VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

POŘÍZENÍ PODROBNÉ A GIGANTICKÉ FOTOGRAFIE A LOKALIZACE V NÍ

CAPTURING OF DETAILED AND VERY LARGE PHOTOGRAPH AND LOCALIZATION WITHIN

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. PAVOL DUBOVEC**
AUTHOR

VEDOUCÍ PRÁCE **prof. Ing. ADAM HEROUT, Ph.D.**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání diplomové práce

154403

Ústav:	Ústav počítačové grafiky a multimédií (UPGM)
Student:	**Dubovec Pavol, Bc.**
Program:	Informační technologie a umělá inteligence
Specializace:	Počítačové vidění
Název:	**Pořízení podrobné a gigantické fotografie a lokalizace v ní**
Kategorie:	Zpracování obrazu
Akademický rok:	2023/24

Zadání:

1. Seznamte se s problematikou sešívání fotografií a s problematikou vyhledávání v obrazových databázích.
2. Navrhněte a prototypujte postup pro pořizování rozsáhlých fotografií planárních povrchů (podlahy, desky stolu atp.).
3. Pořizujte fotografie povrchů, vyhodnocujte vlastnosti vyvinuté metody snímání a iterativně ji vylepšujte.
4. Navrhněte metody lokalizace v pořízených rozsáhlých fotografiích, implementujte je a na datových sadách je iterativně vyhodnocujte a vylepšujte.
5. Na základě vyvinutých postupů a pořízených datových sad vytvořte rozsáhlou datovou sadu výřezů fotografií s věrohodnými anotacemi o jejich překryvech.
6. Zhodnoťte dosažené výsledky a navrhněte možnosti pokračování projektu; vytvořte plakátek a krátké video pro prezentování projektu.

Literatura:

- T. Liao and N. Li, "Single-Perspective Warps in Natural Image Stitching," in IEEE Transactions on Image Processing, vol. 29, pp. 724-735, 2020
- Chen, YS., Chuang, YY. (2016). Natural Image Stitching with the Global Similarity Prior. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) Computer Vision – ECCV 2016. ECCV 2016
- Jiaming Sun et al.: LoFTR: Detector-Free Local Feature Matching with Transformers, CVPR 2021
- Goodfellow, Bengio, Courville: Deep Learning, MIT Press, 2016
- Bharath Ramsundar, Reza Bosagh Zadeh: TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning, O'Reily Media, 2018
- Gary Bradski, Adrian Kaehler: Learning OpenCV; Computer Vision with the OpenCV Library, O'Reilly Media, 2008
- Richard Szeliski: Computer Vision: Algorithms and Applications, Springer, 2011

Při obhajobě semestrální části projektu je požadováno:
body 1. a 2., značné rozpracování bodů 3. a 4.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce:	**Herout Adam, prof. Ing., Ph.D.**
Vedoucí ústavu:	Černocký Jan, prof. Dr. Ing.
Datum zadání:	1.11.2023
Termín pro odevzdání:	17.5.2024
Datum schválení:	9.11.2023

Abstrakt

Cieľom tejto práce bolo vytvoriť veľký obrázok a novú techniku na lokalizáciu fotografie vo väčšom obrázku, aby sa zvýšila rýchlosť a presnosť bežných metód. Navrhovaná technika využíva architektúru CNN na extrakciu *embeddings* z dopytovaného obrázka, ktoré sa použijú na vyhľadávanie v databáze *embeddings* z veľkej fotografie. Boli natrénované dva modely na veľkom súbore údajov: klasifikačný (CE) a dištančný (triplet). Na určenie umiestnenia obrázkov a na generovanie veľkého obrázka sa použili konvenčné metódy. Databáza vkladov sa vytvorila rozdelením veľkej fotografie pomocou natrénovaného modelu. V databáze sa vyhľadá K-najbližších *embeddings* výrezov *query* obrázka. Tieto *embeddings* sa generujú rozdelením *query* fotografie na rovnako veľké časti ako vstupy CNN. Optimálny model homografie sa určí náhodným výberom na základe pozícií štyroch výrezov *query* obrazov a ich zodpovedajúcich pozícií vo veľkom obraze. Ako výsledná pozícia sa vyberie model homografie s najnižším harmonickým priemerom *embedding* vzdialenosti. Homografia sa optimalizuje pomocou párovania šablón, kde je to možné. Metóda vykazuje dostatočnú presnosť a vysokú rýchlosť na testovacích súboroch údajov. Najlepší model dosiahol presnosť top-1 97.71% a presnosť top-3 99.67%. V ďalšom výskume sa budú zisťovať výsledky metódy pri zvyšujúcej sa heterogenite povrchu, možnosti automatizácie vyhľadávania videí na získanie veľkého súboru údajov s fotografiami a jej účinnosť pri lokalizácii fotografií, keď bežné metódy zlyhávajú.

Abstract

The goal of this work was to create a large image and a new technique to localize the photo in the larger image to increase the speed and accuracy of conventional methods. The proposed technique uses CNN architecture to extract embeddings from the queried image which will be used to search the database of embeddings from the large photo. Two models have been trained on a large dataset: based on classification (CE) and distance (triplet) Conventional methods were used to determine the location of the images and to generate the large image. A database of embeddings was created by partitioning the large image using the trained model. The database is searched for the K-nearest embeddings of the cutouts of the query image. These embeddings are generated by dividing the query photo into the same size parts as the CNN inputs. The optimal homography model is determined by random selection based on the positions of the four query image cutouts and their corresponding positions in the big picture. The homography model with the lowest harmonic mean of the embedding distance is selected as the final position. The homography is optimized using template matching where possible. The method shows sufficient accuracy and high speed on test datasets. The best model achieved a top-1 accuracy of 97.71% and a top-3 accuracy of 99.67%. Future research will investigate the performance of the method under increasing surface heterogeneity, the possibility of automating video retrieval to obtain a large dataset with photos, and its effectiveness in locating photos when conventional methods fail.

Kľúčové slová

Lokalizácia obrazu, Odhad homografie, Približné vyhľadávanie, KNN

Keywords

Image Localization, Homography Estimation, Approximate Search, CNN

Citácia

DUBOVEC, Pavol. *Pořízení podrobné a gigantické fotografie a lokalizace v ní*. Brno, 2024. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. Ing. Adam Herout, Ph.D.

Pořízení podrobné a gigantické fotografie a lokalizace v ní

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením pána profesora Herouta. Uviedol som všetky literárne pramene a publikácie z ktorých som čerpal.

.....
Pavol Dubovec
16. mája 2024

Poďakovanie

Ďakujem vedúcemu diplomovej práce prof. Ing. Adamovi Heroutovi Ph.D. za pomoc, rady a inšpiráciu pri písaní tejto práce.

Obsah

1 Úvod **2**

2 Získavanie obrovskej fotografie **3**
2.1 Zošívanie (*stitching*) obrázkov 3
2.2 Určovanie homografie 5
2.3 Metódy určovania homografie 7
2.4 Skreslenie (*warping*) obrazu 15
2.5 Spájanie (*blending*) obrázkov 15
2.6 Nástroje pre prácu s počítačovým videním 17

3 Lokalizácia fotografie v obrovskej fotografii **19**
3.1 Známe prístupy určovania lokalizácie 19
3.2 Neurónové siete 20
3.3 Konvolučné neurónové siete 21
3.4 Vyhľadávanie obrázkov na základe obsahu (*CBIR*) 27
3.5 Nástroje pre vyhľadávanie najbližších susedov 29

4 Riešenie problému zošívania obrázkov **30**
4.1 Aplikácia pre zobrazenie homografie datasetu 30
4.2 Program pre zošívanie obrázkov 32

5 Riešenie problému lokalizácie obrázku v obrovskom obrázku **36**
5.1 Tvorba datasetu 36
5.2 Návrh neurónovej siete pre tvorbu *embeddings* (*encoder*) 41
5.3 Učenie pomocou klasifikácie 43
5.4 Učenie pomocou učenia vzdialeností (*diatance training*) 45
5.5 Vytvorenie databázy *embedings* 47
5.6 Program pre lokalizáciu query obrázku 49

6 Experimenty **55**
6.1 Experimenty s tréningom neurónových sietí 55
6.2 Experimenty pri lokalizácii 59

7 Záver **64**

Literatúra **65**

Kapitola 1

Úvod

Cieľom tejto práce je vyvinúť riešenia dvoch známych problémov počítačového videnia, ktorými sú získanie obrovskej fotografie a lokalizácia. Riešením prvého problému je implementácia metódy na automatické zhotovenie obrovskej fotografie. V súčasnosti existuje veľké množstvo zariadení, ktoré dokážu túto úlohu splniť bez potreby následného spracovania až do veľkosti niekoľkých tisíc pixelov. Aj tento prístup má však svoje limity. V prípade, že sa dosiahne niektoré z fyzikálnych alebo konštrukčných obmedzení (nemožnosť zachytiť celý objekt, blízkosť zariadenia k snímanej oblasti, snímanie v interiéri) je potrebné tento problém ďalej riešiť. Najbežnejšou metódou riešenia tohto problému je spájanie snímok. Riešenie navrhnuté v tomto článku je určené na snímanie interiérových snímok veľmi veľkých plôch v ideálnych podmienkach (pohľad z vtáčej perspektívy, rovnaká výška od snímaného povrchu) s cieľom získať veľmi veľkú fotografiu. Takéto scény nie je možné zachytiť bežnými zariadeniami. Je to spôsobené najmä svetelnými podmienkami, neobvyklým uhlom snímania (pohľad zhora) alebo neobvyklým tvarom samotnej scény. Výsledkom je vytvorenie fotografie interiéru, ktorý by nebolo možné zachytiť prirodzeným spôsobom.

Riešením druhého problému je vyvinúť metódu na určenie homografie daného obrazu vo väčšom obraze. Konvenčná metóda riešenia problému lokalizácie zahŕňa detekciu kľúčových bodov, extrakciu lokálnych deskriptorov v okolí týchto kľúčových bodov, porovnanie extrahovaných znakov z načítaného obrazu so znakmi z veľkého obrazu (mapy) a následné použitie zhodných kľúčových bodov na odhad homografie. Tento prístup je jednoduchý a účinný, ale má niekoľko nevýhod. Metóda je citlivá najmä na geometrické a fotometrické rozdiely, ako aj na nízky počet zhodných bodov, keď je rozdiel vo veľkosti fotografií značný. Z týchto dôvodov sa skúmal nový prístup na určovanie polohy založený na modeli vytvorenom konvolučnou neurónovou sieťou. Táto sieť sa následne využíva kódovanie malých obrázkov do *embeddings*. Z obrovského obrázku je vytvorená *embedding* databáza. Lokalizačný proces zahŕňa rozdelenie vstupnej fotografie na menšie časti, určenie *embedding* pre každú časť a použitie algoritmu náhodného výberu na vytvorenie hypotézy homografie. Následná optimalizácia na úrovni pixelov pomocou porovnávania šablón (*template matching*) spresňuje homografiu. Metóda vykazuje sľubné výsledky z hľadiska rýchlosti aj presnosti.

V nasledujúcich kapitolách sú najprv predstavené oblasti výskumu, návrh jednotlivých častí a riešenie s výsledkami. Kapitola 2 predstavuje stručný prehľad metodiky image stitchingu. V kapitole 3 je predstavená metodika lokalizácie fotografie vo väčšej fotografií. Nasledujú kapitoly vypracovania riešenia zošívania (kapitola 4) a lokalizácie (kapitola 5) obrázkov. Tieto kapitoly sú nasledované kapitolou s experimentami (kapitola 6) a záverom (kapitola 7) so zhrnutím celej práce.

Kapitola 2

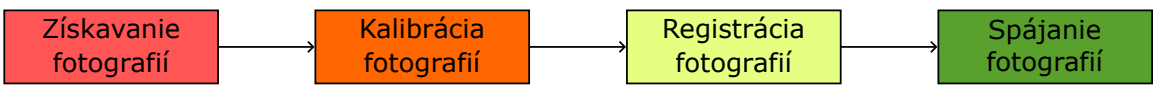
Získavanie obrovskej fotografie

Získanie veľkého obrazu povrchu je pomerne náročná úloha počítačového videnia. Pozostáva z veľkého počtu čiastkových úloh, ktoré sa musia všetky efektívne vyriešiť, aby sa získal kvalitný výsledok. Do určitej veľkosti miestnosti sa táto úloha dá riešiť pomocou snímania širokouhlým objektívom. Avšak veľkosť miestnosti, objekty na scéne a svetelné podmienky pôsobia ako obmedzujúce faktory, ktoré obmedzujú možnosti použitia tejto metódy. Preto sa na získanie obrazu veľmi veľkej plochy najčastejšie používa metóda spájania obrazu (kapitola 2.1).

2.1 Zošívanie (*stitching*) obrázkov

Zošívanie (*stitching*) alebo mozaikovanie (*mosaicing*) obrázkov je proces kombinovania viacerých fotografických snímok s prekrývajúcimi sa zornými poľami s cieľom vytvoriť segmentovanú panorámu alebo snímku s vysokým rozlíšením [69]. Tento proces pozostáva podľa [38, 23] zo štyroch krokov (obrázok 2.1):

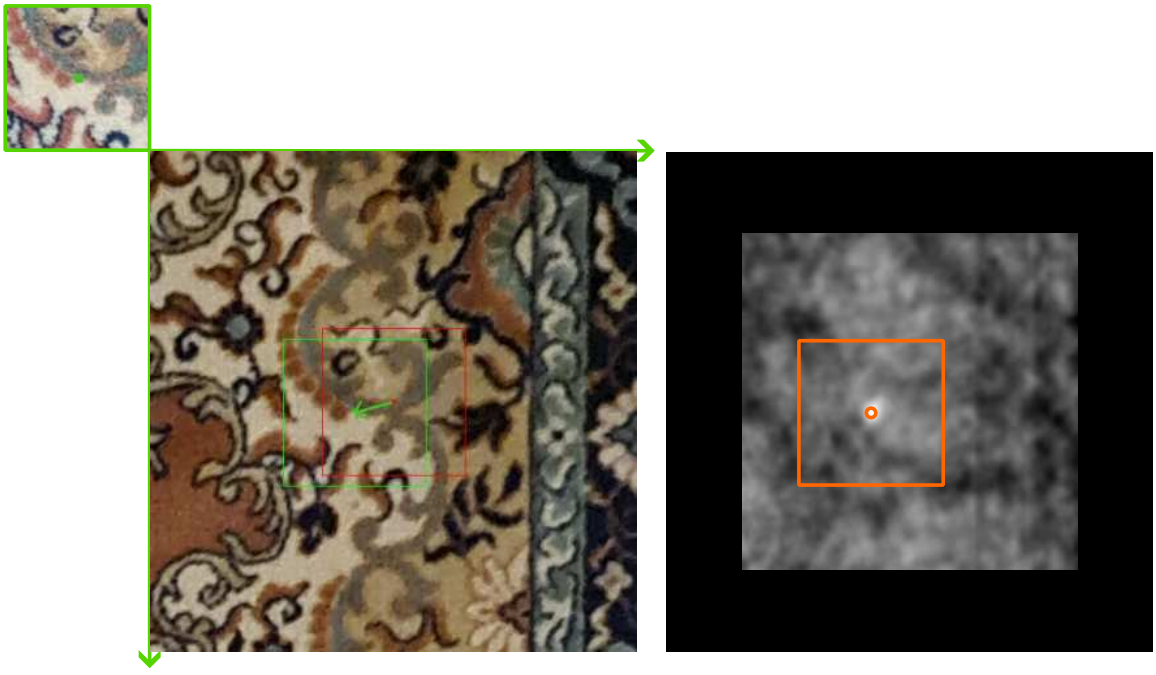
- **Získavanie fotografií** – fáza zaobstarania fotografií pre vykonávanie nasledujúcich operácií. Fotografie môžu byť zachytené rôznymi spôsobmi: snímacím zariadením pohybujúcim sa po sekvenčných smeroch (pohyb paralelný so scénou), otáčajúcim sa okolo svojej vertikálnej osi, prípadne ručným snímacím zariadením [2].
- **Kalibrácia fotografií** – fáza minimalizácie rozdielov medzi ideálnym modelom objektívu a použitou kombináciou fotoaparátu a objektívu.
- **Registrácia fotografií** – fáza tvorby geometrickej zhody medzi obrázkami. Dochádza k vyrovnávaniu dvoch alebo viacerých obrázkov, ktoré sú zachytené z rôznych uhlov pohľadu.
- **Zlučovanie (*blending*) fotografií** – fáza úpravy obrázka tak, aby bol prechod z jedného obrázka do druhého plynulejší.



