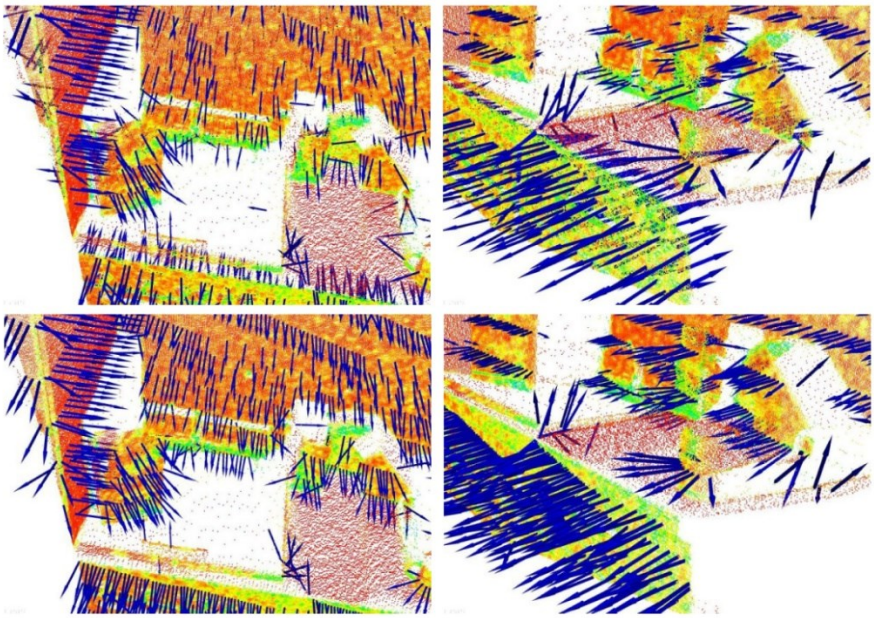
Problém ovšem nastává v případě, kdy není známo správné znaménko normály. Obecně totiž neexistuje matematický způsob, jak znaménko normály řešit. Při použití metody analýzy hlavních komponent (Pearson, 1901) je výsledek nejednoznačný a není konzistentní v celém mračnu bodů. V případě, že je známa doplňková informace, jako je například poloha snímací kamery, je řešení triviální. Výsledkem je konzistentní směr normálů $\vec{n}$, který splňuje podmínku danou rovnicí (5.5).

$$\overrightarrow{n_i} \cdot \left(\mathrm{v_p} - p_i\right) > 0 \tag{5.5}$$

Kde:

- $\mathrm{v_p}$ je poloha snímací kamery
- $\overrightarrow{n_i}$ je i-tý normálový vektor

Ukázka nekonzistentnosti normál $\vec{n}$ bez určení polohy snímací kamery a s polohou snímací kamery $\mathrm{v_p}$ je zobrazena na Obrázek 56.



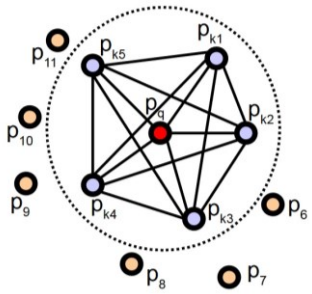
Obrázek 56: Nahoře: Ukázka nekonzistence orientace normálů $\vec{n}$. Dole výsledek po přeorientování normálů $\vec{n}$ na základě polohy kamery v_p. (Zdroj: (Rusu, 2009), upraveno)

5.3.2 Point Feature Histogram (PFH)

Metoda PFH pro odhad výsledného popisu voxelu využívá odhady povrchových normál a geometrické zakřivení v okolí zkoumaného bodu p_q. Přestože tato metoda je velice rychlá a její výpočetní složitost je celkem snadná, nedokáže zachytit velké množství detailů. Tato vlastnost metody je způsobena aproximací *k*-sousedních bodů, tj. pro následné výpočty algoritmus pracuje jen s malým množstvím hodnot.

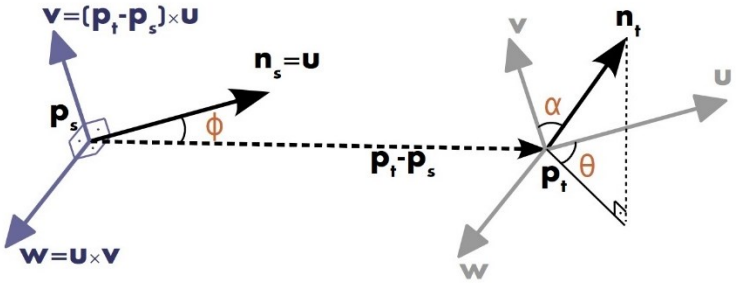
Cílem PFH metody je co nejpřesněji odhadnout geometrické vlastnosti *k*-okolních bodů zobecněním střední křivosti pomocí multidimenzionálního histogramu hodnot. Tento vysoce dimenzionální prostor poskytuje informativní popis pro reprezentaci vzájemných vztahů mezi dvojicemi bodů, dále je neměnný k 6D pozici a je celkem imunní vůči různým úrovním šumu v okolí zkoumaného bodu p_q. Jednoduše řečeno, metoda se snaží co nejlépe zachytit veškeré variace vzorkovaného povrchu tím, že se snaží zachytit veškeré interakce mezi odhadovanými normálami. Z výše zvedeného principu vyplývá, že kvalita výsledného hyperprostoru silně závisí na kvalitě odhadu normál povrchu v každém zkoumaném bodě (Poi211).

Na Obrázek 57 je zobrazen diagram vlivu algoritmu pro zkoumaný bod p_q, který je označen červeně, na okolních k-sousedních bodech, které spadají do zkoumané oblasti, která je definována jako koule s poloměrem r. Výsledný PFH deskriptor je vypočítán jako histogram vztahů mezi jednotlivými dvojicemi bodů v okolí. Výpočetní složitost určení deskriptoru je dána $O(nk^2)$ (Rusu, 2009).



Obrázek 57: Vztah bodů v okolí zkoumaného bodu v případě PFH deskriptoru (Zdroj: (Rusu, 2009))

Pro výpočet relativního rozdílu mezi body p_s a p_t, a jejich normálami $\overrightarrow{n_s}$ a $\overrightarrow{n_t}$ je definován souřadnicový rámec v bodě p_s, který je zobrazen na Obrázek 58, je definován rovnicemi (5.6) a (5.7).



Obrázek 58: Ukázka souřadnicového rámce mezi body (Zdroj: (Rusu, 2009))

$$\begin{aligned} \vec{u} &= \overrightarrow{n_s} \\ \vec{v} &= \vec{u} \times \frac{(p_t - p_s)}{\|p_t - p_s\|^2} \\ \vec{w} &= \vec{u} \times \vec{v} \end{aligned} \tag{5.6}$$

Pomocí souřadnicového systému definovaného vektory $\vec{u}, \vec{v}, \vec{w}$ je možné definovat rozdíl mezi normálami $\overrightarrow{n_s}$ a $\overrightarrow{n_t}$ rovnicí (5.7). Vektor parametrů $\langle \alpha, \phi, \theta, d \rangle$ je vypočítán pro každý z párů v oblasti sousedních bodů, čímž je možné redukovat počet parametrů až na 12 hodnot pro dvojice párů a na 4 jejich normály (Rusu, 2009).

$$\begin{aligned} \alpha &= \vec{v} \cdot \overrightarrow{n_s} \\ \phi &= \vec{u} \cdot \frac{(p_t - p_s)}{d} \\ \theta &= \tan^{-1}(\vec{w} \cdot \overrightarrow{n_t}, \vec{u} \cdot \overrightarrow{n_t}) \end{aligned} \tag{5.7}$$

Kde:

- d je euklidovská vzdálenost bodu p_s od bodu p_t

Pro vytvoření finální reprezentace PFH pro daný bod je množina všech určených vektorů rozdělena do histogramu. Algoritmus následně rozdělí hodnoty každého deskriptoru do *N*-intervalů a určí se počet výskytů daných bodů ve stanovených intervalech. Jak již bylo výše určeno, vektor parametrů $\langle \alpha, \phi, \theta, d \rangle$ se skládá ze tří úhlů mezi normálami. Základní vlastností je, že tyto úhly mezi normálami je možné snadno normalizovat na trigonometrickou kružnici. Rozdělením (binningem) každého intervalu na stejný počet shodných částí, dojde k vytvoření histogramu s b^4 sloupců v prostoru. V určitých případech bylo pozorováno, že je výhodné vynechat rozměr d, například v případě, kdy lokální hustota bodů ovlivňuje hodnotu d (Poi211).

5.3.3 Fast Point Feature Histogram (FPFH)

Jak již bylo zmíněno, teoretická výpočetní náročnost PFH pro dané mračno bodů *P,* s n body a k sousedním bodům je $O(nk^2)$. V disertační práci (Rusu, 2009) a publikaci (Rusu, a další, 2009) byla autorem rozpracována zjednodušující metoda, nazvaná Fast Point Feature Histogram (FPFH). Metoda popisuje geometrické vztahy kolem zkoumaného bodu $\mathrm{p_q}$ pro 3D mračna bodů. Dále autoři provedli optimalizace, které významně snížily výpočetní náročnost algoritmu, ovšem se zachováním původních parametrů i vlastností metody PFH.

Obecný postup zjednodušení je možné zapsat rovnicí (5.8). (Rusu, 2009) , (Fas21), (Rusu, a další, 2009) :

1) Pro zkoumaný bod $\mathrm{p_q}$ a jemu sousední body je vypočítána sada hodnot $\langle \alpha, \phi, \theta \rangle$. Pro zjednodušení bude tato metoda označována Simplified Point Feature Histogram (SPFH).

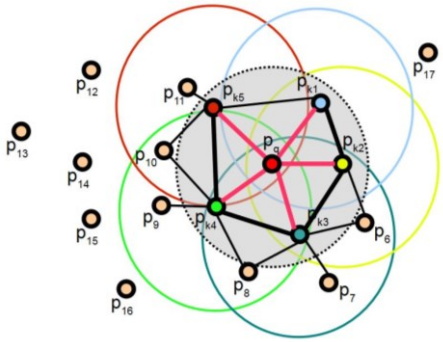
2) Nový výpočet oblasti sousedství pro každý bod. Hodnoty z předchozího kroku jsou použity jako váhy finálního histogramu daných bodů $\mathrm{p_q}$.

$$\mathrm{FPFH}(\mathrm{p_q}) = \mathrm{SPFH}(\mathrm{p_q}) + \frac{1}{k}\sum_{i=1}^{k}\frac{1}{\omega_k} SPFH(p_k) \qquad (5.8)$$

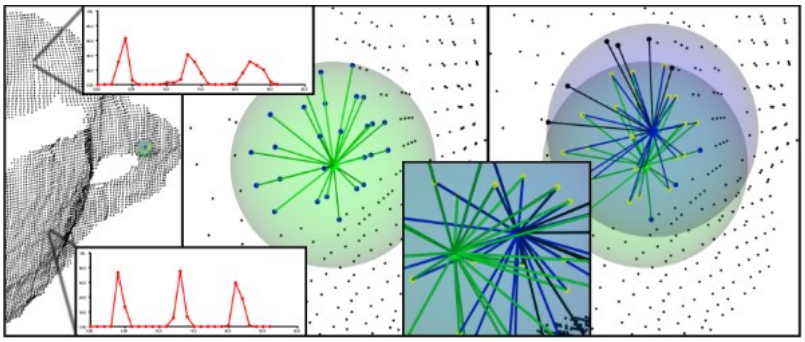
Kde:

- ω_k představuje vzdálenost mezi zkoumaným bodem $\mathrm{p_q}$ a sousedním bodem $\mathrm{p_k}$ v definované metrice

Algoritmus pro zkoumaný bod p_q a jeho nejbližší sousedy vytvoří páry, které jsou v Obrázek 59 označeny červenou spojnicí a odhadne hodnoty SPFH pro všechny body v množině. V následujícím kroku dojde opět k vytvoření párů mezi zkoumaným bodem p_q a jeho nejbližší sousedy. Zároveň dojde k vážení těchto nalezených parametrů hodnotami získanými v předchozím kroku algoritmu. Spojnice bodů, odpovídající dodatečnému váhování jsou vyznačeny černými čárami v Obrázek 59. Tlustou čarou jsou označeny veškeré páry bodů, které byly shodně vyhodnoceny v obou krocích algoritmu (Rusu, 2009), (Fas21).



Obrázek 59: Konstelace bodů se středním bodem p_q a vliv okolních bodů. (Zdroj: (Rusu, 2009))



Obrázek 60: Ukázka odhadu FPFH deskriptoru pro 3D bod. (Zdroj: (Rusu, 2009))

Na Obrázek 60 je zobrazen odhad FPFH deskriptoru pro 3D bod. V levé části obrázku je zelenou barvou označen zkoumaný bod p_q. V zelené kouli o poloměru r jsou zahrnuty všechny sousední body k bodu p_q. Pro každý z párů $\langle p_q, p_k \rangle$, jejichž spojnice je označena v obrázku zelenou čarou, jsou vypočítány dle rovnice (5.6) úhlové parametry $\langle \alpha, \phi, \theta \rangle$. Tento postup je opakován pro každý sousední pixel, jako je zobrazeno v pravé části obrázku. Algoritmus následně provede váhování histogramu dle rovnice (5.8). Výsledný odhadnutý histogram pro vybrané body je zobrazen v levé části pomocí červené křivky (Rusu, 2009).

5.4 Metody registrace mračna bodů

5.4.1 ICP (Iterative Closest Points)

Jednou z nejčastějších metod pro registraci naměřených bodů je metoda ICP, která pracuje s jednotlivými mračny bodů. Algoritmus *ICP* byl představen v publikaci (Besl, a další, 1992). Algoritmus je dále vyvíjen a rozšiřován. Podrobnější informace

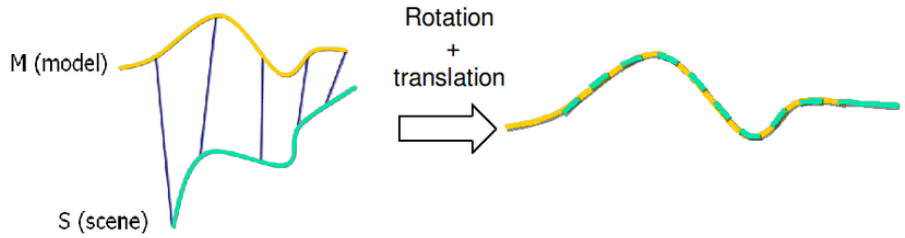
o algoritmu a jeho rozvoji lze získat například z publikací (Rusinkiewicz, a další, 2001) či (Gelfand, a další, 2003),

Vstupní parametry metody jsou dvě mračna bodů popisující například určitý tvar. Cílem metody je nalézt takové parametry transformace označované jako $(R, \boldsymbol{t})$, kde R je rotační matice a $\boldsymbol{t}$ je vektor posunu mezi vstupními mračny bodů, které minimalizují chybovou funkci definovanou vztahem (5.9) (Nuchter, 2009).

$$E(R, \boldsymbol{t}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} \left\| \widehat{m_i} - \left(R\widehat{d_j} + t \right) \right\|^2 \qquad (5.9)$$

Kde:

- $E(R, \boldsymbol{t})$ je chybová funkce
- R je rotační matice
- **t** je vektor posunu
- $w_{i,j}$ jsou nalezené společné body z obou mračen
- $\widehat{m_i}$ je konkrétní testovaný bod z prvního setu dat $\widehat{\mathrm{M}}$(model set) o délce N_m
- $\widehat{d_j}$ je konkrétní testovaný bod z druhého setu dat $\widehat{\mathrm{D}}$(data set) o délce N_d



Obrázek 61: Ukázka principu ICP algoritmu (Zdroj: (Hardy, a další, 2016))

V průběhu jednotlivých iteračních kroků dochází k výpočtu souhlasných bodů $w_{i,j}$ a odpovídající transformaci $(R, \boldsymbol{t})$. Cílem metody je nalézt takovou minimální chybovou funkci $E(R, \boldsymbol{t})$, které odpovídá maximum souhlasných bodů $w_{i,j}$.V každém kroku dojde k vybrání nejbližších bodů v rozsahu d_{max} a jsou vypočítány příslušné transformační parametry $(R, \boldsymbol{t})$ pro minimalizaci chybové $E(R, \boldsymbol{t})$. Algoritmus je ukončen po dosažení maximálního počtu iteračních kroků nebo při dosažení stanovené hranice rozdílu. Výsledné získané parametry jsou zapsány do matice, která je následně využita pro transformaci mračna bodů. Podrobný popis algoritmu je popsán v publikacích (Nuchter, 2009) či (Segal, a další, 2009).

Algoritmus *ICP* má ovšem dva hlavní nedostatky. Prvním je fakt, že algoritmus při zpracování využívá pouze konkrétní body, nikoliv například souvislé plochy či tvary

v okolí zkoumaného bodu. Druhým nedostatkem je vysoká výpočetní náročnost. Díky výše uvedeným nedostatkům vzniklo velké množství modifikací algoritmu, zejména se zaměřením algoritmu na zpracování bodů vůči ploše, z nichž jsou dále v práci popsané jen vybrané.

Jedním z modifikací *ICP* algoritmu je algoritmus *IDC* představený autory, který se primárně zaměřuje na zvýšení rychlosti zpracování (Lu, a další, 1994). Algoritmus *IDC* vyhledává v každé iteraci skupinu společných bodů za využití polárních souřadnic v zadaném intervalu. Při porovnání algoritmů se jeví algoritmus *IDC* více robustnější než *ICP* za předpokladu, že je výraznější počáteční rozdíl v translaci a rotaci mezi mračny bodů. Podrobnější informace o porovnání algoritmu *ICP* a *IDC* je možné najít v publikaci (Minguez, a další, 2006)

Další modifikace *ICP* algoritmu se jmenuje *PLCIP*. Jedná se o modifikaci se zaměřením na minimalizaci metriky mezi zkoumaným bodem a plochou (Point-Plane). Podrobné informace o modifikaci algoritmu je možné nalézt v publikaci (Censi, 2008).

Alternativní přístup je popsán v algoritmu *pIC.* Metoda je zaměřena na počátečním odhadu transformace mračna bodů a na popisu šumu v obou vstupních mračnech bodů. Šum i transformace je brána jako Gaussovský náhodný proces s nulovou střední hodnotou šumu. Kovarianční matice je získání z přesnosti měřicích senzorů a jejich konfigurace. Algoritmus byl porovnán s algoritmem *IDC* i *ICP* a výsledky ukazují, že algoritmus *pIC*, dosahuje konvergence až o 25% rychleji (Montesano, a další, 2005).

5.4.2 NDT (Normal Distribution Transform)

Metoda NDT je modernější metodou ve srovnání s ICP sloužící pro registraci prostorových mračen bodů navržena autory (Biber, a další, 2003), dále rozpracovaná autorem Martinem Magnussonem v jeho disertační práci (Magnusson, 2009). Další podrobné informace o metodě lze získat například v publikaci (Takeuchi, a další, 2006) či (Takubo, a další, 2009).

Algoritmus NDT, jak je patrné i z názvu, využívá pro zpracování data, reprezentovaná modelem normálního rozdělení. V porovnání s algoritmem ICP, metoda nezkoumá jednotlivé body, ale využívá model tvořený normálním rozdělením a pravděpodobnosti shody bodu s plochou, tvořenou mračnem bodů na určité pozici. Tato reprezentace bodů v normálním rozdělení poskytuje reálnou a hladkou reprezentaci

jednotlivých bodů se spojitou derivací 1. a 2. řádu prostřednictvím *Jacobiho* a *Hessovy* matice, což umožňuje použití standardních optimalizačních metod.

Algoritmus se skládá z několika základních kroků. Prvním krokem je při inicializaci algoritmu mračno bodů rozděleno na separované části, tzv. buňky. Pro každou buňku jsou spočteny její základní statistické parametry, jako je kovarianční matice a průměr. V případě, že jsou body z referenčního mračna bodů popsány N-dimenzionálním náhodným procesem, lze definovat pravděpodobnost bodu dle rovnice (5.12) (Biber, a další, 2003).

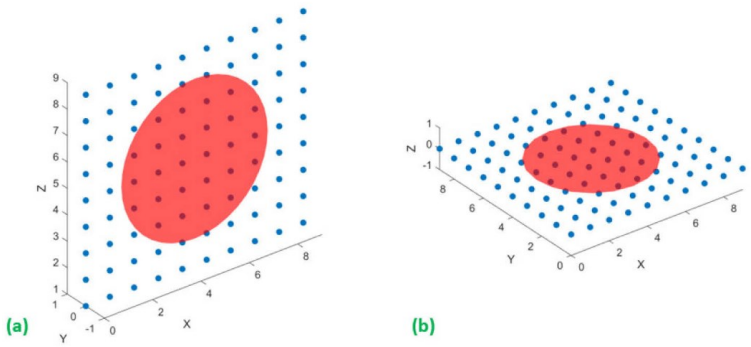
$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{5.10}$$

$$\Sigma = \frac{1}{n}\sum_{i}(x_i - \mu)(x_i - \mu)^T \tag{5.11}$$

$$p(\boldsymbol{x}) = \frac{1}{\sqrt{|\Sigma|}(2\pi)^{\frac{N}{2}}} \exp\left(-\frac{(x-\mu)^T\Sigma^{-1}(x-\mu)}{2}\right) \tag{5.12}$$

Kde:

- μ je střední hodnota bodů
- N je počet bodů v datové sadě $x_{i=1,\ldots,N}$
- Σ je vypočítaná kovarianční matice z datové sady
- $p(\boldsymbol{x})$ je pravděpodobnost bodu



Obrázek 62: Reprezentace buňky v NDT vlastními čísly (a) kolmá stěna, (b) rovina (Chmelař, 2018)

Cílem registrační části algoritmu je nalezení takové transformace $T(p, \mathbf{x})$ pro maximalizaci pravděpodobnosti pozice bodů v referenčním mračnu. Transformační funkce nastavuje orientaci a pozici $\boldsymbol{p}$ bodu $\mathbf{x}$ takovým způsobem, aby se nalezlo maximum pravděpodobnostní funkce Ψ (Magnusson, 2009).

$$\Psi = \prod_{k=1}^{N} p(T(\boldsymbol{p}, \boldsymbol{x_k})) \tag{5.13}$$

Kde:

- Ψ je pravděpodobností funkce
- N je počet bodů v datové sadě $x_{i=1,\dots,N}$
- $\boldsymbol{p}$ je vektor obsahující polohu a rotační úhel. $\boldsymbol{p} = \left[t_x, t_y, t_z, \phi_x, \phi_y, \phi_z\right]^T$
- t_x, t_y, t_z je posun v ose x,y,z
- ϕ_x, ϕ_y, ϕ_z je úhel rotace bodu

$$T_E(\boldsymbol{p}, \boldsymbol{x}) = R_x(\phi_x)R_y(\phi_y)R_z(\phi_z)\boldsymbol{x} + \boldsymbol{t} \qquad (5.14)$$

Kde:

- $T_E(\boldsymbol{p}, \boldsymbol{x})$ je Eulerova transformace
- R_x, R_y, R_z jsou matice směrových kosinů
- $\boldsymbol{t}$ je vektor posun v ose x, y, z

V každém kroku iterace algoritmu je stanovena velikost chybové funkce na základě aktuálního vektoru $\boldsymbol{p}$. Dále jsou aktualizovány Jacobiho a Hessovy matice a za využití Newtonovy optimalizační metody je určen nový odhad vektoru $\boldsymbol{p}$. Algoritmus je ukončen při dosažení hodnoty maximální změny či při překročení počtu stanovených iterací (Magnusson, 2009). Další podrobnosti o algoritmu je možné získat z (Magnusson, a další, 2007) či (Biber, a další, 2003).

V rámci dalších výzkumů byl algoritmus *NDT* více rozvíjen. Například v publikaci (Boughorbel, a další, 2004) je k modelování vstupních mračen bodů použit model Gaussovského rozdělení (*GMM*) a jsou sledovány dva hlavní parametry. Prvním sledovaným parametrem je prostorová vzdálenost mezi body a druhým parametrem je shoda povrchu v okolí bodů. Body jsou převedeny a porovnávány v multidimenzionálním prostoru, který obsahuje nejen prostorové dimenze, ale další dimenzionální parametr, který umožňuje snadnější určení části prostředí. Z těchto dimenzionálních parametrů se určí prostorové momenty a na jejich základě je určena visuální podobnost.

Jedním z dalších modifikací algoritmu NDT je pravděpodobnostní párování bodů. Autoři algoritmu (Burguera, a další, 2007) (Burguera, a další, 2008) se zaměřili na data ze sonarových měření. Sonar, který měl malé úhlové rozlišení, způsoboval problémy s párováním bodů. Při navazující studii (Burguera, a další, 2012) bylo prokázáno, že tato

modifikace algoritmu je aplikovatelná i na metodu ICP, čímž došlo ke vzniku metod *sNDT* a *sICP*.

5.4.3 RANSAC (RANdom SAmple Consensus)

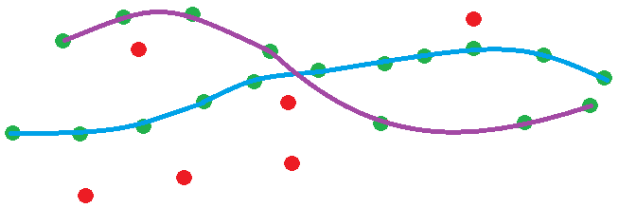
RANSAC metoda je iterativní metoda pro získání parametru modelu z předložených vstupních mračen bodů. Historie metody se datuje od roku 1981, kde byla poprvé popsána v publikaci (Fischler, a další, 1981). Základním parametrem metody je, že metoda není deterministická, tj. produkuje výsledky jen s určitou pravděpodobností. Pravděpodobnost detekce správného výsledku se ovšem zvyšuje s počtem iterací algoritmu (5.15)

$$I_c = \frac{\log(1-P_s)}{\log(1-P_I^N)} \qquad (5.15)$$

Kde:

- I_c je počet iterací
- P_s je pravděpodobnost úspěšného řešení
- P_I je pravděpodobnost vybrání *Inlieru*
- N je počet parametrů zkoumané křivky

Metoda pracuje se dvěma skupinami bodů – vnitřní body (*Inliers*) a odlehlé body (*Outliers*). Vnitřní body (*Inliers)* jsou body patřící hledanému tvaru či se nacházejí v jeho blízkém okolí. Toto okolí se bere v potaz z důvodu potenciálních chyb v datech, způsobených například šumem měření. Odlehlé body (*Outliers*) jsou body, které daný tvar nereprezentují, respektive nespadají do metodou hledaného modelu, a jsou v předložených datech navíc. Úkolem algoritmu je stanovit, který bod ze vstupní množiny bodů patří, do jaké skupiny. Ukázka práce RANSAC algoritmu je zobrazena na Obrázek 63, kde červené body jsou odlehlé body, zelené body jsou vnitřní body. Výsledné modely jsou reprezentovány modrou a fialovou barvou.



Obrázek 63: Ukázka proložení vstupních dat modelem. (Zdroj: Vlastní)

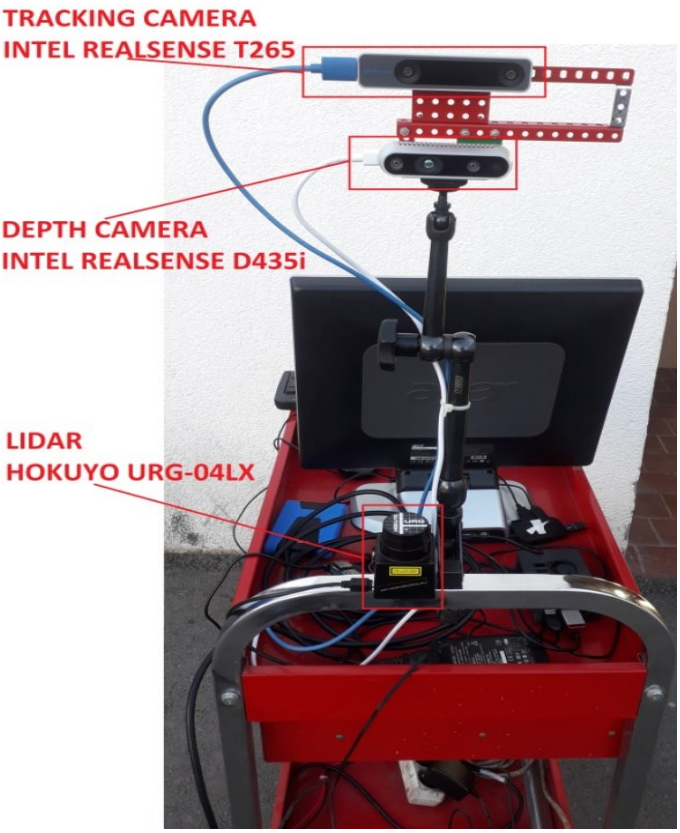
5.4.3.1 Postup algoritmu RANSAC

Při výpočtu se z celé vstupní množiny dat vybírají minimální množiny. Minimální množina je definovaná tak, že obsahuje takový počet bodů, ze kterých se dají vytvořit unikátní parametry modelu. Vytvořené unikátní modely jsou následně testovány vůči celé vstupní množině dat, čímž se vyhodnotí celkový počet aproximovaných bodů. Každý model má přidělené vlastní skóre, což je počet bodů kolik úspěšně aproximuje. V rámci výpočtu dochází postupně k určení nejlepšího modelu pro aproximaci a taktéž se vypočítává pravděpodobnost nalezení nového modelu. Při poklesu pravděpodobnosti nalezení nového modelu pod určitou mez je algoritmus ukončen a je extrahován model s největším skórem. Ve srovnání s jinými algoritmy (např. *ICP* – kapitola 5.4.1 či NDT – kapitola 5.4.2), algoritmus *RANSAC* nemá pevně stanovený konec a délka samotného výpočtu je stanovena pravděpodobnostním prahem, který stanovuje, jaké je množství potenciálních nových modelů k nalezení. Tento práh ovšem výrazně ovlivňuje kvalitu a rychlost výpočtu.

Výhoda algoritmu je celková robustnost metody, hledání jen určitých popsaných objektů, do určité míry odolnost vůči šumu a parametrizace nalezeného objektu. Nevýhodou této metody je ovšem možnost nenalezení optimálního řešení a neschopnost aplikovat na obecné tvary (Fischler, a další, 1981). Další informace o metodě lze nalézt například v publikacích (Schnabel, a další, 2007), (Yaniv, 2010) či (Raguram, a další, 2012).

6 Měřící platforma

Měřící platforma je založena na pohyblivém vozíku, který byl osazen několika druhy senzorů. Nejdůležitějším senzorem je hloubková kamera Intel Realsense D435i, poskytující v rámci měření jak barevný, tak hloubkový snímek. Hloubkové snímky byly předzpracovány dle postupu uvedeném v kapitole 7, barevné snímky zůstaly beze změn. Sady těchto barevných i hloubkových snímků byly použity pro učení i vyhodnocení kvality učení neuronové sítě. Další senzor je Tracking kamera, Intel Realsense T265. Tato kamera obsahuje celkem dvě kamery s čočkou typu rybí oko. Kamery jsou umístěny v definované vzdálenosti. Synchronní snímání a pokročilé V-SLAM algoritmy implementované v ASIC procesoru přímo v senzoru umožňují vytvořit spolehlivý zdroj navigačních dat. Data z *V-SLAM* kamery byla snímána za účelem získání referenčních bodů pro navazující výzkum tvorby mapy okolního prostoru z hloubkových snímků a vyhodnocení kvality výsledku. Veškeré parametry senzorů jsou uvedeny v následujících kapitolách.



Obrázek 64: Umístění senzorů (Zdroj: Vlastní)

6.1 Kamera D435i:

Kamera Intel D435i je širokoúhlá hloubková kamera a byla vybrána pro své parametry, které byly pro práci vyhovující. Dalším aspektem výběru byla podpora ze systému ROS (*Robot Operating System*). Tento systém umožňuje kameru připojit přímo do systému a vyčítat z ní data bez nutnosti externího SW a taktéž umožňuje streamovat data přímo do jiného uzlu, kde jsou naměřená data dále zpracována. Tento princip v práci nakonec nebyl vzhledem k zaměření práce využit, ovšem pro budoucí rozvoj metod bude vhodným zjednodušením práce.

Obrázek 65: Intel Realsense D435i Depth Camera (Zdroj: (Intel, 2019), upraveno)

Tabulka 3: Parametry kamery D435i – snímače

Intel D435i – parametry kamery (Intel, 2019)		
	IR KAMERA	**RGB KAMERA**
Typ snímače	2x OV9282 2x MIPI širokoúhlý	1x OV2740, MIPI CSI-2
Rozlišení snímače	1280x800, 8:5	1920x1080, 16:9
Snímací úhel	H:91.2° V:65.5° D:100.6°	H:69.4° V:42.5° D:77°
Ohnisko	Pevné, 1.93 mm	Pevné, 1.88 mm
Závěrka	Global shutter	Rolling shutter
Formát	10bit RAW	10bit RAW

Tabulka 4: Obecné parametry hloubkové kamery D435i

Hloubková kamera Intel Realsense D435i (Intel, 2019)		
Rozměry kamery	Šířka	90 mm
	Délka	25 mm
	Hloubka	25 mm
CPU	VPU Procesor	Vision procesor D4
	Příkon	2 W
Stereo Depth Module SKU	FOV (VGA, 4:3) @2 m	H:74±3° V:62±1° D:88±3°
	FOV (HD, 16:9) @2m	H:86±3° V:57±1° D:94±3°
	IR projektor	Širokoúhlý
	IR projektor FOV	H:90° V:63° D:99°
	Rozteč IR kamer Baseline	50 mm

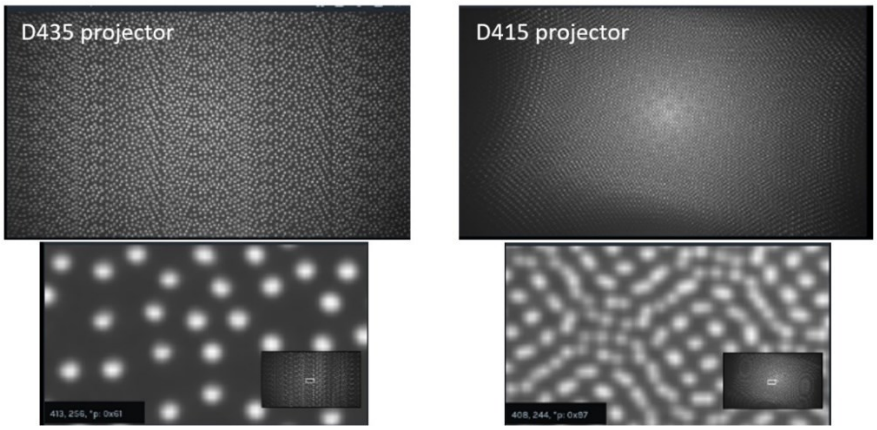
V neposlední řadě byl aspektem výběru i nízká pořizovací cena. Kamera poskytuje hloubková data v ose Z, která je kalibrovaná výrobcem a absolutní chyba je menší než 2 % u vzdálenosti 2 m. Minimální vzdálenost překážky od senzoru je přibližně 28 cm, doporučená vzdálenost překážky je v intervalu 30 cm až 3 m. Maximální vzdálenost, kterou je hloubková kamera schopna zaznamenat je 10 m. Vzdálenost od kamery, přesahující dosah hloubkové kamery je automaticky nastavena na maximální hodnotu dosahu.

Další výhodou kamery je globální závěrka, poskytující ostré snímky při pohybu. Jednou z nevýhod kamery, která velice komplikovala práci při předzpracování dat pro neuronovou síť, bylo odlišné rozlišení senzorů (RGB a IR) a taktéž jejich umístění a SW zpracování na ASIC procesoru. Kalibraci kamery je nutné získat přes geometrický přepočet dle rovnic uvedených v kapitole 7.3, případně je možné použít SW nástroj od výrobce. Tento SW nástroj ovšem je velice nedokonalý. Detailní popis problémů a jejich řešení je uveden v kapitole 7.

Tabulka 5: Parametry kamery D435i – IMU

Intel D435i – parametry IMU (Intel, 2019)		
	Akcelerometr	**Gyroskop**
Rozsah	±4g	±1000°/s
Vzorkovací frekvence	62,5 Hz	200/400 Hz

Kamera Intel RealSense D435i pracuje na principu promítání IR vzoru, který je následně snímán IR kamerou. Z promítaného vzoru a jeho ekvivalentu získaného za využití IR kamery je následně vytvořen hloubkový snímek. Ukázka promítaného vzoru je zobrazena na Obrázek 66.

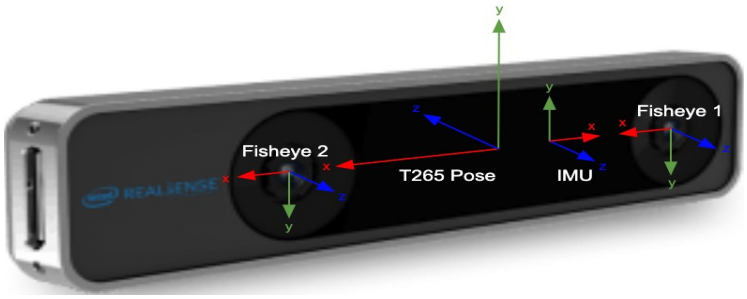


Obrázek 66: Ukázka promítaného IR vzoru kamerou Intel Realsense (Zdroj: (Jepsen, a další))

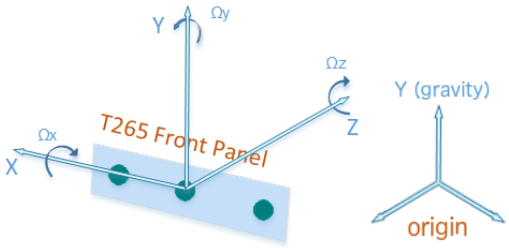
Druhou složkou snímaného obrazu, je nejčastěji RGB snímek. Výsledný snímek, který vznikne je nazýván RGB-D snímek, který obsahuje barevnou i hloubkovou složku. Hloubková složka je nejčastěji reprezentována hodnotou pixelu. Pro další zpracování je výhodné převést surová data z hloubkové kamery do speciální datové struktury pojmenované mračno bodů. Algoritmy převodu RGB-D do mračna bodů jsou detailně popsány v kapitole 7.

6.2 SLAM Kamera T265

Kamera T265 od společnosti Intel je specializovaná kamera obsahující dvě kamery OV2740 s objektivem typu rybí oko. Dále kamera obsahuje inerciální jednotku BMI055, obsahující akcelerometr a gyroskop. Snímky z kamer v koordinaci s daty z inerciální jednotky jsou zpracovány pomocí výkonného ASIC zařízení. Výsledkem je kvalitní určení polohy za využití metody V-SLAM.



Obrázek 67: Ukázka kamery T265 a orientace senzorů v kameře Zdroj: (Intel, 2019)



Obrázek 68: Orientace úhlů v kameře T265 Zdroj: (Intel, 2019)

Kamera byla do systému přidána jako kontrolní zdroj dat pro verifikaci odhadu polohy na základě hloubkových snímků odhadnutých z neuronových sítí.

V kameře, respektive ve výstupu ze V-SLAM (*T265 Pose*), pro někoho možná atypicky, je osa Y umístěna kolmo k zemi. Data z inerciální jednotky a z V-SLAM jsou použita v práci jako zdroj referenčních dat a pro

Tabulka 6: Parametry kamery T265

Tracking Camera Intel Realsense T265 (Intel, 2019)			
Rozměry kamery		Šířka	93.35 mm
		Délka	17,6 mm
		Hloubka	17,13 mm
CPU		VPU Procesor	ASIC Movidius MA215x
		Příkon	2W
Kamera		Typ	2x OV9282
		Rozlišení	848x800
		Ohnisko	Pevné, 1.88mm
		Formát	10bit RAW
		Objektiv	Rybí oko
		Rozteč kamer	64±0.15 mm
		FOV kamery	173°
		Závěrka	Globální
IMU	IMU	Typ	BMI055
		DoF	6
	Akcelerometr	Rozsah	±4g
		f_s	62,5 Hz
	Gyroskop	Rozsah	±2000°/s
		Vzorkovací frekvence	200 Hz

správnou interpretaci hodnot je nutné přepočítat hodnoty na stejný souřadnicový systém.

Na Obrázek 69 je zobrazen výstup z Intel T265 kamery. Z obrázku je patrné, že kamera obsahuje objektiv typu rybí oko, zajišťující úhel snímání jedné kamery 173°.