Obr. 2.1: Postupnosť krokov zošívania obrázkov (*image stitching*).

Metódy zošívania obrázkov je možné klasifikovať z hľadiska domény na:

- **Založené na frekvenčnej doméne** – analyzujú spektrálne informácie obrazov. Tento proces prebieha vo frekvenčnej oblasti a je založený na Fourierovej transformácii (FT). Prekrývajúca sa oblasť medzi dvoma vstupnými obrazmi sa identifikuje vykonaním elementárneho súčinu Fourierovej transformácie jedného vstupného obrazu s komplexnou Fourierovou transformáciou druhého vstupného obrazu [70].
- **Založené na priestorovej doméne** – pracujú priamo s obrazom. Môžeme ich ďalej rozdeliť na:
  - **Párovanie šablón (*template-matching*)** – používajú porovnávanie pixelov (intenzít) v obraze (obrázok 2.2). Osobitný význam majú najmä tieto vlastnosti:
    + Nevyužívajú štrukturálne dáta o obrázku,
    + Nevyžadujú výrazné objekty v scéne,
    + Sú rýchle a jednoduché,
    – Nefungujú dobre na obrázkoch so šumom,
    – Majú obmedzený rozsah konvergencie,
    – Nedokážu spracovať elastické deformácie,
    – Nefungujú dobre pri nelineárnych zmenách svetla.



Obr. 2.2: Tento obrázok znázorňuje metódu párovania šablón. Šablóna (*template*) [obrázok so zeleným ohraničením vľavo] je vyhľadávaná v obrázku. Červený štvorec znázorňuje šablónu pred úpravou pomocou párovania šablón, zatiaľ čo zelený štvorec znázorňuje šablónu po tejto úprave. Zelená šípka označuje vektor pohybu tohto štvorca. Na obrázku vpravo je zobrazená mapa s výsledkom porovnávania, pričom oranžový štvorec s kruhom označuje miesto s najvyššou energiou. V tomto prípade sa používa metóda (2.3).

V súčasnosti sú najrozšírenejšie tieto metódy párovania šablón:

– SSD[1] – je známa aj ako štvorcová euklidovská vzdialenosť (L2). SSD je menej citlivá na šum v obraze v porovnaní s inými technikami. Nie je však vhodná pre detekciu otočených, zmenšených alebo deformovaných objektov. Je tiež citlivá na zmeny jasu v obraze.

$$\mathrm{SSD}(x,y) = \sum_{x',y'} \Big(T\big(x',y'\big) - I\big(x+x',y+y'\big) \Big)^2 \tag{2.1}$$

– NSD[2] – je normalizovaná forma (2.1). Normalizácia umožňuje porovnávať šablóny rôznych veľkostí. NSD je menej citlivá na zmeny jasu v obraze a taktiež je účinnejšia pri identifikácii objektov, ktoré boli otočené ako SSD. Je však tiež do určitej miery náchylná na šum.

$$\mathrm{NSD}(x,y) = \frac{\sum_{x',y'} \Big(T\big(x',y'\big) - I\big(x+x',y+y'\big) \Big)^2}{\sqrt{\sum_{x',y'} T\big(x',y'\big)^2 \cdot \sum_{x',y'} I\big(x+x',y+y'\big)^2}} \tag{2.2}$$

– NCC[3] – Normalizovaná krížová korelácia. Je invariantná voči jasu, odolná voči rotácii, škálovaniu a skresleniu. Je výpočetne náročnejšia ako predchádzajúce metódy a môže byť do určitej miery náchylná na šum. Shou-Der Wei a Shang-Hong Lai navrhli algoritmus vyhľadávania vzorov založený na NCC v roku 2008 [71].

$$\mathrm{NCC}(x,y) = \frac{\sum_{x',y'} \Big(T\big(x',y'\big) \cdot I\big(x+x',y+y'\big) \Big)}{\sqrt{\sum_{x',y'} T\big(x',y'\big)^2 \cdot \sum_{x',y'} I\big(x+x',y+y'\big)^2}} \tag{2.3}$$

– **Metódy využívajúce určovanie homografie** – pre zistenie vzťahu medzi obrazmi (registráciu) využívajú techniku odhadu homografie. Dôležitými vlastnosťami techník spadajúcich do tejto kategórie sú:

+ Invariantnosť voči šumu obrazu,
+ Invariantnosť voči translačným transformáciám,
+ Invariantnosť voči rotačným transformáciám,
+ Invariantnosť voči zmene mierky (len niektoré algoritmy),
+ Efektívna redukcia problému (aj 1000 násobne menšia množina),
− Využívajú štrukturálne dáta o obrázku,
− Sú nevhodné v prípade, ak obrázok obsahuje málo rozdielnych objektov.

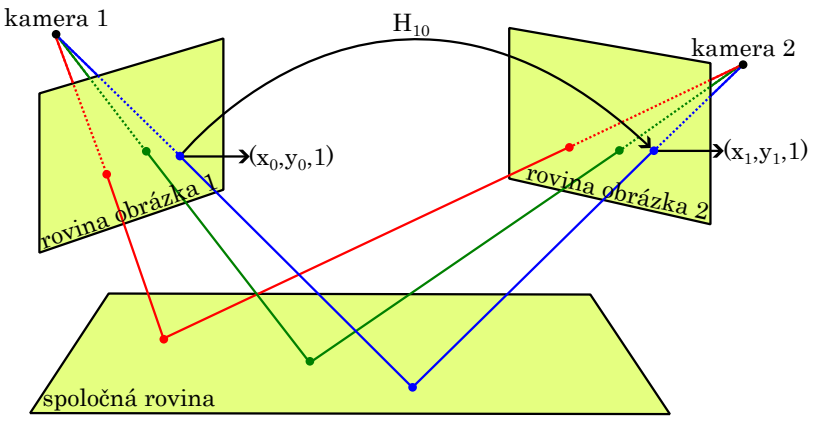
2.2 Určovanie homografie

Odhad homografie je technika používaná v počítačovom videní a spracovaní obrazu na zistenie vzťahu medzi dvoma obrazmi tej istej scény, ale zachytenými z rôznych uhlov pohľadu. Vymedzuje transformačný vzťah z jednej roviny do druhej (obrázok 2.3). Predstavuje kľúčový krok v množstve metód počítačového videnia, vrátane stabilizácie videa, spájanie obrazu, kalibrácia a rekonštrukcia polohy kamery, SLAM a mnohé iné. V kapitole 2.2 je

[1] *SSD – Sum of Squared Difference [TM_SQDIFF v opencv]*
[2] *NSD – Normalized Squared Difference [TM_SQDIFF_NORMED v opencv]*
[3] *NCC – Normalized Cross Corelation [TM_CCORR_NORMED v opencv]*

uvedený stručný prehľad najčastejšie používaných metodík. V ďalšej časti tejto kapitoly sú vymedzené rozdielne metodiky, ktoré sú podmienené využitou technológiou. Z hľadiska množstva zdrojov, delíme techniky odhadu homografie na:

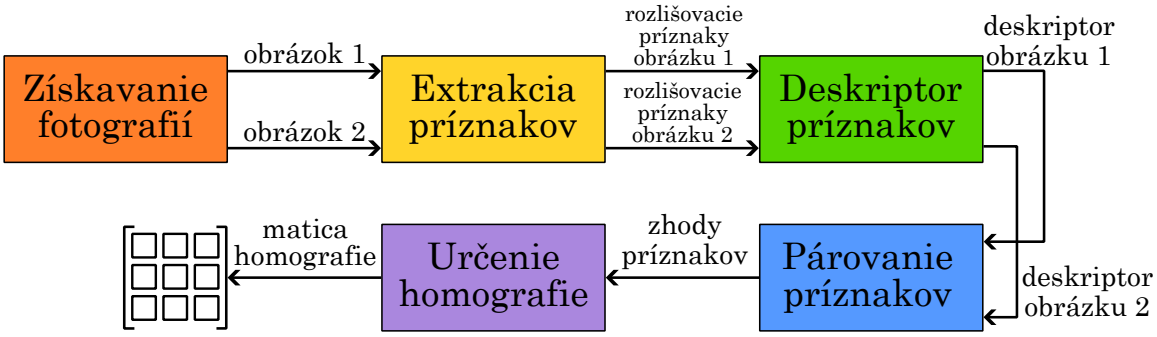
- **Jednozdrojové určovanie homografie** – pre získavanie homografie využíva len jedno a to isté zariadenie z rôznych uhlov pohľadu alebo v rôznom čase.
- **Viaczdrojové určovanie homografie** – pre určovanie homografie využíva množstvo zariadení, ktoré zahŕňajú spektrum rôznych technologických postupov. Patrí k nim fotografia vo viditeľnom spektre, infračervená fotografia, snímky LiDAR a ďalšie metodiky.



Obr. 2.3: Pozorovanie trojrozmerného bodu ležiaceho v rovine sú vzájomne viazané pomocou homografie, ktorá je definovaná pomocou posunu kamery a parametrov roviny. Myšlienka obrázka z [1].

Snímanie fotografií v tejto práci bolo vykonávané výlučne jedným zdrojom, preto sa tento dokument nebude zaoberať určovaním homografií odvodených z viacerých zdrojov. Jednozdrojové metódy je možno z hľadiska použitej technológie rozdeliť na:

- **Metódy založené na príznakoch (*feature-based*)** – najskôr identifikujú príznakové body v obraze pomocou algoritmu extrakcie príznakov, potom sa vypočíta metrika zhodnosti na účely párovania. Následne sa na určenie parametrov matice homografie použije vzťah mapovania medzi spárovanými príznakovými bodmi. Tieto metódy je možno ďalej rozdeliť na:
  - **Konvenčné** – Tradičné metódy sú rozdelené do troch hlavných krokov: detekcia príznakov, párovanie príznakov a určovanie matice homografie. Najskôr sa algoritmy extrakcie príznakov používajú na detekciu príznakových bodov v obraze a na extrakciu deskriptorov okolo týchto príznakových bodov, ktoré sú vo všeobecnosti reprezentované ako vektory. Následne sa vykoná párovanie príznakov. Na takto spárované príznaky sa následne zavolá niektorá z metód na odhad homografických parametrov.
  - **Metódy založené na učení** – využívajú neurónové siete, ktoré nahrádzajú detekciu, deskripciu alebo párovanie príznakov v tradičných algoritmoch. Tradičné metódy sa potom v ďalších krokoch používajú na odhad parametrov homografie. Medzi bežné deskriptory založené na učení patria LIFT [76], SuperGlue [58], či MatchFormer [66].



Obr. 2.4: Súslednosť krokov získavania homografie pre metódy založené na príznakoch. Obdĺžnik reprezentuje krok procesu a text nad šípkou reprezentuje výstup daného kroku.

- ***Deep-learning* metódy** – využívajú jednotný postup odhadu homografie, ktorý je spracovaný modelom hlbokej neurónovej siete. Sieť je schopná pochopiť a spracovať komplexné obrazové korešpondencie.

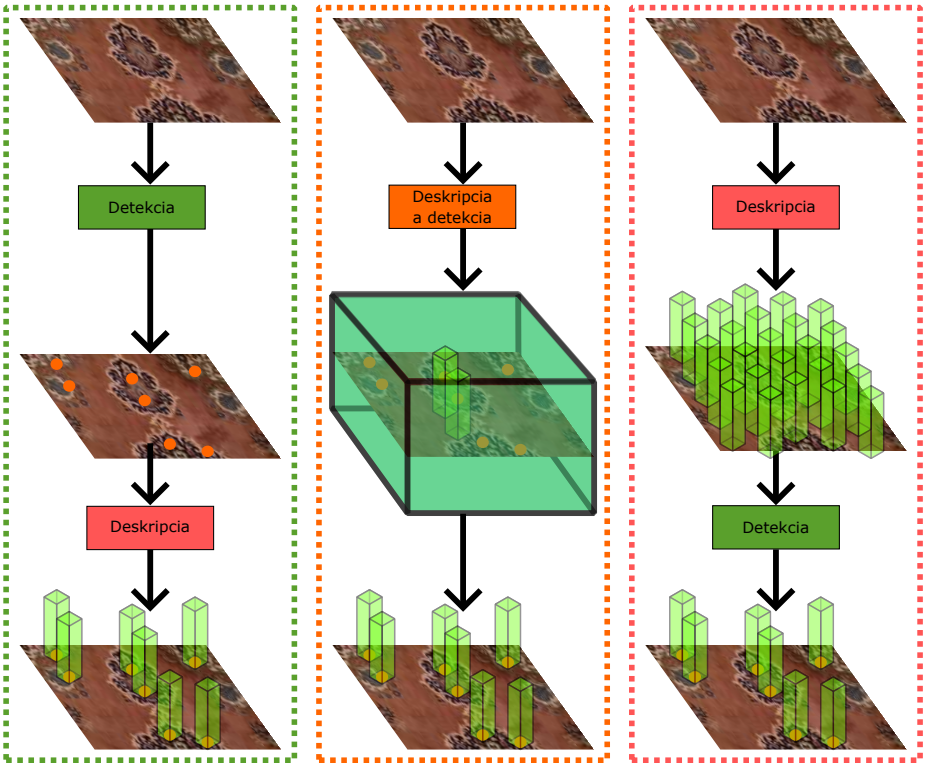
2.3 Metódy určovania homografie

Konvenčné metódy odhadu homografie sa zakladajú na manuálne navrhnutých extraktoroch príznakov. Celý proces určovania homografie konvenčným spôsobom môžeme rozdeliť na **detekciu príznakov**, **deskripciou príznakov**, **párovanie (*matching*) príznakov** a **určenie homografie** (obrázok 2.4).

2.3.1 Extrakcie príznakov (*feature extraction*)

Jedným z podstatných krokov pri určení homografie je proces extrakcie príznakov, ktorý zahŕňa detekciu a následnú deskripciu obrazových príznakov [68]. V konvenčnom prístupe má táto fáza veľký význam. Na určenie zhodných oblastí medzi obrazmi sa používajú len zistené príznaky, na rozdiel od metód porovnávania šablón, ktoré umožňujú výrazné zrýchlenie procesu registrácie obrazu. Príznaky pozorované v obraze môžu byť rohy, čiary, hrany, škvrny alebo iné štruktúry. Existujú tri hlavné prístupy k detekcii a deskripcií príznakov (bližšie znázornené na obrázku 2.5):

- **Detekcia a následná deskripcia (*Detect-then-Describe*)** – prístup lokálneho párovania príznakov, pri ktorom sa kľúčové body najprv detekujú na obrázkoch a následne vykoná deskripcia pomocou deskriptorov príznakov. Vyznačuje sa jasným oddelením fáz detekcie a deskripcie. Detekcia sa zvyčajne vykonáva identifikáciou charakteristických oblastí obrazu, ktoré možno spoľahlivo zistiť v rôznych pohľadoch na tú istú scénu. Po zistení kľúčových bodov sa z políčok so stredom na každom zistenom kľúčovom bode extrahujú deskriptory príznakov. Tieto deskriptory zachytávajú lokálny vzhľad v okolí kľúčových bodov a používajú sa na párovanie. Kľúčovými vlastnosťami *Detect-then-Describe* prístupu sú:
  - \+ jednoduché na pochopenie a ľahká implementácia,
  - \+ umožňuje použitie rôznych detektorov a deskriptorov,
  - − výkonnosť tejto metódy do veľkej miery závisí od účinnosti detektorov kľúčových bodov a deskriptorov prvkov,



Obr. 2.5: Porovnanie rôznych známych *pipeline* založených na detektore pre lokálne párovanie príznakov. Zelené ohraničenie označuje prístup detekcie a následnej deskripcie, oranžové označuje detekcia a deskripcia súčasne a červené označuje deskripcia k detekcii. Obrázok založený na [74].

  - − môže mať problémy v extrémnych podmienkach, ako sú výrazné zmeny uhla pohľadu a oblasti bez / so slabou textúrou.

  Reprezentatívnou metódou tohto prístupu je napríklad **SIFT** nižšie.

- **Detekcia a deskripcia (*Detect-and-Describe*)** – táto metóda spája úlohy detekcie a deskripcie kľúčových bodov v rámci jedného procesu. Integráciou týchto úloh môže model využívať obrazové príznaky nízkej aj vysokej úrovne, čo vedie k zlepšeniu efektívnosti pri párovaní príznakov. Tento prístup eliminuje obmedzenia oddelených detektorov kľúčových bodov a deskriptorov, najmä pri extrémnych zmenách, ako sú zmeny slnečného svitu alebo málo textúrované scény. Kľúčovými vlastnosťami *Detect-and-Describe* prístupu sú:

  - + spája úlohy detekcie a popisu kľúčových bodov v rámci jedného modelu,
  - + dokáže využívať obrazové príznaky nízkej aj vysokej úrovne, čo vedie k zlepšeniu výkonu pri porovnávaní príznakov,
  - − zložitosť týchto metód rastie s počtom kľúčových bodov, čo má negatívny vplyv na škálovateľnosť,
  - − vyžadujú dôsledný návrh, pre dosiahnutie rovnováhy medzi efektívnosťou a účinnosťou pri párovaní príznakov.

  Reprezentatívnou metódou tohto prístupu je napríklad **D2-Net** popísaný nižšie.

- **Deskripcia k detekcii (*Describe-to-Detect*)** – je prístup, v ktorom sa najprv vykoná opis lokálnych oblastí obrazu pomocou deskriptorov príznakov, po ktorom nasleduje detekcia kľúčových bodov na základe týchto deskriptorov. Metóda zahŕňa generovanie veľkého súboru hustých deskriptorov príznakov v celom obraze. Kľúčové body sa vyberajú z hustých deskriptorov na základe meraní nápadnosti. Proces určenia deskriptorov je oddelený od procesu detekcie, čo zvyšuje výkonnosť. Kľúčovými vlastnosťami *Describe-to-Detect* prístupu sú:

  + je možné zachytiť viac detailov vďaka generovaniu veľkého množstva hustých deskriptorov príznakov v celom obraze,
  + je možné využiť informácie o polohe kamery na učenie deskriptorov,
  – absencia detekcie príznakov môže mať za následok väčší prehľadávací priestor, čo môže zvýšiť výpočtovú zložitosť,
  – oddelenie detekcie a deskripcie môže viesť k neoptimálnemu výkonu.

  Reprezentatívnou metódou tohto prístupu je napríklad **D2D** popísaný nižšie.

V ďalšej časti tejto kapitoly budú predstavené reprezentatívne metódy z každého prístupu spomínaného vyššie:

SIFT

Detektor a deskriptor príznakov, ktorý vyvinul David Lowe v roku 1999 [45], ktorý je invariantný voči mierke a rotácii obrazu a odolný voči zmenám uhla pohľadu, šumu a osvetlenia. Algoritmus sa skladá z niekoľkých krokov:

1. **Detekcia extrémov v priestore mierky** – identifikácia potenciálnych bodov záujmu, v ktorých by algoritmus mohol nájsť príznaky. To sa dosiahne hľadaním extrémov (maximálnych a minimálnych bodov) v Gaussovej funkcii rozdielu (*difference-of-Gaussian*) mierky aplikovanej v priestore mierky na sériu vyhladených a prevzorkovaných obrazov.

2. **Lokalizácia kľúčových bodov** – po nájdení potenciálnych kľúčových bodov sa tieto body spresnia. Kľúčové body s nízkym kontrastom a kľúčové body s odozvou na okraje sa vyradia. Na získanie presnejšej lokalizácie extrémov sa používa Taylorov rad rozšírenia priestoru mierky, a ak je intenzita v tomto extréme menšia ako určitá prahová hodnota, je zamietnutý.

3. **Pridelenie orientácie** – je proces, pri ktorom algoritmus využíva okolité pixely daného kľúčového bodu na výpočet veľkosti a smeru gradientu. Tieto informácie sa potom použijú na vytvorenie orientačného histogramu pozostávajúceho z 36 binov, z ktorých každý predstavuje iný uhol orientácie (360 stupňov). Vrcholy v tomto histograme označujú dominantnú orientáciu kľúčového bodu, čo zabezpečuje, že algoritmus je nezávislý od rotácie.

4. **Deskripcia kľúčových bodov** – je vytvorenie charakteristického odtlačku pre každý kľúčový bod. Deskriptor SIFT je trojrozmerné pole obsahujúce 128 prvkov. To sa dosiahne rozdelením okolia 16x16 okolo kľúčového bodu na šestnásť čiastkových blokov 4x4. Pre každý podblok sa vytvorí osembodový orientačný histogram. Spojením týchto histogramov sa získa 128-rozmerný deskriptor SIFT.

D2-Net

Je konvolučná neurónová sieť, ktorá slúži ako deskriptor aj ako detektor príznakov súčasne [16]. Vyvinuli ju Mihai Dusmanu a kol. v roku 2019. Vďaka oneskoreniu fázy detekcie sú kľúčové body, ktoré identifikuje, stabilnejšie ako kľúčové body z tradičných metód. Tréning tohto modelu je možné vykonať pomocou korešpondencií pixelov z rozsiahlych rekonštrukcií *Structure-from-Motion* (SfM) bez potreby dodatočných anotácií. Zjednotenie úloh detekcie a opisu kľúčových bodov v rámci jednej siete umožnila zlepšiť porovnávanie príznakov v náročných snímacích podmienkach.

D2D

Je konvolučná neurónová sieť, ktorá vytvára husté deskriptory. Na základe informatívnosti a zreteľnosti týchto deskriptorov sa potom zisťujú kľúčové body. Bola navrhnutá Y. Tianom v roku 2020 [65]. Metóda zavádza dve miery nápadnosti pre kľúčové body: absolútnu nápadnosť, ktorá hodnotí informatívnosť deskriptora a relatívnu nápadnosť, ktorá hodnotí, ako je deskriptor diskriminačný v rámci svojho priestorového okolia. Táto metóda využíva predtrénované modely KNN, nevyžaduje si žiadne ďalšie učenie a možno ju použiť na akýkoľvek existujúci deskriptor založený na KNN. Zlepšuje výkonnosť párovania rôznych deskriptorov a je prispôsobiteľná rôznym úlohám a datasetom.

2.3.2 Párovanie príznakov (*feature matching*)

Po identifikácii príznakov prostredníctvom extrakcie je ďalším krokom zistenie zodpovedajúcich príznakov medzi jednotlivými snímkami. Tento proces, známy ako párovanie príznakov, zahŕňa hľadanie zhodných príznakov medzi snímkami [8]. Cieľom je identifikovať príznaky na jednom obrázku, ktoré zodpovedajú rovnakému reálnemu miestu na inom obrázku, pri rôznych pohľadoch na scénu alebo objekt (obrázok 2.6). Na riešenie tohto problému možno použiť napríklad tieto metódy:

Párovanie hrubou silou (*Brute Force (BF) matcher*)

Párovanie hrubou silou využíva proces porovnávania daného príznaku z prvej množiny so všetkými ostatnými príznakmi z druhej množiny. Toto porovnávanie sa vykonáva pomocou výpočtu vzdialenosti, pričom výsledkom je príznak z prvej množiny, ktorý je najbližšie ku všetkým ostatným príznakom z druhej množiny [1].

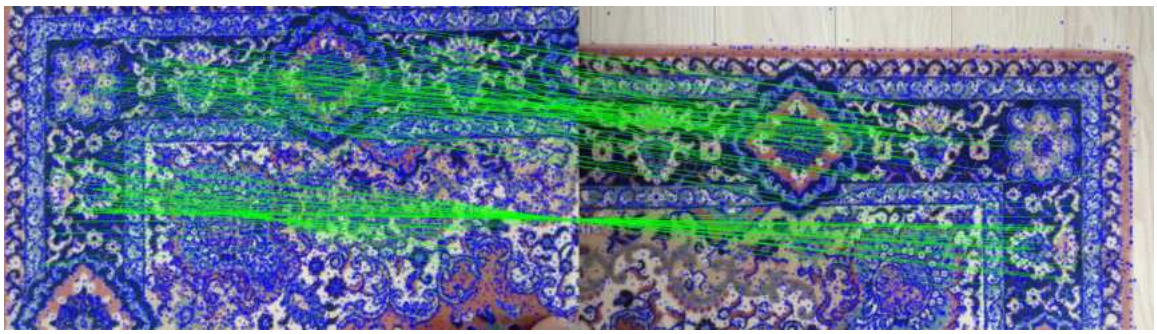
FLANN[4]

FLANN je to algoritmus na vykonávanie rýchleho približného vyhľadávania najbližších susedov (viac v kapitole 3.4.1). V porovnaní s vyhľadávaním hrubou silou [50]:

- \+ poskytuje rýchlejšie vyhľadávanie pomocou algoritmov, ako sú náhodné kd-stromy a hierarchické k-stromy,
- \+ ponúka množstvo algoritmov a parametrov na ladenie výkonu na základe špecifických vlastností datasetu,
- – výsledky vyhľadávania pomocou FLANN sú približné a nemusia mať takú presnosť ako v prípade párovača BF,

[4] **FLANN** – Fast Library for Approximate Nearest Neighbors

– pre dosiahnutie optimálneho výkonu je nutné správne zvoliť parametre a dobre poznať použité algoritmy.



Obr. 2.6: Ukážka párovania príznakov, pri ktorej sa mapujú a párujú zložité vzory na koberci. Zelené čiary symbolizujú zhody príznakov, zatiaľ čo modré kruhy symbolizujú samotné príznaky. Bol využitý extraktor príznakov SIFT (kapitola 2.3.1) a párovač príznakov FLANN (kapitola 2.3.2). Test pomeru bol úmyselne nastavený na 0.5 s cieľom zvýšenia viditeľnosti.

2.3.3 Určenie matice homografie (*homography matrix estimation*)

Pre dosiahnutie veľkej presnosti zarovnania je potrebné nájsť najvhodnejšie korešpondencie prvkov. Cieľom tohto kroku je vypočítať homografiu medzi dvoma obrazmi vzhľadom na súbor kandidátskych zhôd. To znamená, že by odľahlé prvky (*outliers*) mali byť eliminované. Najpoužívanejšími prístupmi pre riešenie tejto úlohy sú RANSAC a DLS. Následne je možné túto homografiu spresniť pomocou metódy, ako je napríklad metóda najmenších stredných štvorcov (LMS) alebo Houghova transformácia.

RANSAC (*RAndom SAmple Consensus*)

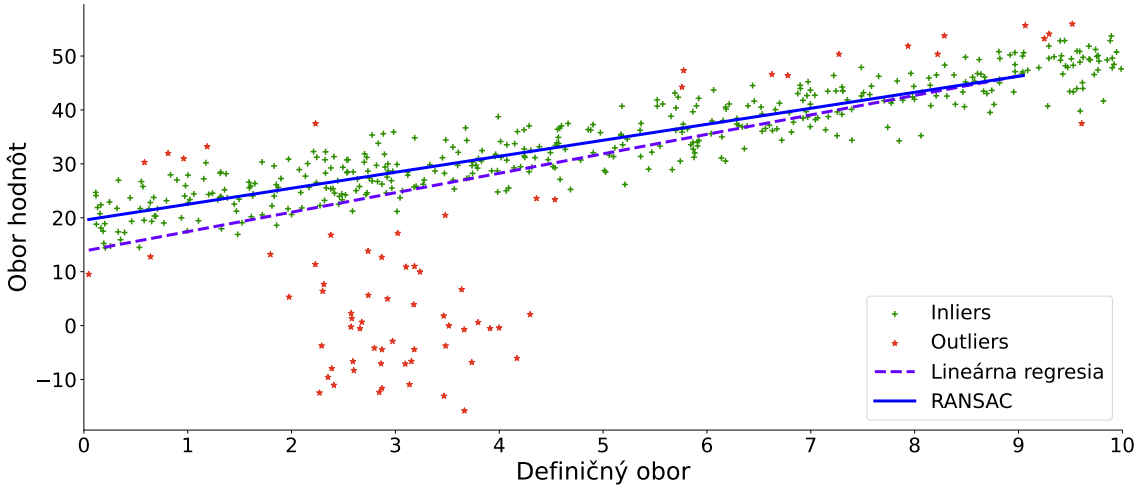
Algoritmus navrhnutý Martinom A. Fischlerom a Robertom C. Bollesom v roku 1981 je iteračná metóda na odhad matematického modelu zo z množiny dát obsahujúcej odľahlé hodnoty. Algoritmus RANSAC identifikuje odľahlé hodnoty v množine dát a odhaduje požadovaný model pomocou dát, ktoré neobsahujú odľahlé hodnoty (obrázok 2.7). Algoritmus sa používa na zabezpečenie robustnosti vzhľadom na možné chybné korešpondencie príznakov [18]. Algoritmus RANSAC pozostáva z nasledujúcich krokov:

1. Náhodne vyberte štyri páry zhodných bodov medzi dvoma obrazmi.
2. Výpočíta homografiu, ktorá najlepšie vyhovuje týmto dvojiciam bodov.
3. Určí *inliers*[5] určeného modelu. Pár sa považuje za súhlasný s homografiou, ak platí:

$$d(Hx, x') < t, \tag{2.4}$$

pričom $d(\dots)$ je euklidovská vzdialenosť, H je homografia, x a x' sú zodpovedajúce body na oboch obrázkoch a t je prahová hodnota.