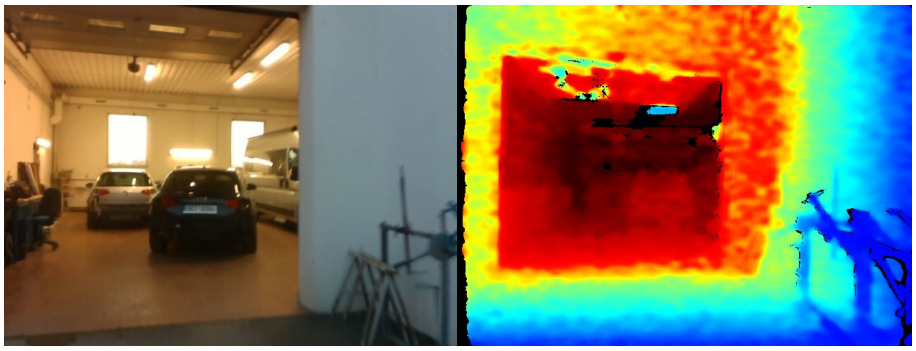
Obrázek 69: Ukázka snímku z T265 Kamery (Zdroj: Vlastní)

6.3 Měření dat

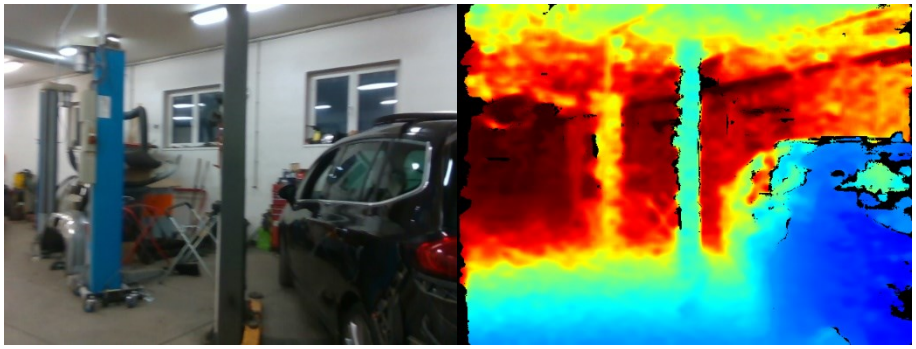
V rámci výzkumu byly provedeny celkem tři sady měření při různých okolních podmínkách. Měření bylo prováděno v uzavřeném areálu autoservisu. Valná většina naměřených dat byla naměřena v interiéru za denního i umělého osvětlení. Minoritní část dat byla naměřena exteriéru mezi budovami za různého počasí – slunečno, zamračeno i soumrak. Tato minoritní data byla měřena experimentálně v rámci přejezdů mezi halami. V průběhu každé sady měření bylo provedeno minimálně 7 sad měření. Celkově bylo naměřeno 24 různých tras. Každá sada měření obsahovala různé trasy v areálu a žádná trasa nebyla stejná. Zároveň mezi měření z důvodu získání co největší variability dat byly významné objekty mezi měřeními přemístěny (automobily, židle, popelnice, stoly apod.).

Měření bylo v areálu autoservisu provedeno s odstupem několika měsíců, zejména vzhledem k požadavkům doučování neuronových sítí během vývoje. Délka záznamu závisela dle zvolené trasy v budově, přičemž nejkratší měření trvalo 4,5 minuty, nejdelší přibližně 12 minut, průměrná délka záznamu byla přibližně 6,5 minuty. Celková doba záznamu byla přibližně 155 minut. Vzniklý objem surových dat ze samotné hloubkové kamery byl obrovský, konkrétně 254 GB. Při započtení záznamů z V-SLAM kamery T265 celkový objem naměřených dat přesáhl 1TB.

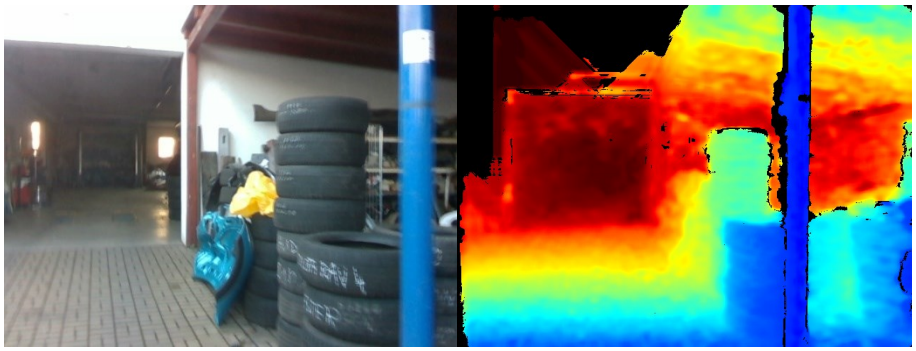
Ukázky z různých měření jsou zobrazeny na Obrázek 70 až Obrázek 73. Snímky obsahují surové snímky z hloubkové kamery jak ve vnitřním, tak vnějším prostředí, včetně různých typů osvětlení scény.



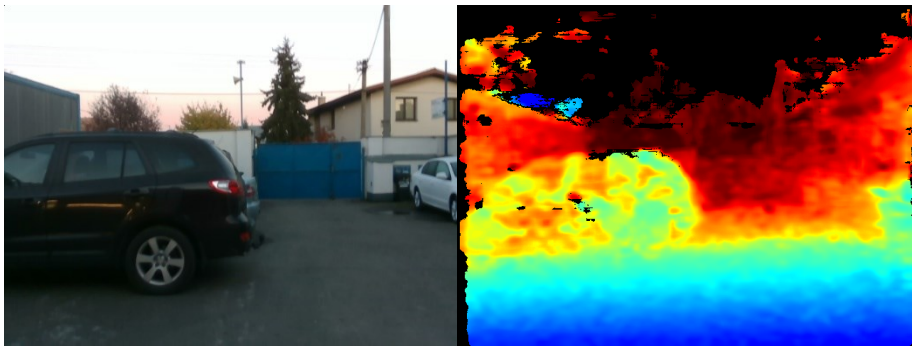
Obrázek 70: Ukázka surových snímků z hloubkové kamery – Přejezd z vnějšího prostředí s přítmím do osvětlené místnosti. Barevný snímek (Vlevo), Hloubkový snímek (Vpravo). (Zdroj: Vlastní)



Obrázek 71 Ukázka surových snímků z hloubkové kamery – Vnitřní prostředí s umělým osvětlením. Barevný snímek (Vlevo), Hloubkový snímek (Vpravo). (Zdroj: Vlastní)



Obrázek 72: Ukázka surových snímků z hloubkové kamery – Vnější prostředí s přírodním. osvětlením. Barevný snímek (Vlevo), Hloubkový snímek (Vpravo). (Zdroj: Vlastní)



Obrázek 73: Ukázka surových snímků z hloubkové kamery – Vnější prostředí s volným prostorem a přírodním osvětlením. Barevný snímek (Vlevo), Hloubkový snímek (Vpravo). (Zdroj: Vlastní)

7 Příprava dat pro zpracování neuronovou sítí

Během měření okolního prostoru jsou z kamer získány snímky, které nejsou vhodné pro přímé zpracování neuronovou sítí. Jedná se zejména o to, že rozlišení a úhly pohledu (HFOV, VFOV) nejsou u RGB a IR senzorů shodné. Výsledkem jsou dvě různé sady snímků, které nejsou koherentní. Algoritmy předzpracování dat na základě kalibračních dat z kamery a rovnic tyto snímky srovnají.

Dalším problémem v naměřených datech jsou chybějící oblasti. Původ těchto prázdných oblastí není vždy stejný, veskrze se ale jedná například o odlesk od lesklých povrchů jako je například sklo. Další možností chybějících dat je svit slunce a oslnění kamery. Tyto prázdné oblasti je nutné nějakým způsobem doplnit. Pro doplnění byly vytvořeny a vyhodnoceny celkem tři metody.

Veškerý postup a popis metod předzpracování snímků je uveden v této kapitole.

7.1 Extrakce naměřených dat

V rámci měření byly veškeré naměřené údaje zaznamenávány v surovém stavu do speciálního souboru, nazývaného RosBag. Tato kapitola detailně popisuje, jak provést vyčtení všech naměřených dat z výše uvedeného souboru jak za využití nástroje od výrobce, tak za pomoci programu Matlab.

7.1.1 Extrakce dat z RosBag souboru za využití nástrojů od výrobce

Datový stream z kamery T265 i D435i je uložen v souboru zvaném Rosbag. RosBag soubor slouží k uložení proudu dat včetně časové značky a dalších informací pro pozdější simulaci a generování identického streamu, který by byl generován fyzickým zařízením. Tento typ souboru vznikl jako standard v open-source komunitě ROS (*Robot Operating System*) (Ros.org, 2021) a již se velice často používá i mimo tento Framework.

Balíček RosBag je možné otevřít nejen za využití platformy ROS, ale nejnovější verze Matlabu jsou schopny s tímto balíčkem taktéž pracovat. Základním problémem ovšem může být velikost záznamu, potažmo velikost operační paměti. Při pokusech otevřít cca 60vteřinový záznam v programu Matlab, který zabíral na pevném disku cca 1 GB, si program Matlab vyžádal přes 60GB paměti RAM. Matlab se snaží celý soubor načíst do paměti a nativně nepracuje se souborem jako s proudem dat. Pro tyto případy společnost

Intel pro práci s RosBag vytvořila API (knihovny librealsense, (GiH21)), který uložené snímky za využití Matlabu extrahuje ze souboru. Bohužel i toto API nefunguje dobře a velice často skript vrací stav, že nalezl konec souboru a dojde k ukončení vyčítání dat, přičemž soubor obsahuje několikanásobně větší množství dat, než bylo skutečně vyčteno. Při řešení s ROS komunitou byly zkoušeny postupy opravy a reindexování souboru RosBag, kdy by mělo dojít k přeskupení dat v souboru v závislosti na časové známce zprávy.

Pro kontrolu a případnou opravu RosBag souboru, reindexaci a dekompresi se používá sekvence příkazů [3]:

```
1) rosbag fix -n old.bag repaired.bag
2) rosbag reindex --output-dir=reindexed *.bag
3) rosbag decompress *.bag
```

Pokud veškeré příkazy proběhly úspěšně, mělo by za využití skriptu dojít k vyčtení uložených snímků ze souboru RosBag. Bohužel v případě této práce i přes různé opravy souboru docházelo k náhodnému čtení zprávu ze souboru a bylo nutné vytvořit imunní skript, který se pokusí vyčíst maximální množství uložených dat. Výhodou oproti programu od Intelu je možnost vyčíst si ze souboru RosBag doplňkové informace, například data z inerciální jednotky, která za využití nástroje od výrobce není možné získat [4]. Problémem je ovšem fakt, že dochází k náhodnému čtení a není vždy správně vyčtena časová známka a pořadí zprávy. Občas se stává, že je vrácen stejný index zprávy a obsah zprávy je jiný. Pokud je potřeba získat pouze hloubková data a RGB snímky, je možné použít sekvenci příkazů níže, která je relativně spolehlivá a funkční.

```
1) cd "C:\Program Files (x86)\Intel RealSense SDK 2.0 (Win7)\tools\"
2) mkdir "H:\RGBD\OutFolder_M1"
3) rs-convert -i "D:\SRC\M1.bag" -p "H:/RGBD/OutFolder_M1/"
```

Výstupem ze skriptu jsou barevné, hloubkové snímky a aditivní informace o streamu dat (časová známka, čas expozice apod.). Dále je potřeba zmínit, že snímky je nutné dále zpracovat (zarovnat). Do této chvíle bylo možné použít bez problémů nástroje od výrobce, ovšem zarovnání je už výrazně složitější. Nástroje od výrobce umí provést srovnání RGB a hloubkového snímku na online streamu a s připojenou kamerou (nástroj *rs-align.exe* či *rs-align-advanced.exe*), což je bohužel pro tento případ nepoužitelné

[3] 3 Sekvenci je nutné spustit na operačním systému (ideálně Ubuntu) s instalovaným ROS Frameworkem verze ROS Melodic Morenia nebo vyšším.

[4] Problémem je od minulé verze knihoven (2.48.0 a dřívější), které při požadavku na vyčtení vyvolají chybu *segmentation fault.* Nově vydaná verze 2.49.0 (Srpen 2021) chybu měla mít již opravenou, ovšem stále není možné informace z inerciální jednotky vyčíst. Tento problém byl vyřešen vytvořením vlastního Python skriptu za využití knihovny *pyrealsense2*, kde je možné s určitými omezeními data vyčíst.

a nástroj slouží spíše jako demo. Pokud uživatel provede extrakci barevných a hloubkových snímků za použití výše zmíněného nástroje (*rs-convert.exe*) tak dojde ke ztrátě informací o transformační matici a bez externího programu není možné provést zarovnání.

7.1.2 Extrakce dat z Rosbag souboru s využitím programu Matlab

V této práci pro vyčítání dat z RosBagu a zároveň zarovnání hloubkového a barevného snímku byl použit program Matlab s využitím C++ API od Intelu. Velkou výhodou je možnost použití vytvořených knihoven což výrazně usnadní práci. Dále je možné provést extrakci doplňkových informací z RosBagu. Jak již bylo zmíněno, program Matlab umí v nejnovějších verzích se souborem pracovat přímo, ovšem práce není úplně jednoduchá a vyžaduje velké systémové zdroje. Nejjednodušší řešení je použití Realsense C++ API, které je možné implementovat do Matlabu.

7.2 Úprava snímků

Úprava snímků byla nezbytná pro použití snímků pro trénování neuronových sítí. Surová data získaná přímo z hloubkové kamery obsahují oblasti, které nejsou validní či určité oblasti mohou chybět.

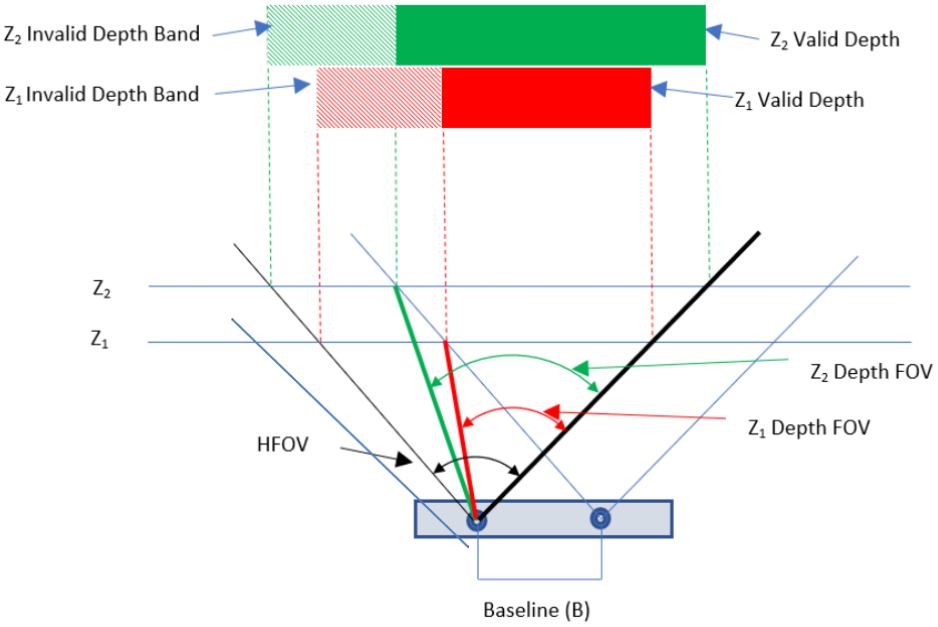
7.2.1 Odstranění nevalidní oblasti hloubkových dat

Surová data získaná přímo ze snímacího zařízení, mají díky konstrukci kamery odlišné rozlišení a taktéž jiné úhly pohledu jednotlivých kamer (barevná a hloubková kamera). Pro odstranění nevalidní oblasti jsou k dispozici základní parametry kamery včetně kalibrací, které jsou uloženy v interní paměti kamery a je možné je kdykoliv vyčíst. Jelikož kamera obsahuje dva infračervené snímače, při zpracování vznikne místo, kde není hloubková informace validní a je nutné tuto oblast ignorovat. Rozkreslení oblasti chybných pixelů je zobrazeno na Obrázek 74. Reálný snímek bez předzpracování a včetně chybné oblasti pixelů je zobrazen na Obrázek 75. Z Obrázek 74 je patrný postup odstranění oblasti nevalidních (chybných) dat v hloubkovém snímku. Za využití rovnice (8.1) lze taktéž vypočítat Depth FOV v jakékoliv vzdálenosti od senzoru.

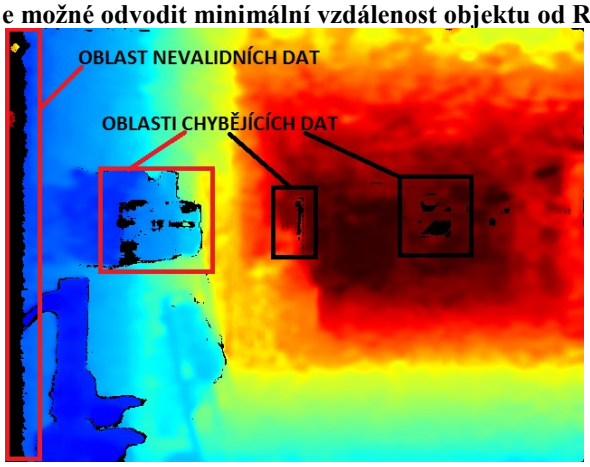
$$\text{Depth FOV} = \frac{\text{HFOV}}{2} + \tan^{-1}\left(\tan\left(\frac{\text{HFOV}}{2}\right) - \frac{\text{B}}{\text{Z}}\right) \qquad (8.1)$$

Kde:

- *Depth FOV* – Horizontální úhel pohledu hloubkového snímače
- *HFOV* – Horizontální úhel pohledu levého snímače
- *B = Baseline*: Rozteč snímačů, v případě kamery D435i = 50 mm.
- *Z* = Vzdálenost scény od hloubkové kamery (dle kalibrace a rozlišení)



Obrázek 74: Ukázka chybějící oblasti dat v hloubkové mapě (Intel, 2019)
Z Obrázek 74 je možné odvodit minimální vzdálenost objektu od RGB-D kamery.



Obrázek 75: Ukázka nevalidní oblasti a chybějících dat v surových hloubkových datech (Zdroj: Vlastní)

Tabulka 7 uvádí předdefinované rozlišení kamery a minimální vzdálenost pro odhad hloubkového snímku.

Tabulka 7: Minimální vzdálenost v závislosti na rozlišení

Minimální vzdálenost v závislosti na rozlišení Zdroj: (Intel, 2019), Upraveno	
Rozlišení	**Minimální vzdálenost (mm)**
1280x720	280
848x480	195
640x480	175
640x360	150
480x270	120
424x240	105

Snímky z kamery tuto oblast obsahují a je nutné na základě nastavení kamery tyto místa v post-processingu odstranit. Výpočet oblasti nevalidních dat lze provést dle rovnice (8.2) a (8.3)

$$\mathrm{DBR} = \frac{\mathrm{B}}{2 \cdot \mathrm{Z} \cdot \tan\left(\frac{\mathrm{HFOV}}{2}\right)} \tag{8.2}$$

$$\text{Invalid Band (Pix)} = \mathrm{HRES} \cdot \mathrm{DBR} \tag{8.3}$$

Kde:

- DBR (Depth Band Ratio) – Poměr chybných pixelů v hloubkovém snímku k užitečnému signálu
- B (Baseline) – Rozteč snímačů, v případě kamery D435i = 50 mm.
- Z – Vzdálenost scény od hloubkové kamery (dle kalibrace)
- HFOV – Horizontální úhel pohledu
- HRES – Horizontální rozlišení snímače

7.2.2 Doplnění chybějících dat v hloubkových datech "Ext Cross Fill"

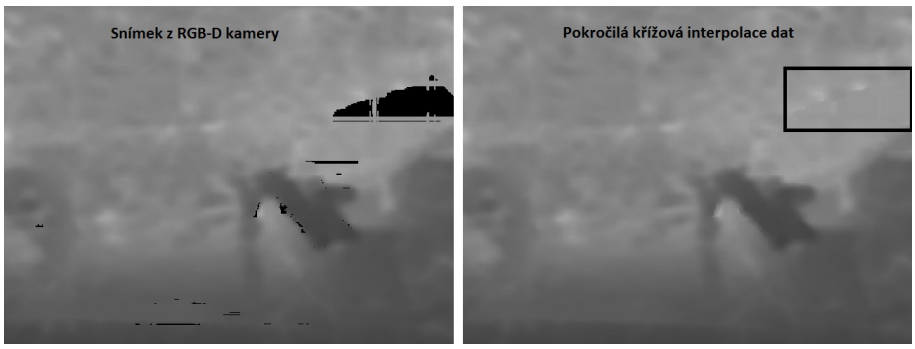
Oblasti chybějících dat, jsou místa, kde hloubková kamera nenašla infračervený vzor, například v případě odlesku od skla, záři světel či slunce. Chybějící data je nutné před použitím v neuronové síti doplnit.

Metoda Ext Cross Fill, rozšířená křížová interpolace slouží k doplnění chybějícího pixelu v závislosti na hodnotě okolních pixelu. Metoda dopočítá vážený aritmetický průměr z okolních pixelů. Vážený aritmetický průměr zajistí co nejoptimálnější odhad hodnoty chybějícího pixelu tím, že vypočítá celkovou délku trasy chybějících pixelů a největší váhu v aritmetickém průměru přiřadí nenulovému pixelu, který je k nulovému pixelu nejblíže. Nejmenší vliv na výslednou hodnotu má poté pixel s největší vzdáleností.

Pseudokód metody Extended cross fill:

Opakuj pro celou matici:

```
KROK1: pix_x,y ≠ 0 ? ANO: Další pixel, NE: KROK2
KROK2: Najdi nejbližší pixely, které mají nenulovou hodnotu
KROK3: Urči absolutní vzdálenost od okolních nenulových pixelů
KROK4: Vypočítej sumu absolutních vzdáleností od nenulových pixelů
KROK5: Urči váhu okolních nenulových pixelů – čím větší vzdálenost, tím menší
       Váha pixelu
KROK6: Vypočítej hodnotu pixelu pix_x,y na základě váženého průměru okolních
       pixelů
```



Obrázek 76: Ukázka křížové interpolace chybějících hloubkový dat (Zdroj: Vlastní)

7.3 Zarovnání RGB a hloubkového snímku

V kapitole popisující vlastnosti RGB-D kamery bylo zmíněno, že kamera D435i má dva různé typy snímače – barevný (RGB) poskytující reálné snímky a dva IR snímače, poskytující hloubkovou mapu. Bohužel tyto senzory nemají shodné rozlišení, což při dalším zpracování bude způsobovat další problémy. Aby bylo možné tyto snímky použít pro učení neuronové sítě, je nutné nejprve snímky zarovnat. Zarovnání využívá specifické parametry kamery, které jsou zapsány do paměti při kalibraci během výroby. Veskrze se jedná o projekci pozice pixelu dle globálního principiálního bodu a matice vnitřních parametrů kamery (*Camera Intrinsic*). Tyto parametry obsahují informace o konkrétním snímači, včetně ohniskové vzdálenosti

V rámci následujících rovnic (8.4) až (8.6) je použita terminologie 2D pozice pixelu $z_c = [u \quad v \quad 1]^{\mathrm{T}}$ a 3D koordináty pixelu v globálních souřadnicích $z_g = [x_w \quad y_w \quad z_w \quad 1]^T$

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R \quad T] \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = M \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{8.4}$$

$$K = \begin{bmatrix} \alpha_x & \gamma & u_0 & 0 \\ 0 & \alpha_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{8.5}$$

$$\alpha_x = f \cdot m_x; a_y = f \cdot m_y \tag{8.6}$$

Kde:

- m_x, m_y je inverze šířky a délky pixelu v projekční rovině
- f je ohnisková vzdálenost
- γ reprezentuje úhel mezi osou X a Y, ideálně rovno nule
- u_0 a v_0 reprezentuje pozici principiálního bodu, ideálně je uprostřed obrazu.

Další parametry, které je nutné pro projekci použít, jsou tzv. vnější parametry kamery (*Camera extrinsic*) *E.*

$$E = \begin{bmatrix} r_{1,1} & r_{1,2} & \mathrm{r}_{1,3} & \mathrm{T}_1 \\ \mathrm{r}_{2,1} & \mathrm{r}_{2,2} & \mathrm{r}_{2,3} & \mathrm{T}_2 \\ \mathrm{r}_{3,1} & \mathrm{r}_{3,2} & \mathrm{r}_{3,3} & \mathrm{T}_3 \\ \mathrm{S}_1 & \mathrm{S}_2 & \mathrm{S}_3 & 1 \end{bmatrix} \tag{8.7}$$

$$C = -R^{-1}P = -R^T P \tag{8.8}$$

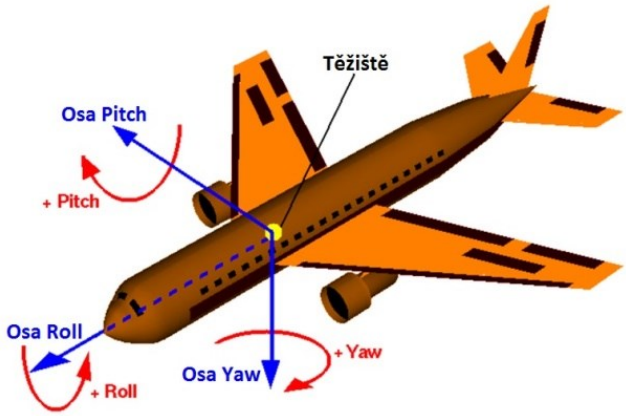
Kde:

- R_{3x3} je matice směrových kosinů (DCM), občas nazývaná rotační matice
- T_{3x1} je pozice počátku vzhledem ke globálním souřadnicím
- S_{1x3} je translační vektor vzhledem ke globálním souřadnicím. Translační vektor se v tomto zápise používá pro transformaci mračna bodů, například při použití metod registrace mračna bodů (ICP, NDT, …)
- C je pozice kamery vzhledem ke globálním souřadnicím

7.3.1 Matice směrových kosinů (DCM)

Obecně lze orientaci tělesa v prostoru popsat vektorem tří úhlových natočení $\Theta_x, \Theta_y, \Theta_z$, kolem jednotlivých os *X, Y, Z*. Tento výhodný zápis otočení tělesa v prostoru zavedl již v polovině 18. století Leonard Euler, po němž se příslušné úhly natočení nazývají Eulerovy úhly. Matice směrových kosinů umožňují transformaci úhlového zrychlení při otáčení bázového prostoru. Dají se velmi výhodně použít pro určení aktuální polohy

robotické platformy v prostoru či provedení rotace mračna bodů. Těchto matic taktéž využívají algoritmy registrace mračna bodů



Obrázek 77: Ukázka Eulerových úhlů (Beran, a další, 2014)

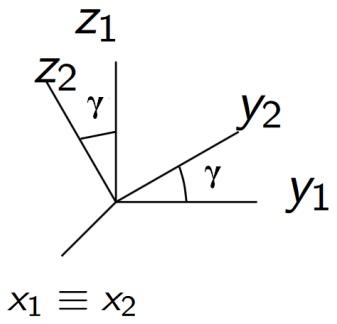
$$\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \tag{8.9}$$

$$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{8.10}$$

$$\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{8.11}$$

$$\mathbf{R} = \mathbf{R}_x(\gamma) \cdot \mathbf{R}_y(\beta) \cdot \mathbf{R}_z(\alpha) \tag{8.12}$$

Výše uvedené matice (8.9), (8.10), (8.11) se nazývají rotačními maticemi, které přísluší rotacím kolem jednotlivých os. Např. pomocí matice (8.9) lze provést rotaci prostoru S_1 kolem osy X, přičemž vzniká nový stavový prostor S_2, pootočený vůči stavovému prostoru S_1 o úhel γ. Rotaci okolo osy X o úhel γ lze vyjádřit graficky, jako je zobrazeno na Obrázek 78



Obrázek 78: Rotace okolo osy X o úhel γ (Bezděk, 2019)

Při rotaci okolo osy x dochází k přechodu z prostoru S_1 do prostoru S_2. Pro zjednodušení v následujících rovnicích bude zobrazena rotace vektoru $\mathbf{V_1}$. Rotace okolo osy X je popsána rovnicí (8.14), rotace okolo osy Y je popsána rovnicí (8.15) a rotace okolo osy Z je definována rovnicí (8.16).

$$\mathbf{V}_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{8.13}$$

$$\mathbf{V}_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \mathbf{R}_x(\gamma) \cdot \mathbf{V}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{8.14}$$

$$\mathbf{V}_3 = \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = \mathbf{R}_y(\beta) \cdot \mathbf{V}_1 = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{8.15}$$

$$\mathbf{V}_4 = \begin{bmatrix} x_4 \\ y_4 \\ z_4 \end{bmatrix} = \mathbf{R}_z(\alpha) \cdot \mathbf{V}_1 = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{8.16}$$

Pro případ rotace vektoru$\mathbf{V_1}$ okolo všech tří os, lze použít matici **R,** která poskytuje rotaci o definované úhly α, β, γ. Tato rotace je popsána rovnicí (8.17).

$$\mathbf{V}_5(\alpha,\beta,\gamma) = \begin{bmatrix} x_5 \\ y_5 \\ z_5 \end{bmatrix} = \mathbf{R}_x(\gamma) \cdot \mathbf{R}_y(\beta) \cdot \mathbf{R}_z(\alpha) \cdot \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{8.17}$$

Využitím rovnic o Eulerových úhlech je možné z rotační matice, respektive transformační matice získat úhly rotace *yaw* (ψ), *pitch* (θ) a *roll* (ϕ) dle rovnice (8.18).

$$\begin{aligned} \psi &= \tan^{-1}\left(\frac{r_{1,2}}{r_{1,1}}\right) \\ \theta &= -\sin^{-1}(r_{1,3}) \\ \phi &= \tan^{-1}\left(\frac{r_{2,3}}{r_{3,3}}\right) \end{aligned} \tag{8.18}$$

7.4 Normalizace hloubkového snímku

Pro zpracování hloubkového snímku bylo nutné provést normalizaci naměřených dat. Hloubková kamera totiž v hloubkovém snímku vrací 16 bit hodnotu pixelu, ve které je zakódována vzdálenost tak, že hodnota pixelu 10 000 je ekvivalentní vzdálenosti 10 000 mm od kamery. Použití 16 bitového čísla pro učení je ovšem nevýhodné, nejen

že při použití 16 bitového čísla je využito pouze přibližně 15 % jeho rozsahu [5]. Pro učení neuronové sítě tedy postačuje využít pouze 8bitové hloubky čísla, tj. hodnoty 0 až 255. Výsledkem je tedy výrazně optimalizovanější stavový prostor, nevýhodou je ovšem snížení rozlišení odhadu polohy a to přibližně 39× (1 mm vs. 39,1 mm). Pro základní odhad hloubkového snímku je krok 3,9 cm bohatě postačující. V případě vyšší přesnosti je možné omezit před učením maximální možnou vzdálenost a tím provést přepočítání dle rovnic uvedených níže. Rovnice (8.19) a (8.20) udávají postup normalizace hloubkového snímku.

$$\Delta s = \frac{D_{MAX}}{2^{bit}-1} = \frac{10000}{255} \sim 39.1\ mm \tag{8.19}$$

$$\mathrm{D_{8bit}} = \mathrm{round}\left(\frac{D_{16bit}}{\Delta s}\right) \tag{8.20}$$

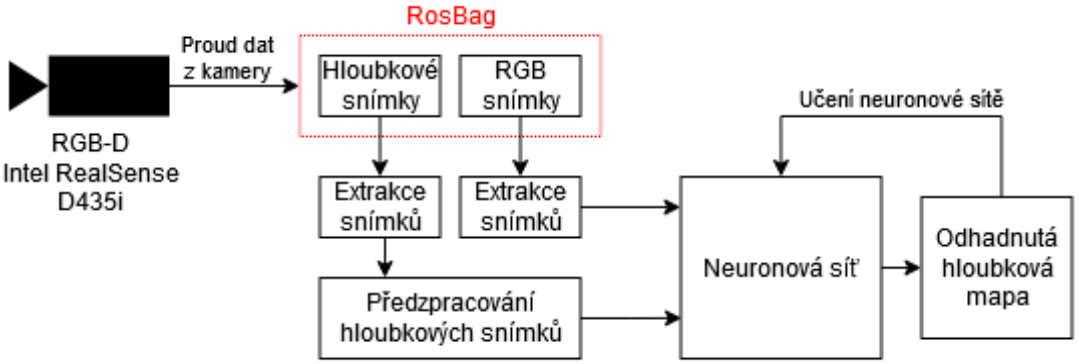
Kde:

- Δs je výsledná hodnota rozlišení
- D_{MAX} je maximální dosah hloubkové kamery v mm
- bit je bitová hloubka čísla, v tomto případě 8
- $\mathrm{D_{8bit}}$ je výsledná normalizovaná vzdálenost
- D_{16bit} je původní vzdálenost získaná z hloubkové kamery v mm
- $round$ je operace zaokrouhlení na nejbližší celé číslo

[5] U použité hloubkové kamery je vzdálenost větší než 10 000 mm automaticky nastavena na přednastavenou hodnotu (dle konfigurace).

8 Trénování neuronových sítí

Veškeré architektury a trénování neuronových sítí bylo vytvořeno v jazyce Python. Základním důvodem byla možnost použití prostředí TensorFlow, které podporuje jazyk Python. Jazyk python je pro trénování neuronových sítí velice přívětivý, protože poskytuje velkou abstrakci od samotného programování konkrétních vrstev a řešení datových typů, místo toho se uživatel může zaměřit přímo na tvorbu dané architektury. V neposlední řadě byl jazyk Python taktéž vybrán z důvodu, že existuje velké množství návodů, jak daný problém správně řešit, včetně ukázkových řešení různých architektur neuronových sítí.



Obrázek 79: Vývojový graf předzpracování hloubkových snímků a učení neuronové sítě (Zdroj: Vlastní)

Na Obrázek 79 je zobrazen celý řetězec, který je nezbytný pro přípravu dat a učení neuronové sítě. Okolní prostředí je snímáno pomocí hloubkové kamery, jejíž výstup je jako proud dat ukládán do speciálního souboru nazývaného *RosBag*. Před samotným učením je nutné provést extrakci ze souboru a předzpracování dat. Předzpracování dat se provádí jen u hloubkových snímků, jelikož surová data jsou nevhodná pro učení neuronové sítě. V předzpracování se provádí doplnění chybějících pixelů a normalizace hloubkového snímku. Podrobný postup předzpracování je detailně popsán v kapitole 7. Výstupem pro učení neuronové sítě je sada dat obsahující normalizovaný hloubkový snímek a korespondující barevný snímek.

8.1 Datová sada

Kvalita učení a složitost neuronové sítě byly klíčové parametry pro další výběr vhodné neuronové sítě.

Učení neuronových sítí probíhalo pro všechny testované neuronové sítě shodně. Nasnímané snímky bylo nutné nejdřív předzpracovat, jak je popsáno v kapitole 7, a zejména snímky decimovat. Z celkové sady surových naměřených snímků (přibližně 150 000 sad) bylo po decimaci použito přibližně 30 % snímků. Decimace byla prováděna ekvidistantně (tj. pro trénovací data byl vybrán každý třetí snímek z celé snímané datové sady).

Tabulka 8: Rozdělení datové sady pro učení a testování neuronové sítě

Typ	Procentuální zastoupení	Počet snímků (RGB+D)
Trénovací data	66,6 %	30 000 dvojic
Validační data	16,6 %	7 500 dvojic
Testovací data	16,6 %	7 500 dvojic
Celkem	100 %	45 000 dvojic

Při zvolené vzorkovací frekvenci 15 snímků za vteřinu, byla po aplikaci decimace snímků časová změna mezi snímky přibližně 0,2 vteřiny (5 fps), což pro výsledné zpracování navigačních dat postačuje, jelikož rychlost pohybu vozíku byla velice pomalá, maximálně 1 m/s, většinou ale méně. Podobně byly decimovány snímky pro validační a testovací datové sady.

Předzpracovaná data byla rozdělena na tři balíčky – trénovací data, validační a testovací data. Konkrétní počty jsou uvedeny v Tabulka 8.

Pro jednodušší sledování výsledků učení byla trénovací datová sada rozdělena na menší sady. Jedním z důvodů rozdělení byla kontrola experimentálních nastavení sítě, zda nedošlo k zablokování sítě v lokálním minimu, což se během experimentů se sítěmi nesčetněkrát stalo. Dalším důvodem byla kontrola kvality výsledků bez zbytečné ztráty času učením špatně nastavené sítě. Trénovací a validační sada byla rozdělena na deset menších balíčků o stejném počtu. Tj. jedna trénovací sada obsahovala 3000 dvojic snímků (3000 RGB snímků a 3000 příslušných hloubkových snímků) a 750 dvojic validačních snímků. Tyto balíčky byly postupně aplikovány na všechny navržené neuronové sítě. Veskrze se dá říct, že každý balíček obsahoval snímky z jedné až tří tras měření (trasy při snímání dat byly různě dlouhé, tj. obsahovaly jiný počet snímků) přičemž například v učení č. 1 byly snímky z prvních dvou měřených tras a třetina ze třetí trasy, učení č. 2

obsahovalo zbylé dvě třetiny ze třetí trasy apod. Stejným postupem byla vytvořena sada validačních snímků. Validační snímky byly různé od trénovacích dat.

8.2 Parametry trénování neuronových sítí

Trénování i testování neuronové sítě bylo prováděno na stroji, jehož parametry jsou uvedeny v Tabulka 9. Výpočty byly prováděny za využití technologie CUDA na grafické kartě. Parametry programového vybavení jsou uvedeny v Tabulka 10.

Tabulka 9: Konfigurace výpočetního počítače

Parametr	
CPU	Intel Core I5-7400T
RAM	16 GB DDR4
GPU	Nvidia GeForce GTX1070 8GB DGGR5 1506MHz GPU clock 1683MHZ CPU clock 1920 CUDA jader

Tabulka 10: Konfigurace software

Parametr	
Operační systém	Windows 10
Ovladače GPU	Nvidia 457.09
Vývojový SW	Pycharm 2020.1 Community
Python	3.7
TensorFlow	2.1
TensorFlow GPU	2.3.1
Keras	2.3.1
Numpy	1.18.4
OpenCV python	4.2.0.34
Kreslení grafu python	Graphwiz 0.16

Veškeré parametry trénování neuronových sítí, až na parametr velikost dávky (*batch*), byly shodné. Konkrétní hodnota velikosti dávky je uvedena v Tabulka 11. Velikost dávkového učení musela být s ohledem na omezenou velikost operační paměti počítače proměnná a hodnota byla určena experimentálně. Všechny neuronové sítě byly nastaveny tak, že dimenze vstupních snímků byly o rozměrech $256 \times 256 \times 3$ pixelů. Každý barevný kanál měl bitovou hloubku 8 bit. Neuronové sítě vytvářely výstupní snímky o shodných rozměrech jako vstupní tj. 256×256, bitová hloubka byla opět 8 bitů.

Hloubkové snímky byly navíc před aplikací do neuronové sítě předzpracované a normalizované. Postup normalizace hloubkového snímku je uveden v kapitole 7.4. Pro lepší učení samotných neuronových sítí byly vstupní snímky před učením navíc normalizovány z původního intervalu $\langle 0{,}255 \rangle$ na interval $\langle 0{,}1 \rangle$. Všechny neuronové sítě vracely taktéž hodnoty hloubkového snímku v intervalu $\langle 0{,}1 \rangle$. Převod na interval $\langle 0{,}255 \rangle$ je prováděn až v postprocessingu.

Pro učení všech neuronových sítí byly použity parametry trénování uvedené v Tabulka 12. Celá datová sada byla rozdělena do 10 sad, z nichž každá sada obsahovala

3000 párů snímků, každá sada obsahovala 750 párů validačních snímků. Pro učení byl nastaven počet epoch na hodnotu 100.

Tabulka 11: Nastavení velikosti dávky zpracování dle konkrétní neuronové sítě

Typ sítě	Podtyp sítě	Velikost dávky	Počet epoch trénování	Rozměr vstupní vrstvy	Rozměr výstupní vrstvy
ResNet	12, 18,50	12	10×100	$256 \times 256 \times 3$	256×256
	101	8			
	200	3			
U-Net	Základní, v.3	8			
FCN	8	12			
SegNet	Není	4			
BiSeNet	Není	2			
RefineNet	Není	4			
PSP-Net	v.3	3			
DenseNet	Není	10			
Xception	Není	3			

Tabulka 12: Parametry trénování

Parametr trénování	Hodnota
Počet opakování učení	10
Počet epoch	100
Počet vzorků trénování v jednom učení	3000 dvojic
Počet vzorků validace v jednom učení	750 dvojic
Optimalizační algoritmus	ADAM
Parametry optimalizačního algoritmu ADAM	$\beta_1 = 0.9,\ \beta_2 = 0.999, \varepsilon = 10^{-7}$
Rychlost učení	10^{-4}
Inicializace vah	Normální rozdělení, $\mu = 0$, $\sigma = \sqrt{\frac{2}{počet\ vstupů\ do\ vrstvy}}$
Kriteriální funkce	Binární křížová entropie (B*inary_cross_entropy*)
Metrika	Přesnost (A*ccuracy*)

Při trénování byl jako optimalizační algoritmus použit algoritmus ADAM, který byl vyhodnocen jako pravděpodobně nejvhodnější pro trénování konvolučních neuronových sítí, (Kingma, a další, 2015), (Nanni, a další, 2021), (Bushaev, 2021).

Parametry optimalizačního algoritmu ADAM byly ponechány defaultní, které jsou přednastaveny ve frameworku TensorFlow (TensorFlow, 2019a). Pro učení byla zpočátku nastavena rychlost učení na hodnotu 10^{-3}, ovšem při experimentování velice často docházelo k zablokování v lokálním minimu. Z výše uvedeného důvodu rychlost učení byla snížena na hodnotu 10^{-4}.

Pro inicializaci vah v jednotlivých vrstvách bylo vybráno normální rozdělení se střední hodnotou rovnou nule. Směrodatná odchylka je vypočítána dle počtu vstupných proměnných do konkrétní vrstvy. Podrobnější informace lze získat v dokumentaci ke frameworku (TensorFlow, 2019b).

Jako chybová funkce byla použita binární křížová entropie (*Binary Cross Entropy*), která byla vyhodnocena jako nejvhodnější chybová funkce pro klasifikaci pixelu (Zhang, a další, 2018), (Ruby, a další, 2020).