[5] ***inliers*** – sú dáta, vyhovujúce modelu. Analogicky ***outliers*** sú dáta, ktoré modelu nevyhovujú.



Obr. 2.7: Graf znázorňuje výsledky lineárnej regresie a RANSAC na syntetických dátach. Súbor dát obsahuje hlavný prúd a niekoľko odľahlých hodnôt. Zelené krížiky predstavujú *inliers*. Červené hviezdičky predstavujú *outliers*. Fialová čiara predstavuje lineárny regresný model, zatiaľ čo tmavomodrá čiara predstavuje model RANSAC. Je zjavné, že model RANSAC je odolnejší voči odľahlým hodnotám a presnejšie modeluje hlavný prúd dát.

4. Uvedené kroky 1. – 3. by sa mali opakovať stanovený počet krát alebo dovtedy, kým sa nesplní zadaná podmienka. Tou môže byť napríklad identifikácia zadaného podielu odľahlých hodnôt alebo dosiahnutie zadanej úrovne spoľahlivosti.

5. Po nájdení najlepšej sady *inliers* sa prepočíta homografia pomocou všetkých týchto odľahlých hodnôt, aby bolo možné získať presnejší model.

V praxi sa prah vzdialenosti t volí empiricky tak, aby pravdepodobnosť, že ide o inlier, bola vysoká (napr. 0.95). Taktiež v praxi sa neskúšajú všetky možné vzorky, kvôli nákladnosti. Namiesto toho sa využije veľký počet vzoriek, tak aby aspoň jedna zo 4 náhodných vzoriek neobsahovala odľahlé hodnoty s vysokou pravdepodobnosťou (napr. 0.99). Existuje tiež pravidlo na ukončenie iterácií v prípade, že je veľkosť množiny zhôd podobná počtu *inliers*, o ktorých sa predpokladá, že sú v množine dát. Dôležitými vlastnosťami algoritmu RANSAC sú:

+ **Robustnosť** – je vysoko odolný voči odľahlým hodnotám,
+ **Adaptabilita** – počet dátových párov je nastavovaný adaptívne, keďže podiel odľahlých hodnôt (*outliers*) sa určuje z každého stavu zhody,
– **Nedeterministickosť** – keďže ide o nedeterministický algoritmus, nie je možné zaručiť, že vo všetkých prípadoch prinesie vyhovujúce výsledky,
– **Empirický prah** – Prah vzdialenosti t sa volí empiricky, čo nemusí byť vždy optimálne.

DLT (*Direct Linear Transform*)

Priama lineárna transformácia (DLT) je lineárny algoritmus, ktorý navrhli Hartley a Zisserman [27], počítajúci homografickú maticu z množiny zhôd medzi dvoma fotografiami.

V prípade líniových zhôd využíva na určenie homografie obmedzenia, poskytnuté zodpovedajúcimi dvojicami línií. Na vyriešenie ôsmich stupňov voľnosti v homografickej matici algoritmus vyžaduje aspoň štyri zodpovedajúce páry línií alebo bodov, z ktorých každý poskytuje dve obmedzenia. DLT je však citlivý na odľahlé hodnoty v dátach. Naopak, algoritmus RANSAC je voči odľahlým hodnotám odolnejší, pretože hľadá model, ktorý vyhovuje väčšine údajov, pričom odľahlé hodnoty ignoruje. Pri výpočte homografickej matice môže použiť DLT aj RANSAC. Keďže sa však často používajú na odlišné úlohy, je možné ich v rámci pipeline použiť spoločne [18, 27]. Vlastnosťami algoritmu DLT môžeme zhrnúť nasledovne:

+ **Efektívnosť** – DLT je jednoduchý a výpočtovo efektívny algoritmus, ktorý je vhodný pre aplikácie vyžadujúce výkon v reálnom čase,
+ **Všestrannosť** – DLT dokáže vypočítať homografickú maticu z množiny zhôd medzi dvoma fotografiami, pričom môže ísť buď o bodové, alebo líniové zhody,
– **Citlivosť na odľahlé hodnoty** – DLT je citlivý na odľahlé hodnoty v dátach, čo môže ovplyvniť presnosť odhadu homografie,
– **Nutnosť normalizácie** – Bežným postupom pri použití DLT je normalizácia údajov pred výpočtom homografie, čo pomáha zmierniť nepriaznivé účinky šumu a chýb kvantifikácie.

Najmenší medián štvorcov (*Least Meadian of Squares (LMS)*)

Algoritmus najmenších stredných štvorcov (LMS) bol pôvodne navrhnutý Peterom Rousseeuwom v roku 1984 [56]. Jedná sa o metódu na spresnenie odhadu homografie, najmä ak počiatočný odhad nie je dostatočne presný. Metóda LMS funguje na základe minimalizácie súčtu štvorcov zvyškov, čo sú rozdiely medzi pozorovanými a predpokladanými hodnotami závislej premennej. V kontexte odhadu homografie je závislou premennou poloha každého príznaku v jednom obraze a predpovedanou hodnotou je poloha príslušného príznaku v druhom obraze predpovedaná pomocou homografie [49]. Aj keď sa jedná o staršiu metódu, zostáva metóda LMS spoľahlivou voľbou. Vlastnosti metódy najmenšieho mediánu štvorcov (LMS) sú:

+ **Spresnenie** – môže účinne spresniť odhad homografie, v prípade ak počiatočný odhad nie je dostatočne presný. Často sa používa v kombinácii s inou metódou určovania homografie,
+ **Robustnosť** – je vysoko odolná voči odľahlým hodnotám, takže je vhodná pre súbory údajov so značným počtom odľahlých hodnôt. Stále však môže byť odľahlými hodnotami ovplyvnená,
+ **Flexibilnosť** – v praxi sa často používa v kombinácii s inými technikami (napríklad RANSAC), ktoré sú navrhnuté tak, aby účinne zvládali určenie homografie veľmi efektívne,
– **Výpočtová náročnosť** – pri veľkých datasetoch môže byť LMS výpočetne náročná,
– **Citlivosť na vysoko vplyvné body** – odolnosť voči odľahlým hodnotám nahrádza vyššou citlivosťou na body s vysokým vplyvom[6].

[6] **vysoko vplyvné body** – ide o body, ktoré majú potenciál výrazne ovplyvniť odhad regresných koeficientov.

Houghova transformácia

Houghova transformácia navrhnutá P. Houghom v roku 1962 [32] zohráva kľúčovú úlohu pri odhade homografie, najmä v kontexte rozpoznávania objektov v neprehľadných scénach. Identifikuje konzistentné zhluky kľúčových bodov, ktoré sa zhodujú v polohe, mierke a orientácii objektu v novom obraze. To sa dosahuje rýchlou identifikáciou vzorov v dátach, ktoré zodpovedajú špecifickému modelu – v tomto prípade modelu objektu definovaného jeho kľúčovými bodmi [6, 26]. Hlavné výhody Houghovej transformácie sú:

+ **Efektivita** – rýchlo identifikuje zhluky príznakov, ktoré patria k jednému objektu, čo je kľúčové pre výkon v reálnom čase,
+ **Robustnosť** – hľadá zhluky príznakov, ktoré sa zhodujú v určitých parametroch, čím sa znižuje vplyv odľahlých hodnôt,
+ **Verifikácia** – po zhlukovaní umožňuje ďalšie verifikovanie prostredníctvom metódy najmenších štvorcov na spresnenie odhadu polohy,
+ **Filtrácia** – metóda filtruje falošné zhody a potvrdzuje správne zhody, čo vedie k identifikácii s vysokou spoľahlivosťou,
– **Výpočtová náročnosť** – môže byť výpočtovo náročná, najmä pre vysokodimenzionálne priestory parametrov,
– **Citlivosť na šum** – môže byť citlivá na šum, pretože kľúčové body môžu byť tesne vedľa seba, čo vedie k nestabilite a falošným detekciám.

2.3.4 Metriky pre určenie presnosti homografie

Po získaní odhadu homografie je potrebné overiť, či je zvolená metóda odhadu homografie dostatočne presná. Je tiež potrebné skontrolovať, či zvolená metóda neobsahuje nejaké závažné nedostatky. Na zistenie týchto kľúčových informácií je nevyhnutné mať k dispozícii súbor presných a spoľahlivých hodnotiacich metrík. Tu sú niektoré z najznámejších:

- **ACE** [40] (***Average Corner Error***) – pozostáva z určenia euklidovskej vzdialenosti medzi pravdivými a odhadovanými rohovými polohami. Menšia hodnota ACE znamená vyššiu úspešnosť odhadu homografie. ACE je definovaná ako:

$$ACE = \frac{\sum_{i=1}^{4} ||x_i - y_i||_2}{4}, \tag{2.5}$$

  kde $||.||$ je euklidovská vzdialenosť, x_i a y_i reprezentujú skutočné a predpovedané súradnice rohov, pričom suma je vydelená 4, pretože homografia má 4 rohy.

- **RMSE** [78] (***Root Mean Square Error***) – je určená rozdielom medzi polohami mapovaných bodov a skutočných bodov po mapovaní bodov v jednom obraze do druhého pomocou odhadovanej homografickej matice (H_{est}). Menšia hodnota RMSE znamená vyššiu úspešnosť odhadu homografie. RMSE je definovaná ako:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|m_i' - H_{\text{est}} m_i\|_2^2}, \tag{2.6}$$

  kde (m_i, m_i') označuje skutočný zodpovedajúci bod, n je počet bodov a H_{est} je odhadovaná matica homografie.

- **PME** [46] (***Point Matching Error***) – je spriemerovaním euklidovských vzdialenosti medzi odhadovanými bodmi a cieľovými bodmi, pričom cieľový bod je často ručne anotovaná súradnica bodu. Menšia hodnota PME znamená vyššiu úspešnosť odhadu homografie. PME je definovaná ako:

$$PME = \frac{\sum_{i=1}^{N} ||x_i - y_i||_2}{N}, \tag{2.7}$$

kde x_i bod i po aplikácii transformácie homografie, y_i je bod anotácie zodpovedajúci bodu i a N je počet anotácií bodov.

2.4 Skreslenie (*warping*) obrazu

Skreslenie (*warping*) obrazu je kľúčovým krokom v procese zošívania obrazu. Po získaní homografickej matice je potrebné skresliť obrazy, aby sa zarovnali na účely zošívania. Hlavným cieľom tejto transformácie je zmena perspektívy obrazu. Obrázky mohli byť nesnímané z rôznych uhlov pohľadu, preto je nutné pred ich spájaním ich dobre zarovnať. Pre vykonanie tejto operácie sa využíva matica homografie získaná pomocou metód určenia homografie (kapitola 2.2). Aplikovaním tejto matice na pixely obrazu môžeme efektívne skresliť obraz tak, aby zodpovedal perspektíve iného obrazu. Výsledkom bude plynulý a vizuálne koherentný spojený obrázok [73].

2.5 Spájanie (*blending*) obrázkov

Spájanie (*blending*) obrazov je technika používaná v oblasti spracovania obrazu a počítačového videnia na účely vytvárania zložených obrazov miešaním dvoch alebo viacerých obrazov. Táto technika vo všeobecnosti vyžaduje extrakciu určitej oblasti, často objektu, zo zdrojového obrazu, ktorý sa potom nanáša na určené miesto na cieľovom obraze. Konečným cieľom tohto procesu je vytvoriť zložený obraz, kombináciou príslušných hodnôt pixelov vstupných obrazov, a to tak aby pôsobil prirodzene [80]. Významnou prekážkou, s ktorou sa pri tejto úlohe stretávame, je potenciálna nepresnosť pri vymedzení orezávaného regiónu. Táto nepresnosť spôsobuje, že je nutné vykonať dve korekcie:

1. Je potrebné upraviť vizuálne charakteristiky orezaného objektu, a tak zabezpečiť jeho kompatibilitu s novým pozadím.

2. Je nutné zaručiť bezproblémovú integráciu okraja orezania so zvyškom obrazu.

Ďalšie kapitoly pojednávajú o najdôležitejších zo spájacích metód metód:

2.5.1 *Alpha blending*

Alpha blending je technika kombinovania obrázka s pozadím, ktorá vytvára dojem čiastočnej alebo úplnej priehľadnosti. Často sa používa v aplikáciách na spracovanie obrazu na prekrytie obrázkov tak, aby bolo prostredníctvom efektu spájania vidieť popredie aj pozadie. Táto technika využíva „alfa“ kanál, čo je špeciálny kanál reprezentujúci úroveň nepriehľadnosti farby v obraze. Z matematického hľadiska možno alpha blending popísať pomocou nasledujúceho vzorca:

$$C_{\text{result}} = \alpha \cdot C_{\text{foreground}} + (1 - \alpha) \cdot C_{\text{background}}, \tag{2.8}$$

kde C_{result} je výsledná farba po zmiešaní, $C_{\text{foreground}}$ je farba obrázka popredia, $C_{\text{background}}$ je farba obrázka na pozadí a α je hodnota alfa, ktorá sa pohybuje od 0 (úplná priehľadnosť) do 1 (úplná nepriehľadnosť).

2.5.2 *Multiband blending*

Multiband blending je technika používaná pri spracovaní obrazu na dosiahnutie plynulej kombinácie viacerých obrazov, pričom zachováva dôležité vlastnosti a detaily oboch obrazov. Vyvinuli ju P. Burt a E. Adelson v roku 1983 [9]. Zahŕňa rozklad obrazov na viacero frekvenčných pásiem a ich samostatné prelínanie pred ich opätovným spojením do konečného obrazu. Tento proces možno zhrnúť takto:

1. Rozklad obrázkov, ktoré sa majú zmiešať, na súbor zložiek filtrovaných pásmovou priepustnosťou pomocou pyramídy s viacerými rozlíšeniami.

2. Zostavenie zložkových obrazov v každom priestorovom frekvenčnom pásme do príslušných pásmových mozaík ich spojením pomocou váženého priemeru v rámci prechodovej zóny úmernej vlnovým dĺžkam zastúpeným v pásme.

3. Skombinujte tieto mozaikové obrazy s pásmovou priepustnosťou ich sčítaním, aby sa získala konečná obrazová mozaika, pričom sa zabezpečí, aby *spline* zodpovedal mierke prvkov v samotných obrazoch.

4. Zabezpečí, aby sa hrubé príznaky, ktoré sa vyskytujú v blízkosti hraníc, postupne prelínali na relatívne veľkú vzdialenosť bez rozmazania alebo zhoršenia jemnejších detailov obrazu v blízkosti hraníc.

Táto metóda je účinná najmä vtedy, keď sa na miešaných obrazoch vyskytujú rôzne množstvá detailov v rôznych mierkach. Proces miešania každého pásma zvlášť umožňuje zachovať veľké prvky a zároveň umožňuje zachytiť malé detaily, ktoré by inak mohli byť stratené.

2.5.3 *Poisson blending*

Poisson blending je technika používaná v počítačovej grafike a spracovaní obrazu na plynulú úpravu oblastí obrazu. Zahŕňa riešenie Poissonových rovníc na interpoláciu oblastí obrazu s plynulými prechodmi [63]. Navrhol ju P. Pérez a kol. v roku 2003 [53]. Pozostáva z nasledujúcich krokov:

1. Vypočet gradientu (rýchlosť zmeny intenzity) zdrojového aj cieľového obrazu v oblasti prelínania. Gradient vyjadruje, ako rýchlo sa menia hodnoty pixelov v rôznych smeroch.

2. Riešenie Poissonovej rovnice pre hodnoty pixelov zdrojového obrazu v oblasti prelínania. Poissonova rovnica zabezpečuje, aby sa gradient zdrojového (zmiešaného) obrazu zhodoval s gradientom cieľového obrazu v tej istej oblasti. To sa dosiahne minimalizáciou rozdielu medzi gradientmi zdrojového a cieľového obrazu.

3. Úprava hodnôt pixelov zdrojového obrazu podľa riešenia Poissonovej rovnice na získanie plynulo spojeného výsledného obrazu. To sa vykoná pridaním diferencie medzi pôvodným a zrekonštruovaným gradientom k cieľovému obrazu.

Poissonova rovnica sa rieši pre každý farebný kanál zvlášť. Táto metóda funguje najlepšie, keď majú zdrojové a cieľové obrázky podobné rozloženie farieb. Ak tomu tak nie je, pred spájaním sa môžu použiť metódy prenosu farieb, aby boli zdrojové a cieľové obrazy kompatibilné.

2.6 Nástroje pre prácu s počítačovým videním

Počítačové videnie je oblasť, ktorá zaznamenala významný pokrok vďaka vývoju rôznych nástrojov a knižníc. Nástroje počítačového videnia možno všeobecne rozdeliť do troch typov:

- **Knižnice pre spracovanie obrázkov a videa** – poskytujú základné možnosti spracovania obrazu a videa. Príklady zahŕňajú OpenCV, PIL a scikit-image.
- **Knižnice pre hlboké učenie** – používajú sa pre trénovanie a nasadzovanie modelov hlbokého učenia, ktoré sú kľúčové pre mnohé úlohy počítačového videnia. Medzi príklady patria TensorFlow a PyTorch.
- **Iné špecializované knižnice** – poskytujú špecializované algoritmy a techniky pre počítačové videnie. Príkladom sú napríklad Dlib a SimpleCV.

Každý z týchto nástrojov má svoje silné a slabé stránky a vhodný výber závisí od konkrétnych požiadaviek na danú úlohu. Po dôkladnom zvážení boli pre túto úlohu vybrané nástroje OpenCV (kapitola 2.6.1) a PyTorch (kapitola 2.6.2). OpenCV je *open-source* softvérová knižnica pre počítačové videnie a strojové učenie, poskytujúca mnoho algoritmov, ktoré možno použiť na rôzne úlohy. Knižnica bola zvolená najmä pre svoju spoľahlivosť a rozšírenosť v oblasti počítačového videnia. PyTorch je známy svojou jednoduchosťou, ľahkým používaním, flexibilitou, efektívnym využívaním pamäte a dynamickými výpočtovými grafmi.

2.6.1 OpenCV

OpenCV[7] je výkonná knižnica určená na počítačové videnie v reálnom čase. Je zložená z vysoko optimalizovaných funkcií jazyka C a niekoľkých tried jazyka C++, ktoré implementujú mnohé populárne algoritmy spracovania obrazu, detekcie objektov, rozpoznávanie tvárí, strojového učenia, či počítačového videnia. Základná štruktúra OpenCV je rozdelená do modulov, z ktorých každý sa zameriava na rôzne aspekty počítačového videnia.

- **core** – definuje základné dátové štruktúry vrátane hustého viacrozmerného poľa Mat a základné funkcie, ktoré používajú všetky ostatné moduly.
- **imgproc** – zahŕňa lineárne a nelineárne filtrovanie obrazu, geometrické transformácie obrazu, konverziu farebného priestoru, histogramy atď.
- **video** – Tento modul zahŕňa algoritmy na odhad pohybu, odčítanie pozadia a sledovanie objektov.
- **calib3d** – obsahuje základné algoritmy geometrie viacerých pohľadov, kalibráciu jednej a stereokamery, odhad polohy objektu, algoritmy stereo korešpondencie a prvky 3D rekonštrukcie.

[7] Open Source Computer Vision Library

- **features2d** – obsahuje detektory významných prvkov, deskriptory a porovnávače deskriptorov.
- **objdetect** – zahŕňa detekciu objektov a inštancií preddefinovaných tried.
- **highgui** – poskytuje ľahko použiteľné rozhranie pre jednoduché možnosti používateľského rozhrania.
- **videoio** – poskytuje ľahko použiteľné rozhranie pre zachytávanie videa a video kodeky.

OpenCV podporuje beh na CPU aj GPU, vďaka čomu je veľmi univerzálny pre rôzne hardvérové konfigurácie. Poskytuje tiež rozhrania pre jazyky ako C++, Python a Java. Základná štruktúra OpenCV je rozdelená do modulov, z ktorých každý sa zameriava na rôzne aspekty počítačového videnia:

2.6.2 Pytorch

Je *open-source*, optimalizovaná tenzorová knižnica na hlboké učenie pomocou GPU a CPU, poskytujúca dve high-level funkcie:

- Výpočet tenzorov (podobne ako NumPy) so silnou akceleráciou prostredníctvom GPU.
- Hlboké neurónové siete vybudované na základe páskového systému autograd.

PyTorch vyvinulo laboratórium AI Research spoločnosti Facebook (FAIR). Na rozdiel od iných knižníc, ktoré predkompilujú statický graf, využíva PyTorch dynamickú výpočtovú grafovú štruktúru, známu ako dynamická neurónová sieť (DNN), čo umožňuje konštrukciu a manipuláciu s grafmi za behu. PyTorch sa skladá z niekoľkých hlavných komponentov, z ktorých každý slúži na iný účel:

- **torch** – základný *namespace*. Obsahuje všetky funkcie a triedy pre tenzorové operácie. Tenzory v prostredí PyTorch sú analogické k ndarrays v NumPy, pričom ich možno využiť aj na GPU pre urýchlenie výpočtov.
- **autograd** – poskytuje triedy a funkcie implementujúce automatickú diferenciáciu ľubovoľných skalárnych funkcií. Požiadavkou je, že tenzory, ktorých gradienty sa majú vypočítať, musia vyžadovať gradienty.
- **torch.jit** – Kompilačný zásobník (TorchScript) na vytváranie serializovateľných a optimalizovateľných modelov z kódu PyTorch.
- **torch.nn** – poskytuje základné komponenty potrebné na konštrukciu a trénovanie neurónových sietí. Definuje súbor modulov analogických k vrstvám neurónových sietí.
- **torch.multiprocessing** – uľahčuje distribuované trénovanie a umožňuje paralelne vykonávať výpočty.
- **torch.utils** – obsahuje pomocné triedy, ako sú napríklad *data loaders*, *trainers* a mnohé ďalšie pomocné funkcie, ktoré možno použiť pri trénovaní neurónovej siete.

Kapitola 3

Lokalizácia fotografie v obrovskej fotografii

3.1 Známe prístupy určovania lokalizácie

Identifikácia menšieho obrázka v rámci väčšieho obrázka sa dá zjednodušiť na otázku určenia homografie menšieho obrazu (*query*) vzhľadom na väčší obraz (mapu). Existujú rôzne metódy na riešenie tohto problému, ktoré sa dajú rozdeliť do troch kategórií: **konvenčné metódy**, ***learning-based*** prístupy a ***deep-learning*** metódy.

Konvenčné metódy sa skladajú z extrakcie a párovania príznakov, po ktorých nasleduje určenie matice homografie. Medzi najznámejšie extraktory patrí SIFT[1] [45]. Yan a kol. zaviedli metódu HEASK[2] [75], ktorá integruje metódy založené na pixeloch a príznakoch. Suárez a kol. predstavili BEBLID[3] [62], binárny deskriptor určený pre zariadenia s obmedzenými výpočtovými zdrojmi.

Learning-based prístupy využívajú neurónové siete na nahradenie extrakcie alebo porovnávania príznakov v tradičných algoritmoch. Príkladom je LIFT[4] [76], ktorý integruje detekciu príznakov, odhad orientácie a deskripcia do jednej diferencovateľnej *pipeline*. Sarlin a kol. navrhli SuperGlue[5] [58], neurónovú sieť, ktorá porovnáva dve sady lokálnych príznakov. MatchFormer[6] [66] je prístup na párovanie medzi jednotlivými sekvenciami.

Deep-learning metódy sú určené pre celé určenie homografie. Medzi supervised metódy patrí Iteratívna homografická sieť (IHN)[7] [10] a RHWF[8] [11], obidve navrhnuté Cao a kol. IHN je architektúra hĺbkového odhadu homografie, ktorá iteračne spresňuje odhad pomocou trénovateľného iterátora s viazanými váhami. RHWF je rámec pre rekurentný odhad homografie, ktorý využíva homograficky riadené deformovanie obrazu a transformátor zamerania (FocusFormer).

[1] **SIFT** – Scale-Invariant Feature Transform
[2] **HEASK** – Homography Estimation Algorithm based on SIFT and KNN
[3] **BEBLID** – Boosted Efficient Binary Local Image Descriptor
[4] **LIFT** – Learned Invariant Feature Transform
[5] **SuperGlue** – SuperGlue: Learning Feature Matching with Graph Neural Networks
[6] **MatchFormer** – MatchFormer: Feature-Matching Transformer for Image Matching
[7] **IHN** – Iterative Homography Network
[8] **RHWF** – Recurrent Homography Estimation Using Homography-Guided Image Warping and Focus Transformer

Medzi unsupervised metódy patrí HVC-Net[9] [79], ktorý využíva hierarchickú konsenzuálnu sieť na odhad homografie a MS2CA-HENet[10] [31], ktorý integruje viacmierkovú a viackanálovú *attention* do odhadu homografie.

GeoFormer[11] [44] je príkladom self-supervised prístupu. Je to transformátor, ktorý je citlivý na geometriu a je trénovaný v režime seba-supervízie.

3.2 Neurónové siete

Množstvo obrazových dát, ktoré je možné v súčasnosti získať, vyššia výpočetná sila počítačov sú hlavnými faktormi, ktoré prispeli k rozvoju prístupov založených na dátach (*data-driven* prístupy). Tento trend je spôsobený najmä účinnosťou týchto prístupov. Keďže klasické techniky počítačového videnia často nedokážu pracovať s komplexnými dátami, potrebujú samostatný extraktor čŕt a aj tak často nedosahujú takých vysokých presností ako data-driven prístupy.

Strojové učenie tento princíp využíva. Na základe dát a očakávaných odpovedí vytvorí pravidlá. Tieto pravidlá je potom možné aplikovať na nové nevidené dáta s cieľom získať pôvodné odpovede [19]. V súčasnosti je najčastejším prístupom strojového učenia deep learning. Jedná sa špecifickú podoblasť strojového učenia, ktorá obsahuje veľké množstvo vnútorných vrstiev vrstiev, pričom počet týchto vrstiev určuje hĺbku modelu. Najdôležitejšie poznatky z histórie neurónových sietí priblíži nasledujúca kapitola.

3.2.1 Zlomové míľniky v oblasti neurónových sietí

Počiatky neurónových sietí datujeme už do 40. rokov 20. storočia. Bol vytvorený model fungovania mozgových neurónov – preceptrón [48]. Je to matematický model, ktorý simuluje fungovanie biologického neurónu. F. Rosenblatt [55] previedol túto myšlienku do hardware implementácie pod názvom „Mark 1 Perceptron“. Pri práci v počítačovom videní vyvinul K. Fukushima Neocognitron [21]. Jednalo sa o prvý návrh modelu hlbokého učenia pomocou konvolučnej neurónovej siete. V polovici 80. rokov prišiel P. Werbos s návrhom, aby sa spätné šírenie (*backpropagation*) [72] použilo pre umelé neurónové siete. V roku 1989 Y. LeCun [41] vytvoril fungujúci systém pre rozpoznávanie ručne písaných poštových smerovacích čísel, využívajúci spätné šírenie a neurónové siete. S. Hochreiter a J. Schmidhuber [30] predstavili LSTM[12], rekurentnú neurónovú sieť, so schopnosťou učiť sa a pamätať si dlhé sekvencie. V roku 2012 došlo k veľkému prielomu, keď A. Krizhevsky a kol. [39] vyvinuli konvolučnú neurónovú sieť AlexNet, ktorá v súťaži ImageNet Large Scale Visual Recognition Challenge preukázala lepší výkon ako všetci predchádzajúci konkurenti. V roku 2014 Ian Goodfellow a kol. predstavili generatívne adverzné siete (GAN) [25]. Jedná sa o triedu strojového učenia, v ktorých dve neurónové siete, generátor a diskriminátor, medzi sebou súťažia, pričom generátor sa snaží generovať dáta, ktoré sú nerozlíšiteľné od skutočných dát, čo sťažuje prácu diskriminátora, ktorý sa snaží rozlišovať medzi generovanými dátami a skutočnými dátami. K. He a kol. [28] v roku 2015 vyvinuli reziduálne siete (ResNet). Reziduálne siete predstavujú špecifický typ modelu hlbokého učenia, ktorý využíva „preskokové spojenia“ alebo „skratky“ na obchádzanie určitých vrstiev, čím zmierňuje problém miznúceho gradientu a umožňuje trénovanie podstatne hlbších sietí. Ďalším míľnikom v

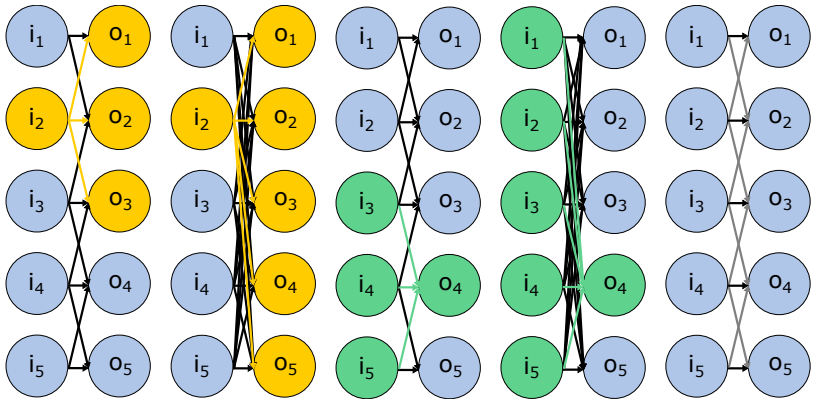
[9] **HVC-Net** – Hierarchical Voting Consensus Network
[10] **MS2CA-HENet** – Multi-Scale and Multi-Channel Attention Homography Estimation Network
[11] **GeoFormer** – Geometry-Aware Self-Supervised Transformer
[12] **LSTM** – Long Short-Term Memory

oblasti neurónových sietí návrh modelu Transformer od A. Vaswani a kolektívu, ktorý slúži ako základ mnohých špičkových modelov v oblasti spracovania prirodzeného jazyka (BERT, GPT). V súčasnosti sa však tento typ neurónovej siete využíva aj v oblasti počítačového videnia. ViT od A. Dosovitskiy a kol. [13] obrázkami zaobchádza ako so sekvenciami pixlov, analogicky k tomu, ako sa so slovami vo vete zaobchádza pri spracovaní prirodzeného jazyka.