Měření snímků pro trénování neuronových sítí bylo provedeno v reálném prostředí autoservisu. Nejednalo se o striktně laboratorní podmínky a snímky byly nedokonalé. Nejčetněji se v datové sadě vyskytovaly lehce rozmazané snímky, které vznikly během otáčení platformy při slabém osvětlení. Měření bylo prováděno jak ve vnitřním prostředí, tak ve vnějším prostředí. Měření bylo taktéž prováděno za různých světelných podmínek (slunečno, zamračeno, umělé osvětlení, přítmí). Pro zvýšení robustnosti učení neuronové sítě, zejména s ohledem na reálné průmyslové prostředí bylo pro trénování použita augmentace dat, jejíž parametry jsou uvedeny v Tabulka 13.

Tabulka 13: Parametry augmentace dat

Operace	Hodnota
Rotace	-10° až 10°
Posun vzoru – Osa X, Y	80% až 120%
Zkosení	80% až 120%
Přiblížení/ Oddálení	80% až 120%
Jas	50% až 150%
Horizontální zrcadlení	ANO
Doplnění pixelů	Dle nejbližší hodnoty
Normalizace snímku	‹0;1›

8.3 Augmentace trénovacích dat

Augmentace trénovacích dat se využívá v procesu učení neuronových sítí pro zajištění větší datové sady v případě, kdy je problematické zajistit velké množství trénovacích vzorů. Dále augmentace trénovacích dat slouží k zajištění robustnosti učení neuronové sítě. V procesu učení je tedy vhodné toto rozšíření využít.

Princip učení neuronové sítě je založen na použití jedinečného vzoru v průběhu učení pouze jedenkrát. Augmentace je založena na definované deformaci vstupního vzoru, čímž dojde k vytvoření N-kopií vstupního vzoru, přičemž každý má jiné parametry – např.

jiný úhel rotace, rozostření, změnu kontrastu apod. Ukázka kombinace metod augmentace dat je zobrazena na Obrázek 80.

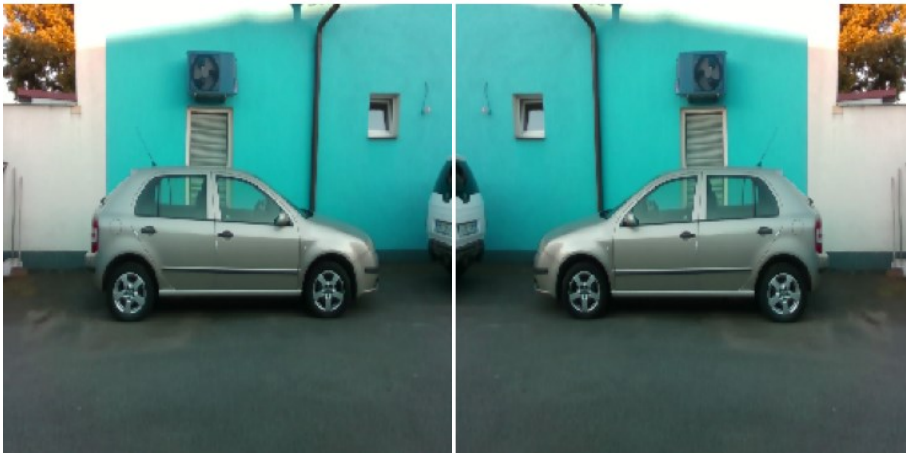
Obrázek 80: Ukázka kombinace augmentace dat. Vlevo originální snímek, vpravo upravený. (Zdroj: Vlastní)

V této kapitole jsou popsány základní použité metody pro trénování neuronové sítě včetně jednoduchého kódu, jak za využití knihovny Keras Tensorflow v jazyce Python nastavit generátor. V případě učení neuronové sítě a použití augmentačních metod je ovšem nutné používat vždy stejné semínko (*seed*) generátoru pro RGB snímek a ty samé metody použít na hloubkový snímek.

8.3.1.1 Zrcadlení vzoru

Metoda zrcadlení je velice jednoduchá metoda, kdy nově vytvořený vzor má shodné rozměry jako vzor vstupní, a v podstatě jde o osovou souměrnost, kterou lze využít ve všech osách definovaného prostoru, v tomto případě se jedná o osy x a y. Tato metoda je jednou z nejbezpečnějších metod pro trénování, protože nedochází ke změně histogramu pixelů v obrázku. Nastavení generátoru v jazyce Python je uvedeno níže. Generátor při uvedeném nastavení provede s každým vzorem otočení okolo osy x a y. Výsledkem je, že generátor pro učení použije základní obrázek bez jakýchkoliv změn a poté je neuronové síti předložen snímek, který obsahuje kombinace zrcadlení – zrcadlení podle osy x, zrcadlení podle osy y a poté zrcadlení podle obou os x a y. Ve výsledku z jednoho snímku je neuronové síti předložen identický snímek ve 4 různých variantách. V případě trénování neuronové sítě pro rozpoznání hloubky je použito pouze horizontální zrcadlení. Vertikální zrcadlení nemá smysl používat, protože podlaha bude vždy dole.

```
        Depth_dataGen = ImageDataGenerator( horizontal_flip = True, vertical_flip =
False)
```



Obrázek 81: Ukázka zrcadlení při augmentaci dat. Vlevo originální snímek, vpravo upravený. (Zdroj: Vlastní)

8.3.1.2 Zkosení vzoru

Metoda zkosení vzoru slouží ke geometrické deformaci obrazu. Generátor vytvoří kosodélník o definovaném procentuálním zkreslení.

V tomto případě není nutné používat extrémní hodnoty zkosení, postačuje hodnota 20 %. Doplnění chybějících pixelů vzniklých geometrickou transformací je opět dle nejbližší hodnoty pixelu.

```
Depth_dataGen = ImageDataGenerator(shear_range = 0.2, fill_mode='nearest')
```

8.3.1.3 Posunutí vzoru

Metoda posunutí obrazu je založena na posunu vstupního vzoru o definované množství pixelů v ose *x* a ose *y*. Posuv obrazu zajistí robustnost sítě a předchází naučení určitého vzoru umístěného v jedné poloze. Dále tato metoda je vhodná pro případ pohybující se scény (například pohyb osob v obraze apod.).

```
Depth_dataGen = ImageDataGenerator(width_shift_range = 0.2, height_shift_range
                = 0.2, fill_mode='nearest')
```

V tomto případě je nastaven pohyb vstupního vzoru v horizontálním i vertikálním směru o ±20% rozlišení vstupního vzoru. Chybějící hodnoty vlivem posuvu jsou doplněny dle hodnoty nejbližšího pixelu. Výsledkem je snímek zobrazený na Obrázek 82, pravá část. Doplněné pixely jsou označeny červeným rámečkem.

Obrázek 82: Ukázka horizontálního posunu při augmentaci dat. Vlevo originální snímek, vpravo upravený. (Zdroj: Vlastní)

8.3.1.4 Rotace vzoru

V případě použití rotace vstupního vzoru je možné nastavit úhel rotace a osu, okolo bude obraz rotován. Tato metoda ovšem vytvoří oblasti, které jsou nedefinované, resp. v původním vzoru neexistují. Je nutné tedy nastavit způsob doplnění pixelů. Nejčastěji se používá metoda doplnění hodnoty dle nejbližšího pixelu, či se použije oříznutí.

V případě učení neuronové sítě pro odhad hloubky nemá smysl používat rotaci vzoru okolo celých 360°, stačí použít rotaci v rozsahu přibližně ±20°. V učení nepředpokládáme, větší náklon. Hodnota ±20° byla zvolena i s ohledem na potenciální otočení kamery. Doplnění chybějících hodnot je dle hodnoty nejbližšího pixelu.

```
Depth_dataGen = ImageDataGenerator(rotation_range = 20, fill_mode='nearest')
```

Obrázek 83: Ukázka rotace o ±20° při augmentaci dat. Vlevo rotace o -20°, vpravo rotace o +20°. (Zdroj: Vlastní)

8.3.1.5 Změna měřítka vzoru

Změna měřítka slouží k zvětšení/zmenšení vstupního vzoru dle požadovaného měřítka. Pro interpolaci pixelů existuje několik využívaných metod. Nejčastěji využívanou

je metoda typu nejbližší pixel, dále lineární doplnění kubické či bikubické interpolace. Výsledný obraz si zachová vstupní dimenze vzoru tím, že v případě zvětšení dojde k oříznutí výsledného vzoru. V případě zmenšení je použita některá z výše uvedených metod doplnění chybějících pixelů.

V tomto případě učení neuronové sítě nemá smysl používat přiblížení vzoru ve velkém rozsahu. Byla zvolena hodnota 0.2. Generátor použije měřítko v rozsahu 1±0.2 z čehož vyplývá, že vstupní vzor je zmenšen až na 80% původní hodnoty a zvětšen na 120% původního rozměru.

```
Depth_dataGen = ImageDataGenerator(zoom_range = 0.2)
```

Obrázek 84: Ukázka přiblížení o ±20% při augmentaci dat. Vlevo originální snímek, vpravo po úpravě. (Zdroj: Vlastní)

8.3.1.6 Změna jasu vzoru/ Změna barvy vzoru

Změna jasu vzoru je jednou z důležitých metod. Při měření vstupních dat je složité zaručit shodné světelné podmínky. Při měření trénovacích dat v reálném prostoru docházelo občas k velkým změnám jasu (např. přejezdy mezi budovami). Neuronová síť by si měla poradit i s těmito změnami, proto je vhodné zařadit změnu jasu. Ve výsledku je neuronová síť „nucena“ se učit zejména na tvarech než na barvě jednotlivých objektů.

V tomto případě učení neuronové sítě nemá smysl používat extrémní rozsah změny jasu vstupního vzoru, hodnota 1±0.5, což je ekvivalent snížení jasu na 50% původní hodnoty a zvýšení jasu na 150% původní hodnoty, se zdá být dostačující.

```
Depth_dataGen = ImageDataGenerator(brightness_range = [0.5, 1.5])
```

Obrázek 85: Ukázka změny jasu při augmentaci dat. Vlevo originální snímek, vpravo upravený. (Zdroj: Vlastní)

8.3.1.7 Přidání šumu

Přidání šumu opět způsobuje zvýšení robustnosti trénované neuronové sítě. Aditivní šum zajišťuje lepší generalizaci vytvořeného modelu, protože algoritmus se neučí pouze na jeden konkrétní prvek v obraze, ale je schopen se vypořádat s lehkými změnami. Nejpoužívanějším typem aditivního šumu je šum typu „*salt and pepper*“ či bílý Gaussovský šum.

9 Popis úprav neuronových sítí a vyhodnocení výsledků

V rámci výzkumu bylo nutné základní modely sítí navržených jejími autory upravit. Úpravy modelů neuronových sítí v rámci tohoto výzkumu jsou popsány v této kapitole.

Důvodů, v některých případech celkem rozsáhlých úprav, bylo několik. Při tvorbě základních modelů neuronových sítí byly stanoveny následující požadavky, které modely sítí musely splnit.

1) Pro vstup neuronové sítě bude použit RGB snímek
2) Výstup neuronové sítě bude hloubkový snímek v odstínech šedi
3) Bude použita 8bit hloubka výstupní vrstvy. Hodnota pixelu bude korespondovat s normalizovanou vzdáleností
4) Rozměry výstupní vrstvy musí být shodné s rozměrem vstupní vrstvy. Z důvodu výpočetní náročnosti a zejména omezené velikosti operační paměti na grafické kartě a v počítači byl zvolen rozměr 256x256

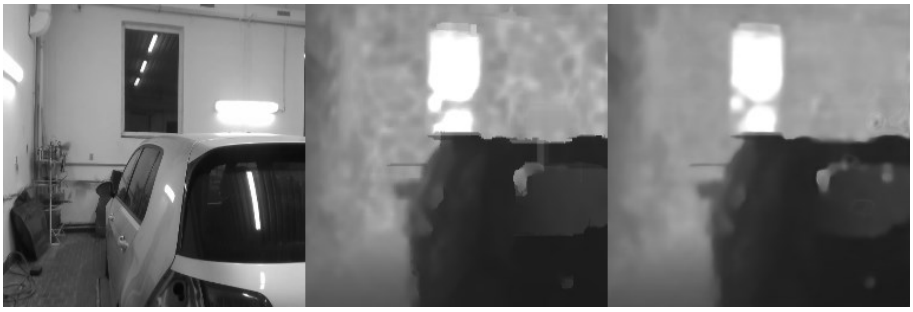
Při zkoumání bylo zjištěno, že nejsou navrženy architektury neuronových sítí, které by splňovaly výše uvedené podmínky. Používané modely veskrze poskytují pouze segmentační mapu s malým počtem kategorií, případně se jedná o zejména o klasifikační neuronové sítě pro rozpoznávání scény. Zkoumané modely neuronových sítí jsou detailně popsány v kapitole 4.2.

9.1 Úpravy a výsledky sítě PSPNet

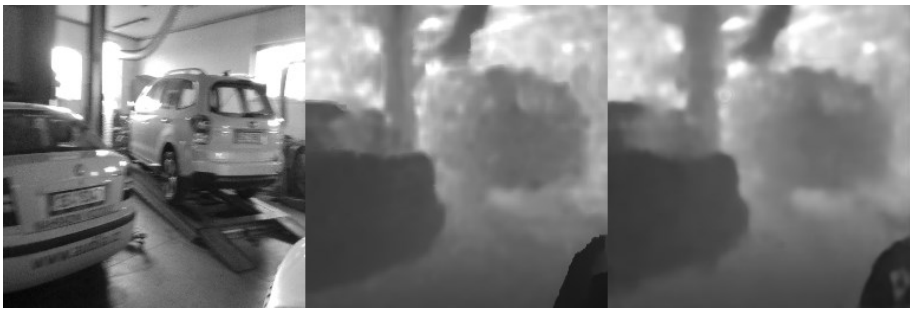
Pro splnění výše definovaných požadavků musel být základní model sítě *PSPNet* modifikován následujícím způsobem: Jako mateřská síť byla použita struktura sítě Resnet101, pro její kvalitní výsledky a zároveň menší výpočetní náročnost než například Resnet200 či U-Net. Matice příznaků ze sítě Resnet101 byla následně zpracována čtyřmi paralelními větvemi, z nichž každá větev obsahovala jinou velikost konvolučních matic. Jelikož byly u všech neuronových sítí zachovávány stejné dimenze výstupních snímků jako vstupních, které byly 256x256 pixelů, není možné přímo aplikovat výstup sítě *ResNet101* do struktur *PSPNet.* Před aplikací bylo nutné použít vrstvu *AveragePool2D* pro přizpůsobení, jelikož velikost matic byla zvolena dle návrhu (Zhao, a další, 2017) jako čtyř-úrovňové matice o rozměru 1×1, 2×2, 3×3 a 6×6. Obecně, rozměr matic

je možno zvolit téměř jakýkoliv, ovšem s ohledem na dimenzi vstupní matice a celočíselný výsledek po dělení rozměrem konvoluční matice.

Obrázek 86: Ukázka výsledku učení neuronové sítě PSPNet. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)



Obrázek 87: Ukázka výsledku učení neuronové sítě PSPNet. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)



Obrázek 88: Porovnání výsledků sítě PSPNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě PSPNet (Zdroj: Vlastní)

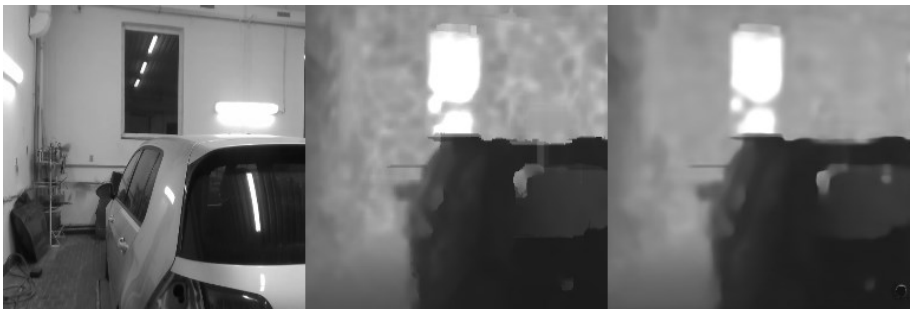
Obrázek 89: Porovnání výsledků sítě PSPNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě PSPNet (Zdroj: Vlastní)

Na Obrázek 86 je zobrazen průběh učení dle epoch a je patrný postupný progres odhadu výsledné hloubkové mapy. Na Obrázek 86 až Obrázek 89 jsou zobrazeny výstupy z testovací množiny dat. Tyto snímky nebyly neuronové síti během učení přeloženy, tak jsou velice dobrou ukázkou kvality učení neuronové sítě. Na Obrázek 89 je vidět postupný vývoj odezvy neuronové sítě na vstupní RGB snímek průběhu epoch učení. Po první epoše je vidět, že kvalita odezvy sítě je velice špatná a není možné cokoliv v obraze rozpoznat. Po čtyřech epochách je již možné rozpoznat i hrubý obraz automobilu, ale bez jakýchkoliv detailů. Výsledný snímek je velice kvalitní a téměř odpovídá originálu, který ovšem obsahuje velké množství šumu. Neuronová síť tyto vysokofrekvenční složky, jako je šum, není schopna odhadnout a výsledný snímek tento šum má výrazně potlačen. Podrobnější výsledky získané vyhodnocením statistických parametrů jsou popsány v kapitole 10.

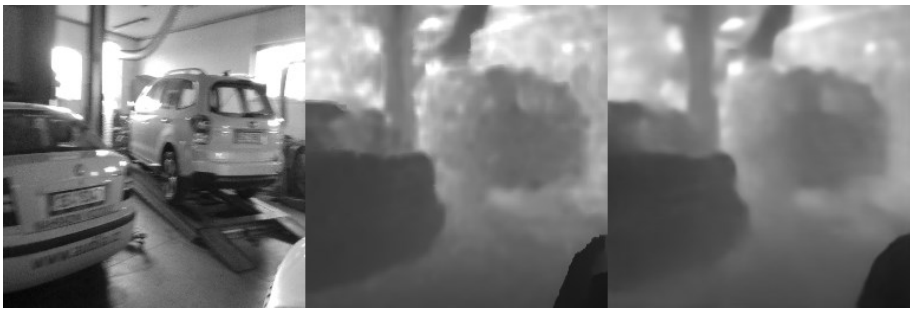
9.2 Úpravy a výsledky sítě SegNet

Základní struktura sítě *SegNet* je podobná struktuře *U-Net*. Pro splnění výše stanovených požadavků provést změny zejména v klasifikační části neuronové sítě, a to způsobem popsaným v této kapitole. Základní model neuronové sítě poskytuje pouze omezené množství klasifikačních tříd, proto bylo nezbytné počet klasifikačních tříd rozšířit. Dalším problémem základní sítě je menší výstupní rozměr než matice vstupní. Tento problém byl odstraněn vhodným nastavením vnitřních koeficientů sítě, jako je například změna pixelového posunu (*stride*) v nastavení konvoluční vrstvy a volbou správných kombinací převzorkování (*upsampling*) a konvoluční neuronové sítě. Model sítě byl nesčetněkrát upraven v rámci výzkumu. Zejména bylo nutné zvolit správnou aktivační funkci u konvolučních vrstev. Nejčastěji používaná aktivační funkce *sigmoida*, musela být v rámci sítě téměř vždy nahrazena aktivační funkcí *ReLu*. Při ponechání aktivační funkce *sigmoida*, docházelo k zablokování sítě v lokálním minimu a síť se nebyla schopna naučit a použitými optimalizačními algoritmy nebylo možné se z lokálního minima vymanit.

Obrázek 90: Ukázka výsledku učení neuronové sítě SegNet. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)



Obrázek 91: Porovnání výsledků sítě SegNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě SegNet (Zdroj: Vlastní)



Obrázek 92: Porovnání výsledků sítě SegNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě SegNet (Zdroj: Vlastní)

Obrázek 93: Porovnání výsledků sítě SegNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě SegNet (Zdroj: Vlastní)

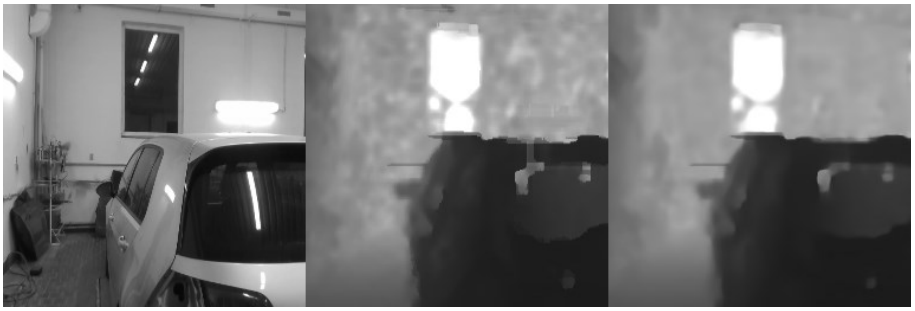
Na Obrázek 90 až Obrázek 93 jsou zobrazeny výstupy z testovací množiny dat. Tyto snímky nebyly neuronové síti během učení přeloženy, tak jsou velice dobrou ukázkou kvality učení neuronové sítě. Na Obrázek 90 je vidět postupný vývoj odezvy neuronové sítě na vstupné RGB snímek průběhu epoch učení. Po první epoše je vidět že kvalita odezvy sítě je relativně kvalitní, a dokonce je vidět i hrubý tvar automobilu. Po čtyřech epochách

je již možné rozpoznat i oblast s oknem, ale nikoliv kvalitní. Výsledný snímek je velice kvalitní a téměř odpovídá originálu, který ovšem obsahuje velké množství šumu. Neuronová síť tyto vysokofrekvenční složky, jako je šum, není schopna odhadnout a výsledný snímek tento šum má výrazně potlačen. Podrobnější výsledky získané vyhodnocením statistických parametrů jsou popsány v kapitole 10.

9.3 Úpravy a výsledky sítě BiSeNet

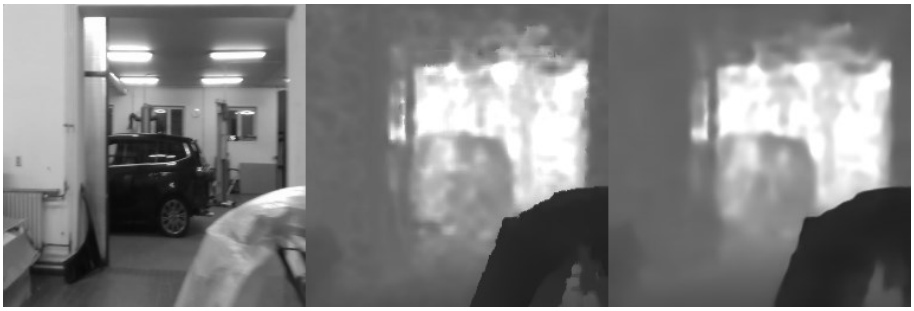
Architektura sítě *BiSeNet* byla pro plnění výše uvedených požadavků upravena jen minimálně. Jednalo se zejména o úpravy ve vstupní a výstupní vrstvě pro přizpůsobení architektury sítě vstupním snímkům. Ostatní parametry byly zachovány tak, jak byly navrženy autory architektury sítě (Yu, a další, 2018).

Obrázek 94: Ukázka výsledku učení neuronové sítě BiSeNet. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)



Obrázek 95: Porovnání výsledků sítě BiSeNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě BiSeNet (Zdroj: Vlastní)

Obrázek 96: Porovnání výsledků sítě BiSeNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě BiSeNet (Zdroj: Vlastní)

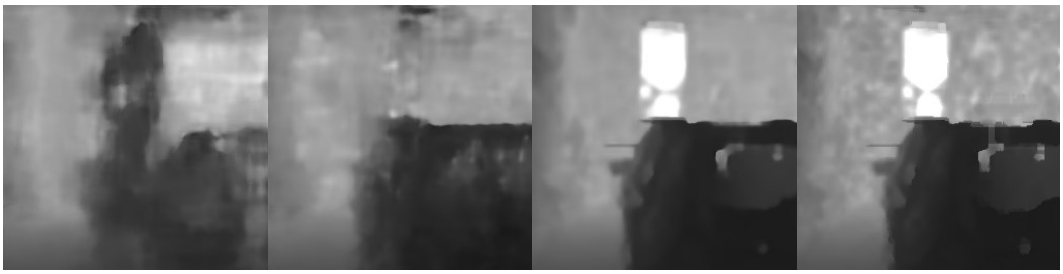


Obrázek 97: Porovnání výsledků sítě BiSeNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě BiSeNet (Zdroj: Vlastní)

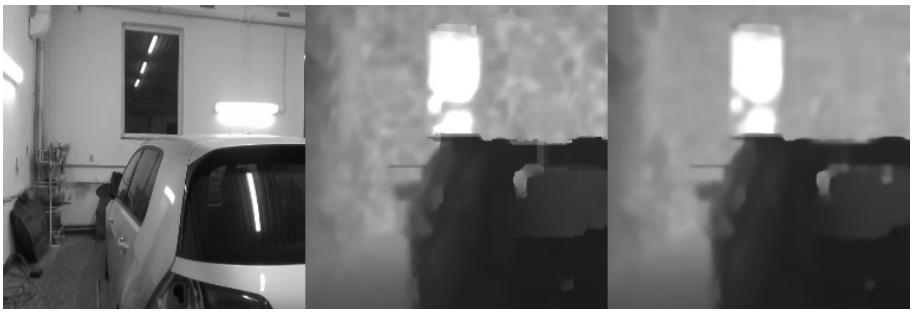
Na Obrázek 94 je zobrazen pokrok učení neuronové sítě *BiSeNet* u shodného vstupního snímku. Tento snímek nebyl síti při učení překládán, pouze byl vyhodnocen za využití koeficientů učení po jedné epoše (snímek úplně vlevo), po čtyřech epochách (snímek druhý zleva) a po devíti epochách (třetí zleva). Snímek úplně vpravo je korespondující snímek získaný z hloubkové kamery. Z Obrázek 94 je patrné, že neuronová síť *BiSeNet* po první epoše provedla celkem kvalitní odhad, jelikož je patrný hrubý obrys automobilu. Po čtyřech epochách je už obrys automobilu zřetelný, ovšem vzdálenější objekty, v tomto případě okno není rozpoznáno. Po devíti epochách je výsledná hloubková mapa téměř identická s hloubkovou mapou změřenou pomocí hloubkové kamery. Pro porovnání výsledku učení je na Obrázek 95 zobrazen vstupní barevný snímek, který je zde zobrazen v odstínech šedi (vlevo), uprostřed je snímek z hloubkové kamery a vpravo je hloubkový snímek vytvořený neuronovou sítí *BiSeNet*. Pro ilustraci výsledků sítě byly přidány Obrázek 96 a Obrázek 97.

9.4 Úpravy a výsledky sítě FCN

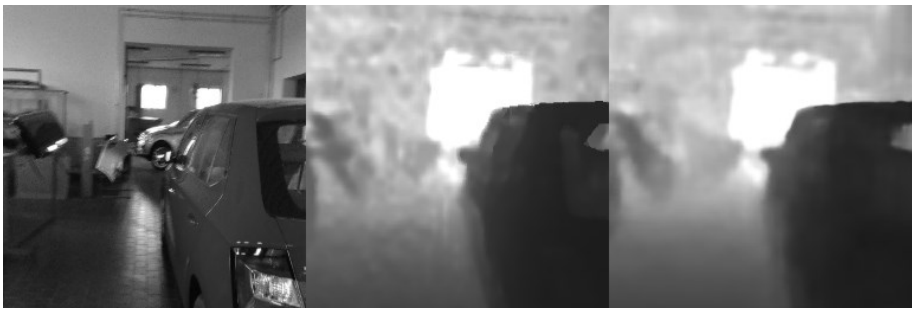
V této práci byla síť *FCN* upravena z důvodu výše stanovených požadavků, které základní architektura nesplňovala. Základní změna byla taková, že jako první vrstvy byly použity identické a reziduální bloky ze sítě pro lepší extrakci příznaků ze vstupního snímku. Dále byla architektura upravena tak, že počet kategorií byl stanoven na celkem 256. Následně byly na výstup sítě přidány konvoluční vrstvy, které prováděly postupné převzorkování na výsledný rozměr. Důvodem pro postupné převzorkování byl fakt, základní *FCN* síť používá jednorázové převzorkování, které bylo pro aplikaci odhadu hloubkové mapy nevhodné a způsobovalo malé rozlišené sítě. Veškeré úpravy byly experimentální, ale kombinací těchto úprav vznikla síť, která je schopna velice dobře odhadovat hloubku z barevného vstupního snímku.



Obrázek 98: Ukázka výsledku učení neuronové sítě FCN. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)



Obrázek 99: Porovnání výsledků sítě FCN. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě FCN (Zdroj: Vlastní)



Obrázek 100: Porovnání výsledků sítě FCN. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě FCN (Zdroj: Vlastní)

Obrázek 101: Porovnání výsledků sítě FCN. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě FCN (Zdroj: Vlastní)

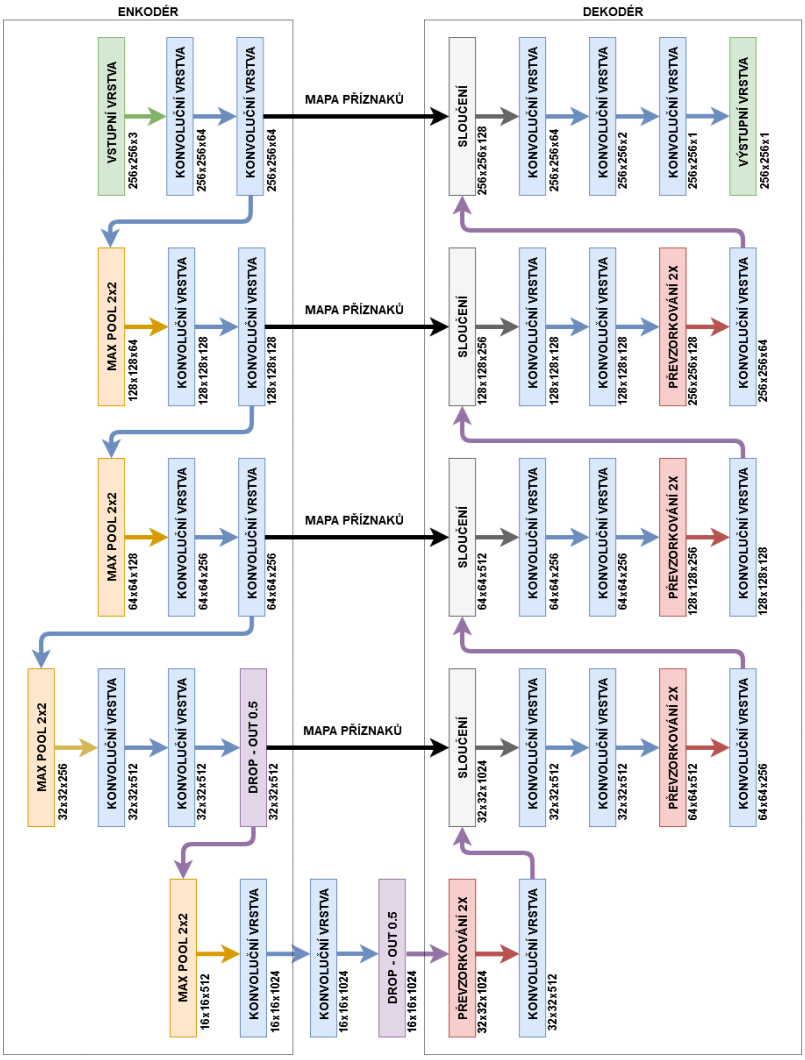
Na Obrázek 100 je zobrazen pokrok učení neuronové sítě *FCN* u shodného vstupního snímku. Tento snímek nebyl síti při učení překládán, pouze byl vyhodnocen za využití koeficientů učení po jedné etapě (snímek úplně vlevo), po čtyřech etapách (snímek druhý zleva) a po devíti etapách (třetí zleva). Snímek úplně vpravo je korespondující snímek získaný z hloubkové kamery. Na Obrázek 101 je zobrazen výstup

ze sítě *FCN*. Na Obrázek 100 a Obrázek 101 jsou zobrazeny náhodné snímky z testovací množiny dat. Tyto snímky nebyly neuronové síti během učení přeloženy, tak jsou velice dobrou ukázkou kvality učení neuronové sítě. Na Obrázek 101 je vidět postupný vývoj odezvy neuronové sítě na vstupné RGB snímek průběhu epoch učení. Po první epoše je vidět že kvalita odezvy sítě je velice slabá a není ani možné rozpoznat jakýkoliv tvar. Po čtyřech epochách je již možné rozpoznat hrubý tvar automobilu, avšak ve výsledném snímku chybí odhad oblasti s oknem, který byla schopna síť odhadnout až po 9 epochách. Výsledný snímek je velice kvalitní a téměř odpovídá originálu. Podrobnější výsledky získané vyhodnocením statistických parametrů jsou popsány v kapitole 10

9.5 Úpravy a výsledky sítě U-Net

Základní architektura sítě nesplňovala stanovené požadavky a bylo nezbytné architekturu upravit způsobem popsaným v této kapitole. Základní struktura byla zachována, ovšem byl přidán větší počet U-Net bloků pro lepší pokrytí řešeného problému. Upravená architektura dále měla upraveny zejména výstupní vrstvy, aby výstupní mapa měla shodné rozměry se vstupním vzorem (256x256 pixelů). Následně jako aktivační funkce byla použita funkce *ReLu*. Během učení bylo pozorováno, že pokud byla použita funkce, která měla limitaci (*Tanh*, Sigma funkce), docházelo vždy k limitaci a nedošlo k naučení sítě ani při velkém množství pokusů. Při použití aktivační funkce, která není shora ohraničená, došlo téměř vždy k procesu učení. *ReLU* funkce je nejvhodnější aktivační funkcí, zejména u výstupních vrstev.

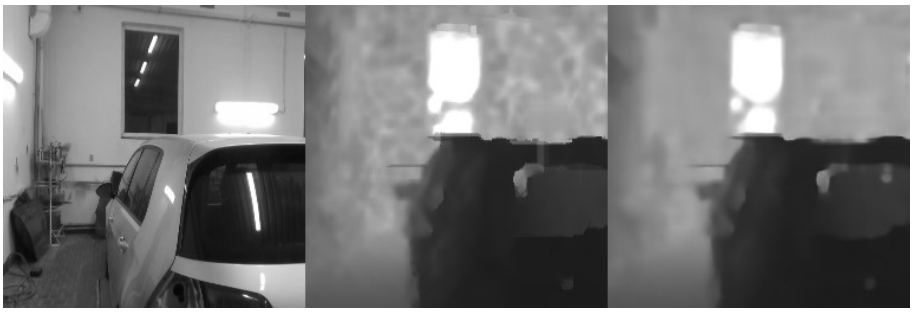
Upravená architektura použitá pro řešení problému odhadu hloubkové mapy na základě vstupního RGB snímku je zobrazena na Obrázek 102.



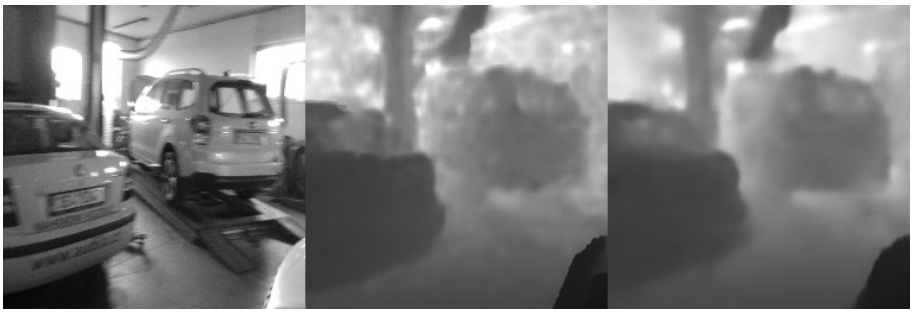
Obrázek 102: Struktura sítě U-Net (Zdroj: Vlastní)

Obrázek 103: Ukázka výsledku učení neuronové sítě U-Net. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)

Na Obrázek 103 je zobrazeno porovnání pokroku učení po první, čtvrté a deváté epoše učení. Z Obrázek 103 je patrné, že síť *U-Net* již po první epoše učení byla schopna celkem kvalitně odhadnout obrysy automobilu. Po deváté epoše je patrné, že hloubková mapa odhadnutá neuronovou sítí *U-Net* je minimálně odlišná od hloubkové mapy vytvořené z hloubkové kamery.



Obrázek 104: Porovnání výsledků sítě U-Net. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě U-Net (Zdroj: Vlastní)



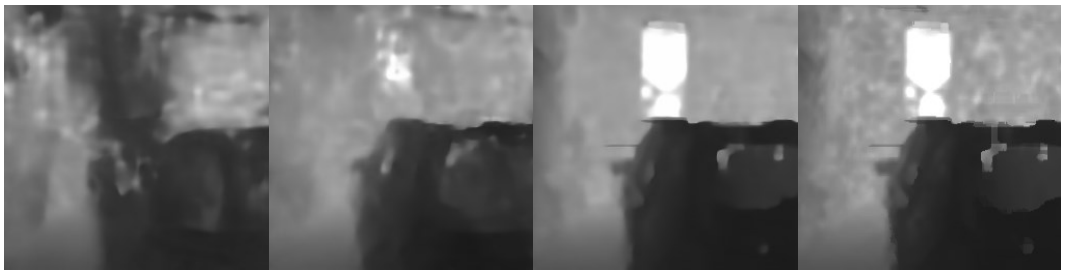
Obrázek 105: Porovnání výsledků sítě U-Net. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě U-Net (Zdroj: Vlastní)

Obrázek 106: Porovnání výsledků sítě U-Net. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě U-Net (Zdroj: Vlastní)

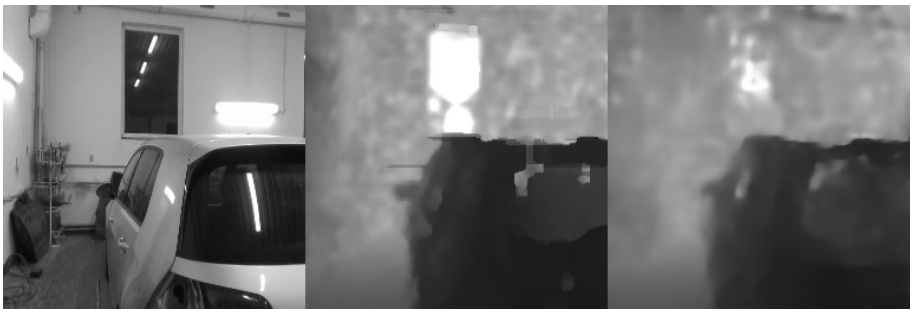
Na Obrázek 103 až Obrázek 106 je zobrazeno porovnání výsledků učení sítě *U-Net*. Síť velice dobře taktéž filtruje šum vzniklý snímáním hloubkové kamery a vytváří čistější výsledky, které jsou vhodnější pro další zpracování v navigaci. Podrobnější výsledky získané vyhodnocením statistických parametrů jsou popsány v kapitole 10.

9.6 Úpravy a výsledky sítě U-Net3

Základní popis neuronové sítě *U-Net3*, která byla navržena autorem práce je popsán v samostatné kapitole 4.2.10. Síť byla navržena jako robustnější varianta neuronové sítě *U-Net*, s větším počtem trénovatelných koeficientů pro kvalitnější popsání hloubkové mapy. Struktura sítě byla navržena do velké hloubky pro extrakci maximálního množství detailů pro zvýšení přesnosti odhadu hloubkové mapy.



Obrázek 107: Ukázka výsledku učení neuronové sítě U-Net3. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)



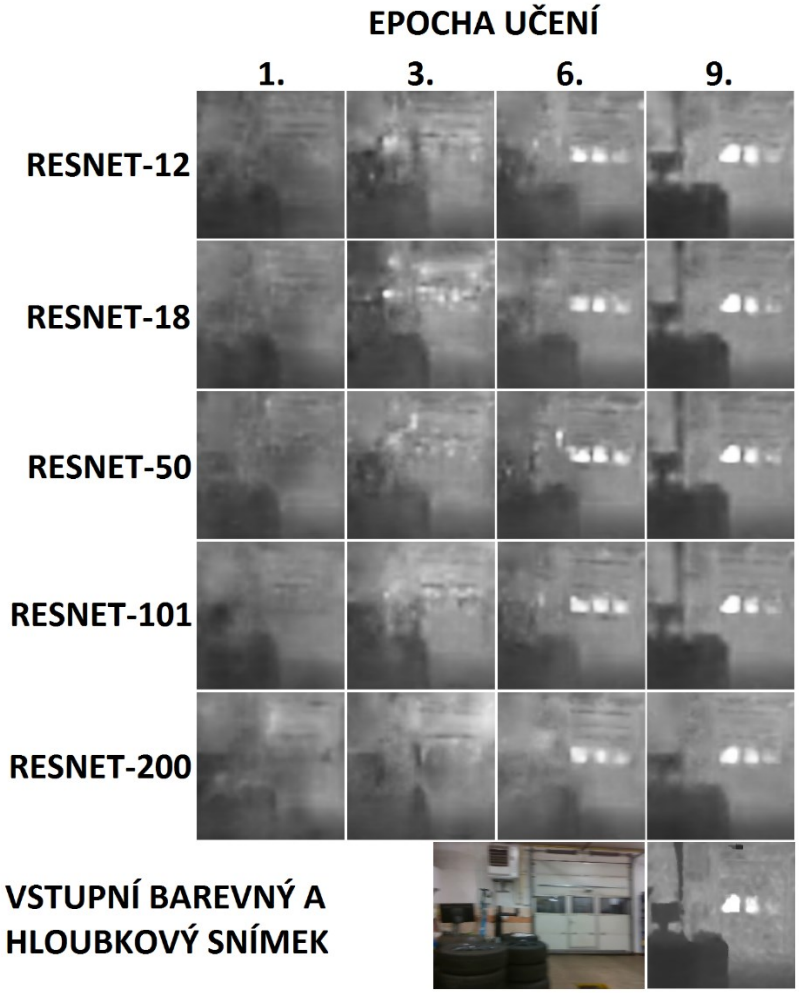
Obrázek 108 : Porovnání výsledků sítě U-Net3. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě U-Net (Zdroj: Vlastní)

Obrázek 109: Porovnání výsledků sítě U-Net3. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě U-Net (Zdroj: Vlastní)

Obrázek 110: Porovnání výsledků sítě U-Net3. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě U-Net (Zdroj: Vlastní)

9.7 Úpravy a výsledky sítě ResNet

V rámci výzkumu bylo testováno celkem 5 variant architektury neuronové sítě *ResNet*, od nejjednodušší varianty sítě *ResNet-12*, obsahující celkem 20,84 milionu trénovatelných koeficientů po nejsložitější variantu *ResNet-200*, která má 76,27 milionu trénovatelných koeficientů.