3.3 Konvolučné neurónové siete

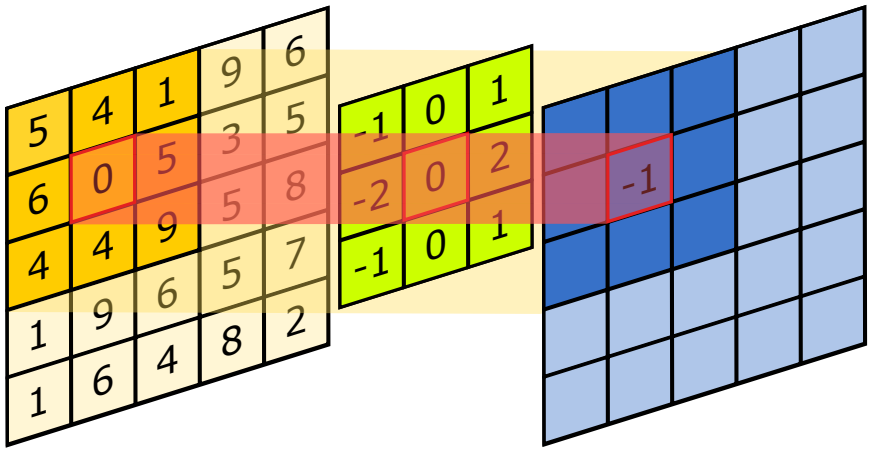
Konvolučné neurónové siete (CNN) sú triedou modelov hlbokého učenia, ktoré sa používajú najmä na spracovanie obrazu a úlohy počítačového videnia. Sú navrhnuté tak, aby sa automaticky adaptívne učili priestorové hierarchie príznakov z obrazov alebo videa. Konvolúcia (obrázok 3.2) využíva štyri dôležité myšlienky, ktoré môžu pomôcť zlepšiť systém strojového učenia [24]:

- **Riedke interakcie** – neurónové siete nemusia pre každý výstup použiť všetky svoje vstupy súčasne. Každá jednotka siete pripojená len k malej oblasti vstupu.
- **Zdieľanie parametrov** – neurónové siete používajú rovnaký parameter pre viac ako jednu funkciu v modeli (redukuje pamäťovú stopu) (obrázok 3.1).
- **Ekvivariantné reprezentácie** – zmena vstupu, spôsobí reprezentácie rovnakým spôsobom, čo pomáha sieti rozpoznávať objekty bez ohľadu na ich polohu alebo orientáciu v obraze.
- **Možnosť pracovať s vstupmi s premenlivou veľkosťou** – konvolúcia nie je závislá od veľkosti vstupu.



Obr. 3.1: V prípade, že vstupný neurón i_3 vznikne konvolúciou s jadrom so šírkou 3, tak ovplyvní iba tri výstupy (o_1, o_2, o_3). Ak je však operáciou násobenie matíc, vzniká husté (*dense*) spojenie a ovplyvnené sú všetky výstupy. Ak o_4 vzniká konvolúciou so šírkou 3 vplyv majú iba vstupy i_3, i_4, i_5 (recepčné pole). Ak je operáciou násobenie matíc spojitosť už nie je riedka, takže všetky vstupy ovplyvňujú výstup. Posledná časť obrázka znázorňuje zdieľanie parametrov. Jediný parameter používa na všetkých vstupných miestach (čierna šípka).

V nasledujúcich kapitolách budú vysvetlené najdôležitejšie termíny konvolučných neurónových sietí, viažuce sa k tejto práci. Kapitola 3.3.1 opisuje základné vrstvy, ktoré sa



Obr. 3.2: Vizuálne znázornenie operácie konvolúcie. Žltá matica znázorňuje maticu mapy príznakov (vstupnú maticu), na ktorú sa aplikuje operácia konvolúcie. Zelená matica znázorňuje konvolučné jadro. Ide o menšiu maticu, ktorá sa používa pre operáciu konvolúcie. Je posúvaná nad vstupnou maticou. Každý prvok konvolučného jadra interaguje s jedným prvkom mapy príznakov. Modrá matica znázorňuje výsledok operácie konvolúcie. Prvok matice predstavuje výsledok jednej konvolučnej operácie na vstupe.

nachádzajú vo väčšine sieťových architektúr. V kapitole 3.3.2 je vysvetlená úloha aktivačných funkcií a sú tiež objasnené najznámejšie z nich. Priblíženie štruktúry reziduálnych neurónových sietí, vyžitých v tejto práci prináša kapitola 3.3.5. Táto práca sa obmedzuje na skúmanie učenia s učiteľom, pretože iné paradigmy učenia sú mimo rozsahu tejto práce.

3.3.1 Vrstvy konvolučných neurónových sietí

Základnou jednotkou neurónovej siete je vrstva, ktorá slúži ako modul na spracovanie dát. Túto vrstvu si možno predstaviť ako filter dát, pričom vstupom je jeden alebo viacero tenzorov[13], ktoré z nej vychádzajú ako jeden alebo viacero tenzorov v užitočnejšej forme. Dochádza k extrakcii reprezentácie vstupných dát s cieľom získať ich reprezentáciu, ktorá je pre daný problém zmysluplnejšia. Väčšina techník hlbokého učenia spočíva v zreťazení jednoduchých vrstiev, ktoré vykonávajú určitú formu postupnej filtrácie [19]. Nasledujúca časť kapitoly priblíži najdôležitejšie z vrstiev.

- **Konvolučná vrstva** – Základný stavebný prvok konvolučných neurónových sietí. Aplikuje konvolúciu (obrázok 3.2) na vstupný signál zložený z niekoľkých vstupných rovín. Operáciu konvolúcie možno vyjadriť v rámci jednej pozície

$$S(i,j) = \sum_m \sum_n I(i-m, j-n) \cdot K(m,n), \tag{3.1}$$

pričom $S(i,j)$ je výstup konvolúcie, $I(i-m, j-n)$ predstavuje vstupný obraz a $K(m,n)$ je jadro. Suma je vypočítaná zo všetkých platných hodnôt m a n. Veľkosť jadra je zvyčajne relatívne malá hodnota. Hodnoty v konvolučných jadrách sú učené pomocou optimalizátora (kapitola 3.3.4). Okrem vstupných a výstupných kanálov a veľkosti jadra majú na konvolučnú vrstvu významný vplyv ďalšie tri hlavné parametre[14]:

[13] **tenzor** – pole čísel usporiadaných do pravidelnej mriežky s premenlivým počtom osí [24].
[14] https://pytorch.org/docs/stable/index.html

  - **stride** – počet pixelov, o ktoré sa konvolučné jadro posunie cez vstupnú maticu.
  - **padding** – pridávanie vrstiev núl do vstupnej matice s cieľom zachovať priestorové rozmery výstupu.
  - **dilation** – riadi rozstupy medzi hodnotami v jadre, čo umožňuje sieti pracovať vo viacerých mierkach a mať širšie zorné pole.

- **Plne prepojená vrstva** – Vrstva, v ktorej každý neurón je prepojený so všetkými neurónmi predchádzajúcej vrstvy (*fully connected – FC*). Táto vrstva sa zvyčajne umiestňuje na koniec architektúry konvolučnej neurónovej siete. Slúži ako klasifikátor, ktorý preberá high-level príznaky naučené konvolučnými vrstvami a používa ich na určenie konečnej výstupnej triedy [3, 19].

3.3.2 Aktivačné funkcie

Aktivačná funkcia je základnou zložkou učenia neurónových sietí, ktorá umožňuje modelu pochopiť aj nelineárne funkcie. Bez tejto schopnosti by model nebol schopný pochopiť vzájomný vzťah medzi akýmikoľvek dvoma vstupnými premennými. Začlenením nelinearity do modelu môžeme rozšíriť jeho schopnosť reprezentovať nelineárne funkcie. To sa dosiahne použitím nelineárnej transformácie na vstupné dáta [24]. Nasledujúca časť približuje aktivačné funkcie spomenuté ako perspektívne v štúdií [42] a funkciu sigmoid (obrázok 3.3):

1. **softsign** – predstavuje alternatívu k hyperbolickému tangensu (*tanh*) v rámci triedy sigmoidných funkcií. Bola predstavená v roku 1993 D. Elliottom [17]. V porovnaní s *tanh* je jeho vyhodnocovanie relatívne nenáročné, pretože neobsahuje exponenciálne funkcie. Táto funkcia je definovaná pomocou rovnice

$$\text{softsign}(x) = \frac{x}{1 + |x|} \quad \text{H(f): } (-1, 1) \tag{3.2}$$

2. **Swish** – je interpoláciu medzi ReLU a škálovanou lineárnou. Minimum funkcie je $ca \approx -0,278/a$ pre a $\neq 0$ a $ca = \infty$ pre $a = 0$. Meno Swish funkcia dostala v roku 2017 [54]. Štúdia [60] uvádza, že funkcia swish pomáha zmierniť problém miznutia gradientu počas spätného šírenia. Táto funkcia je definovaná pomocou rovnice

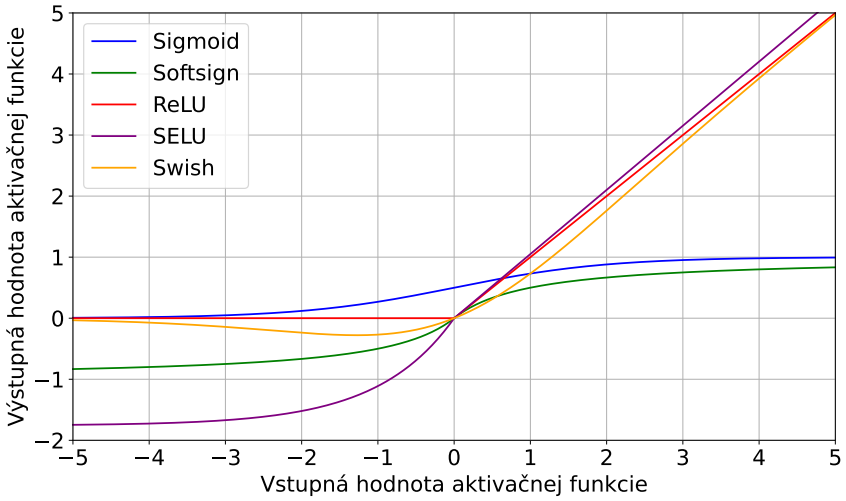
$$\text{Swish}(x) = x\,\text{sigmoid}(ax) = \frac{x}{1 + e^{-ax}} \quad \text{H(f): } < -c_a, \infty) \tag{3.3}$$

3. **ReLU** (Rektifikovaná lineárna jednotka) – je aktivačná funkcia, ktorá je definovaná kladnou časťou svojho vstupu. Umožňuje, aby kladné vstupy zostali nezmenené, zatiaľ čo záporné vstupy sú mapované na nulu. Funkcia ReLU a jej derivácie sú výpočtovo efektívne na implementáciu. Vynašiel ju K. Fukushima v roku 1969 [20]. Táto funkcia je definovaná pomocou rovnice

$$\text{ReLU}(x) = \begin{cases} 0, & \text{ak } x \leq 0 \\ x, & \text{ak } x > 0 \end{cases} \quad \text{H(f): } < 0, \infty) \tag{3.4}$$

4. **SELU** – je aktivačná funkcia, so samonormalizáciou (*self-normalisation*),

$$\text{SELU}(x) = \begin{cases} \lambda x, & \text{ak } x > 0 \\ \lambda\alpha(e^x - 1), & \text{ak } x \leq 0 \end{cases} \quad \text{H(f): } < -\alpha, \infty) \tag{3.5}$$



Obr. 3.3: Porovnanie aktivačných funkcií.

pričom $\lambda \approx 1.0507$ a $\alpha \approx 1.6732$ sú preddefinované škálovacie parametre. Keďže hodnota funkcie môže klesnúť pod 0, táto funkcia môže konvergovať rýchlejšie ako ReLU. Výpočtová zložitosť výpočtu exponenciálnej funkcie však znamená, že preferovanou je stále ReLU. S myšlienkou tejto funkcie prišiel G. Klambauer v roku 2017 [37]. Táto funkcia je definovaná pomocou rovnice (3.5).