Obrázek 111: Porovnání kvality výsledků u různých variant sítě ResNet. (Zdroj: Vlastní)

Pro porovnání kvality výsledků byl vytvořen Obrázek 111. Na Obrázek 111 první řádek zobrazuje postupný progres učení všech testovaných variant sítě *ResNet*. Z výsledků je patrné, že neuronová síť je schopna velice dobře odhadnout hloubku po 9. epoše učení.

Z výsledků taktéž vychází, že ze všech architektur sítě *ResNet* dosáhla síť *Resnet-200* nejhorších výsledků. Na Obrázek 111 je na ukázkovém snímku tento fakt patrný. Podrobnější výsledky získané vyhodnocením statistických parametrů jsou popsány v kapitole 10

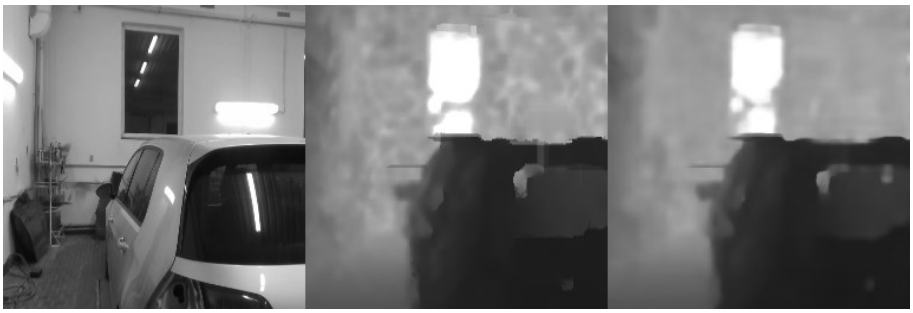
9.8 Úpravy a výsledky sítě Xception

Architektura sítě *Xception* byla pro potřeby této práce upravena. Standardní výstup neuronové sítě *Xception* je vektor, například o 2048 kategorií. Pro potřeby této práce byla architektura přepracována. Základní konvoluční bloky, uvedené v publikaci (Chollet, 2017) byly veskrze ponechány až výstupní blok, který musel být kompletně přepracován, aby bylo možné získat správné dimenze výstupní matice. Dále bylo nutné

vyměnit některé aktivační funkce. Ve většině případů při použití aktivační funkce *Sigma* či *TanH*, docházelo k uvíznutí sítě v lokálním minimu a ani při opakovaném spuštění učení nebylo možné dále provést učení. Při výměně aktivační funkce za *ReLu*, bylo možné provést učení neuronové sítě.

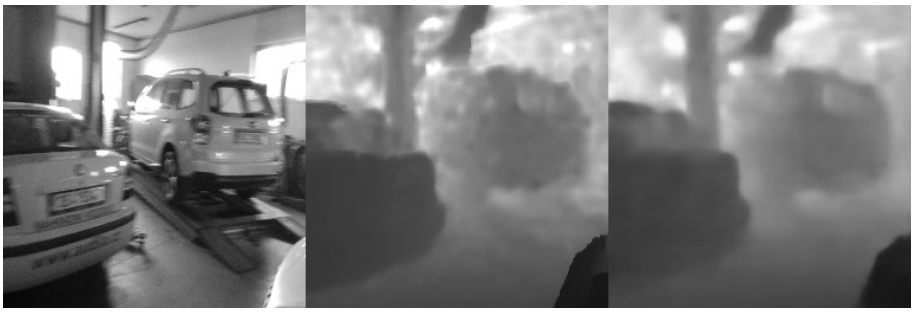
Obrázek 112: Ukázka výsledku učení neuronové sítě Xception. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)

Na Obrázek 112 je zobrazen pokrok učení neuronové sítě *Xception* po první, čtvrté a deváté epoše učení. Z Obrázek 112 je patrné, že po první epoše síť odhadla hrubé obrysy automobilu, ovšem navíc síť vygenerovala objekt v místě okna, který ovšem odhadla jako blízký objekt, nikoliv jako vzdálený, jak je tomu ve skutečnosti. Po uplynutí deváté epochy je patrné, že síť odhadla obrys automobilu, ovšem v místě okna je chybně vytvořena plocha.



Obrázek 113: Porovnání výsledků sítě Xception. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě Xception (Zdroj: Vlastní)

Obrázek 114: Porovnání výsledků sítě Xception. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě Xception (Zdroj: Vlastní)



Obrázek 115:Porovnání výsledků sítě Xception. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě Xception (Zdroj: Vlastní)

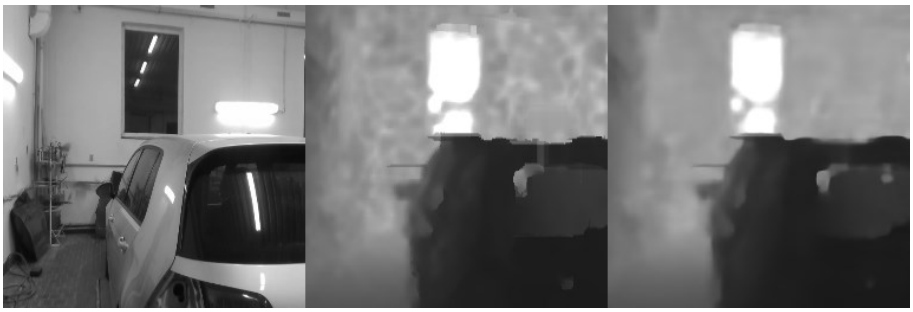
Z Obrázek 114 a Obrázek 115 je patrné, že síť *Xception* velice dobře odhaduje hloubku na základě vstupního barevného snímku. Síť *Xception* byla dále použita pro další experimentování v rámci práce. Podrobnější výsledky získané vyhodnocením statistických parametrů jsou popsány v kapitole 10.

9.9 Úpravy a výsledky sítě RefineNet

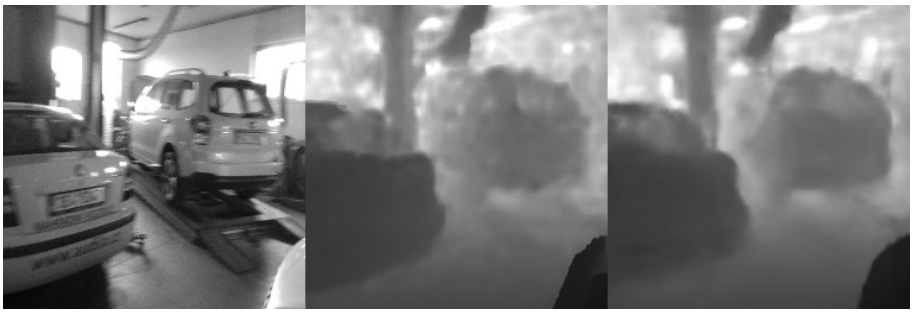
V rámci práce byla síť upravena. Základní síť totiž provádí sémantickou segmentaci, typicky s omezeným počtem odhadovaných segmentů. Základní model sítě *RefineNet* byl navržen tak, že dimenze výstupního snímku nejsou shodné s dimenzemi vstupního snímku. Typicky jsou použity sítě, které mají výstupní dimenzi čtvrtinovou vůči vstupnímu snímku. Pro potřeby práce bylo nutné síť upravit tak, aby byla schopna odhadovat hloubku pixelu s 8-bit rozlišením (jako u všech ostatních sítí) a taktéž aby dimenze vstupního a výstupního snímku byly shodné.

V rámci práce došlo k implementaci upravené struktury sítě *RefineNet* a ukázka výsledků je uvedena na Obrázek 118 a Obrázek 119.

Obrázek 116: Ukázka výsledku učení neuronové sítě RefineNet. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)



Obrázek 117: Porovnání výsledků sítě RefineNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě RefineNet (Zdroj: Vlastní)



Obrázek 118: Porovnání výsledků sítě RefineNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě RefineNet (Zdroj: Vlastní)

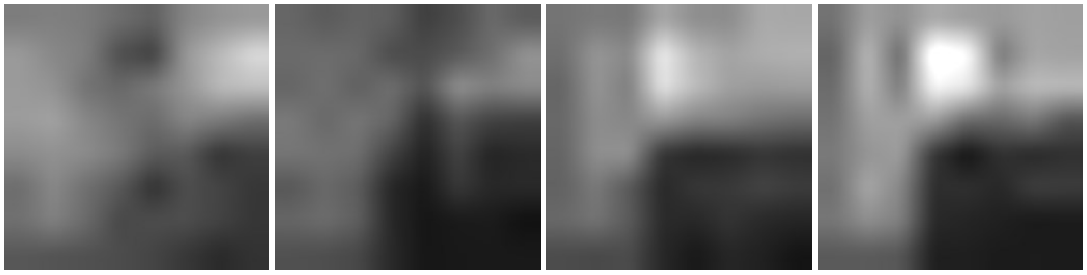
Obrázek 119: Porovnání výsledků sítě RefineNet. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě RefineNet (Zdroj: Vlastní)

Podrobnější výsledky získané vyhodnocením statistických parametrů jsou popsány v kapitole 10.

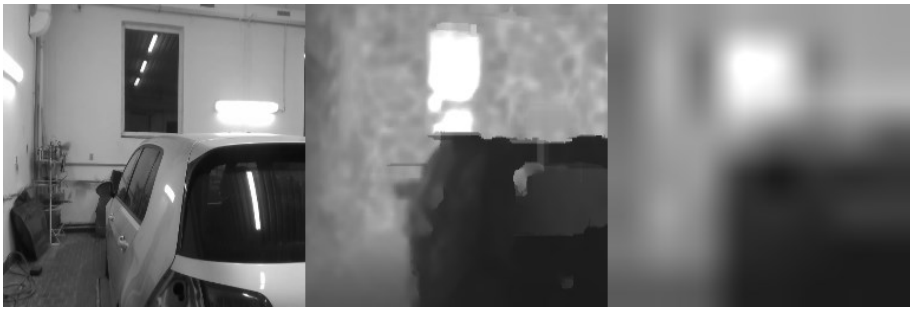
9.10 Úpravy a výsledky sítě DenseNet

V práci byly testovány různé konfigurace sítě (*DenseNet-169*, *DenseNet-201* či *DenseNet-264*), ale díky extrémní náročnosti výpočtů byla zvolena nejméně náročná *DenseNet-121* napojená na mateřskou síť *Resnet-101*. U sítě *DenseNet-121* je možné použít téměř jakoukoliv mateřskou síť. Z Obrázek 122 je patrné, že síť je schopna se naučit odhad hloubky, ovšem při stávajícím nastavení nebyly výsledky uspokojivé. Obrysy objektů jsou neostré, avšak odhad vzdálenosti není úplně špatný. Pro další použití sítě je nutné se sítí dále experimentovat pro dosažení kvalitnějších výsledků, například

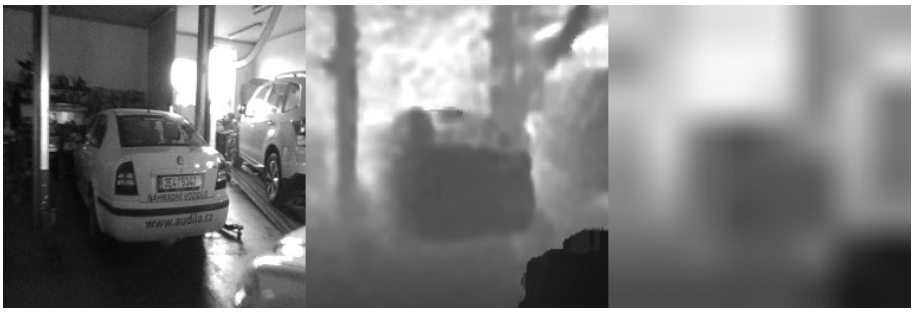
použitím rozsáhlejší sítě DenseNet-300. Pro další experimenty nebyla již síť *DenseNet* použita.



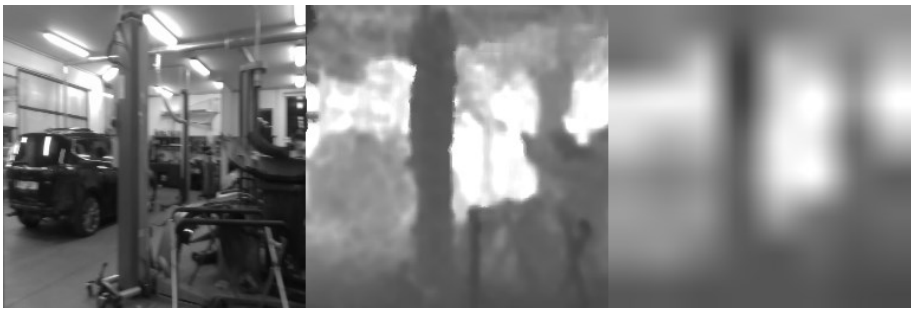
Obrázek 120: Ukázka výsledku učení neuronové sítě DenseNet. Vlevo po první epoše učení, druhý zleva po čtyřech epochách učení, třetí zleva po 9 epochách učení, vpravo skutečný snímek z hloubkové kamery (Zdroj: Vlastní)



Obrázek 121: Porovnání výsledků sítě DenseNet-121. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě DenseNet (Zdroj: Vlastní)



Obrázek 122: Porovnání výsledků sítě DenseNet-121. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě DenseNet (Zdroj: Vlastní)

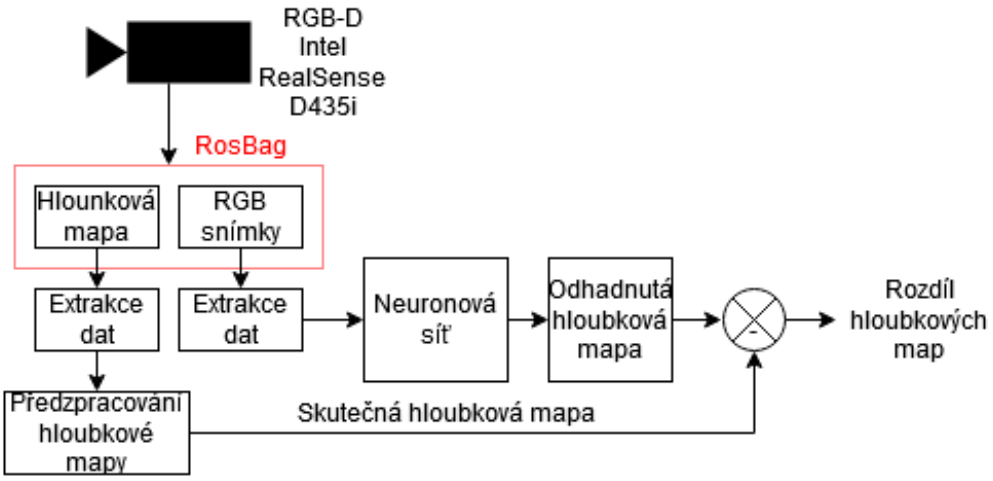


Obrázek 123: Porovnání výsledků sítě DenseNet-121. Vlevo vstupní snímek, uprostřed snímek z hloubkové kamery, vpravo výstup ze sítě DenseNet (Zdroj: Vlastní)

Podrobnější výsledky získané vyhodnocením statistických parametrů jsou popsány v kapitole 10.

10 Analýza výsledků a výběr vhodného modelu sítě

Všechny výsledky uvedené v této kapitole byly získány vyhodnocením hloubkových map generovaných testovanými neuronovými sítěmi, jejichž předlohy nebyly předloženy neuronové síti během učení. Je nutno zmínit, že neuronová síť měla k dispozici během učení podobné snímky ze stejných budov, jen v jiném kontextu – natočení, trase, jiné konstelaci překážek (zejména automobily a jiné snadno přemístitelné objekty) Tyto výsledky jsou detailně rozebrány v této kapitole.



Obrázek 124: Ukázka řetězce vyhodnocení chyby hloubkové mapy (Zdroj: Vlastní)

Na Obrázek 124 je zobrazen řetězec pro vyhodnocení chyby hloubkové mapy. Chyba konkrétního snímku, resp. pixelu je vyhodnocena jako rozdíl mezi odhadnutým snímkem pomocí neuronové sítě a skutečným snímkem, získaným pomocí hloubkové kamery. Rozdílový snímek je následně detailně vyhodnocován. Popis různých metod vyhodnocení je rozepsán v této kapitole.

Pro zhodnocení kvality odhadu hloubky neuronovými sítěmi byly stanoveny základní statistické parametry MAE a MSE. Tyto parametry byly stanoveny porovnáním každého originálního a cílového snímku s odhadem sítě.

$$MAE = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}|y_{i,j}-x_{i,j}|}{n \cdot m} \tag{10.1}$$

$$MSE = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(y_{i,j}-x_{i,j})^2}{n \cdot m} \tag{10.2}$$

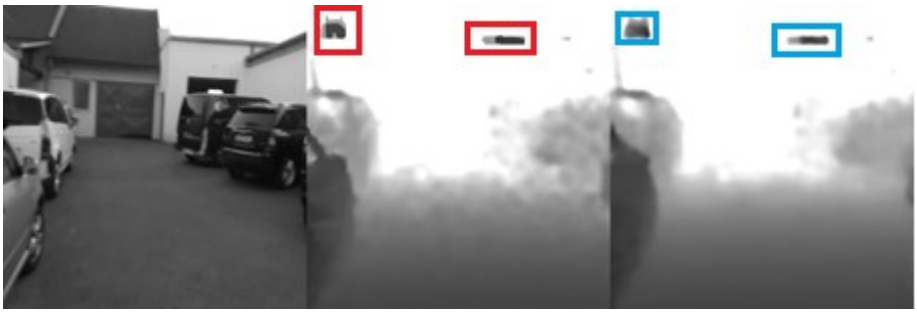
Kde:

- n, m jsou rozměry snímku, v našem případě 256x256
- $y_{i,j}$ je hodnota pixelu skutečného hloubkového snímku
- $x_{i,j}$ je hodnota pixelu odhadnutého hloubkového snímku z neuronové sítě

Vyhodnocení výsledků neuronových sítí bylo provedeno za využití statistických parametrů uvedených rovnicemi (10.1) a (10.2). Tyto parametry byly vypočítávány v každém snímku testovací sady.

Parametr MSE je vypočítán v rámci celého snímku. Slouží pro odhad chyby učení neuronové sítě. Pro lepší vyhodnocení je proveden součet chyb MSE v rámci celého datasetu. Chyba MAE slouží pro výpočet absolutní odchylky odhadnuté hodnoty pixelu vůči skutečné hodnotě pixelu. Tato hodnota určuje, jak hodně se neuronová síť „netrefila“ při svém odhadu. Pro lepší vyhodnocení byl opět vypočítán součet MAE pro vyhodnocení kvality neuronové sítě.

Je potřeba zmínit, že je nutné ke každému parametru přistupovat individuálně. Jedná se zejména o lokální chyby určení v rámci jednoho snímku. Jednat se může zejména o nesprávné vyhodnocení snímku na základě slabého osvětlení, odlesku od skla či zrcadla, či obyčejného svitu slunce do senzoru hloubkové kamery. Tyto okolnosti způsobují veliké problémy a byla snaha tyto chybné snímky z trénovacích dat odstranit.

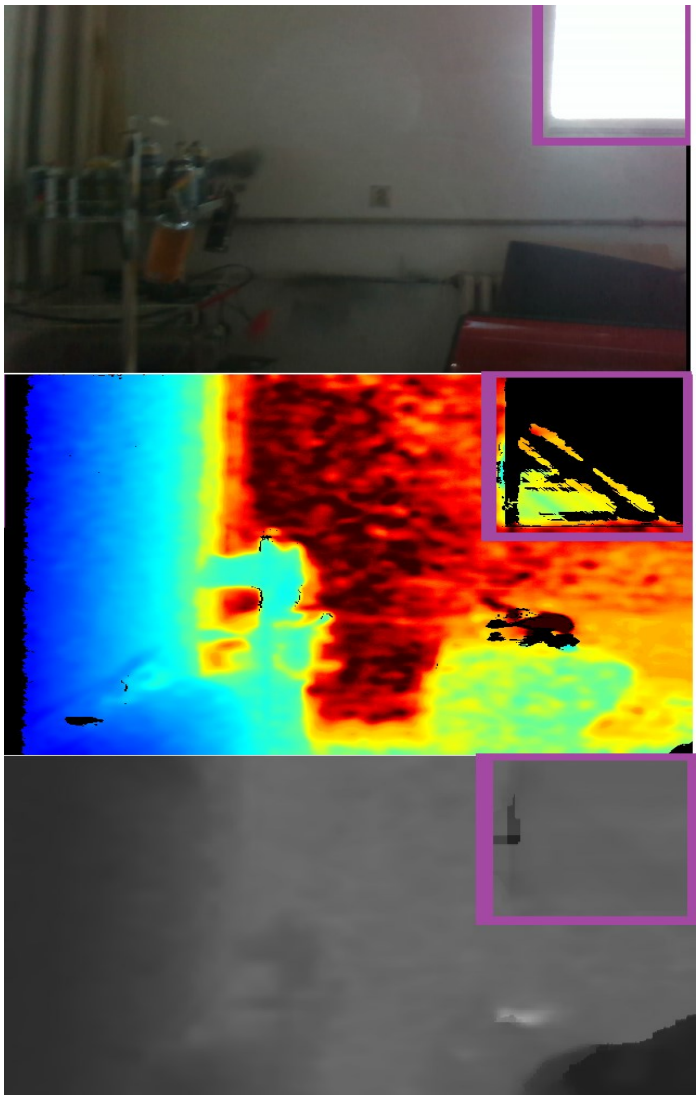


Obrázek 125: Ukázka chyb v trénovacích datech (červený rámeček) a odezva sítě (modrý rámeček). (Zdroj: Vlastní)

Obrázek 126: Ukázka chyb v trénovacích datech (červený rámeček) a odezva sítě (modrý rámeček). (Zdroj: Vlastní)

Ukázka chybných oblastí ve vstupních datech je zobrazena na Obrázek 125. Zde došlo k přehlédnutí chyby ve vstupních snímcích vlivem drobného svitu slunce do senzoru hloubkové kamery, čímž došlo k vytvoření imaginárního blízkého objektu v trénovacích datech. Jelikož tato chyba se vzhledem ke slunečnému počasí objevila v několika hloubkových snímcích, síť se tyto chyby naučila také. Další chyby vstupních dat z hloubkové kamery jsou patrné na Obrázek 126. Zde byla snímaná scéna chybně interpretována hloubkovou kamerou vlivem odlesku od laku automobilu. Nejvíce patrné to je v místě upevnění registrační značky, kde došlo k vytvoření imaginární vzdálenosti větší než 10 metrů (bílá barva).

Je nutné poznamenat, že podmínky měření snímků nejsou ideální, jelikož některé scény získané za využití RGB-D kamery byly na hraně maximálního dosahu, tj. vzdálenost od kamery byla 10 m nebo dokonce vyšší. Hodnota vzdálenosti větší než 10 m byla automaticky nastavena na maximální vzdálenost. Základním problémem je i fakt, že snímky získané hloubkovou kamerou, mají různou úroveň šumu v závislosti na snímané scéně. Jelikož se jednalo o reálné průmyslové prostředí s různou intenzitou osvětlení scény, zároveň zde byly okolní vlivy, které nebylo možné během měření ovlivnit. Jedná se například o sluneční svit, který je schopen kameru úplně oslnit.



Obrázek 127: Ukázka surových dat z oslnění RGB-D kamery. Nahoře: RGB snímek, uprostřed: Surový hloubkový snímek z RGB-D kamery, Dole: Výsledná předzpracovaná a normalizovaná data předkládaná k učení neuronové síti. (Zdroj: Vlastní)

Během měření byla trasa v průmyslovém prostředí stanovena tak, aby k oslnění kamery docházelo minimálně. Při postprocessingu ovšem bylo několik snímků oslnění kamery odhaleno a bylo nutné tyto snímky manuálně odstranit, protože by docházelo ke špatnému učení neuronové sítě. Drobné oslnění kamery ovšem bylo ponecháno pro zvýšení robustnosti učení neuronové sítě. Ukázka částečně oslněného snímku včetně výsledného snímku po opravě je zobrazena na Obrázek 127. Horní část Obrázek 127 ukazuje RGB snímek, na kterém je zobrazeno částečné oslnění sluncem skrz okno. Prostřední část Obrázek 127 zobrazuje surová data bez jakéhokoliv zpracování. Fialovým rámečkem je zobrazena oslněná část. Dolní část Obrázek 127 zobrazuje již zpracovaný snímek po aplikaci zarovnání dat a doplnění chybějících bodů metodou "*Ext Cross Fill*", popsané v kapitole 7.2.2. Spodní část je výsledný snímek, pomocí kterého dochází k učení neuronové sítě. Stejným postupem bylo provedeno předzpracování celé datové sady, tj. trénovací, validační i testovací datové sady. RGB snímky nebyly upravovány, jen bylo sníženo rozlišení snímků vstupní vrstvou neuronové sítě.

10.1 Výběr vhodného modelu

Výběr vhodného modelu neuronové sítě je velice netriviální proces, protože sítě nejsou rovnocenné a bylo nutné zavést řadu kritérií, na jejichž základě je možné rozhodnout, který model neuronové sítě bude pro další zkoumání použit.

Zkoumané parametry:

- Časová náročnost učení jednoho snímku – Časová náročnost učení byla získána jako průměr doby učení celé sady získaný z metody, provádějící učení po ukončení dávky učení. Výsledný čas vyhodnocení jednoho snímku byl vypočítán jako suma času nezbytná pro vyhodnocení všech snímků dělená počtem snímků. Do času učení se nezahrnuje doba ukládání koeficientů na pevný disk. Výpočet časové náročnosti byl prováděn na počítačové sestavě, jejíž parametry jsou uvedeny v Tabulka 9.
- Časová náročnost vyhodnocení jednoho snímku – Časová náročnost vyhodnocení jednoho snímku byla získána jako rozdíl času zpracování jedné dávky v jazyce Python. Výpočet časové náročnosti byl prováděn na počítačové sestavě, jejíž parametry jsou uvedeny v Tabulka 9. Při výpočtu času je vždy nutné ignorovat prvních několik průběhů učení z důvodu alokace paměťového prostoru. Z výsledků je většinou patrné, že čas zpracování prvních přibližně dvou až tří dávek vyhodnocení má výrazně delší čas zpracování než zbylých několik

desítek až stovek dávek. Výsledný čas vyhodnocení jednoho snímku byl vypočítán jako suma času nezbytná pro vyhodnocení všech snímků dělený počtem snímků.

- Složitost sítě – Parametr pro hrubý odhad paměťové náročnosti dané architektury. Odhaduje se z počtu trénovatelných i statických parametrů sítě.
- Přesnost dosažení výsledků – statistické parametry MAE, MSE a jejich součty v rámci celého testovaného datasetu.

V Tabulka 14 jsou uvedeny základní parametry zkoumaných architektur neuronových sítí. Z tabulky je patrné, že bylo zastoupeno široké spektrum sítí s různou složitostí a taktéž s různou dobou potřebnou pro učení a zpracování jednoho snímku.

Tabulka 14: Parametry neuronových sítí

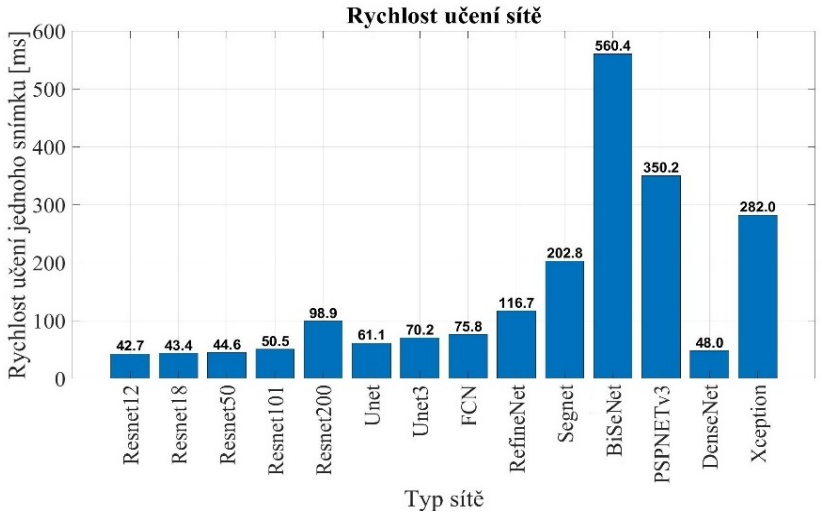
Typ sítě	Podtyp sítě	Trénovatelné parametry sítě	Statické parametry sítě	Celkem parametrů	Průměrný čas učení jednoho vstupního snímku [ms]	Průměrný čas zpracování jednoho vstupního snímku [ms]
ResNet	12	20 844 129	4 848	20 848 977	42,7	8,9
	18	22 237 057	5 136	22 242 193	43,4	9,3
	50	30 036 513	5 808	30 042 321	44,6	10,3
	101	43 658 729	7 152	43 692 881	50,5	12,1
	200	76 277 153	10 800	76 287 953	98,9	16,8
U-Net	Základní	31 032 837	0	31 032 837	61,1	15,8
	V.3	88 278 117	0	88 278 117	70,2	24,2
FCN	8	79 606 817	19 648	79 626 465	75,8	33,5
SegNet	Není	25 891 365	15 872	25 907 237	202,8	57,6
BiSeNet	Není	42 454 393	59 008	42 513 401	560,4	231,5
RefineNet	Není	105 912 339	20 640	105 932 979	116,7	23,7
PSP-Net	Není	36 420 697	58 360	36 479 057	350,2	145,9
DenseNet	Není	47 417 089	29 216	47 446 305	48	11,2
Xception	Není	35 264 053	90 416	35 354 469	282.0	85,7

V Tabulka 15 je uveden čas, který byl zapotřebí k samotnému učení konkrétní neuronové sítě. Tento čas je vypočítán z průměrné doby učení jednoho snímku. Z tabulky je patrné, že samotné učení zabralo čas přesahující 1700 strojových hodin, což je v přepočtu 71 dní čistého strojového času. Bohužel během učení se velice často stalo vlivem extrémní paměťové náročnosti k havárii trénovacích algoritmů vlivem vyčerpání systémových prostředků a učení se muselo opakovat. Dále čas uvedený v Tabulka 15 nezohledňuje časy ukládání výsledků a inicializaci struktur nezbytných k učení. Při zahrnutí všech časů je odhad celkového času učení již připravených neuronových přibližně 150 dní. Čas učení byl stanoven na základě parametrů uvedených v Tabulka 11, která je uvedena v kapitole 8 a dle časů učení uvedených v Tabulka 14.

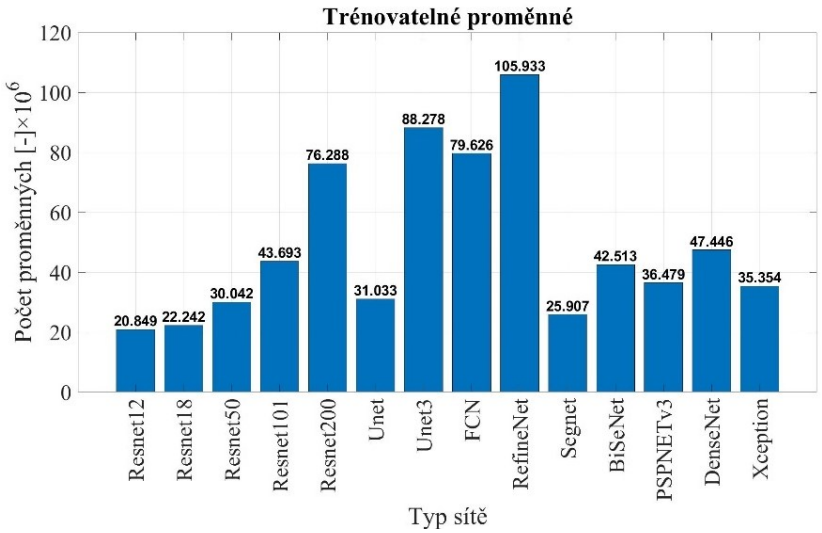
Tabulka 15: Doba učení neuronové sítě

Typ sítě	Podtyp sítě	Doba učení [hod]
ResNet	12	35,54
	18	36,20
	50	37,19
	101	42,06
	200	82,45
U-Net	Základní	50,93
	V.3	58,49
FCN	8	63,19
SegNet	Není	97,29
BiSeNet	Není	169,03
RefineNet	Není	466,99
PSP-Net	Není	291,83
DenseNet	Není	40,02
Xception	Není	234,97
Součet [hod]		**1 706,18**

Na Obrázek 128 je zobrazen čas potřebný pro naučení jednoho snímku v závislosti na druhu učené sítě. Tento parametr je odvozen z průměrné doby učení jedné epochy a velikosti dávky učení. Do času se nezapočítává čas potřebný pro inicializaci datových struktur neuronové sítě a čas potřebný pro uložení souborů na pevný disk. Na první pohled je možné poznamenat, že časy učení jednoho snímku jsou vzhledem ke složitosti některých architektur sítí velice nízké. Důvodem je ovšem fakt, že pro učení neuronových sítí byla použita pouze velice malá vstupní matice o rozměrech 256×256×3, což v dnešní době FullHD či 4K kamer je velice malé rozlišení. Pro učení neuronových sítí je samozřejmě možné takto veliké rozlišení použít, ovšem zásadní informaci pro učení vysoké rozlišení vstupního snímku nepřinese, naopak extrémně zvedne počet trénovatelných proměnných nezbytných pro naučení neuronové sítě a taktéž významně prodlouží čas učení či zpracování.

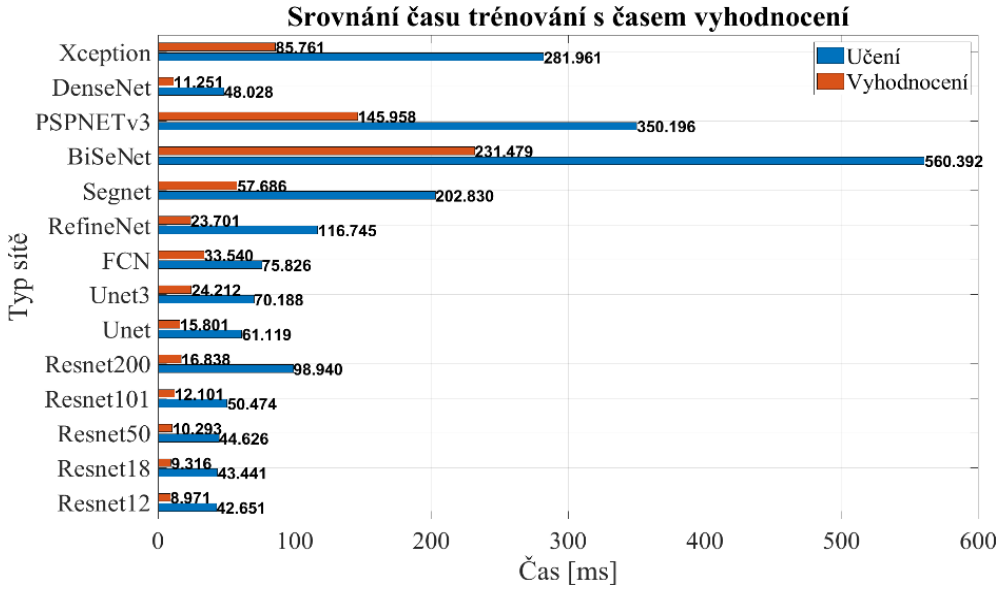


Obrázek 128: Srovnání rychlosti učení v závislosti na typu neuronové sítě (Zdroj: vlastní)



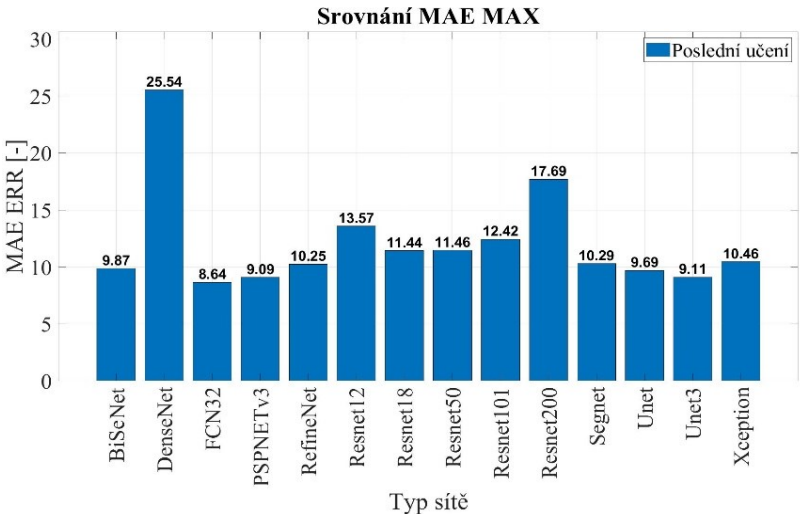
Obrázek 129: Srovnání počtu trénovatelných parametrů v závislosti na typu neuronové sítě (Zdroj: Vlastní)

Z Obrázek 129 je patrná složitost testovaných sítí. Z výsledků je taktéž patrné, že rozdíl v počtu trénovatelných proměnných byl až pětinásobný mezi nejsložitější sítí (*RefineNet*) a nejjednodušší sítí (*ResNet-12*). Nezávislý pozorovatel by mohl namítnout, že větší množství trénovatelných proměnných znamená větší kvalitu výsledného snímku. Opak může být ovšem pravdou, jelikož nezávisí jen na počtu trénovatelných parametrů, ale taktéž na architektuře.

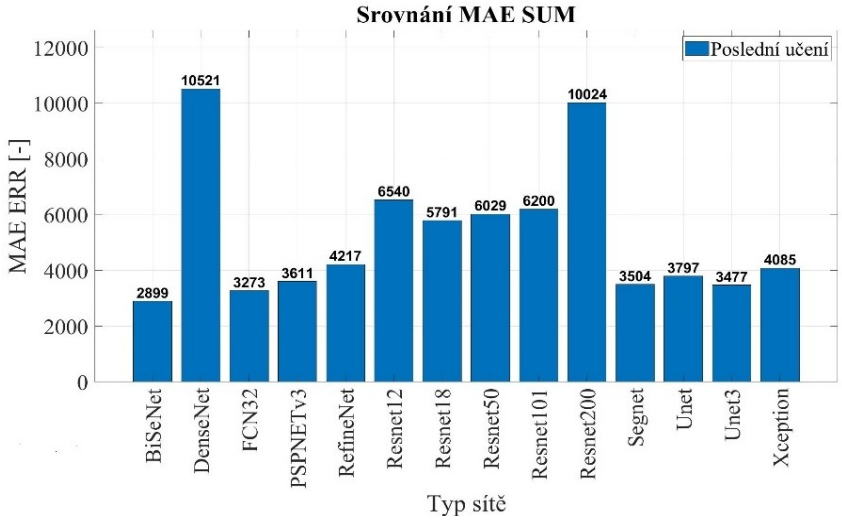


Obrázek 130: Srovnání času učení s časem zpracování jednoho snímku (Zdroj: Vlastní)

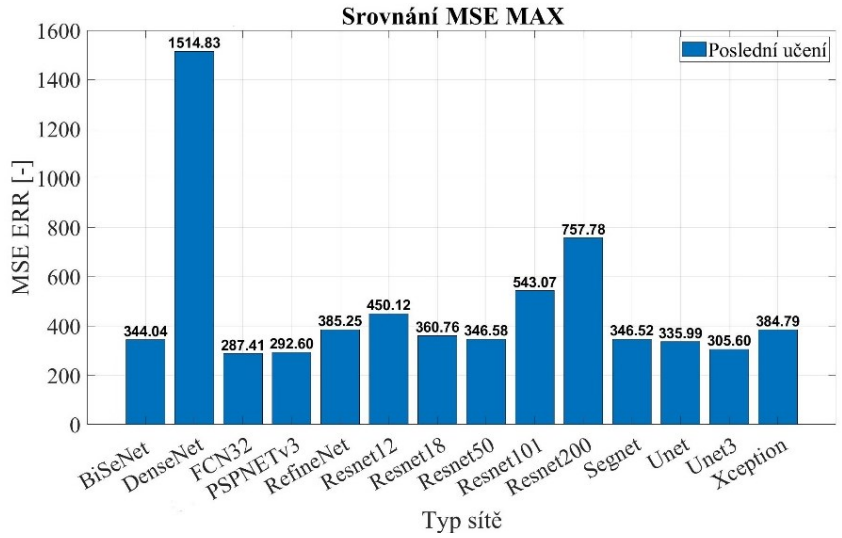
Na Obrázek 130 je zobrazeno porovnání rychlosti učení a rychlosti zpracování jednoho snímku dle typu zvolené neuronové sítě. Z výsledků zobrazených na Obrázek 130 je patrné, že většina neuronových sítí by byla vhodná pro zpracování odhadu hloubkového snímku v reálném zařízení, protože doba potřebná pro zpracování je v řádu maximálně několika desítek milisekund. Složitější sítě mají dobu zpracování v řádu několika desítek milisekund a pro některé aplikace by nemusely být vhodné. Konkrétně se jedná o neuronovou síť *BiSeNet*, která dosahuje velké přesnosti odhadu výsledného snímku, ovšem rychlost zpracování je extrémně pomalá.



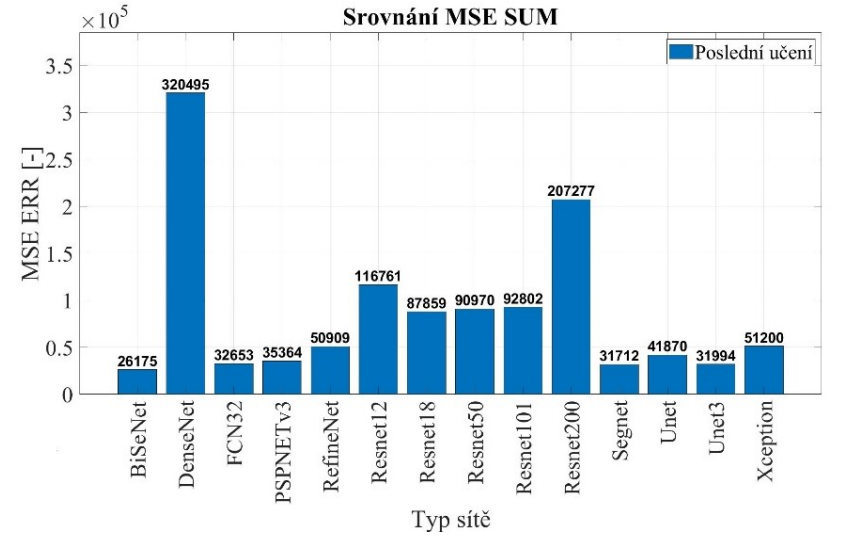
Obrázek 131: Ukázka maximální absolutní chyby z celého datasetu (Zdroj: Vlastní)



Obrázek 132: Ukázka součtu maximální absolutní chyby z celého datasetu (Zdroj: Vlastní)



Obrázek 133: Srovnání maximální chyby MSE v rámci celého datasetu (Zdroj: vlastní)



Obrázek 134: Srovnání součtu chyby MSE v rámci celého datasetu. (Zdroj: vlastní)

Na Obrázek 134 jsou zobrazeny výsledky parametru MSE v rámci všech testovaných neuronových sítí. Z výsledků je patrné že pro odhad hloubky na základě

vstupního RGB snímku je nejlepší testovaná síť *BiSeNet*, druhá nejlepší *SegNet* a třetí nejlepší byla vyhodnocena autorem práce navržená síť *U-Net3*. Tyto sítě jsou výpočetně výrazně složitější než sítě například typu *ResNet*-18 či *Resnet-50*, jak je uvedeno v Tabulka 14.

10.1.1 Srovnání výsledků neuronových sítí se shodnou vahou parametrů

Pro každý z těchto parametrů bylo stanoveno pořadí pro každý zkoumaný model neuronové sítě. Čím lepšího výsledku neuronová síť v dané kategorii dosáhla, tím více bodů získala, tj. čím menší chybu MSE či MAE daná síť měla, tím více získala bodů v dané kategorii. Obdobně tomu je u kategorie rychlost učení a rychlost vyhodnocení. Výsledné pořadí bylo vyhodnoceno na základě součtu bodů získaných v jednotlivých kategoriích. Srovnání monitorovaných parametrů dle přidělených bodů u testovaných neuronových sítí je uvedeno v Tabulka 16.

Tabulka 16: Srovnání kvality testovaných neuronových sítí (více je lépe).

Typ sítě		Rychlost učení	Rychlost zpracování	Složitost sítě	MAE MAX	Σ MAE	MSE MAX	Σ MSE	Σ
ResNet	12	**14**	**14**	**14**	3	3	4	3	55
	18	13	13	13	6	6	7	6	64
	50	12	12	11	5	5	8	5	58
	101	10	10	6	4	4	3	4	41
	200	6	8	4	2	2	2	2	26
U-net		9	9	10	11	9	11	9	**68**
U-Net3		8	6	2	12	12	12	12	64
FCN		7	5	3	**14**	13	**14**	11	67
RefineNet		5	7	1	9	7	5	8	42
Segnet		4	4	12	8	11	9	13	61
Bisenet		1	1	7	10	**14**	10	**14**	57
PSPNET_v3		2	2	8	13	10	13	10	58
DenseNet-121		11	11	5	1	1	1	1	31
Xception		3	3	9	7	8	6	7	43

Z Tabulka 16 je patrné, že nejlepšího komplexního bodového zisku dosáhla neuronová síť typu *U-Net*. Z výsledků je taktéž patrné, že síť *U-Net* není absolutně nejlepší v odhadu výsledných snímků, ovšem kvalitativně dosahuje stejných výsledků jak ve složitosti sítě a náročnosti na učení a vyhodnocení, tak velice kvalitními výsledky. Druhou, velice kvalitní architekturou sítě se ukázala síť typu FCN. Z výsledků je patrné, že FCN má nejlepší kvalitu odhadu výsledných snímkům ovšem její architektura je velice složitá a rychlost vyhodnocení výsledného snímkuje poloviční oproti síti *U-Net*. Velice

kvalitní výsledky poskytuje i autorem práce navržená síť *U-Net3*, která dosahuje výborných výsledků v oblasti kvality vyhodnocení, ovšem je náročná na čas učení i vyhodnocení snímku a z hlediska složitosti sítě je druhou nejsložitější použitou sítí, jak je možné vyčíst z Tabulka 14, kde jsou uvedeny detaily o složitosti architektury neuronové sítě a taktéž doba učení a vyhodnocení jednoho vstupního snímku.

Z Tabulka 16 je taktéž patrné, že velice kvalitních výsledků dosáhla neuronová síť BiSeNet, ovšem rychlost učení i zpracování snímku je nejpomalejší ve srovnání s nejrychlejší architekturou sítě *ResNet-18* je přibližně 13× pomalejší v učení a přibližně 26× pomalejší ve zpracování jednoho snímku, jak je patrné z Tabulka 14.

Z výsledků uvedených v Tabulka 16 vyplývá, že neuronová síť *U-Net*, která dosáhla v hodnocení nejlepších komplexních výsledků bez použití vah má jeden z nejmenších počtů trénovatelných proměnných. Druhá nejlepší síť, *FCN*, dosáhla téměř totožného bodového zisku, ovšem má 2× více trénovatelných parametrů. Jako třetí nejlepší byla vyhodnocena neuronová síť *U-Net3* a *ResNet-18*, které dosáhly shodného počtu bodů. Při pohledu do tabulky je ovšem patrné, že neuronová síť *ResNet-18* dosáhla velkého bodového zisku díky rychlosti učení, rychlosti zpracování i díky malé složitosti sítě, avšak v kvalitě výsledků byla podprůměrná. Naopak neuronová síť *U-Net3* získala malý součet bodů v kategoriích rychlosti učení, rychlosti zpracování i složitosti. Velké množství bodů naopak získala z kvality odhadu. Z výše uvedených důvodů bylo nutné stanovit váhu zkoumaných kritérií pro objektivnější vyhodnocení kvality konkrétní neuronové sítě pro daný problém.

10.1.2 Srovnání výsledků neuronových sítí s váhovým parametrem

Pro každou zkoumanou architekturu a každý zkoumaný parametr bylo stanoveno pořadí sítě. Toto pořadí je patrné v Tabulka 16. Čím lepší pořadí, tím více daná architektura v kategorii získala bodů. Je zjevné, že některé parametry jsou nesrovnatelné a není úplně vhodné používat shodné vážené metriky pro všechny kategorie. Ze zkoumaných parametrů byly vybrány takové, které mají pro zaměření této práce a vyhodnocení hloubkové mapy vyšší a nižší váhu.

V rámci práce byla stanovena kritéria vážené metriky dle požadavků práce. Ze zkoumaných parametrů dostaly nejvyšší prioritu parametry *MAE* a *MSE*, určující kvalitu učení neuronové sítě v průběhu celého datasetu. Čím nižší je součet chyb *MAE*

i *MSE*, tím vyšší metriku pro danou kategorii daná architektura získala. Druhou nejvyšší metriku měla rychlost vyhodnocení jednoho snímku na výpočetním systému, jehož parametry jsou uvedeny v Tabulka 9. Naopak nejnižší váženou metriku získaly parametry složitost sítě a rychlost učení. Koeficienty vážené metriky stanovené dle kritérií stanovených pro tuto práci jsou uvedeny v Tabulka 17. Výsledné bodové ohodnocení za využití koeficientů vážené metriky je uvedeno v Tabulka 18.

Z Tabulka 18 je patrné, že při aplikaci součtu vážených kategorií, dosáhla nejlepších výsledků síť *FCN*. Body navíc síť získala při snížení vážené metriky složitosti sítě a rychlosti učení. Druhou nejlepší sítí je modifikovaná síť *U-Net3*, jejíž struktura byla autorem práce upravena. Detaily úprav a navržená architektura jsou popsány v kapitole 4.2.10. Vyhodnocení kvality odhadu neuronové sítě vybraných oblastí

Tabulka 17: Vážená metrika jednotlivých kategorií (vyšší číslo znamená vyšší váhu daného parametru)

	Rychlost učení	Rychlost zpracování	Složitost sítě	MAE MAX	Σ MAE	MSE MAX	Σ MSE
Vážená metrika	0,25	1	0,25	1,5	1,5	1,5	1,5

Tabulka 18: Srovnání kvality testovaných neuronových sítí na základě vážené metriky (více je lépe).

Typ sítě		Rychlost učení	Rychlost zpracování	Složitost sítě	MAE MAX	Σ MAE	MSE MAX	Σ MSE	Σ
ResNet	**12**	**3,5**	**14**	**3,5**	4,5	4,5	6	4,5	40,5
	18	3,25	13	3,25	9	9	10,5	9	57
	50	3	12	2,75	7,5	7,5	12	7,5	52,25
	101	2,5	10	1,5	6	6	4,5	6	36,5
	200	1,5	8	1	3	3	3	3	22,5
U-Net		2,25	9	2,5	16,5	13,5	16,5	13,5	73,75
U-Net3		2	6	0,5	18	18	18	18	80,5
FCN		1,75	5	0,75	**21**	19,5	**21**	16,5	**85,5**
RefineNet		1,25	7	0,25	13,5	10,5	7,5	12	52
SegNet		1	4	3	12	16,5	13,5	19,5	69,5
BiSeNet		0,25	1	1,75	15	**21**	15	21	75
PSPNET_v3		0,5	2	2	19,5	15	19,5	15	73,5
DenseNet		2,75	11	1,25	1,5	1,5	1,5	1,5	21
Xception		0,75	3	2,25	10,5	12	9	10,5	48

10.2 Srovnání kvality neuronových sítí na vybraných oblastech

V rámci výběru vhodné neuronové bylo provedeno vyhodnocení základních statistických parametrů jako je průměr μ, směrodatná odchylka σ, či maximální a minimální hodnota pixelu ve zkoumané oblasti. Tyto hodnoty jsou následně převedeny do reálných vzdáleností postupem uvedeným v kapitole 7.4. Z výše uvedených statistických veličin je taktéž možné odhadnout kvalitu učení neuronové sítě. Oblast pro zkoumání byla vybrána manuálně na základě vstupních snímků a byla vybrána tak, aby obsahovala rovnou plochu, například zeď či sloup, tj. oblast s minimálním rozptylem hodnoty pixelu v originální hloubkové mapě, pro maximálně věrohodné vyhodnocení kvality odhadu výsledku neuronové sítě.

10.2.1 Testovací snímek M7-292

Obrázek 135: Ukázka testované oblasti – snímek z barevné a hloubkové kamery (Zdroj: Vlastní)

Obrázek 136: Srovnání vyhodnocení vybraného snímku M7-292. Zleva: U-Net, U-Net3, ResNet50, DenseNet. (Zdroj: Vlastní)

Z Tabulka 19 je patrné, že hloubková kamera změřila polohu sloupu v průměrné vzdálenosti 216,29 cm se směrodatnou odchylkou 2.91 cm od snímače. Rozdíl maximální a minimální hodnoty vzdálenosti Δ byl ve zkoumané oblasti 15,69 cm. Při porovnání výsledků je patrné, že celkově odhad oblasti z testovaných neuronových sítí nebyl špatný při řešení takto komplikovaného, problému jako je odhad hloubky na základě barevného snímku.

Tabulka 19: Srovnání statistických veličin z vybrané oblasti snímku M7-292.png

	μ [cm]	|Δμ| [cm]	σ [cm]	Min [cm]	Max [cm]	Δ [cm]
RGB-D kamera	216,29	0,00	2,91	207,84	223,53	15,69
BiSeNet	211,94	4,35	4,41	200,00	223,53	23,53
DenseNet	292,75	76,46	69,58	176,47	505,88	329,41
FCN	218,29	2,01	7,39	196,08	243,14	47,06
PSPNET_v3	218,45	2,17	6,19	203,92	235,29	31,37
RefineNet	207,21	9,08	7,69	188,24	231,37	43,14
ResNet-12	216,80	0,52	16,24	149,02	274,51	125,49
ResNet-18	226,31	10,02	18,21	133,33	274,51	141,18
ResNet-50	224,50	8,21	19,37	129,41	286,27	156,86
ResNet-101	213,54	2,74	21,10	117,65	286,27	168,63
ResNet-200	241,00	24,72	36,81	156,86	325,49	168,63
SegNet	224,37	8,09	3,59	211,76	235,29	23,53
U-Net	214,09	2,20	4,60	196,08	227,45	31,37
U-Net3	224,77	8,48	8,85	200,00	250,98	50,98
Xception	214,52	1,77	8,34	192,16	247,06	54,90

Nejhorší sítí v odhadu byla síť *DenseNet*, která má i jednoznačně nejhorší parametry v testování chyb v rámci celého datasetu i porovnání vyhodnocení konkrétní oblasti. Porovnáním dat v Tabulka 19 je taktéž možné zjistit, že relativně nejlépe si vedla již dříve zmiňovaná neuronová síť U-Net, která v oblasti určila průměrnou vzdálenost 214 cm se směrodatnou odchylkou 4,6 cm. Z Tabulka 19 je taktéž patrné, že většina testovaných neuronových sítí dosáhla kvalitních výsledků, konkrétně *BiSeNet*, *FCN*, *PSPNET_v3, SegNet*, *U-Net*, či *Xception*. Extrémní nepřesnosti naopak dosáhla síť *DenseNet.*

10.2.2 Testovací snímek M7-2545

Obrázek 137: Ukázka testované oblasti – snímek z barevné a hloubkové kamery (Zdroj: Vlastní)

Obrázek 138: Srovnání vyhodnocení vybraného snímku M7-2545. Zleva: U-Net, U-Net3, ResNet50, DenseNet. (Zdroj: Vlastní)

Na Obrázek 138 je vidět porovnání výsledků z vybraných neuronových sítí dle kvality výsledků. Levý obrázek je výstup z neuronové sítě U-Net, která dosáhla v kategorii shodných vah nejlepší výsledek, v kategorii s váženou metrikou byla třetí nejlepší sítí. Druhá zleva je výsledek sítě U-Net3, která dosáhla velice dobrých výsledků. Třetí zleva je pro porovnání přidána neuronová síť ResNet50, která při učení získala 6. pozici ve vyhodnocení bodů, tj. byla ve výsledném hodnocení bez využití vah vyhodnocena jako průměrná. Vpravo je zobrazena dle výsledků absolutně nejhorší neuronová síť DenseNet, jejíž architektura je sice schopna částečně odhadnout hloubku a hrubé obrysy objektů, ovšem kvalita je velice slabá. Z Obrázek 137 a Obrázek 138 je taktéž patrné, že do testovacího datasetu byl omylem přidán i snímek, obsahující oslněné hloubkové kamery externím zdrojem světla. Podobné artefakty se občas ve zpracovaných datech z neuronových sítí objevují.

Tabulka 20: Srovnání statistických veličin z vybrané oblasti snímku M7-2545.png

	μ [cm]	|Δμ| [cm]	σ [cm]	Min [cm]	Max [cm]	Max-Min [cm]
RGB-D kamera	173,91	0,00	3,96	164,71	180,39	15,69
BiSeNet	170,73	3,19	7,53	152,94	184,31	31,37
DenseNet	222,00	48,09	51,17	125,49	368,63	243,14
FCN	178,20	4,28	8,33	160,78	203,92	43,14
PSPNET_v3	177,84	3,93	4,36	168,63	188,24	19,61
RefineNet	170,06	3,85	7,46	156,86	184,31	27,45
ResNet-12	171,47	2,44	22,26	113,73	239,22	125,49
ResNet-18	188,46	14,55	12,18	137,25	215,69	78,43
ResNet-50	171,51	2,40	13,86	109,80	223,53	113,73
ResNet-101	182,00	8,09	12,68	145,10	243,14	98,04
ResNet-200	205,43	31,52	21,03	156,86	266,67	109,80
SegNet	174,10	0,19	3,58	160,78	180,39	19,61
U-Net	179,13	5,22	5,48	164,71	196,08	31,37
U-Net3	175,35	1,44	5,62	156,86	188,24	31,37
Xception	171,77	2,14	5,18	156,86	184,31	27,45

10.2.3 Testovací snímek M7-2454

Obrázek 139: Ukázka testované oblasti – snímek z barevné a hloubkové kamery (Zdroj: Vlastní)

Obrázek 140: Srovnání vyhodnocení vybraného snímku M7-2454. Zleva: U-Net, U-Net3, FCN, DenseNet. (Zdroj: Vlastní)

Na Obrázek 140 je vidět srovnání snímků ze sítě *U-Net, U-Net3, FCN a DenseNet.* Kvalita sítě *FCN, U-Net a U-Net3* je na první pohled patrná. Síť *FCN* v tomto snímku dosáhla velice dobré přesnosti odhadu průměrné hodnoty i směrodatné odchylky. Z Tabulka 21 je taktéž patrné, že v tomto konkrétním snímku dosáhla lepších parametrů architektura *BiSeNet* a *FCN* lepších parametrů než *U-Net*. Velice kvalitních výsledků taktéž dosáhla autorem práce navržená architektura *U-Net3*.

Tabulka 21: Srovnání statistických veličin z vybrané oblasti snímku M7-2454.png

	μ [cm]	|Δμ| [cm]	σ [cm]	Min [cm]	Max [cm]	Max-Min [cm]
RGB-D kamera	323,10	0,00	5,64	313,73	345,10	31,37
BiSeNet	323,12	0,01	5,94	305,88	341,18	35,29
DenseNet	322,37	0,74	34,32	231,37	427,45	196,08
FCN	323,49	0,38	7,66	305,88	341,18	35,29
PSPNET_v3	321,39	1,71	9,04	301,96	341,18	39,22
RefineNet	319,74	3,36	6,46	301,96	337,25	35,29
Resnet12	316,65	6,46	18,27	290,20	360,78	70,59
Resnet18	341,87	18,76	11,19	317,65	376,47	58,82
Resnet50	324,36	1,26	8,43	309,80	352,94	43,14
Resnet101	336,36	13,25	14,39	309,80	376,47	66,67
Resnet200	328,65	5,55	24,49	282,35	384,31	101,96
Segnet	318,42	4,69	7,91	301,96	337,25	35,29
Unet	322,70	0,40	8,72	309,80	345,10	35,29
Unet3	312,81	10,29	7,07	298,04	329,41	31,37
Xception	322,39	0,71	7,89	305,88	349,02	43,14

10.3 Shrnutí výsledků

Z výsledků uvedených v této kapitole byl potvrzen předpoklad, že neuronová síť U-Net je vhodnou architekturou pro sémantický odhad hloubkové mapy na základě vstupního barevného snímku. Mimo předpokládanou architekturu sítě *U-Net* vynikly taktéž architektury *U-Net3* a *FCN*, které byly dle kritérií dokonce lepší. Všechny tyto architektury ovšem dosáhly velice podobných výsledků a jsou použitelné pro zpracování dat v robotické platformě v online podobě. Doba potřebná pro vyhodnocení jednoho snímku o rozměrech 256×256 pixelů byla u výše zmíněných architektur maximálně 33,5 ms, při vyhodnocení na výpočetní sestavě, jejíž konfigurace je uvedena v Tabulka 9.

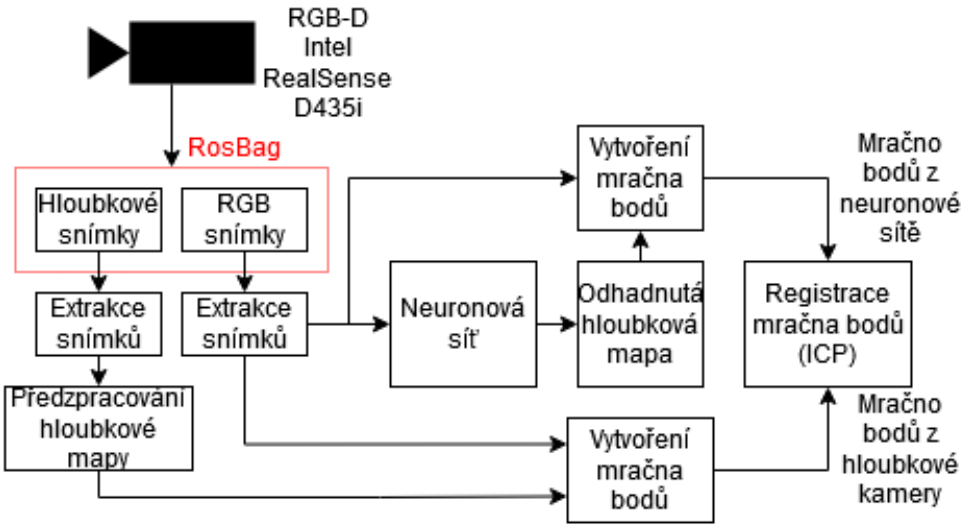
Autor práce předpokládal, že neuronová síť *ResNet-12*, *ResNet-18* a *ResNet-50* budou slabé neuronové sítě z hlediska velice malého počtu trénovatelných koeficientů. V hodnocení dle vážené metriky ze všech testovaných variant dosáhla největšího bodového ohodnocení varianta *ResNet-18*. Složitější architektury *ResNet-101* a *ResNet-200* dosáhly dle vážené metriky dokonce horších výsledků než varianta *ResNet-18.*

Počet epoch učení byl záměrně stanoven na stejný počet, aby bylo možné provést srovnání kvality učení za stejných podmínek. Rizikem tohoto požadavku byla možnost přeučení jednodušších architektur neuronových sítí a možnost nedoučení složitějších. Během učení byla ztrátová funkce přísně sledována, která se při učení již neměnila a je tedy možné uvažovat, že všechny architektury sítí byly pravděpodobně dostatečně naučeny na problematiku odhadu hloubkové mapy, což se i dle srovnání výsledků zdařilo.

Z výše uvedených grafů bylo taktéž zjištěno, že neuronová síť *DenseNet* není pro aplikaci odhadu hloubkové mapy vhodná. Architektura byla schopna detekovat pouze hrubé obrysy objektů, jak je patrné na Obrázek 122 a Obrázek 123, ovšem kvalita výsledného snímku je ve srovnání s ostatními snímky velice slabá. Pro případné další použití by bylo nutné architekturu sítě *DenseNet* pro odhad hloubkového snímku minimálně přepracovat, spíše však architektura není úplně vhodná na řešení tohoto problému.

11 Vygenerování a zarovnání mračna bodů

Tato kapitola se zabývá vytvořením mračna bodů z vygenerovaných hloubkových map z neuronových sítí. Tato vytvořená mračna je následně možné dále zpracovat pro vytvoření výsledné mapy okolního prostoru za využití metod SLAM. Pro ověření kvality hloubkových snímků získaných z neuronových sítí byla vytvořena mračna bodů, která byla porovnána s mračnem bodů vygenerovaných z dat získaných z hloubkové kamery. V práci byly testovány dvě metody registrace mračna bodů – *ICP* a *NDT*. Popis metod práce s mračnem bodů a jejich registraci je popsán v kapitole 5.



Obrázek 141: Ukázka řetězce registrace mračna bodů a jejich porovnání (Zdroj: Vlastní)

Na Obrázek 141 je zobrazen řetězec pro vygenerování dvou sad mračen bodů. První sada mračna bodů je vytvořena z předzpracovaných hloubkových map pomocí hloubkové kamery. Totožný postup předzpracování hloubkových map je použita pro přípravu dat pro učení neuronové sítě a celý postup je popsán v kapitole 7.

Druhá sada mračna bodů je vytvořena za použití shodného RGB snímku jako v první sadě, ovšem pro vytvoření konkrétního mračna je použita odhadnutá hloubková mapa pomocí neuronové sítě.

Sady mračen bodů jsou následně registrovány pomocí metody *ICP*. Výsledky z registrace mračen bodů jsou detailně popsány v této kapitole.

11.1 Tvorba mračna bodů

Mračno bodů je možné vytvořit několika způsoby a záleží jen na volbě uživatele, jaký přístup zvolí. Autor práce zvolil přístup přes jazyk Python a Matlab. Výsledný přístup byl zvolen použití jazyka Python. Důvodů bylo několik. Prvním důvodem je fakt, program Matlab poskytuje velké množství funkcí pro práci s mračnem bodů, ovšem předpokládá se již existující soubor. Vytvoření kvalitního mračna bodů je v Matlabu výrazně složitější, a to z důvodu, že funkce programu Matlab neumožnují načíst soubor s vnitřními parametry kamery (*Camera Intrinsic*). Matlab podporuje v době psaní této práce načtení těchto parametrů pouze z Kinectu a to pouze za předpokladu připojené RGB-D kamery. Dalším a největším důvodem byl fakt, že pro jazyk Python existují rozsáhlé knihovny pro práci s mračny bodů nazvané Open3D (Zhou, a další, 2018), (Ope21). Tyto knihovny mají otevřený zdrojový kód včetně rozsáhlé dokumentace a implementace do vývojového prostředí je za splnění požadavků velice jednoduchá. Tyto knihovny poskytují funkce, kde je možné při tvorbě mračna bodů předložit soubor či matici s parametry kamery.

Výsledkem po provedení výše uvedených příkazů je vytvořené mračno bodů, které je možné dále zpracovat v jakémkoliv jiném programu a za využití různých knihoven. Pro další zpracování byl použit program Matlab, zejména z důvodu kvalitnějšího zobrazení výsledných mračen bodů a jejich ukládání.

11.2 Porovnání vygenerovaných mračen pro vybrané neuronové sítě

V kapitole 10 bylo provedeno vyhodnocení kvality vyhodnocení snímků získaných z testovaných neuronových sítí. Z velké skupiny byly na základě stanovených kritérií vybrány tři nejlepší neuronové sítě, tj. *FCN*, *U-Net3* a *U-Net*. *U-Net* jako taková dopadla v hodnocení dle vážené metriky jako třetí nejlepší, ovšem její bodové hodnocení bylo s neuronovou sítí *BiSeNet* hraniční. Při porovnání byla rychlost zpracování sítě *U-Net* mnohonásobně větší než u sítě *BiSeNet*. Pro porovnání byla přidána neuronová síť, která dopadla ze všech testovaných architektur nejhůře – *DenseNet*.

Všechny hloubkové snímky získané z neuronových sítí mají rozměry 256×256 pixelů. Pro lepší přehlednost a taktéž porovnání výsledků získaných z hloubkové kamery, byly hloubkové snímky převzorkovány na rozlišení 640×480 pixelů.

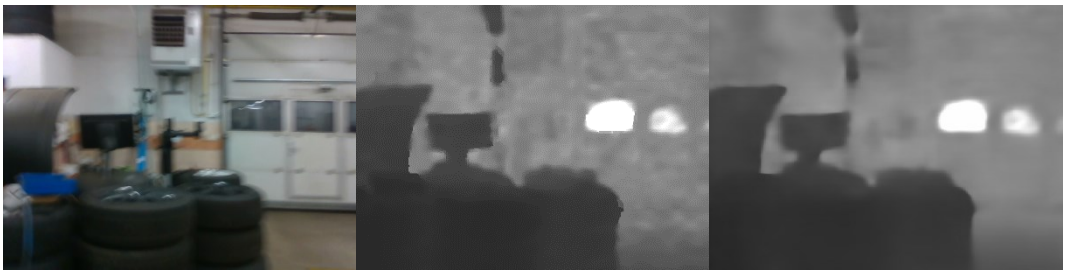
V rámci testování byly vybrány různé ukázkové snímky pro představu, jak kvalitně došlo k učení neuronových sítí. Pro porovnání byly vybrány shodné scény pro neuronové sítě, které byly vyhodnoceny jako nejlepší. Tato vygenerovaná mračna bodů z neuronové sítě byla následně registrována metodou *ICP* s mračnem bodů vygenerovaným z hloubkové kamery. Výsledkem je porovnání mračen bodů pro představu, jak se neuronové sítě trefily ve svém odhadu. Dále byly vytvořeny rozdílové hloubkové snímky pro vyhodnocení míst, kde jsou výraznější změny v rozdílu vzdálenosti odhadnuté neuronovou sítí a změřené hloubkovou kamerou.

Ve všech testovaných neuronových sítí se projevuje vlastnost neuronových sítí, kde se neuronová síť chová jako filtr typu dolní propust. Je patrné, že největší rozdíly jsou v místech, kde dochází k výrazné změně vzdálenosti, jako jsou například hrany objektů. Neuronová síť má problém takto rychlé změny pokrýt. V místech, kde jsou celistvé plochy bez výrazných změn vzdálenosti, tento problém nevzniká a neuronová síť je schopna odhadnout vzdálenost velice přesně. V případě těchto rovných ploch je možné si všimnout rozdílů mezi hloubkovým snímkem z hloubkové kamery a výstupním hloubkovým snímkem z neuronové sítě, které jsou uvedeny na Obrázek 142, Obrázek 154, Obrázek 166 či Obrázek 178. Z obrázků je patrné, že na rovných plochách kamera generuje šum, projevující se fluktuací odhadnuté vzdálenosti. Tato vlastnost kamery je nejvíce patrná ve větších vzdálenostech. Jedná se o fluktuace ve vyšších jednotkách, občas i menších desítkách centimetrů. Neuronová síť ze své podstaty filtru typu dolní propust tyto vysokofrekvenční složky obrazu filtruje, což je v určitých místech výhoda, jako například v místě rovné celistvé plochy, ale v místech výrazných změn vzdálenosti vznikají výraznější špičky, jak je patrné na Obrázek 146, Obrázek 158, Obrázek 170 či Obrázek 182.

Ze všech rozdílových snímků byl vytvořen histogram a v histogramu jsou vyznačeny intervaly spolehlivosti pokrývající 68,27%, 95,45% a 99,73% všech hodnot. Pro základní odstranění extrémních hodnot (občas nazývaných anglicky *outliers*) se nejčastěji používá hranice 95,45%, což odpovídá hustotě normálního náhodného rozdělení s $\mu \pm 2\sigma$. V případě rozdílového snímku se ovšem nedá normální rozdělení použít, jelikož data nemají normální rozdělení a rozdělení bodů silně závisí na snímané scéně. Pro ilustraci rozdělení dat a případné odstranění odlehlých hodnot jsou tyto procentuální hranice použitelné.

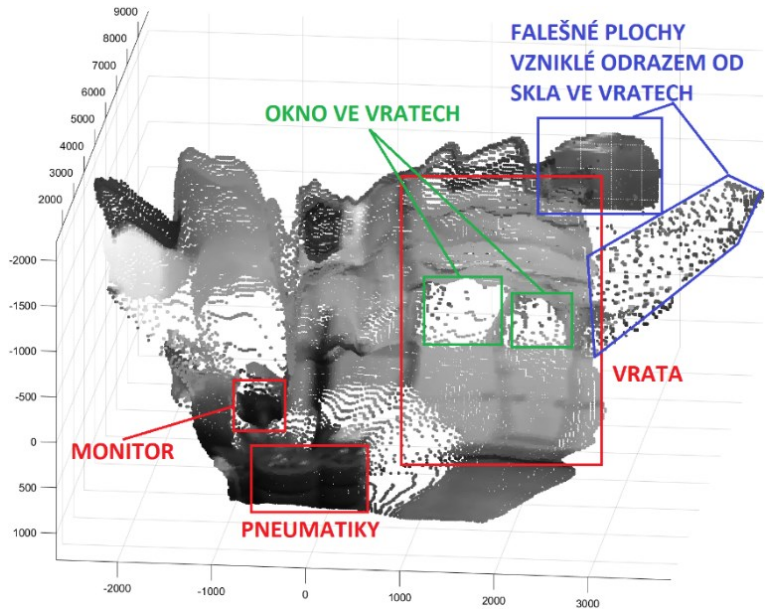
11.2.1 U-Net3: Výsledky registrace mračna bodů

11.2.1.1 Analýza snímku M7_2694



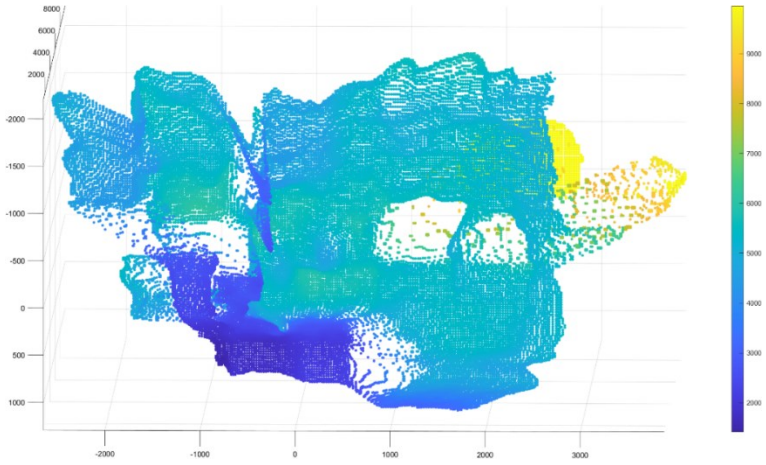
Obrázek 142: Vstupní snímky M7_2694 pro vytvoření mračna bodů. Vlevo: Snímek z barevné kamery. Uprostřed: Výstup z hloubkové kamery. Vpravo: Výstup ze sítě U-Net3. (Zdroj: Vlastní)

Na Obrázek 142 jsou zobrazeny vstupní snímky nezbytné pro vytvoření mračna bodů. Konkrétně se jedná o výstup z autorem navržené neuronové sítě U-Net3. Výsledek po vytvoření mračna bodů je uveden na Obrázek 143.



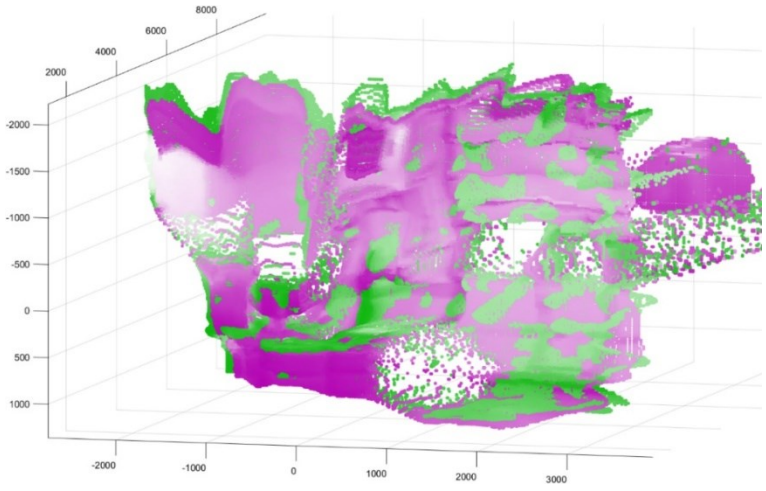
Obrázek 143: Vytvořené mračno bodů ze snímku M7_2694 z neuronové sítě U-Net3. (Zdroj: Vlastní)

Na Obrázek 143 je zobrazen výstup z neuronové sítě U-Net3. Pro lepší přehlednost byl každý voxel obarven hodnotou odpovídající pixelu v barevném snímku převedeného do odstínů šedi. Z obrázku je patrné, že neuronová síť odhadla hloubkový snímek velice dobře. Ve snímku jsou patrné i chyby, vzniklé chybějícími daty v místech, kde je ve vratech sklo. Hloubková kamera při snímání trénovacích dat není schopna lesklé plochy správně nasnímat. V tomto případě se jedná o místa, kde hloubková kamera změřila, že vzdálenost je větší než maximální rozsah, tudíž v mračnu bodů má konkrétní voxel nastavenou maximální vzdálenost, tj. 10 metrů.



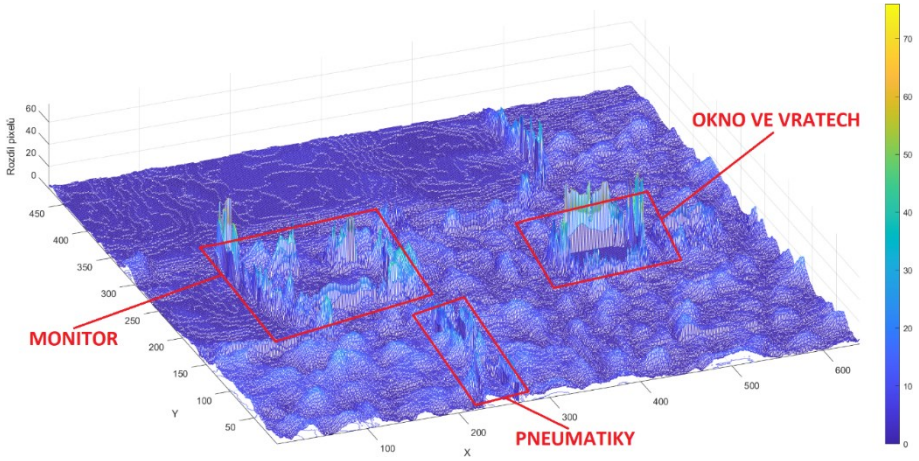
Obrázek 144: Vytvořené mračno bodů ze snímku M7_2694 z neuronové sítě U-Net3. Přebarveno dle vzdálenosti voxelu (Zdroj: Vlastní)

Pro lepší přehled odhadnuté hloubky, bylo provedeno přebarvení všech voxelů dle odhadnuté vzdálenosti a výsledné mračno bodů je zobrazeno na Obrázek 144. Studená modrá barva značí blízké objekty, teplá žlutá barva značí vzdálené objekty.

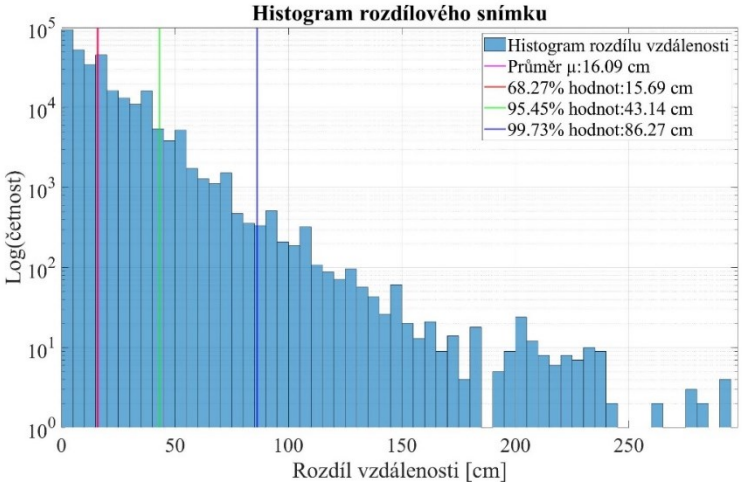


Obrázek 145: Srovnání mračna bodů vytvořené z hloubkové kamery (zelená barva) a z neuronové sítě U-Net3 (fialová barva). (Zdroj: Vlastní)

Na Obrázek 145 je zelenou barvou zobrazeno porovnání mračna bodů vygenerovaných z hloubkové kamery, fialovou barvou je zobrazeno mračno bodů vygenerovaných z neuronové sítě U-Net3. Z obrázku je patrné, že mračna bodů se liší, ale rozdíl je velice malý. Tento rozdíl je zobrazen na Obrázek 146. Rozdíly jsou patrné zejména v okolí oken ve vratech, u monitoru počítače a v jeho okolí a taktéž na hraně pneumatik.