5. **sigmoid** (logistic) – transformuje svoj vstup na rozsah medzi 0 a 1, čo sa využíva najmä v problémoch binárnej klasifikácie, kde sa výstup neurónovej siete interpretuje ako pravdepodobnosť. V praxi sa však bežne nepoužíva kvôli problémom, ako sú miznúce gradienty. Založená na základe preceptrónu od F. Rosenblatta [55]. Táto funkcia je definovaná pomocou rovnice

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad \text{H(f): } (0, 1) \tag{3.6}$$

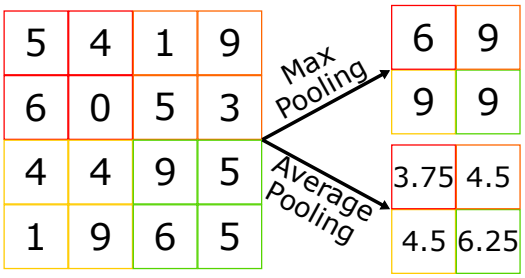
3.3.3 Funkcia pooling-u

Pooling nahrádza výstup siete v určitom mieste súhrnnou štatistikou blízkych výstupov. Zvyčajne sa aplikuje po aktivačnej funkcii. Hlavným cieľom pooling-u je zmenšiť priestorové rozmery ($h \times w$) vstupného objemu pre následnú konvolučnú vrstvu bez toho, aby sa ovplyvnila hĺbka. Najrozšírenejšie formy pooling-u (obrázok 3.4) sú:

- **Max pooling** – Vypočíta maximálnou hodnotou okolia zvolenej obdĺžnikovej veľkosti (typicky 2×2), teda zachováva najvýznamnejšie prvky vstupu.
- **Average pooling** – Vypočíta maximálnou hodnotou okolia zvolenej obdĺžnikovej veľkosti, teda získa hladšiu verziu vstupu s nižším rozlíšením.

3.3.4 Optimalizácia na báze gradientu a stratové funkcie

V kontexte hlbokého učenia majú zásadný význam dva hlavné pojmy: optimalizácia na základe gradientu a stratová funkcia. Stratová funkcia umožňuje kvantifikovať kvalitu ľubovoľnej množiny váh (W). Naopak, cieľom optimalizácie je identifikovať množinu váh, ktorá



Obr. 3.4: Príklad výpočtu pooling funkcie pre max a average pooling metódy pre down-sampling, pri využití stride 2.

minimalizuje stratovú funkciu. Tieto koncepty spoločne umožňujú proces učenia, a tým pomáhajú sieti vytvárať presné odpovede. Snahou teda je, nájsť smer váhového vektoru v prostredí váhového priestoru. Túto zmenu nie je potrebné hľadať, pretože je možné vypočítať optimálny smer, ktorým sa má váhový vektor zmeniť. Tento smer je zaručene smer najstrmšieho zostupu a súvisí s gradientom (vektorom derivácií) stratovej funkcie [57, 24].

Zostup po gradiente (*gradient descent*) je algoritmus pre minimalizáciu stratovej funkcie. Algoritmus iteratívne počíta gradient a v cykle vykonáva aktualizáciu parametrov. Najčastejšie využívaná verzia mini-batch, ktorá vykoná aktualizáciu pre každú minidávku n trénovacích vzoriek. Táto verzia má stabilnejšiu konvergenciu a je veľmi efektívna vďaka využitiu optimalizovaných maticových optimalizácií. Vzorec výpočtu zostupu po gradiente vo verzii mini-batch je nasledovný:

$$p_{n+1} = p_n - \eta \nabla f_B(p_n), \tag{3.7}$$

kde p_n a p_{n+1} sú staré a nové hodnoty parametrov, η je rýchlosť učenia a $\nabla f_B(p_n)$ predstavuje predstavuje gradient stratovej funkcie f vypočítaný pre minidávku B dát, vyhodnotený pri parametroch p_n. V súčasnosti sa však častejšie využíva optimalizátor **Adam**. Adam má adaptívnu rýchlosť učenia pre každý parameter, čo môže byť obzvlášť výhodné pre datasety s chýbajúcimi hodnotami alebo modely s mnohými parametrami. Je efektívny z hľadiska pamäte aj výpočetného výkonu a robustný na rôznych úlohách. Obsahuje mechanizmus korekcie skreslenia pomáha znížiť počiatočné skreslenie smerom k nule, čím zlepšuje počiatočné fázy trénovania modelu [57]. Výpočet optimalizátoru Adam zobrazuje vzorec (3.8):

$$\begin{aligned} m_n &= \beta_1 m_{n-1} + (1-\beta_1)\nabla f(p_n), \\ v_n &= \beta_2 v_{n-1} + (1-\beta_2)(\nabla f(p_n))^2, \\ \hat{m_n} &= \frac{m_n}{1-\beta_1^n}, \\ \hat{v_n} &= \frac{v_n}{1-\beta_2^n}, \\ p_{n+1} &= p_n - \eta \frac{\hat{m_n}}{\sqrt{\hat{v_n}} + \epsilon}, \end{aligned} \tag{3.8}$$

kde m_n a v_n sú odhady prvého momentu a druhého momentu gradientov, β_1, β_2 sú faktory zabúdania, je gradient stratovej funkcie v p_n, $\hat{m_n}$ a $\hat{v_n}$ sú verzie m_n a v_n korigované o skreslenie, p_{n+1} a p_n sú staré a nové hodnoty parametrov, η je miera učenia a ϵ je malá konštanta, ktorá sa používa, aby sa zabránilo deleniu nulou. Adam vyvinul v roku 2014 D. Kingma a J. Ba [36]. Pre úspešnú optimalizáciu je účinnosť akéhokoľvek optimalizačného algoritmu do veľkej miery závislá od výberu stratovej funkcie. Tieto funkcie majú za

úlohu merať zhodu medzi predikciou a skutočnou cieľovou hodnotou (GT). Výsledná strata je priemerom všetkých strát jednotlivých údajov. Účelová funkcia je priemerom stratovej funkcie celej trénovacej množiny, ktorá obsahuje niekoľko trénovaných dát [64]. Pre účely tejto práce sa obmedzíme, len na predstavenie použitých stratových funkcií:

1. **Cathegorical Cross-Entropy (CCE) Loss Function** – je funkcia používaná pri úlohách klasifikácie viacerých tried, ktorá meria rozdielnosť medzi predpovedaným a skutočným rozložením pravdepodobnosti. Je definovaná ako priemerná záporná logaritmická pravdepodobnosť skutočnej triedy a je daná vzorcom (3.9):

$$L_{CE}(y, \hat{y}) = -\sum_{i} y_i \log(\hat{y}_i), \tag{3.9}$$

   pričom y_i je skutočná hodnota pre triedu i, $\hat{y}_i$ je predpokladaná pravdepodobnosť pre triedu i a suma sa vzťahuje na všetky triedy i.

2. **Triplet Loss Function** – funkcia najčastejšie používaná pre učenie rozoznávania tvárí a učenie embedding-ov. Tripletovú stratovú funkciu predstavili F. Schroff a kol. v roku 2015 v práci [59]. Hlavnou myšlienkou tejto funkcie je naučiť rozoznávať pozitívny pár obrázkov (dva obrázky rovnakého objektu) od negatívneho páru (dva obrázky rôzneho objektu). Výpočet triplet stratovej funkcie zobrazuje nasledujúci vzorec:

$$L_{Triplet}(a, p, n) = \max\Big(0, ||f(a) - f(p)||^2 - ||f(a) - f(n)||^2 + \alpha\Big), \tag{3.10}$$

   pričom a je *anchor* (referenčný vstup), p je pozitívny príklad tej istej triedy ako *anchor*, n je negatívny príklad rozdielnej triedy ako *anchor*, f je príznaková reprezentácia príkladu a α je tolerancia, ktorá sa používa na oddelenie kladných a záporných dvojíc.

3. **Quadruplet Loss Function** – W. Chen a kol. [12] vyvinuli *quadruplet* stratovú funkciu s cieľom odstrániť obmedzenia trojnásobnej stratovej funkcie pre reidentifikáciu osôb. *Quadruplet* stratová funkcia bola navrhnutá tak, aby zvyšovala medzitriednu variabilitu (rozdiely medzi rôznymi triedami) a znižovala vnútrodielnu variabilitu (rozdiely v rámci tej istej triedy). Vďaka tomu má lepšiu schopnosť generalizácie (dokáže lepšie fungovať na nevidených dátach).

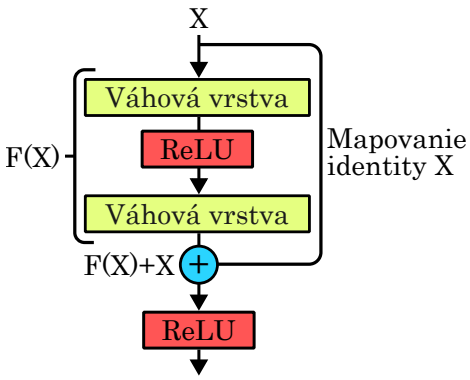
$$L_{Quadruplet}(a, p, n, n') = \max(0, ||f(a) - f(p)||^2 - ||f(n) - f(n')||^2 + \alpha), \tag{3.11}$$

   pričom n a n' sú záporné príklady inej triedy. Ostatné symboly majú rovnaký význam ako v prípade rovnice (3.10).

3.3.5 Reziduálne siete

Reziduálne siete boli navrhnuté ako odpoveď na problém miznúceho gradientu. Vo svojej práci K. He a kol. [28] poukázali na problém trénovania veľmi hlbokých neurónových sietí. Pri porovnaní dvoch takmer identických modelov, ktoré sa líšili iba v hĺbke siete, dosahovala hlbšia sieť horšie výsledky, a to aj počas tréningu. Autori zdôraznili, že táto degradácia nie je spôsobená pretrénovaním, ale skôr náročnosťou optimalizácie veľmi hlbokých sietí. Tento problém je známy ako problém **miznúceho gradientu**. Tento problém spôsobuje,

že v niektorých prípadoch bude gradient takmer nulový, čo účinne zabráni zmene hodnoty váhy a v najhoršom prípade môže spôsobiť zastavenie ďalšieho trénovania neurónovej siete. ResNet zavádza koncept „skratky“, ktorý umožňuje priame spätné šírenie gradientu do predchádzajúcich vrstiev. Architektúra reziduálnej siete sa skladá z reziduálnych blokov. Reziduálny blok obsahuje preskokové („skratkové“) spojenia, ktoré obchádzajú jednu alebo viacero vrstiev. Počas priameho prechodu reziduálny blok prijíma vstup, aplikuje súbor transformácií a potom pridá pôvodný vstup späť k výsledku týchto transformácií. Pri spätnom prechode umožňujú tieto preskokové spojenia priamy prechod gradientu do predchádzajúcich vrstiev. To pomáha zmierniť problém miznúceho gradientu, ktorý je v hlbokých sieťach bežný [28, 77]. Obrázok 3.5 približuje architektúru reziduálneho bloku.



Obr. 3.5: Stavebný blok reziduálnej siete. Diagram znázorňuje tok dát a hlavné komponenty reziduálneho bloku, pričom X je vstupná vrstva, $F(x)$ je transformovaný vstup, ku ktorému sa pridá pôvodný vstup X po mapovaní identity. Obrázok založený na [28].

3.4 Vyhľadávanie obrázkov na základe obsahu (*CBIR*)

Vyhľadávanie obrázkov na základe obsahu (*Content-Based Image Retrieval, CBIR*) je technika používaná na vyhľadávanie obrázkov vo veľkej databáze na základe ich vizuálneho obsahu, na rozdiel od metadát, ako sú označenia alebo popisky v prípade vyhľadávania obrázkov na základe textu (*TBIR*). Táto technika sa využíva v medicíne (pomáha pri diagnostike chorôb), elektronických obchodoch (vyhľadanie vizuálne podobných produktov na základe obrázku), či monitorovaní (identifikácia podozrivých osôb). Základnou koncepciou *CBIR* je extrahovať množinu príznakov z obrázku, ako je farba, textúra, tvar alebo iné vizuálne charakteristiky, a využiť tieto príznaky na identifikáciu iných obrázkov v databáze, ktoré vykazujú vizuálnu podobnosť. Tento proces je možné rozdeliť na dve úlohy:

1. **Extrakcia príznakov** – Z obrázka extrahujú príznaky, je možné použiť rôzne techniky ako napr. histogram farieb, detektor hrán, prípadne hlboké neurónové siete (tento prístup je bližšie popísaný v kapitole 2.3.1).

2. **Meranie podobnosti** – Po extrakcii príznakov sa vypočíta podobnosť medzi príznakmi dopytovaného (*query*) obrázka a podobnosťami podoblastí v rámci veľkého obrázka. Tieto podoblasti slúžia ako potenciálne zhody.

3. **Lokalizácia** – Podoblasť s najvyšším skóre podobnosti zodpovedá lokalizovanej polohe query obrázka.

V rozsiahlom spektre metód vyhľadávania obrázkov sme sa v našej práci rozhodli zamerať na prístup hlbokého učenia CBIR. Toto rozhodnutie je motivované niekoľkými dôvodmi najmä [15]:

1. **Automatické učenie príznakov** – využívajú prístupy založené na dátach pre učenie kľúčových príznakov. Takto získané príznaky, môžu obsahovať komplexné informácie, ktoré nie sú pre konvenčné prístupy viditeľné.

2. **Odolnosť voči zmenám** – sú odolnejšie voči zmenám mierky, rotácie a svetelných podmienok.

3. **Vysoká efektivita** – dosahujú vyššiu presnosť a efektívnosť vyhľadávania ako konvenčné prístupy.

4. **Sémantické porozumenie** – konvolučné neurónové siete dokážu porozumieť sémantickému obsahu obrázkov, čo umožňuje vyhľadanie nielen vizuálne podobných obrázkov, ale aj obrázkov, ktoré sémanticky s vyhľadávaným obrázkom súvisia.

Pri lokalizácii veľkých obrázkov v malých však vzniká problém s rýchlosťou a náročnosťou presného vyhľadávania. V scenároch zahŕňajúcich veľké obrazové databázy sa proces presného porovnávania príznakov stáva výpočtovo náročným a nepraktickým. Preto je potrebné nájsť metódu na zníženie objemu údajov, ktoré sa majú prehľadávať. Na vyriešenie tohto problému možno použiť algoritmy pre približné vyhľadávanie najbližších susedov.

3.4.1 Algoritmy pre približné vyhľadávanie najbližších susedov (ANNS)

Pri vyhľadávaní podobnosti je základnou úlohou identifikovať entitu, ktorá je najbližšie k zadanej požiadavke. Vo vysokodimenzionálnych priestoroch však prekliatie dimenzionality(*curse of dimensionality*[15]) spôsobuje, že presné vyhľadávanie podľa najbližšieho suseda je výpočtovo veľmi nákladné [43]. V dôsledku toho sa približné metódy stali vhodnou alternatívou vďaka svojej efektívnosti. Približné vyhľadávanie podľa najbližšieho suseda umožňuje efektívne znížiť počet potenciálnych zhôd. Tieto metódy dosahujú kompromis medzi presnosťou a efektívnosťou, čo umožňuje rýchle zníženie počtu potenciálnych zhôd pri zachovaní uspokojivého stupňa presnosti. Nie je zaručené, že približné metódy nájdu práve najbližšieho suseda, ale pri mnohých úlohách to nepredstavuje problém. Približné vyhľadávanie sa skladá z niekoľkých fáz:

1. **Transformácia vektorov** – je fáza, pri ktorej sa na vstupné vektory sa aplikujú transformácie. Tieto transformácie majú za úlohu zvýšenie efektivity vyhľadávania.