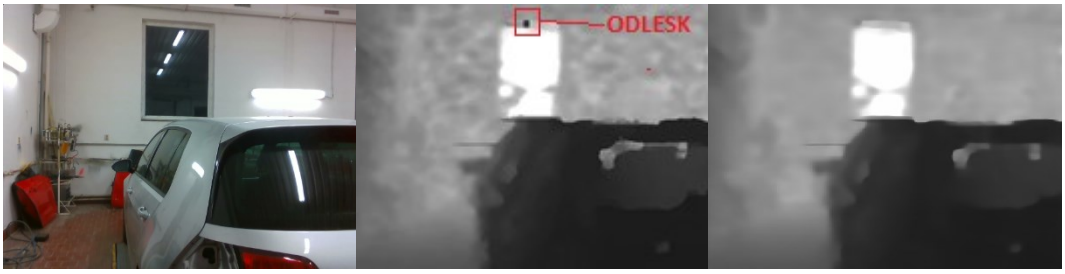
Obrázek 146: Rozdílový snímek M7_2694 mezi hloubkovou mapou získanou z hloubkové kamery a hloubkovou mapou získané z neuronové sítě U-Net3 (Zdroj: Vlastní)



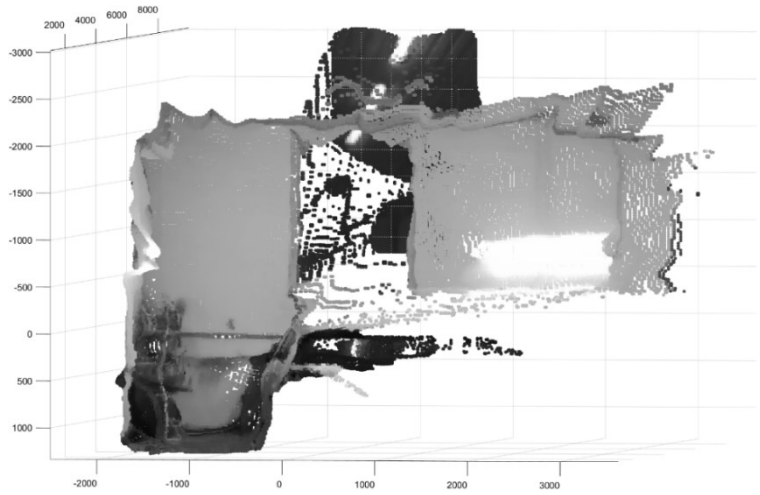
Obrázek 147: Histogram rozdílového snímku M7_2694 sítě U-Net3 (Zdroj: Vlastní)

Na Obrázek 147 je zobrazen histogram četnosti rozdílu pixelů z rozdílového snímku zobrazeného na Obrázek 146. Četnost je zobrazena v logaritmickém měřítku. Z Obrázek 147 je patrné, že přibližně 68% veškerých odhadnutých bodů má maximální chybu odhadu vzdálenosti menší než 15,69 cm a přibližně 95% odhadnutých bodů má chybu menší než 43,14 cm. Základním problémem v tomto vyhodnocení jsou odlehlé body, které vznikají na hranách objektů, v tomto případě v okolí oken či monitoru. Jedná se ovšem o lokální chyby, které je možné při pozdějším zpracování mračna bodů filtrovat. Při vyjádření procentuální chyby, je možné vyhodnotit, že pro přibližně 68% odhadnutých bodů na rozsahu 10 metrů má odhad neuronové sítě *U-Net* chybu menší než 1,56% a pro 95% odhadnutých bodů má chybu menší než 4,3%.

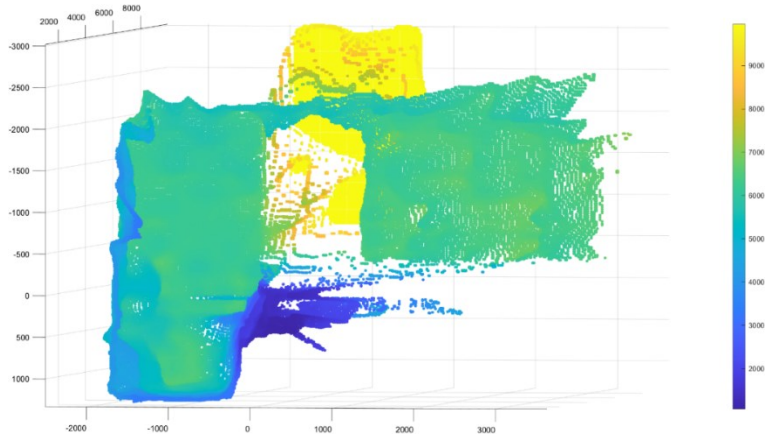
11.2.1.2 Analýza snímku M7_2396



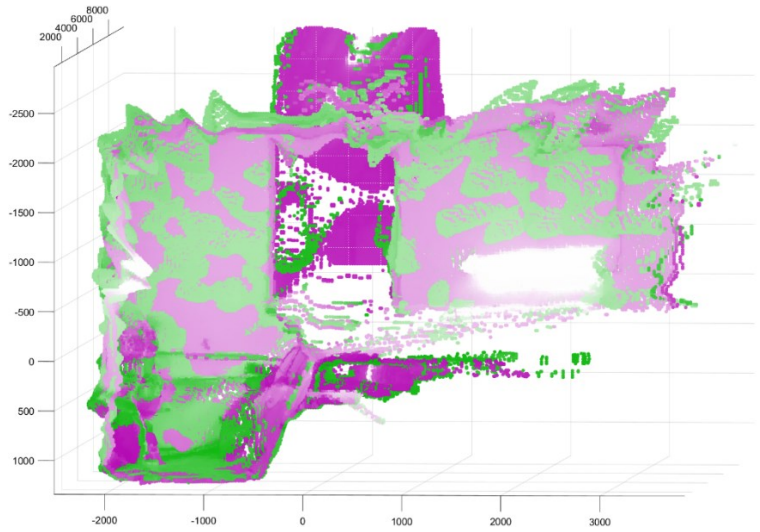
Obrázek 148: Vstupní snímky M7_2396 pro vytvoření mračna bodů. Vlevo: Snímek z barevné kamery. Uprostřed: Výstup z hloubkové kamery. Vpravo: Výstup ze sítě U-Net3. (Zdroj: Vlastní)



Obrázek 149: Vytvořené mračno bodů ze snímku M7_2396 z neuronové sítě U Net3. (Zdroj: Vlastní)

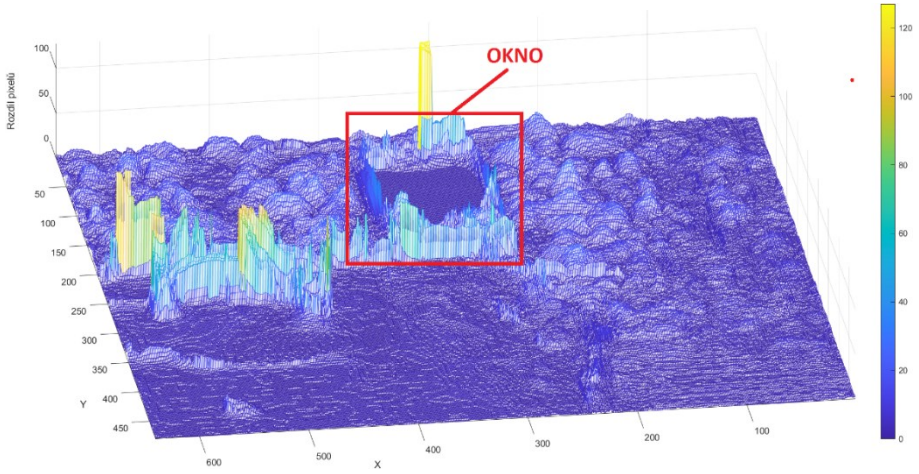


Obrázek 150: Vytvořené mračno bodů ze snímku M7_2396 z neuronové sítě U-Net3. Přebarveno dle vzdálenosti voxelu (Zdroj: Vlastní)



Obrázek 151: Srovnání mračna bodů vytvořené z hloubkové kamery (zelená barva) a z neuronové sítě U-Net3 (fialová barva). (Zdroj: Vlastní)

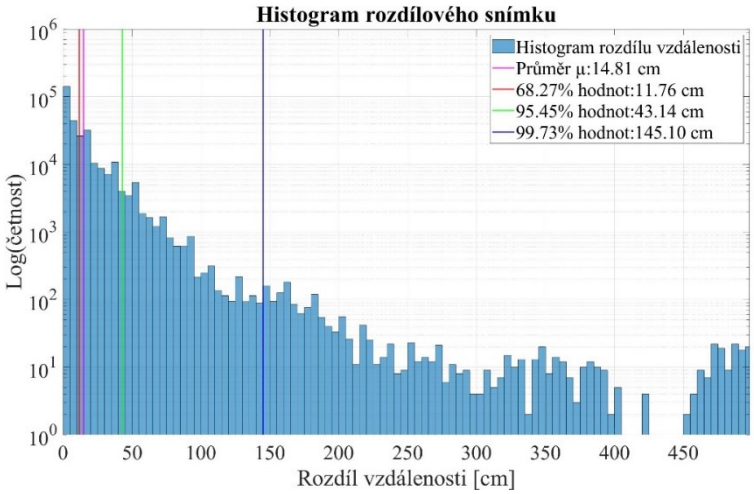
Na Obrázek 151 je zobrazeno srovnání vytvořeného mračna bodů získaného ze snímků z hloubkové kamery (zelená barva) a snímků vytvořených z hloubkového snímku vytvořeného neuronovou sítí *U-Net3*. Z obrázku je patrné, že došlo k velice kvalitní shodě s minimálními rozdíly. Pro přehlednost jsou tyto rozdíly zobrazeny jako rozdílový snímek na Obrázek 152.



Obrázek 152: Rozdílový snímek M7_2396 mezi hloubkovou mapou získanou z hloubkové kamery a hloubkovou mapou získané z neuronové sítě U-Net3 (Zdroj: Vlastní)

Rozdílový snímek zobrazený na Obrázek 152 ukazuje kritická místa, kde má neuronová síť *U-Net3* problém správně odhadnout hloubku daného voxelu. Jedná se zejména o místa, kde dochází ke skokové změně vzdálenosti, jako je například okno nebo střecha automobilu. Místo nad oknem, kde je patrná žlutá špička, ovšem není úplně chyba vyhodnocení neuronové sítě. Při porovnání snímků na Obrázek 148 je patrné, že v místě, kde je okno, došlo k odrazu světla ze zářivek od skla. Hloubková kamera tento odraz zaznamenala jako místo s malou vzdáleností. Tento bod je patrný na Obrázek 148

jako černá skupina bodů v horní části okna. Neuronová síť ovšem tuto velkou změnu ignorovala a vytvořila vlastní odhad hloubky, který je v tomto konkrétním případě přesnější než skutečný snímek z hloubkové kamery.

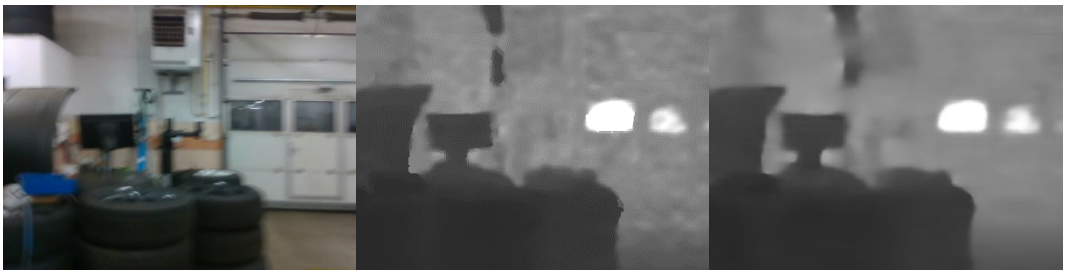


Obrázek 153: Histogram rozdílového snímku M7_2396 sítě U-Net3 (Zdroj: Vlastní)

Na Obrázek 153 je zobrazen histogram četnosti rozdílu pixelů z rozdílového snímku zobrazeného na Obrázek 152. Četnost je zobrazena v logaritmickém měřítku. Z histogramu je patrné, že přibližně 68% veškerých odhadnutých bodů má maximální chybu odhadu vzdálenosti menší než 11,76 cm a přibližně 95% odhadnutých bodů má chybu menší než 43,14 cm. Z histogramu je patrné, že neuronová síť na vzdálenosti do 10 metrů pro přibližně 68% bodů provedla velice kvalitní odhad s chybou menší než přibližně 1,1% a pro přibližně 95% je chyba odhadu vzdálenosti menší než 4,3%.

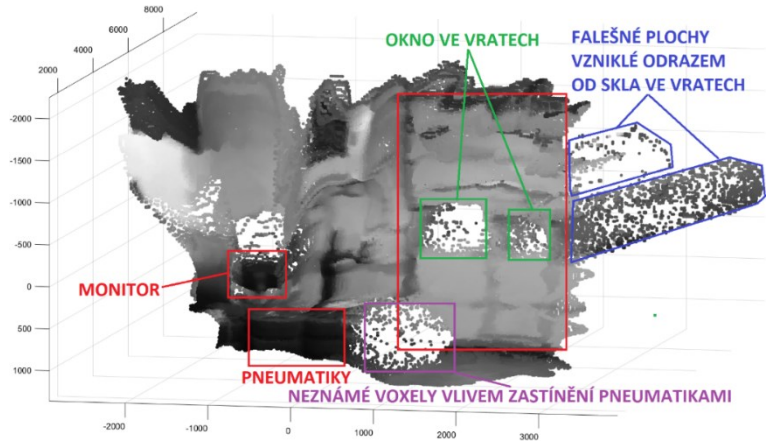
11.2.2 U-Net: Výsledky registrace mračna bodů

11.2.2.1 Analýza snímku M7_2694



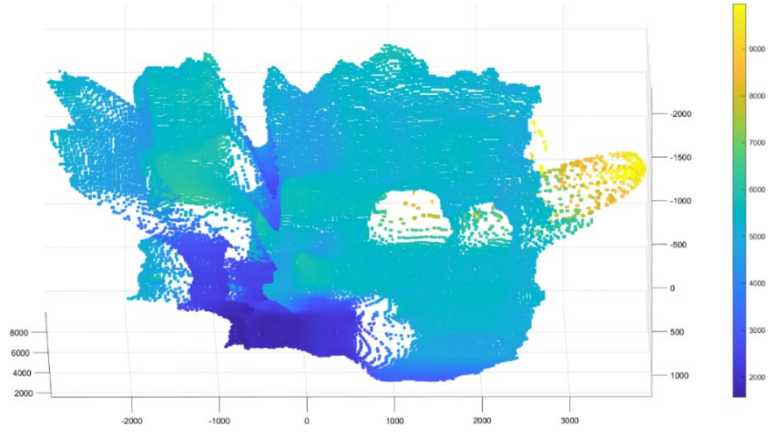
Obrázek 154: Vstupní snímky M7_2694 pro vytvoření mračna bodů. Vlevo: Snímek z barevné kamery. Uprostřed: Výstup z hloubkové kamery. Vpravo: Výstup ze sítě U-Net. (Zdroj: Vlastní)

Na Obrázek 154 jsou zobrazeny vstupní snímky nezbytné pro vytvoření mračna bodů. Snímek vpravo zobrazuje výstup z neuronové sítě *U-Net*. Z obrázku je patrné, že neuronová síť U-Net velice dobře odhadla hloubkovou mapu. Výsledek po vytvoření mračna bodů je uveden na Obrázek 155.



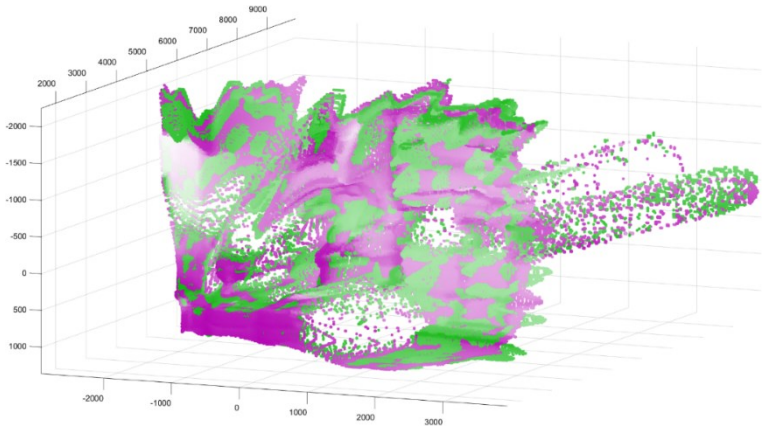
Obrázek 155: Vytvořené mračno bodů ze snímku M7_2694 z neuronové sítě U-Net. (Zdroj: Vlastní)

Na Obrázek 155 je zobrazen výstup z neuronové sítě U-Net. Pro lepší přehlednost byl každý voxel obarven hodnotou odpovídající pixelu v barevném snímku převedené do odstínů šedi. Z obrázku je patrné, že neuronová síť odhadla hloubkový snímek velice dobře. Ve snímku jsou patrné i chyby, vzniklé chybějícími daty v místech, kde je ve vratech sklo, jako je tomu u téměř všech neuronových sítí. Při porovnání sítě *U-Net* a *U-Net3* jsou patrné rozdíly zejména v oblastech falešných ploch.

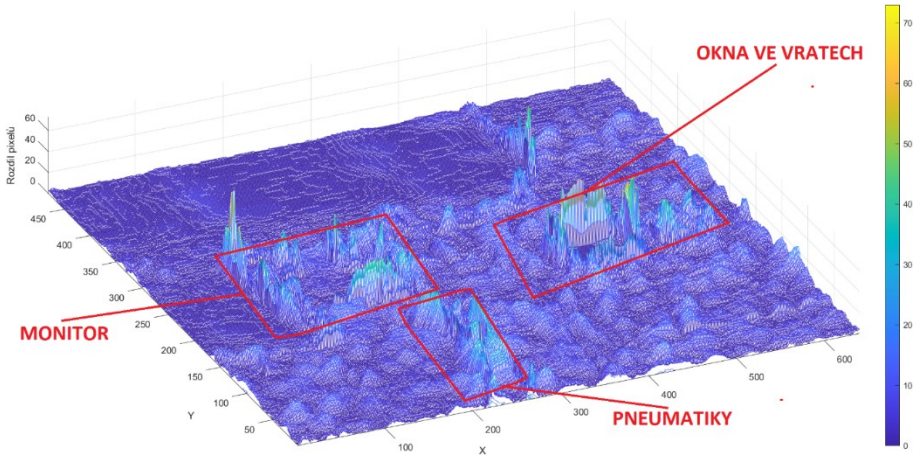


Obrázek 156: Vytvořené mračno bodů ze snímku M7_2694 z neuronové sítě U-Net. Přebarveno dle vzdálenosti voxelu (Zdroj: Vlastní)

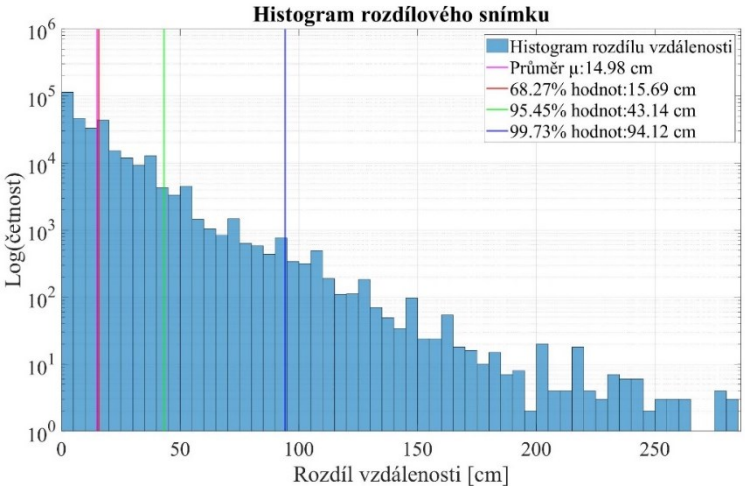
Pro lepší přehlednost je na Obrázek 156 snímek, jako je zobrazen na Obrázek 155, jen s čelním pohledem z důvodu lepší přehlednosti. Barva voxelu byla stanovena dle vzdálenosti. Studená modrá barva značí blízké voxely, teplá žlutá barva značí vzdálené voxely.



Obrázek 157: Srovnání mračna bodů vytvořené z hloubkové kamery (zelená barva) a z neuronové sítě U-Net (fialová barva). (Zdroj: Vlastní)



Obrázek 158: Rozdílový snímek M7_2694 mezi hloubkovou mapou získanou z hloubkové kamery a hloubkovou mapou získané z neuronové sítě U-Net (Zdroj: Vlastní)

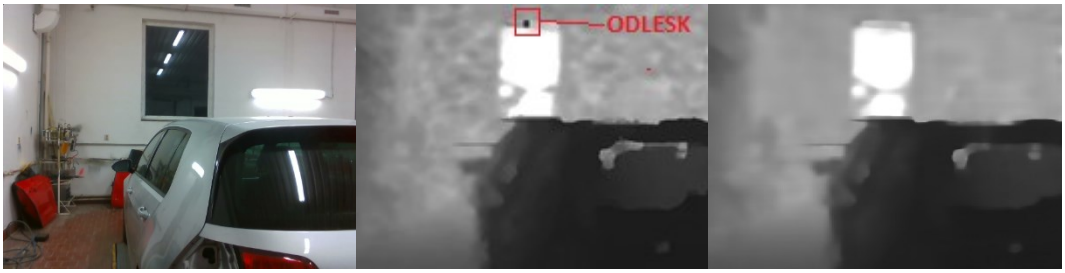


Obrázek 159: Histogram rozdílového snímku M7_2694 sítě U-Net (Zdroj: Vlastní)

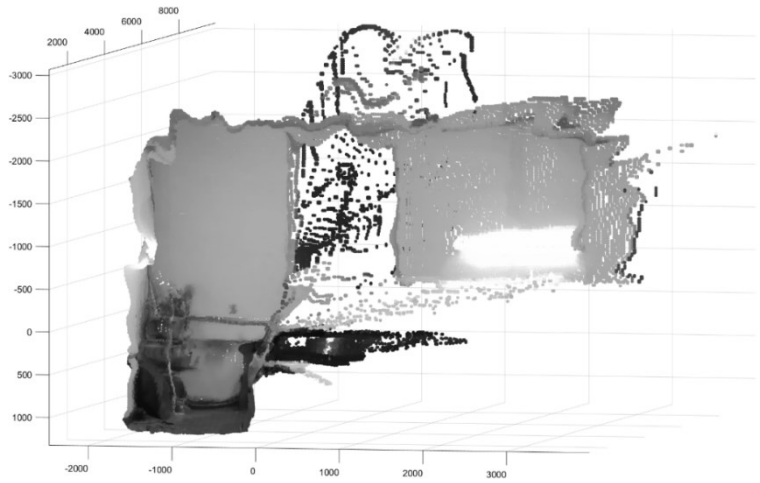
Z histogramu zobrazeného na Obrázek 159 je možné vyčíst, že neuronová síť *U-Net* pro daný snímek provedla odhad vzdálenosti pro přibližně 68% bodů s chybou menší než 15,96 cm, což na vzdálenosti rozsahu 10 metrů odpovídá chybě menší než 1,59%,

pro přibližně 95% bodů je maximální chyba odhadu vzdálenosti 43,13 cm, což odpovídá chybě menší než 4,3% na vzdálenosti 10 metrů.

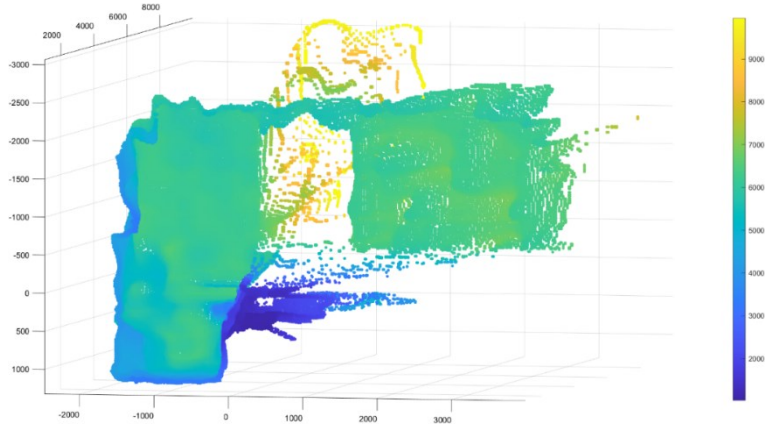
11.2.2.2 Analýza snímku M7_2396



Obrázek 160: Vstupní snímky M7_2396 pro vytvoření mračna bodů. Vlevo: Snímek z barevné kamery. Uprostřed: Výstup z hloubkové kamery. Vpravo: Výstup ze sítě U-Net. (Zdroj: Vlastní)



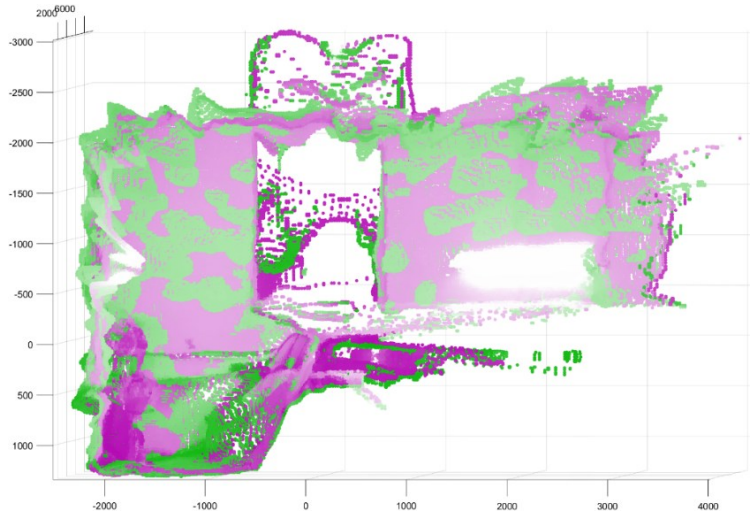
Obrázek 161: Vytvořené mračno bodů ze snímku M7_2396 z neuronové sítě U-Net. (Zdroj: Vlastní)



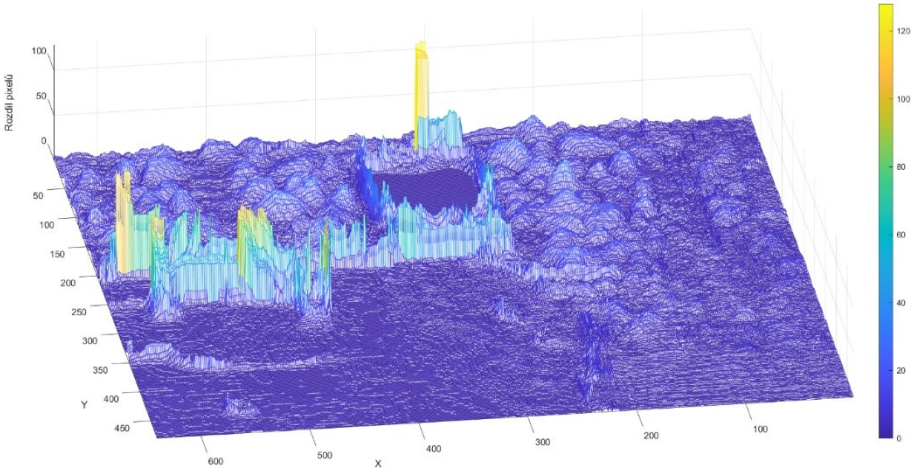
Obrázek 162: Vytvořené mračno bodů ze snímku M7_2396 z neuronové sítě U-Net. Přebarveno dle vzdálenosti voxelu (Zdroj: Vlastní)

Pro lepší přehlednost je na Obrázek 162 snímek, jako je zobrazen na Obrázek 161. Barva voxelu byla stanovena dle vzdálenosti. Studená modrá barva značí blízké voxely, teplá žlutá barva značí vzdálené voxely. Pro porovnání je na Obrázek 163 zobrazeno,

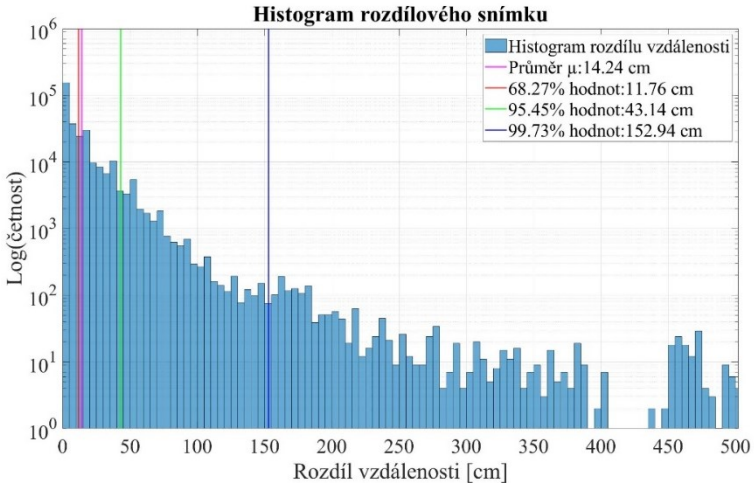
jaké jsou rozdíly mezi mračnem bodů získaným z hloubkové kamery a mračnem bodů vygenerovaným na základě hloubkového snímku z neuronové sítě *U-Net*. Z obrázku je patrné, že obě mračna jsou téměř shodná, což potvrzuje i rozdílový snímek Obrázek 164. Rozdíly jsou patrné, jako u všech ostatních neuronových sítí na hranách objektů, kde síť není schopna pokrýt rychlou impulsní změnu v obraze.



Obrázek 163: Srovnání mračna bodů vytvořené z hloubkové kamery (zelená barva) a z neuronové sítě U-Net (fialová barva). (Zdroj: Vlastní)



Obrázek 164: Rozdílový snímek M7_2396 mezi hloubkovou mapou získanou z hloubkové kamery a hloubkovou mapou získané z neuronové sítě U-Net (Zdroj: Vlastní)



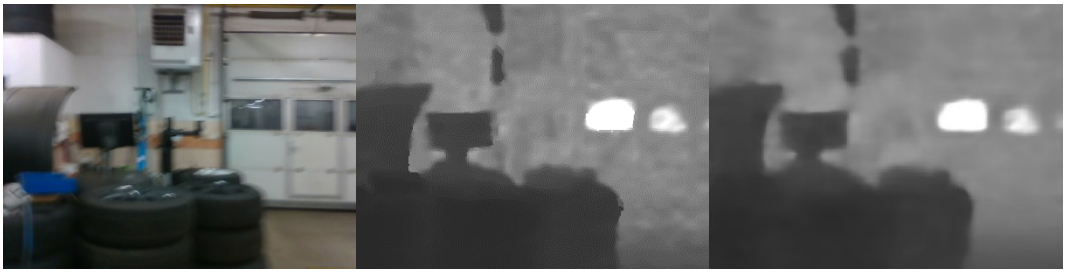
Obrázek 165: Histogram rozdílového snímku M7_2396 sítě U-Net (Zdroj: Vlastní)

Z histogramu zobrazeného na Obrázek 165 je možné vyčíst, že neuronová síť *U-Net* pro daný snímek provedla odhad vzdálenosti pro přibližně 68% bodů s chybou menší než 11,76 cm, což na vzdálenosti rozsahu 10 metrů odpovídá chybě menší než 1,17%, pro přibližně 95% bodů je maximální chyba odhadu vzdálenosti 43,14 cm, což odpovídá chybě menší než 4,3% na vzdálenosti 10 metrů.

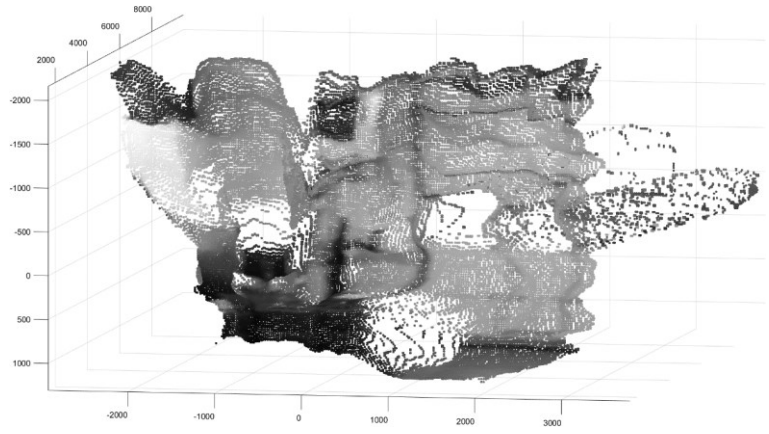
11.2.3 FCN: Výsledky registrace mračna bodů

11.2.3.1 Analýza snímku M7_2694

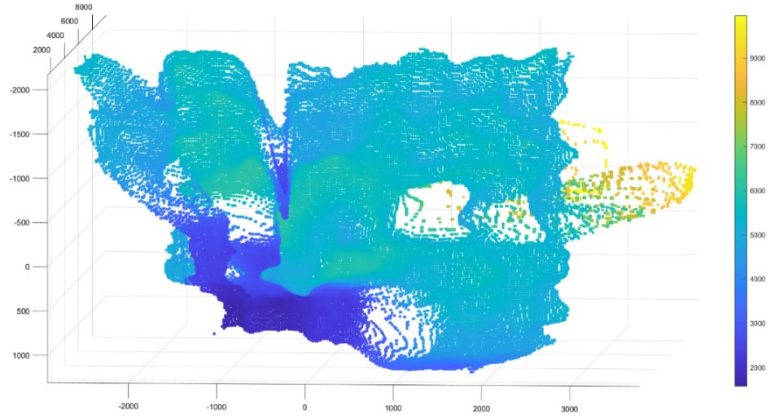
Neuronová síť *FCN* ve vyhodnocení dosáhla největšího počtu bodů. Při porovnání výsledků na Obrázek 166 je patrné, že odhad hloubky je velice přesný a není téměř patrný rozdíl. Tento fakt je patrný taktéž na Obrázek 170, kde zobrazen rozdílový snímek.



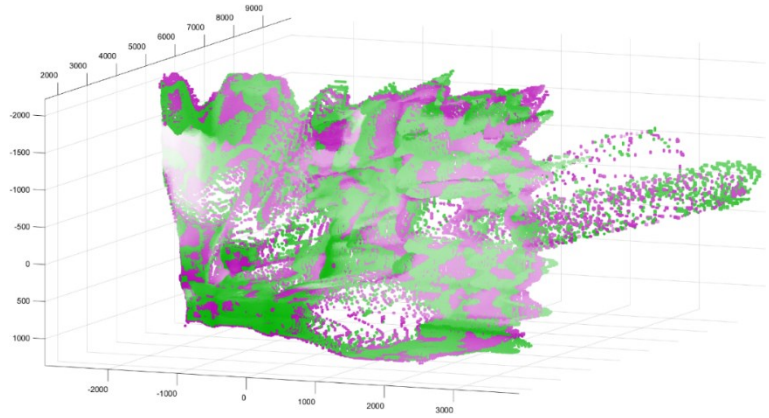
Obrázek 166: Vstupní snímky M7_2694 pro vytvoření mračna bodů. Vlevo: Snímek z barevné kamery. Uprostřed: Výstup z hloubkové kamery. Vpravo: Výstup ze sítě U-Net. (Zdroj: Vlastní)



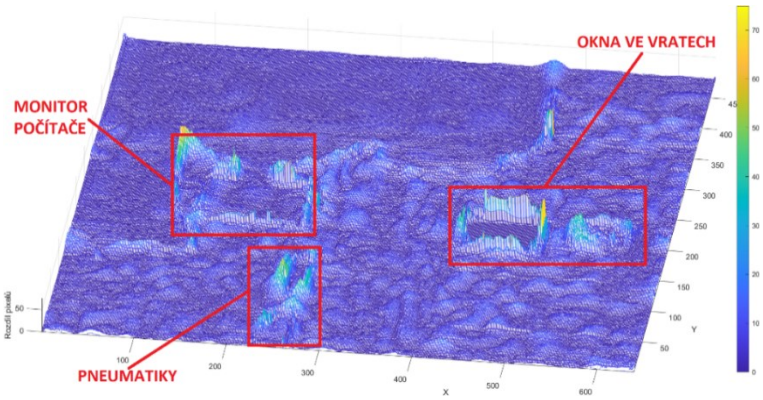
Obrázek 167: Vytvořené mračno bodů ze snímku M7_2694 z neuronové sítě FCN. (Zdroj: Vlastní)



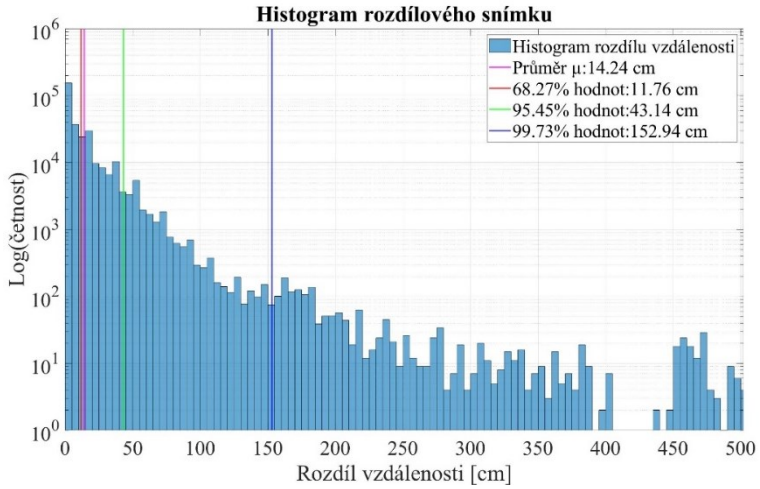
Obrázek 168: Vytvořené mračno bodů ze snímku M7_2694 z neuronové sítě FCN. Přebarveno dle vzdálenosti voxelu (Zdroj: Vlastní)



Obrázek 169: Srovnání mračna bodů vytvořené z hloubkové kamery (zelená barva) a z neuronové sítě FCN (fialová barva). (Zdroj: Vlastní)



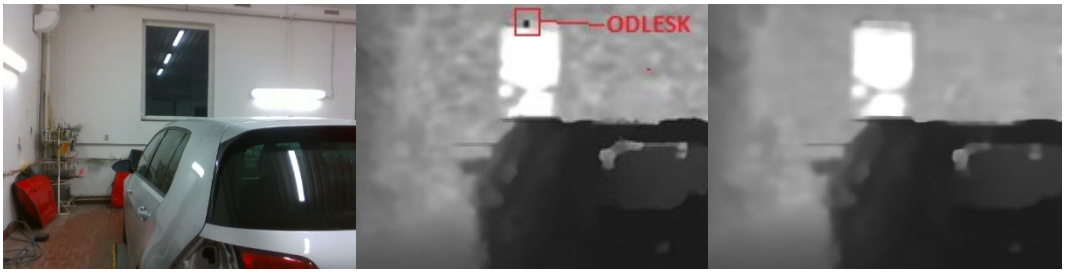
Obrázek 170: Rozdílový snímek M7_2694 mezi hloubkovou mapou získanou z hloubkové kamery a hloubkovou mapou získané z neuronové sítě FCN (Zdroj: Vlastní)



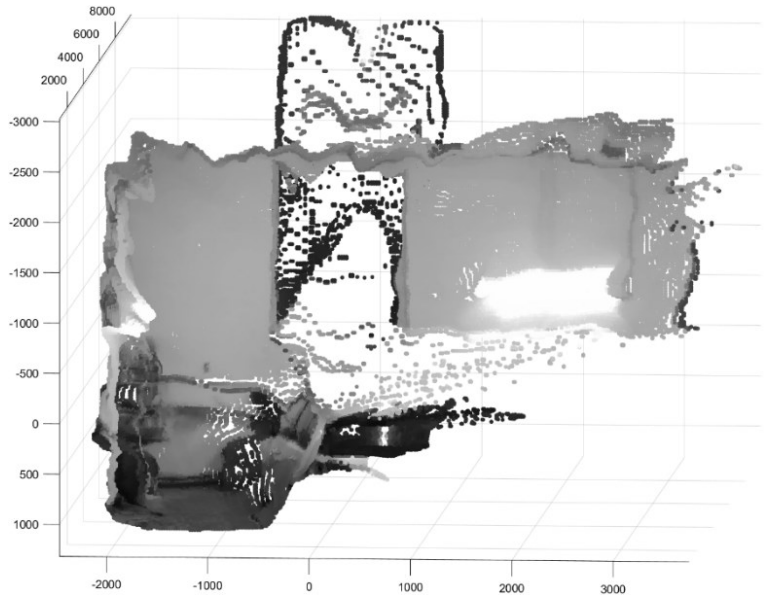
Obrázek 171: Histogram rozdílového snímku M7_2694 sítě FCN (Zdroj: Vlastní)

Na Obrázek 170 je zobrazen rozdílový snímek zobrazující odlišně odhadnuté vzdálenosti od neuronové sítě FCN vzhledem k hloubkové kameře. Z obrázku jsou patrné místa, kde dochází ke skokové změně vzdálenosti, a to zejména v okolí pneumatik či oken. Okna jsou specifická, protože hloubková kamera není schopna v tomto místě správně určit vzdálenost a zobrazuje většinou chybnou vzdálenost. Rozdílový snímek byl dále analyzován a výsledný histogram je zobrazen na Obrázek 171. Z histogramu je patrné, že přibližně 68% všech odhadnutých vzdáleností má menší chybu než 15,69 cm, což odpovídá chybě 1,5% na rozsahu 10 metrů a přibližně 95% odhadnutých vzdáleností má chybu menší než 43,17 cm, což odpovídá chybě odhadu menší než 4,3% na rozsahu 10 metrů.

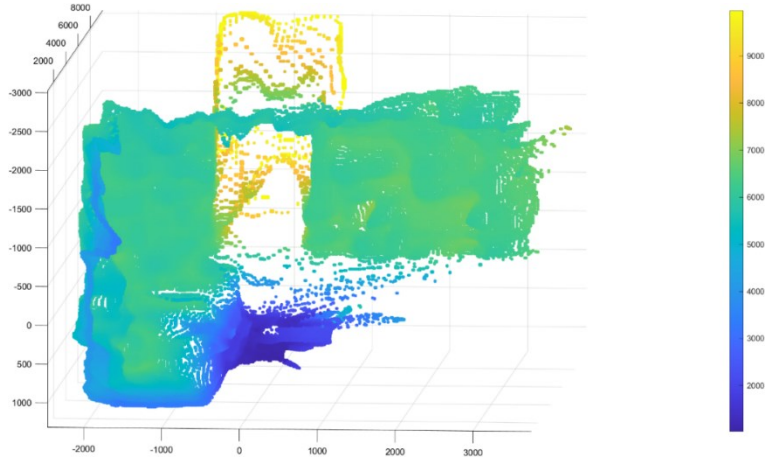
11.2.3.2 Analýza snímku M7_2396



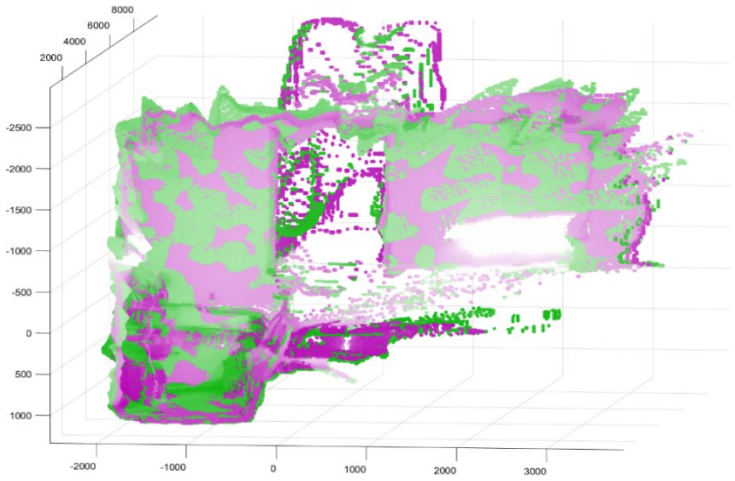
Obrázek 172: Vstupní snímky M7_2396 pro vytvoření mračna bodů. Vlevo: Snímek z barevné kamery. Uprostřed: Výstup z hloubkové kamery. Vpravo: Výstup ze sítě FCN. (Zdroj: Vlastní)



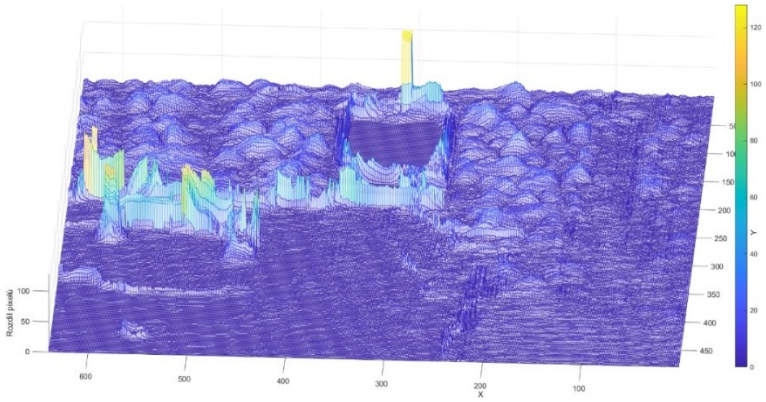
Obrázek 173: Vytvořené mračno bodů ze snímku M7_2396 z neuronové sítě FCN. (Zdroj: Vlastní)



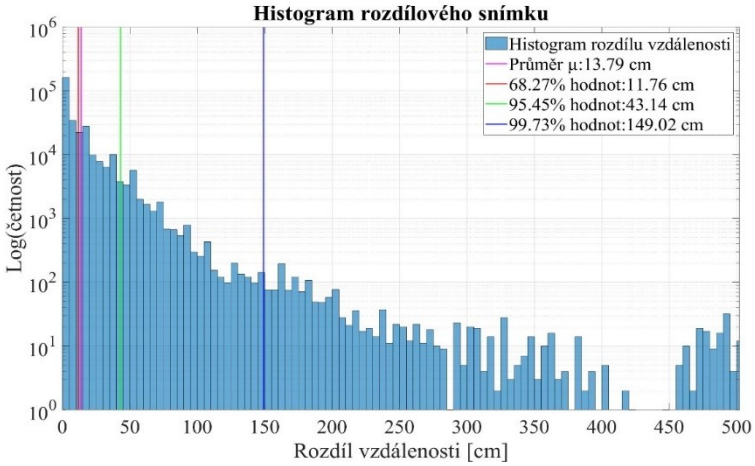
Obrázek 174: Vytvořené mračno bodů ze snímku M7_2396 z neuronové sítě FCN. Přebarveno dle vzdálenosti voxelu (Zdroj: Vlastní)



Obrázek 175: Srovnání mračna bodů vytvořené z hloubkové kamery (zelená barva) a z neuronové sítě FCN (fialová barva). (Zdroj: Vlastní)



Obrázek 176: Rozdílový snímek M7_2396 mezi hloubkovou mapou získanou z hloubkové kamery a hloubkovou mapou získané z neuronové sítě FCN (Zdroj: Vlastní)



Obrázek 177: Histogram rozdílového snímku M7_2396 sítě FCN (Zdroj: Vlastní)

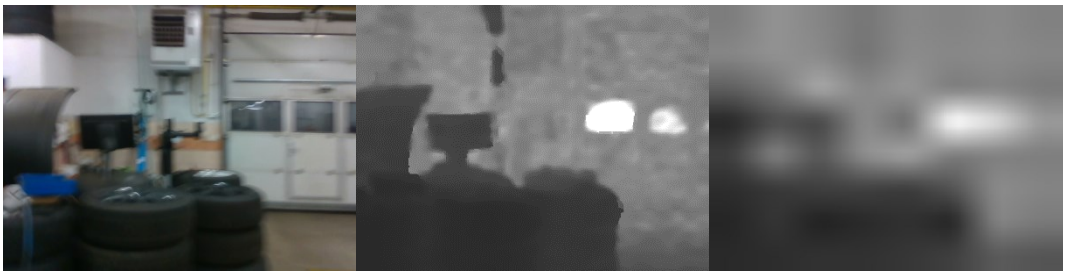
Na Obrázek 176 je zobrazen rozdílový snímek zobrazující odlišně odhadnuté vzdálenosti od neuronové sítě FCN vzhledem k hloubkové kameře. Z obrázku jsou patrné hrany objektů, zejména je patrné okno a střecha automobilu. Rozdílový snímek byl dále analyzován a výsledný histogram je zobrazen na Obrázek 177. Z histogramu je patrné, že přibližně 68% všech odhadnutých vzdáleností má menší chybu než 11,76 cm,

což odpovídá chybě odhadu menší než 1,1% na rozsahu 10 metrů a přibližně 95% odhadnutých vzdáleností má chybu menší než 43,14 cm, což odpovídá chybě odhadu vzdálenosti menší než 4,3% na rozsahu 10 metrů.

11.2.4 DenseNet: Výsledky registrace mračna bodů

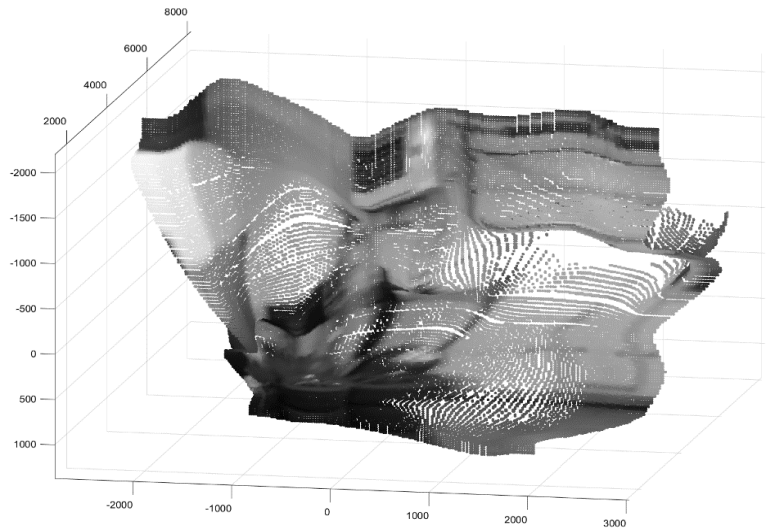
Výsledky neuronové sítě *DenseNet* byly přidány pro srovnání s ostatními neuronovými sítěmi. Neuronová síť *DenseNet* byla vyhodnocena jako nejhorší testovaná architektura. Síť je schopna odhadnout v hloubkové mapě hrubé obrysy objektů, ale přesnost odhadu je velice malá.

11.2.4.1 Analýza snímku M7_2694

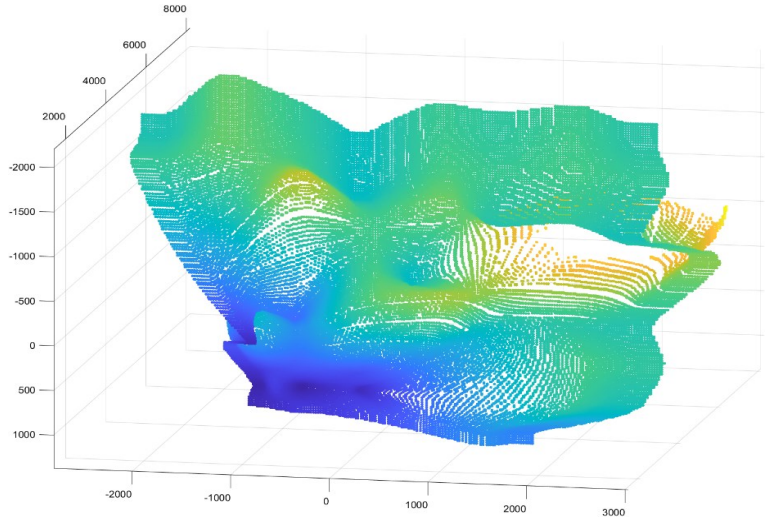


Obrázek 178: Vstupní snímky M7_2694 pro vytvoření mračna bodů. Vlevo: Snímek z barevné kamery. Uprostřed: Výstup z hloubkové kamery. Vpravo: Výstup ze sítě DenseNet. (Zdroj: Vlastní)

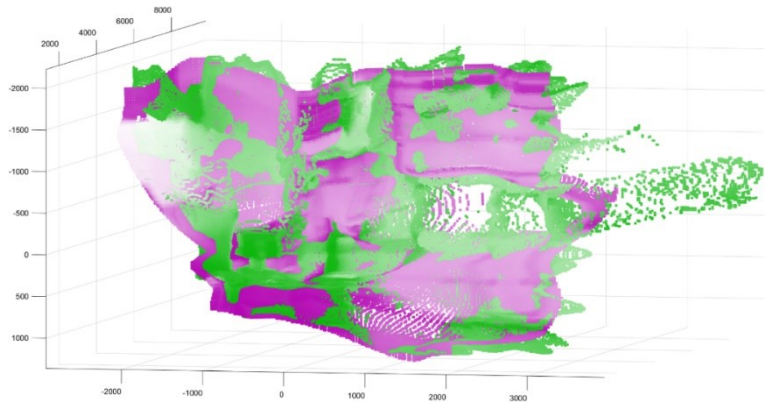
Z Obrázek 178 je patrné, že výsledný odhad hloubky je velice špatný. Jsou částečně vidět pouze okraje objektů, ovšem jakýkoliv detail není patrný. Při převodu takovéto hloubkového snímku bude výsledné mračno bodů deformované a jakékoliv další zpracování, například pomocí algoritmu *ICP* či *NDT* pravděpodobně selže či bude vytvořená mapa deformovaná. Ukázka takovéto deformace mračna bodů je zobrazena na Obrázek 179 a Obrázek 180.



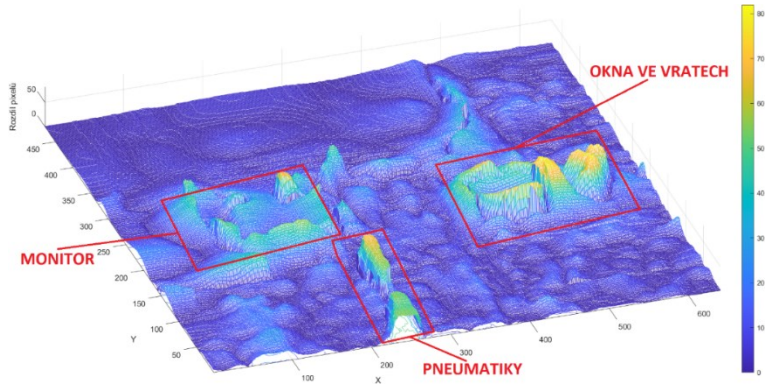
Obrázek 179: Vytvořené mračno bodů ze snímku M7_2694 z neuronové sítě DenseNet. (Zdroj: Vlastní)



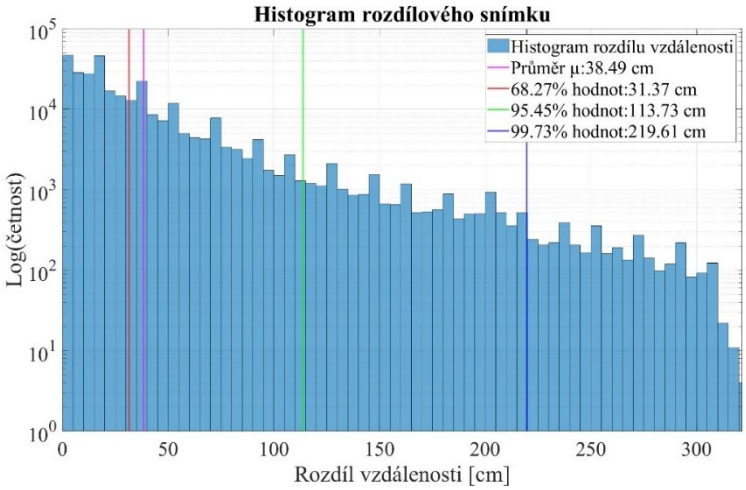
Obrázek 180: Vytvořené mračno bodů ze snímku M7_2694 z neuronové sítě DenseNet. Přebarveno dle vzdálenosti voxelu (Zdroj: Vlastní)



Obrázek 181: Srovnání mračna bodů vytvořené z hloubkové kamery (zelená barva) a z neuronové sítě DenseNet (fialová barva). (Zdroj: Vlastní)



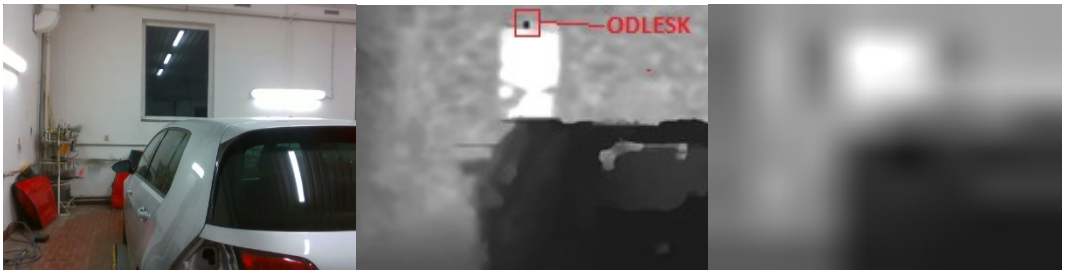
Obrázek 182: Rozdílový snímek M7_2694 mezi hloubkovou mapou získanou z hloubkové kamery a hloubkovou mapou získané z neuronové sítě DenseNet (Zdroj: Vlastní)



Obrázek 183: Histogram rozdílového snímku M7_2694 sítě DenseNet (Zdroj: Vlastní)

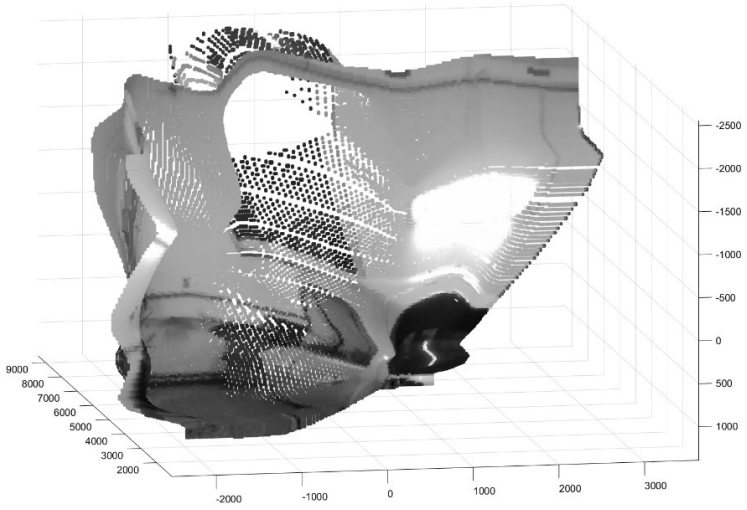
Z Obrázek 182 je patrné rozložení rozdílů mezi hloubkovým snímkem získaným z hloubkové kamery a hloubkovým snímkem vytvořeným neuronovou sítí DenseNet. Výsledky jsou velice špatné a je patrné, že neuronová síť měla problém pokrýt jakékoliv větší změny v obraze. Pro ilustraci rozložení chyb odhadu vzdálenosti je na Obrázek 183 zobrazen histogram. Z histogramu je patrné, že 68,27% bodů má chybu odhadu vzdálenosti menší než 31,37 cm, což odpovídá chybě 3,1% a pro přibližně 95% bodů je chyba odhadu vzdálenosti 113,73 cm, což odpovídá chybě 11,3% na rozsahu 10 metrů.

11.2.4.2 Analýza snímku M7_2396

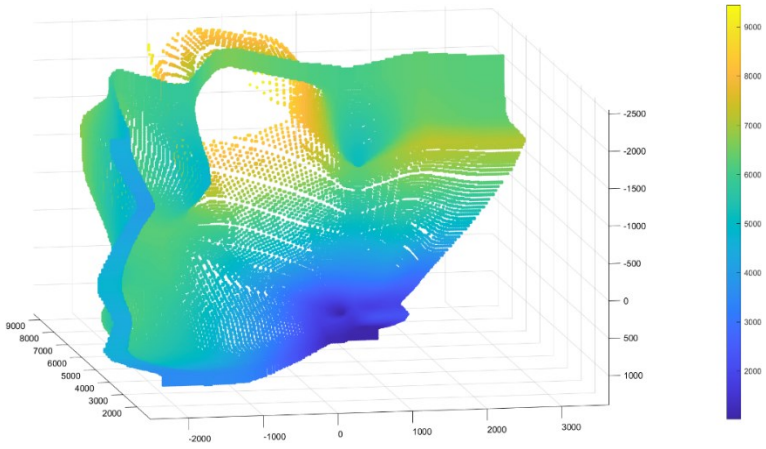


Obrázek 184: Vstupní snímky M7_2396 pro vytvoření mračna bodů. Vlevo: Snímek z barevné kamery. Uprostřed: Výstup z hloubkové kamery. Vpravo: Výstup ze sítě DenseNet. (Zdroj: Vlastní)

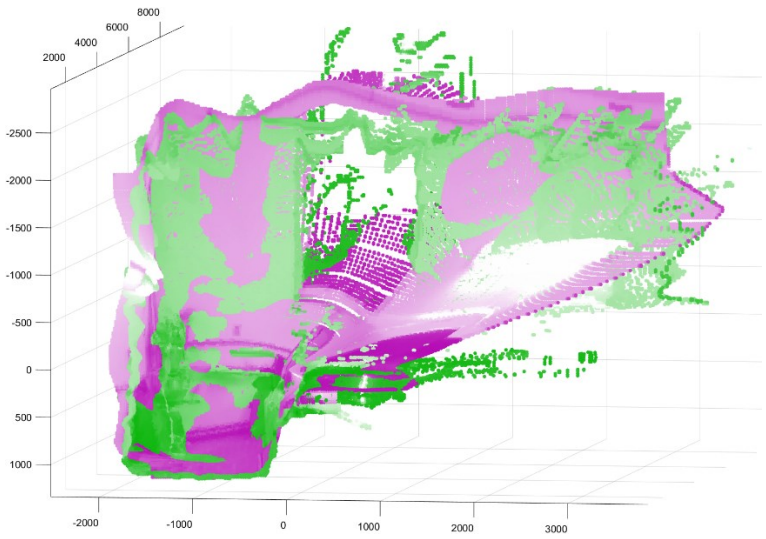
Na Obrázek 184, pravá část je zobrazen výstup z neuronové sítě *DenseNet.* Z výsledků je patrné, že odhad hloubky je velice špatný. Pokud se takto vytvořený hloubkový snímek použije při tvorbě mračna bodů, dojde k deformaci prostorového snímku, jak je zobrazeno na Obrázek 185 a Obrázek 186. Taktéž při porovnání mračna bodů vytvořeného z hloubkové kamery a mračna bodů vytvořeného za využití snímků z neuronové sítě *DenseNet* je patrný velký rozdíl, jak je zobrazeno na Obrázek 187.



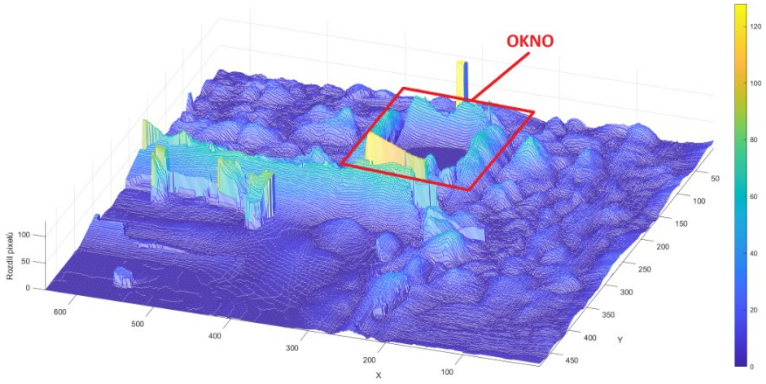
Obrázek 185: Vytvořené mračno bodů ze snímku M7_2396 z neuronové sítě DenseNet. (Zdroj: Vlastní)



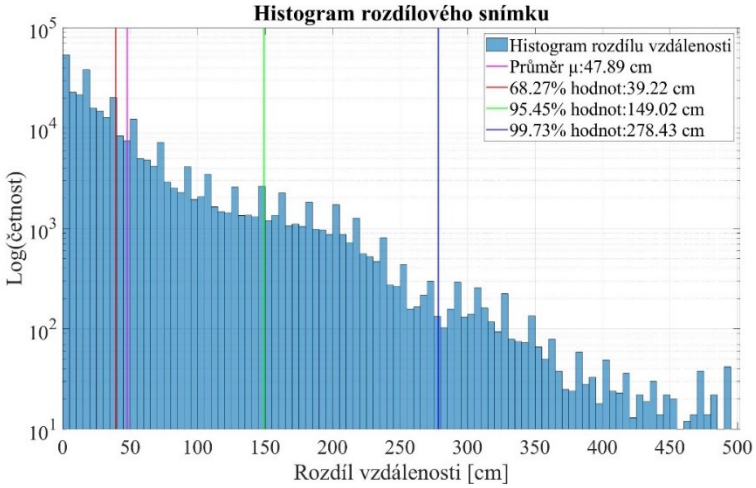
Obrázek 186: Vytvořené mračno bodů ze snímku M7_2396 z neuronové sítě DenseNet. Přebarveno dle vzdálenosti voxelu (Zdroj: Vlastní)



Obrázek 187: Srovnání mračna bodů snímku M7_2396 vytvořené z hloubkové kamery (zelená barva) a z neuronové sítě DenseNet (fialová barva). (Zdroj: Vlastní)



Obrázek 188: Rozdílový snímek M7_2396 mezi hloubkovou mapou získanou z hloubkové kamery a hloubkovou mapou získané z neuronové sítě DenseNet (Zdroj: Vlastní)



Obrázek 189: Histogram rozdílového snímku M7_2396 sítě DenseNet (Zdroj: Vlastní)

Z histogramu zobrazeného na Obrázek 189 je patrné, že neuronová síť *DenseNet* dosáhla chyby v odhadu vzdálenosti pro přibližně 68% všech odhadnutých bodů s chybou menší než 39,22 cm, což odpovídá chybě 3,9% a pro přibližně 95% všech odhadnutých bodů s chybou menší než 149,02 cm, což odpovídá chybě 14,9%. Tato chyba je již opravdu enormní ve srovnání s ostatními neuronovými sítěmi.

11.2.5 Vyhodnocení přesnosti odhadu hloubky

Pro vyhodnocení přesnosti odhadu neuronových sítí byly hloubkové snímky získané z vybraných neuronových sítí porovnány s hloubkovým snímkem z hloubkové kamery a výsledný rozdílový snímek byl statisticky vyhodnocen.

Statistické hodnoty jako průměr a směrodatná odchylka byly vyhodnoceny z 2500 snímků, shodných pro každou testovanou neuronovou síť. Výsledky vyhodnocení intervalů jsou uvedeny v Tabulka 22. Hranice 95,45% znamená, že dané procento hodnot má chybu odhadu vzdálenosti menší než hodnota uvedená v tabulce.

Tabulka 22: Srovnání výsledků neuronových sítí z testovací sady 2500 snímků

	DenseNet	FCN	U-Net	U-Net3
μ [cm]	41,41	13,35	15,40	14,20
Interval spolehlivosti 68,27% [cm]	34,62	12,71	14,37	13,38
σ pro 68,27%[cm]	12,85	3,20	3,40	3,18
Interval spolehlivosti 95,45% [cm]	121,93	38,51	44,28	38,97
σ pro 95,45%[cm]	43,62	9,37	11,36	10,21
Interval spolehlivosti 99,73% [cm]	259,72	114,11	121,97	106,84
σ pro 99,73% [cm]	75,99	40,39	40,62	36,92

Z tabulky vyplývá, že nejlépe hodnocená neuronová síť *FCN* má pro přibližně 95% všech hodnot chybu odhadu vzdálenosti menší než 38,51±9,37 cm, což odpovídá chybě odhadu na celém rozsahu 10 metrů přibližně 3,85±0.9%. Druhá nejlepší síť, autorem práce navržená síť *U-Net3*, má pro přibližně 95% všech hodnot chybu odhadu vzdálenosti menší než 38,97±10,21 cm, což odpovídá chybě odhadu na celém rozsahu 10 metrů přibližně 3,9±1%. Veškeré přesnosti odhadu vzdálenosti dle vybrané neuronové sítě jsou uvedeny v Tabulka 23.

Tabulka 23: Srovnání přesnosti neuronových sítí

	DenseNet	FCN	U-Net	U-Net3
Chyba pro interval 68,27%	≤3,46%	≤1,27%	≤1,44%	≤1,34%
Chyba pro interval 95,45%	≤12,19%	≤3,85%	≤4,43%	≤3,90%
Chyba pro interval 99,73%	≤25,97%	≤11,41%	≤12,20%	≤10,68%

Z výsledků v Tabulka 23. je patrné, že všechny testované sítě, až na síť *DenseNet*, která byla do testů přidána pro srovnání, jsou použitelné pro vytvoření kvalitního a přesného mračna bodů. Z výsledků je taktéž patrné, že testované neuronové sítě dosáhly při intervalu spolehlivosti 95,45% chyby menší než 4,4% při maximálním rozsahu 10 metrů.

12 Závěr

Práce se zabývá návrhem a testováním vhodných architektur neuronových sítí pro odhad hloubkové mapy na základě vstupního barevného snímku. Výstupem práce byl stanoven cíl vygenerovat mračno bodů na základě hloubkových snímků získaných z testovaných architektur. Toto mračno bodů musí být použitelné pro stávající algoritmy registrace mračen bodů jako je například *ICP* či *NDT.*

V práci byly provedeny celkem 3 sady měření, které byly provedeny každá jiný den s odstupem i několika měsíců. Během této doby došlo k rozsáhlejším změnám v konstelaci objektů v místnostech (jiné druhy automobilů, změna umístění heverů, stavební úprava místnosti). Celkem bylo naměřeno 24 různých tras uvnitř místností i ve vnějším prostředí. Tyto trasy obsahují různé intenzity osvětlení (jasno, umělé osvětlení, soumrak). Celkový objem dat vygenerovaných samotnou hloubkovou kamerou přesáhl 254GB s celkovou dobou záznamu přes 155 minut. Počet surových naměřených snímků z hloubkové kamery dosahoval hodnoty 140 000 párů (barevný a hloubkový snímek). Tento objem dat ovšem nezahrnuje data z V-SLAM kamery. Při započtení dat z V-SLAM kamery bylo zaznamenáno celkové množství dat přesahujících 1TB.

Celkem bylo provedeno přepracování a trénování 10 různých architektur sítí. U architektury neuronové sítě *ResNet* byla provedena analýza celkem 5 variant sítě, které se lišily jen ve složitosti. V rámci práce byla jedna architektura navržena kompletně vlastní na bázi sítě *U-Net*.

Autorem práce navržena architektura sítě byla nazvána pracovním názvem *U-Net3*, jelikož vychází k architektury *U-Net* a využívá podobných principů. Číslovka 3 v názvu sítě označuje trojnásobnou složitost ve srovnání se základní architekturou *U-Net.* Vývoj neuronové sítě byl zdlouhavý a velké množství variant bylo špatných a výsledkem byla neschopnost sítě *U-Net3* se cokoliv naučit. Několik variant se ukázalo slibných a ve výsledku byla vybrána nejlepší varianta, která byla posléze vyhodnocena a poskytuje velice kvalitní výsledky. Při srovnání výsledků s ostatními testovanými architekturami bylo zjištěno, že autorem navržená architektura neuronové sítě *U-Net3* byla vyhodnocena jako druhá nejlepší testovaná architektura.

Architektura sítě *U-Net3* je výrazně složitější než základní architektura *U-Net* což je znát na počtu trénovatelných parametrů, kde neuronová síť *U-Net3* má přibližně 88,27 milionu trénovatelných parametrů proti přibližně 31,03 miliónu trénovatelných

parametrů u architektury sítě *U-Net*. Rozdíl ve složitosti sítě, potažmo počtu trénovatelných parametrů je markantní, přibližně 2,84×. Rychlost učení i vyhodnocení jednoho snímku v případě neuronové sítě U-Net3 nezaznamenal však tak vysoký nárůst. Při srovnání rychlosti učení jednoho snímku byla doba nezbytná k učení jednoho snímku u sítě *U-Net* 61,1 ms a pro síť *U-Net3* byla doba nezbytná k učení jednoho snímku 70,2ms. Jedná se tedy o nárůst času nezbytného k učení jednoho snímku o přibližně 14%. V případě času nezbytného k vyhodnocení jednoho snímku sítí *U-Net* bylo změřeno 15,8 ms, oproti času nezbytnému k vyhodnocení neuronové sítě *U-Net3* 24,2 ms. Nárůst doby nezbytné k vyhodnocení byl tedy přibližně o 53%. Tento nárůst doby nezbytné k vyhodnocení jednoho snímku neznamená v dnešní době zásadní problém, jelikož doba 24,2 ms je postačující k aplikaci v reálném čase. Autorem navržená architektura má do budoucna rozhodně potenciál k dalšímu zlepšování.

Zbylých 9 architektur bylo taktéž nutné výrazně upravit. Základní úprava spočívala zejména v přizpůsobení architektury sítě, aby síť generovala stejný rozměr snímku jako je vstupní obrázek, což u všech sítí znamenalo přepracování výstupních vrstev. Základní bloky typické pro danou architekturu byly ovšem ponechány. Dále u všech neuronových sítí, bylo nezbytné změnit téměř všechny aktivační funkce z používaných aktivačních funkcí autorem dané architektury, veskrze se jednalo o změnu aktivační funkce z nejčastěji používané *sigmoidy či TanH* na aktivační funkci *ReLU.*

Zkoumané architektury byly rozmanité a poskytovaly velký rozptyl doby učení, doby vyhodnocení i počtu trénovatelných parametrů. Nejjednodušší neuronovou sítí byla síť *ResNet-12*, která obsahovala 20,84 milionu trénovatelných parametrů. Nejsložitější sítí z pohledu počtu trénovatelných parametrů byla síť *RefineNet*, obsahující 105,9 milionu trénovatelných parametrů. Z hlediska času učení byly u testovaných architektur neuronových sítí taktéž patrné výrazné rozdíly. Nejrychleji bylo možné provést učení jednoho snímku u nejjednodušší sítě *ResNet-12* za dobu 42,7 ms, nejpomaleji trvalo učení jednoho snímku neuronové síti *BiSeNet*, která potřebovala k 560,4 ms. Obdobné výsledky byly i u času nezbytného k vyhodnocení jednoho snímku. Nejlépe z hlediska času vyhodnocení dopadla neuronová síť *ResNet-12,* která časem vyhodnocení jednoho snímku strávila 8,9 ms. Naopak nejhůře dopadla již zmíněná síť *BiSeNet*, které vyhodnocení jednoho snímku trvalo 231,5 ms. Většině sítí trvalo ovšem vyhodnocení jednoho snímku méně než 100 ms, což je doba, která je bez problémů použitelná pro aplikace v reálném

čase za předpokladu využití stejného či výkonnějšího hardware, jak je uvedeno v Tabulka 9.

Celková doba samotného učení všech neuronových sítí přesáhla 71 dní čistého strojového času. Výpočet byl proveden z průměrné doby učení jednoho snímku. Tato doba ovšem nereflektuje skutečné časové náklady učení, jako čas nezbytný na inicializaci samotné neuronové sítě a ukládání dat na pevný disk. Dále během učení se velice často stalo, že došlo vlivem extrémní paměťové náročnosti k vyčerpání systémových prostředků a k havárii učení, které bylo posléze nezbytné opakovat. Celková doba samotného učení se započtením prodlevami a opakovaným spuštěním byla přibližně 150 dní. Tento čas znamenal jen samotné učení již odladěných neuronových sítí a závisí na výkonnosti použitého hardware.

V práci byly všechny architektury testovány a statistiky vyhodnoceny. Výsledkem bylo zajímavé zjištění, že většina testovaných sítí je dle definovaných požadavků vhodná pro použití v oblasti odhadu hloubkové mapy na základě RGB snímku. Výjimkou byla architektura *DenseNet*, která ve vyhodnocení byla velice slabá. Neuronová síť byla schopna se naučit základní obrysy objektů v barevném snímku, ovšem hloubková mapa byla velice nepřesná a při tvorbě mračna bodů docházelo k deformacím obrazu. Tyto deformace způsobují selhání registračních algoritmů.

Ostatní testované architektury byly schopny se naučit velice kvalitně a záleží pouze na požadavcích konkrétní aplikace, která architektura neuronové sítě bude vhodná. V práci byly tyto kriteriální požadavky stanoveny a z výsledků byly vybrány tři nejlepší neuronové sítě. Nejlépe dopadla podle kritéria vážené metriky neuronová síť *FCN* (85,5 bodů). Druhou nejlepší sítí byla autorem práce navržená architektura *U-Net3* (80,5 bodů). Třetí nejlepší sítí byla stanovena dle kritéria vážené metriky neuronová síť *U-Net* (73,75 bodů). Nejhůře dopadla již zmiňovaná neuronová síť *DenseNet* (21 bodů), o trochu lépe dopadla neuronová síť *ResNet-200* (22,5 bodů).

V práci bylo provedeno vyhodnocení základních statistických parametrů u všech zkoumaných sítí. Základní parametry byly stanoveny funkce *MSE* a *MAE* a jejich suma v rámci celého datasetu. Dále byla vyhodnocována maximální hodnota této funkce ve všech testovaných snímcích. Tyto parametry stanovily míru kvality učení konkrétní neuronové sítě.

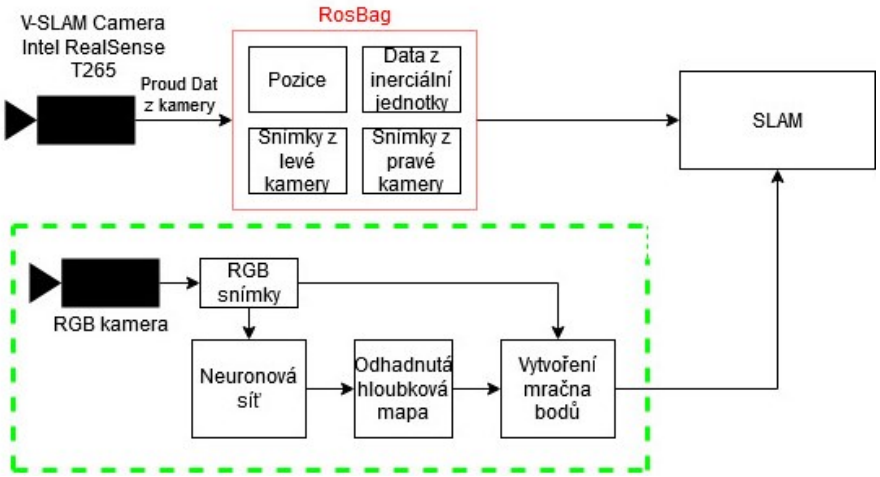
Pro lepší ukázku kvality učení neuronových sítí byly u vybraných snímků provedeny doplňkové statistické analýzy. Tyto analýzy byly provedeny u snímků, které obsahovaly určitou rovnou plochu s co nejmenším rozptylem, jako je například sloup heveru či zeď. Na této ploše byly vyhodnoceny statistické parametry μ, σ, minimum a maximum. Z výsledků je patrné, že většina neuronových sítí vyhodnotila vzdálenost velice přesně. Nejlepší neuronové sítě byly schopny dosáhnout odchylky průměrné hodnoty v řádu jednotek centimetrů od skutečné vzdálenosti změřené hloubkovou kamerou se směrodatnou odchylkou taktéž několik centimetrů. Pro porovnání byly tytéž statistické parametry vyhodnoceny u hloubkové kamery.

Mračno bodů, potřebné pro praktickou aplikaci zkoumaných neuronových sítí, bylo vytvořeno za využití knihoven *Open3D* (Ope21) v jazyce Python. Knihovna *Open3D* byla zvolena z praktických důvodů, jelikož obsahuje velké množství metod pro práci s mračny bodů. Vytvořená mračna bodů byla posléze detailně analyzována a vyhodnocena. Pro přehled, jaká místa jsou pro neuronové sítě kritická k vyhodnocení a v jakých místech dochází k největší diferenci mezi skutečným snímkem a odhadnutou vzdáleností neuronovou sítí, byl vytvořen rozdílový snímek, který byl převeden do prostorové podoby a histogramu. Z tohoto snímku byly taktéž stanoveny základní statistické parametry μ, σ, a intervaly spolehlivosti 68,27%, 95,45% a 99,73%, které byly vypočítány z celého rozdílového snímku.

Pro maximálně věrohodné stanovení procentuální chyby odhadu konkrétní neuronové sítě bylo použito 2500 testovacích snímků, které byly použity jen pro vyhodnocení neuronovými sítěmi a tyto snímky nebyly použity při učení. Pro každý z těchto snímků byl vytvořen rozdílový snímek. Z této sady rozdílových snímků bylo následně provedeno vyhodnocení přednosti odhadu hloubkového snímku za využití tří intervalů spolehlivosti. U architektur sítí, které byly vyhodnoceny jako nejlepší, byly posléze provedeny detailní analýzy rozdílových hloubkových snímků. Výsledkem je zjištění, že neuronová síť *FCN* dosáhla pro interval spolehlivosti 95,45% maximální chyby odhadu hloubkového snímku 38,51±9,37 cm, což na rozsahu 10 metrů odpovídá chybě 3,85% pro interval spolehlivosti 95,45%. Autorem navržená architektura neuronové sítě *U-Net3*, dosáhla velice podobných výsledků. Pro interval spolehlivosti 95,45% je maximální chyba 38,97±10,21 cm, což odpovídá chybě na rozsahu 10 metrů přibližně 3,9%. Z rozdílových snímků je ovšem patrné, že chyba je na rovných plochách výrazně menší, ovšem oblasti, kde dochází ke skokové změně vzdálenosti, jsou pro všechny

testované neuronové sítě problematické a z podstaty jejich chování jako filtr typu dolní propust tyto skokové změny vzdálenosti nejsou schopny spolehlivě pokrýt. Tyto skokové změny způsobí výrazné špičky v rozdílovém snímku, což samozřejmě výrazně zhoršuje přesnost odhadu hloubkového snímku.

Vybrané neuronové sítě, zejména pak *FCN* a *U-Net3* jsou použitelné pro další zpracování při využití mračna bodů pro simultánní lokalizaci a mapování (SLAM). Za předpokladu požadavku vyšší rychlosti učení i zpracování je možné použít i neuronové sítě ze skupiny architektury *ResNet*, například variantu *ResNet-18* či *ResNet-50*, které nedosáhly tak kvalitních výsledků jako detailně testované sítě *FCN, U-Net3* či *U-Net*, ovšem pro aplikace s malým důrazem na přesnost odhadu hloubkového snímku jsou použitelné. Naopak absolutně nepoužitelnou sítí se ukázala architektura *DenseNet,* která sice byla schopna odhadnout hloubkovou mapu, ve které jsou patrné hrubé obrysy objektů, ovšem přesnost je velice špatná.



Obrázek 190: Ukázka návrhu systému pro autonomní navigaci v neznámém prostoru

Na Obrázek 190 je zobrazen blokový návrh systému, využívající neuronové sítě pro odhad hloubkové mapy, které byly popsány v této práci. Systém se dělí na dvě části. První část systému je založena na V-SLAM kameře T265, která byla použita pro sběr dat v této práci. Kamera poskytuje celkem přesné navigační údaje, které jsou získávány pomocí dvou kamer s čočkou typu rybí oko a integrované inerciální jednotky. Druhá část systému je založena na odhadu hloubkové mapy, respektive mračna bodů za využití naučené neuronové sítě. Výsledné mračno bodů je pomocí metod registrace mračna bodů (např. *ICP* či *NDT*) registrováno, je proveden odhad pohybu robotické platformy. Výsledná poloha může být za využití pokročilých algoritmů korigována na základě odhadnuté polohy pomocí V-SLAM kamery.

13 Seznam odborných publikací autora

13.1 Publikace v odborných časopisech

1. REJFEK Luboš, Tan NGUYEN, Pavel CHMELAŘ, **Ladislav BERAN**, T. Tran PHUONG. Neural networks application for processing of the data from the FMICW radars. Symmetry, 2019, 11(10), 1308
2. **BERAN, Ladislav**, Luboš REJFEK, Pavel CHMELAŘ a David MATOUŠEK. Position estimation of robotic platform using optical flow. Journal of Fundamental and Applied Sciences. 2018, 10(3S 2018), 507-520. ISSN 1112-9867.
3. CHMELAŘ, Pavel, Luboš REJFEK, **Ladislav BERAN**, Natalia CHMELAŘOVÁ a Martin DOBROVOLNY. Point Cloud Plane Visualization by Using Level Image. Journal of Fundamental and Applied Sciences. 2018, 10(3S 2018), 547-560. ISSN 1112-9867.
4. REJFEK, Luboš, Natalia CHMELAŘOVÁ, Martin DOBROVOLNY, **Pavel CHMELAŘ** a Ladislav BERAN. Filtration of the FMICW radar output signals by the advanced windows. Journal of Fundamental and Applied Sciences. 2018, 10(3S 2018), 521-535. ISSN 1112-9867.
5. HORÁČEK, Michal, **Ladislav BERAN** a Luboš REJFEK. Desk games digitalization using Radon transformation and Color segmentation. Journal of Fundamental and Applied Sciences. 2018, 10(3S 2018), 481-493. ISSN 1112-9867.
6. VANÍČEK Petr, **Ladislav BERAN** Navigation of robotics platform in unknown spaces using LIDAR, Raspberry PI and Hector SLAM. Journal of Fundamental and Applied Sciences. 2018, 10(3S 2018), 494-506. ISSN 1112-9867.
7. REJFEK Luboš, Zbyšek MOŠNA, **Ladislav BERAN**, Ondrej FIŠER a Martin DOBROVOLNÝ. Automatic detection of the unknown number point targets in FMICW radar signals. International Journal of ADVANCED AND APPLIED SCIENCES [online]. 2017, 4(11), 116-120 [cit. 2019-06-27]. DOI: 10.21833/ijaas.2017.011.018. ISSN 2313626X. Dostupné z: http://www.science-gate.com/IJAAS/V4I11/Rejfek.html
8. **BERAN, Ladislav**, Pavel CHMELAŘ, Luboš REJFEK. An Increase in Estimation Accuracy Position Determination of Inertial Measurement Units. *MATEC Web of Conferences* [online]. 2016, **75** [cit. 2019-06-27]. DOI:

10.1051/matecconf/20167505001. ISSN 2261-236X. Dostupné z: http://www.matec-conferences.org/10.1051/matecconf/20167505001

9. **BERAN, Ladislav**, Pavel CHMELAŘ, Luboš REJFEK. Image Processing Methods Usable for Object Detection on the Chessboard. *MATEC Web of Conferences* [online]. 2016, **75** [cit. 2019-06-27]. DOI: 10.1051/matecconf/20167503004. ISSN 2261-236X. Dostupné z: http://www.matec-conferences.org/10.1051/matecconf/20167503004
10. CHMELAŘ, Pavel, **Ladislav BERAN**, Luboš REJFEK. The Depth Map Construction from a 3D Point Cloud. *MATEC Web of Conferences* [online]. 2016, **75** [cit. 2019-06-27]. DOI: 10.1051/matecconf/20167503005. ISSN 2261-236X. Dostupné z: http://www.matec-conferences.org/10.1051/matecconf/20167503005
11. REJFEK, Luboš, **Ladislav BERAN**, Pavel CHMELAŘ, Natalia CHMELAŘOVÁ, Viktor PEK a Ondřej FIŠER. Analysis of radar signals by PSD methods. International Journal of Advanced and Applied Science. 2016, 2(12), 62-66. ISSN 2313-3724.
12. CHMELAŘ, Pavel, Luboš REJFEK, **Ladislav BERAN** a Martin DOBROVOLNY. A Point Cloud Decomposition by the 3D Level Scanning for Planes Detection. International Journal of Advanced and Applied Science. 2016, 2(12), -. ISSN 2313-626X.
13. REJFEK, Luboš, Zbyšek MOŠNA, **Ladislav BERAN**, Pavel CHMELAŘ, Natalija CHMELAŘOVÁ a Pavel ROZSÍVAL. Comparison of Digisonde and CDSS measurement for the monitoring of the existence of the Ionospheric communication channel. *International Journal of Advanced and Applied Science*. 2016, **2**(12), 67-71. ISSN 2313-626X.

13.2 Konferenční publikace

1. DOLEŽEL Petr, Dominik ŠTURSA, Daniel HONC, Jan MERTA, Veronika ROZSIVALOVÁ, **Ladislav BERAN** a Ivo HORA. Counting Livestock with Image Segmentation Neural Network. In: 2020 Advances in Intelligent Systems and Computing – 15th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2020), 2020, pp. 237-244, doi: 10.1007/978-3-030-57802-2_23.

2. PITAŠ Karel, Luboš REJFEK, Tan N. NGUYEN, **Ladislav BERAN,** Phuong TRAN a Ondřej FIŠER. FMICW Radar Target Classification By Neural Network. In: 2020 30th International Conference Radioelektronika (RADIOELEKTRONIKA), 2020, pp. 1-5, doi: 10.1109/RADIOELEKTRONIKA49387.2020.9092342. Dostupné z: https://ieeexplore.ieee.org/document/9092342

3. JAROŠ Ondrej a **Ladislav BERAN**. Design of an Electronically Steered Antenna Array in the X band. In: 2019 Conference on Microwave Techniques (COMITE) [online]. IEEE, 2019, 2019, s. 1-6 [cit. 2019-06-27]. DOI: 10.1109/COMITE.2019.8733593. ISBN 978-1-5386-9337-7. Dostupné z: https://ieeexplore.ieee.org/document/8733593/

4. CHMELAŘ, Pavel, **Ladislav BERAN**, Natalija CHMELAŘOVA a Lubos REJFEK. Advanced plane properties by using level image. In: 2018 28th International Conference Radioelektronika (RADIOELEKTRONIKA) [online]. IEEE, 2018, 2018, s. 1-6 [cit. 2019-06-27]. DOI: 10.1109/RADIOELEK.2018.8376365. ISBN 978-1-5386-2485-2. Dostupné z: https://ieeexplore.ieee.org/document/8376365/

5. CHMELAŘ, Pavel, **Ladislav BERAN**, Lubos REJFEK a Natalija CHMELAŘOVA. The point cloud visualisation for rotary optical rangefinders. In: *2017 27th International Conference Radioelektronika (RADIOELEKTRONIKA)* [online]. IEEE, 2017, 2017, s. 1-6 [cit. 2019-06-27]. DOI: 10.1109/RADIOELEK.2017.7937589. ISBN 978-1-5090-4591-4. Dostupné z: http://ieeexplore.ieee.org/document/7937589/

6. REJFEK, Lubos, Ondrej FISER, David MATOUSEK, **Ladislav BERAN** a Pavel CHMELAŘ. Correction of received power for Doppler measurements by FMICW radars. In: 2017 International Symposium ELMAR [online]. IEEE, 2017, 2017, s.

104-110 [cit. 2019-06-27]. DOI: 10.23919/ELMAR.2017.8124446. ISBN 978-953-184-225-9. Dostupné z: http://ieeexplore.ieee.org/document/8124446/

7. REJFEK, Luboš, Ondrej FISER, David MATOUSEK, **Ladislav BERAN** a Pavel CHMELAŘ. Sensitivity analysis of PCDR35 radar. In: 2017 International Symposium ELMAR [online]. IEEE, 2017, 2017, s. 111-114 [cit. 2019-06-27]. DOI: 10.23919/ELMAR.2017.8124447. ISBN 978-953-184-225-9. Dostupné z: http://ieeexplore.ieee.org/document/8124447/
8. MATOUSEK, David a **Ladislav BERAN**. Comparison of positive and negative Dickson charge pump and Fibonacci charge pump. In: 2017 International Conference on Applied Electronics (AE) [online]. IEEE, 2017, 2017, s. 1-4 [cit. 2019-06-27]. DOI: 10.23919/AE.2017.8053595. ISBN 978-8-0261-0642-5. Dostupné z: http://ieeexplore.ieee.org/document/8053595/
9. **BERAN, Ladislav**, Pavel CHMELAŘ a Luboš REJFEK. Navigation of Robotics Platform Using Advanced Image Processing Navigation Methods. Proceedings of the 5th Eccomas Thematic Conference on Computational Vision and Medical Image Processing. 1. Philadelphia: Taylor & Francis, 2016, s. 341-346. ISBN 978-1-138-02926-2.
10. REJFEK, Lubos, **Ladislav BERAN**, Pavel CHMELAŘ, Ondrej FISER, Tomas ZALABSKY a Michal REZNICEK. Checking of automatically scaled ionograms by infinite Impulse Response filters. In: *2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA)* [online]. IEEE, 2016, 2016, s. 273-276 [cit. 2019-06-27]. DOI: 10.1109/RADIOELEK.2016.7477346. ISBN 978-1-5090-1674-7. Dostupné z: http://ieeexplore.ieee.org/document/7477346/
11. CHMELAŘOVA, Natalija, Pavel CHMELAŘ, **Ladislav BERAN** a Luboš REJFEK. Improving precision of laser line detection in 3D range scanning systems. In: *2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA)* [online]. IEEE, 2016, 2016, s. 207-212 [cit. 2019-06-27]. DOI: 10.1109/RADIOELEK.2016.7477409. ISBN 978-1-5090-1674-7. Dostupné z: http://ieeexplore.ieee.org/document/7477409/
12. REJFEK, Luboš, Jan PIDANIC, **Ladislav BERAN**, Pavel CHMELAŘ, Tomas ZALABSKY a Michal REZNICEK. Checking of wrongly scaled ionograms by using of virtual height of ionospheric layers. In: *2016 International Symposium ELMAR*

[online]. IEEE, 2016, 2016, s. 81-84 [cit. 2019-06-27]. DOI: 10.1109/ELMAR.2016.7731759. ISBN 978-953-184-221-1. Dostupné z: http://ieeexplore.ieee.org/document/7731759/

13. REJFEK, Luboš, Zbyšek MOSNA, Jaroslav URBAR, Daniel KOUBA, **Ladislav BERAN**, Pavel CHMELAŘ, Tomas ZALABSKY a Michal REZNICEK. Comparison of Digital Filters and GNSS for checking of automatically scaled ionograms. 2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA). IEEE, 2016, 2016, 2(12), 277-281. DOI: 10.1109/RADIOELEK.2016.7477341. ISBN 978-1-5090-1674-7. ISSN 2313-3724. Dostupné z: http://ieeexplore.ieee.org/document/7477341/

14. **BERAN, Ladislav**, Pavel CHMELAŘ a Luboš REJFEK. Navigation of robotics platform using monocular visual odometry. In: *2015 25th International Conference Radioelektronika (RADIOELEKTRONIKA)* [online]. IEEE, 2015, 2015, s. 213-216 [cit. 2019-06-27]. DOI: 10.1109/RADIOELEK.2015.7129012. ISBN 978-1-4799-8117-5. Dostupné z: http://ieeexplore.ieee.org/document/7129012/

15. **BERAN, Ladislav**, Pavel CHMELAŘ, Luboš REJFEK, Jaroslav CHUM a Zbyšek MOŠNA. Comparison of devices for monitoring of the ionosphere at the observatory Pruhonice. In: *2015 Conference on Microwave Techniques (COMITE)* [online]. IEEE, 2015, 2015, s. 1-4 [cit. 2019-06-27]. DOI: 10.1109/COMITE.2015.7120327. ISBN 978-1-4799-8121-2. Dostupné z: http://ieeexplore.ieee.org/document/7120327/

16. REJFEK, Luboš, **Ladislav BERAN** a Ondrej FISER. Correction of radar received signal. In: *2015 25th International Conference Radioelektronika (RADIOELEKTRONIKA)* [online]. IEEE, 2015, 2015, s. 191-194 [cit. 2019-06-27]. DOI: 10.1109/RADIOELEK.2015.7129006. ISBN 978-1-4799-8117-5. Dostupné z: http://ieeexplore.ieee.org/document/7129006/

17. CHMELAŘ, Pavel, **Ladislav BERAN** a Nataliia KUDRIAVTSEVA. The laser color detection for 3D range scanning using Gaussian mixture model. In: *2015 25th International Conference Radioelektronika (RADIOELEKTRONIKA)* [online]. IEEE, 2015, 2015, s. 248-253 [cit. 2019-06-27]. DOI: 10.1109/RADIOELEK.2015.7129023. ISBN 978-1-4799-8117-5. Dostupné z: http://ieeexplore.ieee.org/document/7129023/

18. CHMELAŘ, Pavel, **Ladislav BERAN**, Lubos REJFEK a Nataliia KUDRIAVTSEVA. Effective lens distortion correction for 3D range scannig systems. In: *2015 57th International Symposium ELMAR (ELMAR)* [online]. IEEE, 2015, 2015, s. 37-40 [cit. 2019-06-27]. DOI: 10.1109/ELMAR.2015.7334490. ISBN 978-953-184-209-9. Dostupné z: http://ieeexplore.ieee.org/document/7334490/

19. **BERAN, Ladislav**, Pavel CHMELAŘ a Martin DOBROVOLNY. Navigation of robotic platform with using inertial measurement unit and Direct Cosine Matrix. In: Proceedings ELMAR-2014 [online]. IEEE, 2014, 2014, s. 1-4 [cit. 2019-06-27]. DOI: 10.1109/ELMAR.2014.6923322. ISBN 978-953-184-199-3. Dostupné z: http://ieeexplore.ieee.org/document/6923322/

20. CHMELAŘ, Pavel, **Ladislav BERAN** a Nataliia KUDRIAVTSEVA. Projection of point cloud for basic object detection. In: *Proceedings ELMAR-2014* [online]. IEEE, 2014, 2014, s. 1-4 [cit. 2019-06-27]. DOI: 10.1109/ELMAR.2014.6923303. ISBN 978-953-184-199-3. Dostupné z: http://ieeexplore.ieee.org/document/6923303/

14 Bibliografie

Zhao Hengshuang [a další] Pyramid Scene Parsing Network [Konference] // CVPR 2017 - Conference on Computer Vision and Pattern Recognition 2017. - Honolulu, Hawai : [autor neznámý], 2017.

Artificial Intelligence & Autopilot [Online]. - Tesla. - 21. 09 2021. - https://www.tesla.com/AI.

Ascar Davix X., Judson D. a Pittala Suresh Kumar SegNet Approach for Vehicle License Plate Localization [Konference] // 2020 Seventh International Conference on Information Technology Trends (ITT). - Abu Dhabi, United Arab Emirates : IEEE, 2020.

Badrinarayanan Vijay, Kendall Alex a Cipolla Roberto SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation [Článek] // IEEE Transactions on Pattern Analysis and Machine Intelligence. - 2017. - 39. - 10.1109/TPAMI.2016.2644615 : Sv. 12.

Bellman Richard Dynamic programming and stochastic control processes [Článek] // Information and Control. - [místo neznámé] : Information and Control, 09 1958. - 3 : Sv. 1.

Bengio Yoshua, Goodfellow Ian J. a Courville Aaron Deep Learning [Kniha]. - [místo neznámé] : MIT Press, 2016. - ISBN: 9780262035613.

Beran Ladislav, Chmelař Pavel a Dobrovolný Martin Navigation of robotic platform with using inertial measurement unit and Direct Cosine Matrix [Konference] // Proceedings ELMAR-2014. - Zadar : ELMAR, 2014. - stránky 1-4. - doi: 10.1109/ELMAR.2014.6923322.

Besl P. J. a McKay Neil D. A method for registration of 3-D shapes [Článek] // IEEE Transactions on Pattern Analysis and Machine Intelligence. - [místo neznámé] : iEEE, 1992. - 2 : Sv. 14. - stránky 239-256.

Bezděk Aleš Souřadnicové systémy [Online]. - Astronomický ústav AV ČR, 2019. - 06. 06 2019. - http://www.asu.cas.cz/~bezdek/prednasky/tg4/jednotlive_prezentace/2_Souradnicove_soustavy_(TG4).pdf.

Biber Peter a Straßer Wolfgang The normal distributions transform: A new aproach to laser scan matching [Konference] // Proceedinsg of the IEEE International conference on inteligent robots and system (IROS). - Las Vegas, USA : IEEE, 2003.

Bistron Marta a Piotrowski Zbigniew Artificial Intelligence Applications in Military Systems and Their Influence on Sense of Security of Citizens [Článek] // Electronics / editor Rudas Imre J. a Liu Jun. - [místo neznámé] : MDPI, 2021. - 7 : Sv. 10.

Boughorbel Faysal [a další] Gaussian Felds: a new criterion for 3D rigid registration [Článek] // Pattern Recognition. - 10. 02 2004. - 7 : Sv. 37. - stránky 1567-1571.

Bryson Arthur E. A gradient method for optimizing multi-stage allocation processes [Článek] // Proc. Harvard Univ. Symposium on digital computers and their applications. - [místo neznámé] : Hardvard University, 1961. - 72.

Budhiman Arief, Suyanto Suyanto a Arifianto Anditya Melanoma Cancer Classification Using ResNet with Data Augmentation [Konference] // 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI). - Yogyakarta, Indonesia : IEEE, 2019.

Burguera Antoni, Gonzales Yolanda a Oliver Gabriel A Probabilistic Framework for Sonar Scan Matching Localization [Článek] // Advanced Robotics. - [místo neznámé] : Advanced Robotics, 2012. - 11 : Sv. 22. - stránky 1223-1241.

Burguera Antoni, Gonzales Yolanda a Oliver Gabriel Probabilistic Sonar Scan Matching for Robust Localization [Konference] // Proceedings 2007 IEEE International Conference on Robotics and Automation. - Rome, Italy : IEEE, 2007.

Burguera Antoni, Gonzales Yolanda a Oliver Gabriel The likelihood field approach to sonar scan matching [Konference] // 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. - Nice, France : IEEE, 2008.

Bushaev Vitali ADAM - latest trends in deep learning optimization [Online] // Towards Data Science. - 2021. - https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c.

Cauchy Augustin Luis Méthode générale pour la réolution des systémes d'équations simultanées [Článek] // Comptes Rendus Hebd. Séances Acad. Sci. - 18. 10 1847. - 1. - 25 : Sv. 10. - stránky 536-538.

Censi Andrea An ICP variant using a point-to-line metric [Konference] // 2008 IEEE International Conference on Robotics and Automation. - Pasadena, CA, USA : IEEE, 2008.

CleanTechnica.com [Online] // Tesla's New HW3 Self-Driving Computer — It's A Beast (CleanTechnica Deep Dive). - CleanTechnica.com. - 20. 09 2021.

Curry Haskell B. The method of steepest descent for non-linear minimization process [Článek] // Quarterly of applied mathematics. - [místo neznámé] : Brown University, 10 1944. - 3 : Sv. 2. - stránky 258-261.

Datacenterdynamics.com [Online] // Tesla details pre-Dojo supercomputer, could be up to 80 petaflops. - Datacenterdynamics.com. - 20. 09 2021. - https://www.datacenterdynamics.com/en/news/tesla-detail-pre-dojo-supercomputer-could-be-up-to-80-petaflops/.

Delaunay Boris Sur la sphère vide. A la mémoire de Georges Voronoï [Konference] // Bulletin de l'Académie des Sciences de l'URSS: Classe des sciences mathématiques et na. - 1934. - Sv. 6.

Deng Zhuo [a další] Classification of Breast Cancer Based on Improved PSPNet [Konference] // 2021 IEEE/ACIS 6th International Conference on Big Data, Cloud Computing, and Data Science (BCD). - Zhuhai, China : IEEE, 2021.

Doležel Petr Modelování a řízení nelineární soustavy s využitím umělých neuronových sítí = Disertační práce. - Pardubice : Univerzita Pardubice, 2011.

Doughferty Geoff Digital Image Processing for Medical Applications [Kniha]. - [místo neznámé] : California State University, Channel Island, 2009. - ISBN: 9780521860857.

Duchi John, Hazan Elad a Singer Yoram Adaptive Subgradient Methods for Online Learning and Stochastic Optimization [Článek] // The Journal of Machine Learning Research. - 7. 11 2011. - 12. - stránky 2121-2159 .

Erwandi Reynold a Suyanto Suyanto Improved Residual Neural Network for Breast Cancer Classification [Konference] // 2020 3rd International Conference on Information and Communications Technology (ICOIACT). - Yogyakarta, Indonesia : IEEE, 2020.

Fabien Mael XCeption Model and Depthwise Separable Convolutions [Online] // Deep Neural Networks. - 2021. - 13. 10 2021. - https://maelfabien.github.io/deeplearning/xception/#.

Fast Point Feature Histograms (FPFH) descriptors [Online] // Point Cloud Library 0.0 documentation. - PointCloudLibrary. - 01. 12 2021. - https://pcl.readthedocs.io/projects/tutorials/en/latest/fpfh_estimation.html#fpfh-estimation.

Fischler Martin A. a Bolles Robert C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography [Článek] // Communications of the ACM. - 1981. - 24 : Sv. 6. - stránky 381-395.

Fossaceca John M. a Young Stuart H. Artificial intelligence and machine learning for future army applications [Konference] // Proceedings Volume 10635, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR IX. - Orlando, Florida, United States : SPIE, 2018.

Fu Xiaomeng a Qu Huiming Research on Semantic Segmentation of High-resolution Remote Sensing Image Based on Full Convolutional Neural Network [Konference] // 2018 12th International Symposium on Antennas, Propagation and EM Theory (ISAPE). - Hangzhou, China : IEEE, 2018.

Gandhi Rohith A look at Gradient Descent and RMSprop Optimizers [Online] // Towards Data Science. - 19. 06 2018. - 05. 05 2021. - https://towardsdatascience.com/a-look-at-gradient-descent-and-rmsprop-optimizers-f77d483ef08b.

Gavin Henri P. The Levenberg-Marquardt algorithm fornonlinear least squares curve-fitting problems [Referát] / Department of Civil and Environmental Engineering ; Duke University. - [místo neznámé] : Department of Civil and Environmental Engineering Duke University, 2020. - https://people.duke.edu/~hpgavin/ce281/lm.pdf.

Gelfand Natasha [a další] Geometrically Stable Sampling for the ICP Algorithm [Konference] // Fourth International Conference on 3-D Digital Imaging and Modeling. - Banff, AB, Canada : IEEE, 2003.

Ghosal Deepanway a Kolekar Maheshkumar H. Music Genre Recognition Using Deep Neural Networks and Transfer Learning [Konference] // Proc. Interspeech 2018. - Hyderabad, India : [autor neznámý], 2018. - stránky 2087-2091.

GiHub - IntelRealsense/librealsense: Intel Realsense SDK [Online]. - 01. 03 2021. - https://github.com/IntelRealSense/librealsense.

Glorot Xavier a Bengio Yoshua Understanding the difficulty of training deep feedforward neural networks [Konference] // Proceedings of the13th International Conferenceon Artificial Intelligence and Statistics (AISTATS) 2010. - Sardinia, Italy : JMLR: W&CP 9, 2010. - Sv. 9. - Dostupné z: http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf.

Groman Martin Diplomová práce: Tvorba umělé neuronové sítě pro výpočet termodynamických veličin [Online]. - Vysoké učení technické v Brně. Fakulta strojního inženýrství, 2019. - 22. 03 2021. - Dostupné z: http://hdl.handle.net/11012/175381.

Guosheng Lin [a další] RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation [Článek] // Computer Vision and Pattern Recognition. - [místo neznámé] : arxiv.org, 25. 11 2016. - https://arxiv.org/abs/1611.06612#.

Hardy Jeremy [a další] Navigation and Control for Micro Aerial Vehicles in GPS-Denied Environments [Konference] // 2016 IEEE/ION Position, Location and Navigation Symposium (PLANS). - Savannah, GA, USA : IEEE, 2016. - 978-1-5090-2042-3.

Hart Matthew Tesla's Autopilot 'Vision' Looks Like TERMINATOR HUD [Online] // Nerdlist.com. - 6. 2 2020. - 21. 09 2021. - https://nerdist.com/article/teslas-autopilot-vision-terminator-hud/.

Haykin Simon Neuroal Networks: a Coprehensive Foundation. Second Edition [Kniha]. - [místo neznámé] : Pearson Education Inc., 1999. - ISBN: 81-7808-300-0.

He Kaiming [a další] Deep Residual Learning for Image Recognition [Konference] // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). - Las Vegas, NV, USA : IEEE, 2016.

Hebb Donald O. Organization of Behavior: A neropsychological Theory [Kniha]. - New York : McGill Univentgli, 1949.

Hinton Geoffrey E. A Practical Guide to Training Restricted Boltzmann Machines [Oddíl knihy] // Lecture Notes in Computer Science: Neural Networks: Tricks of the Trade. - [místo neznámé] : Springer Berlin Heidelberg, 2012. - Sv. DOI: 10.1007/978-3-642-35289-8_32.

Hopfield John Neural networks and physical systems with emergent collective computational abilities [Článek] // Procceedings of the National Academy od Science. - USA : [autor neznámý], 1982. - 8 : Sv. 79.

Hopfield John Neurons With Graded Response Have Collective Computational Properties Like Those of Two-State Neurons [Článek] // Procceedings of the National Academy od Science. - USA : [autor neznámý], 1984. - 10 : Sv. 81.

Huang Gao [a další] Densely Connected Convolutional Networks [Konference] // Computer Vision and Pattern Recognition 2017. - 2017. - Sv. online.

Chen Hongshun a Lu Shilin Building Extraction from Remote Sensing Images Using SegNet [Konference] // 2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC). - Xiamen, China : IEEE, 2019.

Chmelař Pavel Systém automatického mapování 3D prostoru [Kniha]. - Pardubice : Univerzita Pardubice, 2018. - Disertační práce.

Chollet Francois Xception: Deep Learning with Depthwise Separable Convolutions [online] = Computer Vision and Pattern Recognition. - [místo neznámé] : Google, Inc., 2017.

Intel Intel® RealSense™ Tracking Camera [Online] // Datasheet: Intel® RealSense™ Tracking Camera T265, Intel® RealSense™ Tracking Module T261. - Intel, Září 2019. - Revision 004. - 31. 1 2021. - https://www.intelrealsense.com/wp-content/uploads/2019/09/Intel_RealSense_Tracking_Camera_Datasheet_Rev004_release.pdf.

Intel Intel®RealSenseTMD400Series Product Family [Online] // Intel®RealSense™ Vision Processor D4, Intel®RealSense™ Vision Processor D4 Board, Intel®RealSense™ Depth Module D400, Intel®RealSense™ Depth Module

D410,Intel®RealSense™ Depth Module D415, Intel®RealSense™Depth Camera D415, Intel®RealSense™ Depth Module D. - Intel, 01 2019. - Revision 005. - 31. 01 2021. - https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf.

Jepsen Andres Grunnet [a další] Projectors for D400 Series Depth Cameras [Online] // Intel Realsense Documentation. - Intel. - 01. 12 2021. - https://www.intelrealsense.com/wp-content/uploads/2019/03/WhitePaper_on_Projectors_for_RealSense_D4xx_1.0.pdf.

Kelley Henry J. Gradient Theory of Optimal Flight Paths [Článek] // American Rocket Society. - [místo neznámé] : American Rocket Society, 1960. - 30. - Sv. 10.

Kingma Diederik P a Ba Jimmy Lei ADAM: A METHOD FORSTOCHASTICOPTIMIZATION [Konference] // ICLR 2015. - San Diego : [autor neznámý], 2015.

Kingma Diederik P. a Ba Jimmy Adam: A Method for Stochastic Optimization [Konference] // International Conference on Learning Representations (ICLR 2015). - San Diego : arXiv.org, 2015.

Le Cunn Yann [a další] Efficient BackProp [Kniha] = DOI: 10.1007/3-540-49430-8_2. - Berlin : Springer Berlin Heidelberg, 1998. - str. 9. - Dostupné z: http://yann.lecun.com/exdb/publis/pdf/lecun-98b.pdf. - ISBN 978-3-540-65311-0.

LeCun Yann, Cortes Corinna a Burges Christopher J.C. MNIST hadwritten digit database [Online] // The MNIST Database of handwritten digits. - 2021. - 01. 10 2021. - http://yann.lecun.com/exdb/mnist/.

Lera Gabriel a Pinzolas Miguel Neighborhood based Levenberg-Marquardt algorithm for neural network training [Článek] // IEEE Transactions on Neural Networks. - [místo neznámé] : Neural Networks, 2002. - 13. - Sv. 5. - stránky 1200-1203. - 10.1109/TNN.2002.1031951.

Linnainmaa Seppo he representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors. Master's Thesis. [Kniha]. - Helsinky : University Helsinky, 1970.

Linnainmaa Seppo Taylor expansion of the accumulated rounding error [Článek] // BIT Numerical Mathematics. - [místo neznámé] : BIT Numerical Mathematics, 1976. - 2 : Sv. 16. - stránky 146-160.

Lisa Lab Deep Learning Tutorial, Release 0.1 [Kniha]. - Montreal : University of Montreal, 2015. - Dostupné z: http://deeplearning.net/tutorial/deeplearning.pdf.

Liu Angbang [a další] A Deep Fully Convolution Neural Network for Semantic Segmentation Based on Adaptive Feature Fusion [Konference] // 2018 5th International Conference on Information Science and Control Engineering (ICISCE). - Zhengzhou, China : IEEE, 2018.

Long Jonathan, Shelhamer Evan a Darrell Trevor Fully Convolutional Networks for Semantic Segmentation [Článek] // IEEE Transactions on Pattern Analysis and Machine Intelligence . - [místo neznámé] : IEEE, 2016. - 4 : Sv. 39. - stránky 640-651.

Lu Feng a Milios Evangelos Robot pose estimation in unknown environments by matching 2D range scans [Konference] // 1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. - Seattle, WA, USA : IEEE, 1994.

Magnusson Martin The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection [Kniha]. - [místo neznámé] : Örebro Universit, 2009. - 978-91-7668-696-6.

Magnusson Martin, Lilienthal Achim a Duckett Tom Scan Registration for Autonomous Mining Vehicles Using 3D-NDT [Článek] // Journal of Field Robotics. - [místo neznámé] : Wiley Periodicals, Inc., 2007. - 10 : Sv. 24. - stránky 803-827.

Maur Pavel Delaunay Triagulation in 3D [online]. - Plzeň : University of West Bohemian in Pilsen, 2002.

Minguez Javier, Montesano Luis a Lamiraux Florent Metric-Based Iterative Closest Point Scan Matching for Sensor Displacement Estimation [Článek] // IEEE Transactions on Robotics. - [místo neznámé] : IEEE, 2006. - 5 : Sv. 22. - stránky 1047-1054.

Minsky Marvin a Papert Seymour Perceptrons [Kniha]. - [místo neznámé] : M.I.T Press, 1969.

Montesano Luis, Minguez Javier a Montano Luis Probabilistic scan matching for motion estimation in unstructured environments [Konference] // 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. - Edmonton, AB, Canada : IEEE, 2005.

Nanni Loris, Maguolo Gianluca a Lumini Alessandra Exploiting Adam-like Optimization Algorithms to Improve the Performance of [Konference] // MIDL 2021 Conference . - 2021.

National Museum of American History [Online] // Perceptron, Mark I. - National Museum of American History. - 20. 09 2021. - https://americanhistory.si.edu/collections/search/object/nmah_334414.

Nguyen Vinh The Determinants of Intention to use Google Lens [Článek] // Special Issue on Learning Systems and Innovation in Education – iJIST. - [místo neznámé] : International Journal of Information Science & Technology, 2021. - 2 : Sv. 5. - ISSN: 2550-511.

Nielsen Michael Neural Networks and Deep Learning [Kniha]. - [místo neznámé] : Determination Press, 2015.

Nilsson Nils John Learning Machines [Kniha]. - New York : McGrawHill, 1965.

Nuchter Andreas 3D Robotic Mapping - The Simultaneous Localization and Mapping Problem with Six Degrees of Freedom [Článek] // Springer Tracts in Advanced Robotics. - [místo neznámé] : Springer, 2009. - 52. - 978-3-540-89883-2.

Open3D: A Modern Librady for 3D Data processing [Online] // Open3D. - 20. 12 2021. - http://www.open3d.org/.

Özkan Coskun a Erbek Filiz Sunar The Comparison of Activation Functions for Multispectral Landsat TM Image Classification [Konference] = DOI: 10.14358/PERS.69.11.1225 // Photogrammetric Engineering & Remote Sensing. - 2003. - Sv. 69(11). - Dostupné z: https://www.asprs.org/wp-content/uploads/pers/2003journal/november/2003_nov_1225-1234.pdf. - ISSN 00991112.

Pearson Karl On lines ans Planes of Closest Fit to Systems of Points in Space [Článek] // Philosophical Magazine and Journal of Science. - London : [autor neznámý], 1901. - 11 : Sv. 2. - stránky 559-572.

Point Cloud Library 1.12.0-dev documentation [Online] // Estimation Surface Normals in a PointCloud. - 01. 12 2021. - https://pointclouds.org/documentation/tutorials/normal_estimation.html.

Point Feature Histogram (PFH) Desctriptors [Online] // Point Cloud Library 0.0 Documentation. - PointCloud Library. - 01. 12 2021. - https://pcl.readthedocs.io/projects/tutorials/en/latest/pfh_estimation.html.

Raguram Rahul [a další] USAC: A Universal Framework for Random Sample Consensus [Článek] // IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE. - [místo neznámé] : IEEE, 2012. - 8 : Sv. 35. - stránky 2022-2038.

Rismiyati Rismi [a další] Xception Architecture Transfer Learning for Garbage Classification [Konference] // 2020 4th International Conference on Informatics and Computational Sciences (ICICoS). - Semarang, Indonesia : IEEE, 2020.

Ronneberger Olaf, Fischer Philipp a Brox Thomas U-Net: Convolutional Networks for Biomedical Image Segmentation [Konference] // International Conference on Medical Image Computing and Computer-Assisted Intervention. - 2015.

Ros.org www.ros.org [Online] // ROS - Robot Operating System. - 2021. - https://www.ros.org/.

Rossenblatt Frank [a další] MARK I PERCEPTRON OPERATOR'S MANUAL (PROJECT PARA) [online]. - New York : Cornell aeronautical laboratory, inc., 1960.

Ruby Usha a Yendapalli Vamsidhar Binary cross entropy with deep learning technique for Image classification [Článek] // International Journal of Advanced Trends in Computer Science and Engineering. - 2020. - 4 : Sv. 9.

Rumelhart David E., Hinton Geoffrey E. a Williams Ronald J. Learning representations by back-propagating errors [Článek] // Nature. - 09. 10 1986. - 323.

Rusinkiewicz Szymon a Levoy Marc Efficient variants of the ICP algorithm [Konference] // Proceedings Third International Conference on 3-D Digital Imaging and Modeling. - Quebec City, QC, Canada : IEEE, 2001.

Rusu Radu Bogdan a Cousins Steve 3D is here: Point Cloud Library (PCL) [Konference] // IEEE International Conference on Robotics and Automation (ICRA). - Shanghai, China : IEEE, 2011.

Rusu Radu Bogdan Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments [Kniha]. - München : Institut für Informatik der Technischen Universität München, 2009. - Sv. Disrtační práce.

Rusu Radu Bogdan, Blodow Nico a Beetz Michael Fast Point Feature Histograms (FPFH) for 3D registration [Konference] // 2009 IEEE International Conference on Robotics and Automation. - Kobe,Japan : iEEE, 2009. - eISSN: 1050-4729 .

Satya Muhammad Ferianda a Suyanto Suyanto Music Source Separation Using Generative Adversarial Network and U-Net [Konference] // 2020 8th International Conference on Information and Communication Technology (ICoICT). - Yogyakarta, Indonesia : IEEE, 2020.

Segal Aleksandr V., Haenhel Dirk a Thrun Sebastian Generalized-ICP [Konference] // Robotics: Science and Systems V. - Seattle, USA : [autor neznámý], 2009.

Shantanu Ingle a Phute Madhuri Tesla Autopilot : Semi Autonomous Driving, an Uptick for Future Autonomy [Článek] // International Research Journal of Engineering and Technology (IRJET). - 2016. - 03 : Sv. 09.

Sharma Mohini a Sau Paresh Chandra Blood Vessel Segmentation using SegNet [Konference] // 2019 4th International Conference on Information Systems and Computer Networks (ISCON). - Mathura, India : IEEE, 2020.

Schmidhuber Juergen Deep Learning in Neural Networks: An Overview [Článek] // Neural Networks. - 08. 10 2014. - Sv. 61. - stránky 85-117.

Schnabel Ruwen, Wahl Roland a Klein Reinhard Efficient RANSAC for Point-Cloud Shape Detection [Článek]. - [místo neznámé] : Blackwell Publishing, 2007. - 2 : Sv. 26. - stránky 214-226.

Sinčák Peter a Andrejková Gabriela Neurónové siete: Inžiniersky prístup (1. diel) [Kniha]. - [místo neznámé] : Technická Univerzita Košice, 1996.

SNARC Maze Solver – Minsky / Edmonds (American) - 1951 [Online] // Cyberneticszoo.com: A history of cybernetic animals and early robots. -

Cyberneticszoo.com. - 17. 09 2021. - http://cyberneticzoo.com/mazesolvers/1951-maze-solver-minsky-edmonds-american/.

Srivastava Nitish [a další] Dropout: A Simple Way to Prevent Neural Networks fromOverfitting [Článek] // Journal of Machine Learning Research / editor Bengio Yoshua. - 2014. - 15. - stránky 1929-19. - ISSN 1533-7928.

Takeuchi Eijiro a Tsubouchi Takashi A 3-D Scan Matching using Improved 3-D Normal Distributions Transform for Mobile Robotic Mapping [Konference] // 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. - Beijing, China : IEEE, 2006. - stránky 3068-3073.

Takubo Tomohito [a další] NDT scan matching method for high resolution grid map [Konference] // 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. - St. Louis, MO, USA : IEEE, 2009.

TensorFlow TensorFlow Core v2.1.0 [Online] // TensorFlow Core v2.1.0: tf.keras.optimizers.Adam. - 2019a. - https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Adam.

TensorFlow TensorFlow Core v2.1.0 - tf.keras.initializers.HeNormal [Online] // TensorFlow. - 2019b. - 12. 10 2019. - https://www.tensorflow.org/api_docs/python/tf/keras/initializers/HeNormal.

Tomono Masahiro Robust 3D SLAM with a stereo camera based on an edge-point ICP algorithm [Konference] // 2009 IEEE International Conference on Robotics and Automation. - Kobe, Japan : IEEE, 2009.

Tsang Sik-Ho Review: DenseNet — Dense Convolutional Network (Image Classification) [Online] // towardsdatascience.com. - 25. 11 2018. - 01. 11 2021. - https://towardsdatascience.com/review-densenet-image-classification-b6631a8ef803.

Urmee Progya Paromita [a další] Real-time Bangla Sign Language Detection using Xception Model with Augmented Dataset [Konference] // 2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE). - Bangalore, India : IEEE, 2019.

Wang Chi-Feng A Basic Introduction to Separable Convolutions [Online] // towards data science. - towards data science, 14. 08 2018. - 13. 10 2021. -

https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728.

Widrow Bernard An adaptive "ADALINE" Neuron using chemical "MEMRISTORS" [Kniha]. - Stanford : Stanford Electrotechnics Laboratories, 1960.

Woo Boyeong a Lee Myungeun Comparison of tissue segmentation performance between 2D U-Net and 3D U-Net on brain MR Images [Konference] // 2021 International Conference on Electronics, Information, and Communication (ICEIC). - Jeju, Korea (South) : IEEE, 2021.

Wu Xizhi [a další] An Xception Based Convolutional Neural Network for Scene Image Classification with Transfer Learning [Konference] // 2020 2nd International Conference on Information Technology and Computer Application (ITCA). - Guangzhou, China : IEEE, 2020.

Yaniv Ziv Random Sample Consensus (RANSAC) Algorithm, A generic implementation [Článek] // The Insight Journal. - 2010.

Yu Changqian [a další] BiSeNet V2: Bilateral Network with Guided Aggregation for Real-time Semantic Segmentation [Článek] // International Journal of Computer Vision. - [místo neznámé] : Springer, 2021. - 129. - stránky 3051–3068.

Yu Changqian [a další] BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation [Konference] // Computer Vision – ECCV 2018. - Munich, Germany : ECCV, 2018. - Sv. XIII. - stránky 334-349.

Zhang Zhilu a Sabuncu Mert R. Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels [Konference] // 32nd Conference on Neural Information Processing Systems (NeurIPS 2018). - Montreal, Canada : arXiv.org, 2018.

Zhao Shida [a další] Region segmentation of sheep ribs based on fully convolutional neural network [Konference] // 2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL). - Chongqing, China : IEEE, 2020.

Zheng Xiaoxiong a Chen Tao Segmentation of High Spatial Resolution Remote Sensing Image based On U-Net Convolutional Networks [Konference] // IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium. - Waikoloa, HI, USA : IEEE, 2020.

Zhou Qian Yi, Park Jaesik a Koltun Vladlen Open3D: A Modern Library for 3D Data Processing [Článek] // arXiv:1801.09847. - [místo neznámé] : arXiv, 2018.