2. **Kódovanie vektorov** – je fáza, pri ktorej sa vektory kódujú s cieľom skonštruovať skutočný index alebo graf pre vyhľadávanie.

3. **Nevyčerpávajúce vyhľadávanie (*non-exhaustive search*)** – je fáza, ktorá zahŕňa hľadanie približných najbližších susedov v rámci vytvoreného indexu alebo grafu.

Metódy pre približné vyhľadávanie najbližších susedov môžeme rozdeliť na [43]:

- ***Hashing-based metódy*** – transformujú dáta na reprezentáciu s nízkou dimenziou, čím umožňujú, aby každý údaj bol reprezentovaný krátkym kódom (označovaným ako hash kód). V rámci tejto triedy existujú dve hlavné podkategórie:

[15] **Curse of dimensionality** – s rastúcimi dimenziami sa vlastnosti dát stávajú zložitejšími [52].

  - ***Locality-sensitive hashing* (LSH)** – prístup nezávislý na dátach, ktorý garantuje istú teoretickú kvalitu, efektívnosť aj veľkosť výsledkov, a to i v najhoršom prípade. Táto kategória zahŕňa metódy, ako sú SRS [61], QALSH [33], FALCONN [4].
  - ***Learning to hash* (L2H)** – využíva rozloženie dát pre dosiahnutie vyššej efektivity generovaním špecifických hašovacích funkcií za cenu garancií poskytovanými v prípade LSH. Táto kategória zahŕňa metódy, ako sú OPQ [22] a NSH [51].

- **Metódy založené na delení** – rozdeľujú celý vysokodimenzionálny priestor na niekoľko disjunktných oblastí. Medzi najznámejšie metódy patria Annoy [7] alebo Flann [50].

- **Metódy založené na grafoch** – Metódy založené na grafoch vytvárajú graf susedstva, v ktorom každému dátovému prvku zodpovedá uzol a hrany spájajúce niektoré uzly definujú susedské vzťahy. Zakladajú sa na myšlienke, že sused suseda je pravdepodobne tiež sused. Najúspešnejšími metódami v súčasnosti sú HNSW [47], NGT-QG [35, 34].

3.5 Nástroje pre vyhľadávanie najbližších susedov

V oblasti približného vyhľadávania podľa najbližších susedov (ANNS) je k dispozícii niekoľko knižníc, z ktorých každá má svoje silné a slabé stránky. Medzi významné z nich patria Annoy (*Approximate Nearest Neighbors Oh Yeah*), FLANN (*Fast Library for Approximate Nearest Neighbors*), HNSW (*Hierarchical Navigable Small World*) a Faiss (*Facebook AI Similarity Search*).

Výber správnej knižnice pre úlohy ANNS závisí od rôznych faktorov, ako je efektívnosť, výkon, škálovateľnosť a flexibilita. Po dôkladnom zvážení bola na túto úlohu vybraná knižnica Faiss. Toto rozhodnutie je motivované testovaním jednotlivých knižníc a na základe výsledku porovnávacieho testu [5].

3.5.1 Feiss

Faiss je knižnica pre približné vyhľadávanie podľa najbližších susedov (ANNS), ktorá je určená na správu veľkých kolekcií *embedding* vektorov vo vektorových databázach. Tieto databázy sa stávajú čoraz dôležitejšími v dôsledku rýchleho rozvoja aplikácií umelej inteligencie a zodpovedajúceho nárastu počtu *embeddings*. *Embeddings* sú vektorové reprezentácie zvyčajne vytvorené neurónovou sieťou. Mapujú vstupné mediálne položky do vektorového priestoru, v ktorom lokalita kóduje sémantiku vstupu. Tento spôsob ukladania a vyhľadávania komplexných údajov sa objavil s rozmachom hlbokého učenia.

Faiss poskytuje rozsiahly súbor nástrojov indexovacích metód a súvisiacich primitív. Tieto nástroje sa používajú na vyhľadávanie, zhlukovanie, kompresiu a transformáciu vektorov, vďaka čomu je nástroj Faiss mimoriadne vhodný na vyhľadávanie vektorovej podobnosti, čo je základná funkcia vektorových databáz.

Základnou štruktúrou Faiss je index, ktorý môže uchovávať množstvo databázových vektorov, ktoré sa postupne pridávajú. Keď sa počas vyhľadávania do indexu vloží query vektor, index vráti databázový vektor, ktorý je najbližšie ku query vektoru z hľadiska zvolenej vzdialenosti. Knižnica umožňuje vyhľadávanie pomocou CPU alebo GPU [14].

Kapitola 4

Riešenie problému zošívania obrázkov

4.1 Aplikácia pre zobrazenie homografie datasetu

Koncept homografie je vo svojej podstate abstraktný, takže je náročné predstaviť si mapovanie obrazov, na ktoré sa homografia aplikuje. Na vyriešenie tohto problému som vyvinul aplikáciu HomographyViewer, ktorá poskytuje základný prehľad o súbore údajov obsahujúcom obrázky so známymi homografiami. Aplikácia je vytvorená v prostredí PyQT6[1] v kombinácií s základnou prácou v prostredí OpenGL[2], pretože sa používa prostredie GLWidget. Aplikácia nebola dokončená a je v rannej fáze vývoja, keďže práca bola upriamená na inú činnosť. Táto aplikácia v aktuálnej podobe umožňuje:

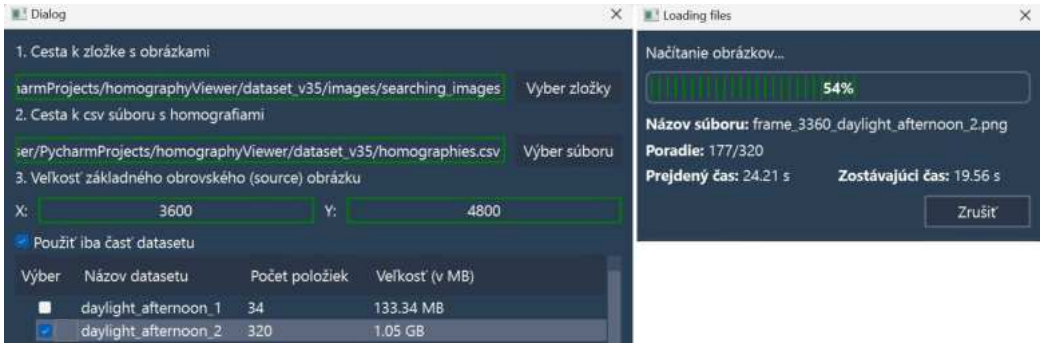
- Obrázky sa môžu zobrazovať v trojrozmernom prostredí, kde sa správajú ako plocha.
- Otáčanie (ťahanie RMB[3]), označovanie (kliknutie LMB[4])), pohyb kamery (šípky) (viď obrázok 4.2).
- Zoom do subpixelovej presnosti.
- Zobrazene farby po kliknutí (kliknutie LMB).
- Uloženie obsahu GLWidgetu ako obrázok.
- Načítanie, len malej časti súboru údajov so zobrazením základných informácií o ňom (veľkosť, počet snímok) (viď obrázok 4.1).

4.1.1 Budúci rozvoj aplikácie pre zobrazenie homografie datasetu

Vylepšenia tejto aplikácie GUI môžu byť predmetom ďalšej práce. Ako ilustračný príklad takýchto vylepšení môže slúžiť implementácia nasledujúcich funkcií:

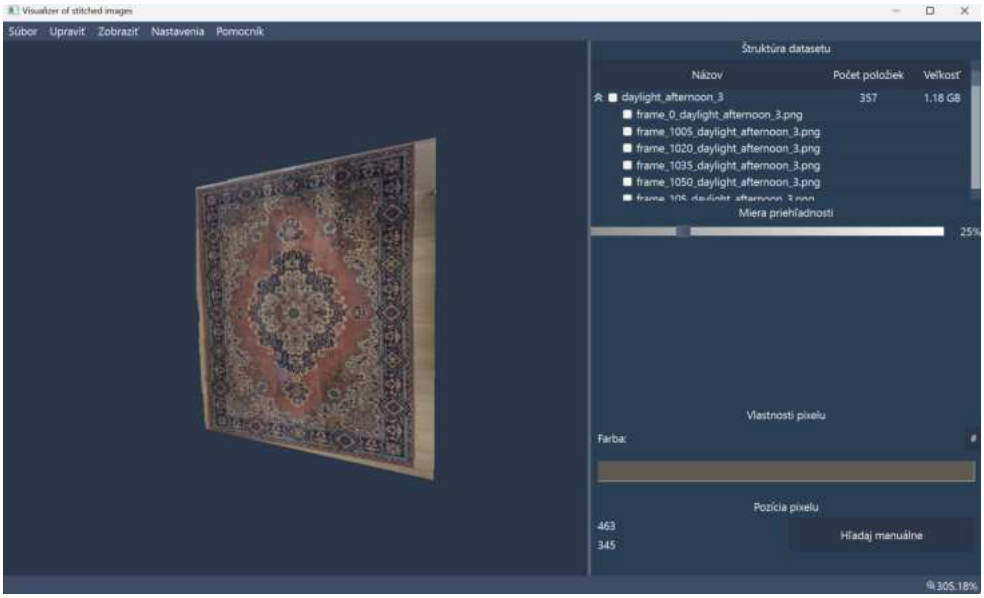
- (ťažké) Umožniť užívateľovi do scény pridať svetelný zdroj a tak umožniť vytvorenie verných augmentácií snímok.

[1] https://doc.qt.io/qtforpython-6/
[2] https://www.opengl.org/
[3] *Right Mouse Button* – pravé tlačidlo myši
[4] *Left Mouse Button* – ľavé tlačidlo myši



Obr. 4.1: Dialógové okná aplikácie. Okno vľavo zobrazuje dialógové okno pre výber snímok datasetu a homografií k nim. Používateľ si môže zvoliť, ktoré podmnožiny datasetu si chce zobraziť. Druhé dialógové okno (*progress bar*) zobrazuje načítanie zvoleného datasetu s informáciami o tom, ktorá časť je načítaná aktuálne.

- Implementácia možnosti skrytia a odstránenia označených snímok po ich výbere.
- Vykreslenie ohraničenia okolo vyznačených snímok.
- Možnosť pridať ďalšie datasety do zobrazenia vedľa seba pre možnosť porovnania.
- Zobrazenie informácií o snímku po kliknutí naň (zobrazenie detailov).



Obr. 4.2: Aplikácia grafického používateľského rozhrania (GUI) umožňuje vizualizáciu homografie snímok (*framov*) datasetu vo obrovskom obraze. Umožňuje zobrazenie snímok datasetu na správnej pozícii. Pri načítaní je možné vybrať dataset, ktorý sa má zobraziť (súbor údajov jedného typu svetelných podmienok sa zvyčajne vytvára z jedného videa).

4.2 Program pre zošívanie obrázkov

Tradičným prístupom k zachytávaniu rozľahlých scén je použitie fotoaparátu so širokouhlým objektívom. S narastajúcimi rozmermi scény je čoraz náročnejšie zachytiť celú scénu tradičným spôsobom. Využitie širokouhlého objektívu je však obmedzené niekoľkými významnými obmedzeniami:

- **Spôsobujú kreslenie** – skreslenie objektívu môže spôsobiť, že rovné čiary sa môžu javiť ako krivky. Existujú tri základné typy skreslenia objektívu: súdkovité skreslenie, vankúšové skreslenie a fúzové skreslenie.
- **Obmedzené rozlíšenie snímača** – rozlíšenie snímačov je obmedzené, a preto je na zachytenie veľkej scény vo vysokom rozlíšení potrebné použiť kvalitné zariadenie. Takéto vybavenie je však často veľmi drahé.
- **Obmedzené zorné pole** – zorné pole kamery (FOV) je obmedzené. Dokonca ani špecializované objektívy typu rybie oko nedokážu zachytiť celú scénu.

Tieto problémy sa dajú zmierniť alebo úplne odstrániť využitím spájania obrazov (*image stitching*).

+ **Spôsobujú menšie kreslenie** – použitie štandardného objektívu umožňuje zachytiť viacero snímok scény, ktoré sa potom spoja. Tento proces znižuje skreslenie a vedie k presnejšiemu zobrazeniu scény.
+ **Nezávislé od rozlíšenia snímača** – použitie kamery so štandardným rozlíšením umožňuje zachytiť viacero snímok scény, ktoré sa potom spoja do jedného obrazu s vysokým rozlíšením. Tento proces umožňuje použiť snímač s nižšou kvalitou, než by bolo inak možné, a výsledkom je vysokokvalitný obraz.
+ **Simulácia neobmedzeného zorného poľa** – spájanie snímok umožňuje používateľovi spájať scény z viacerých uhlov, čím sa vytvorí jedna veľká snímka. Je preto možné vytvárať aj 360-stupňové panorámy.

Z týchto dôvodov bol zvolený prístup spájania obrazov. Nasledujúce časti opisujú implementáciu vytvárania veľkého obrazu z menších obrazov pomocou knižnice openCV (kapitola 2.6.1). Tieto kroky sa opakujú pre všetky obrázky, ktoré si vyžadujú zlúčenie. Vždy sa zlúči predtým vytvorená časť zlúčeného obrázka (čiastkový výsledok) a nasledujúci obrázok z priečinka.

4.2.1 Extrakcia snímok videa

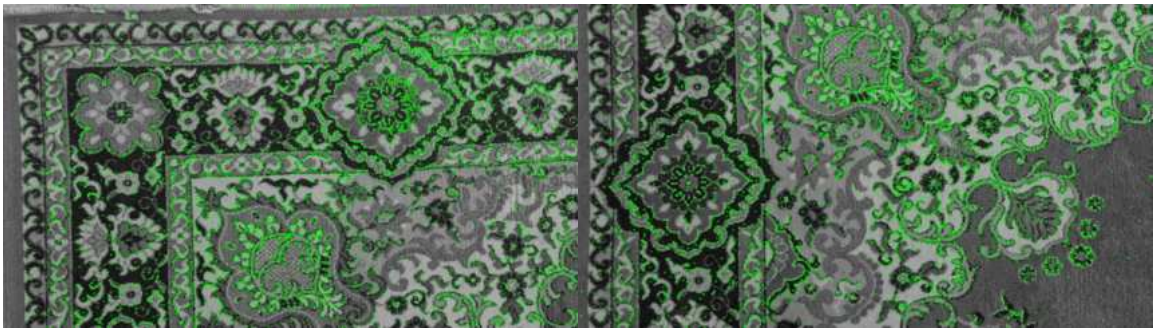
Prvá fáza tohto procesu zahŕňa vzorkovanie snímok z videa (obrázok 4.3). Toto sa vykonáva takou rýchlosťou, ktorá poskytuje dostatočné prekrývanie medzi po sebe idúcimi snímkami pre proces spájania. Extrahované snímky sa uložia do adresára na ďalšie spracovanie.

4.2.2 Extrakcia príznakov pomocou SIFT

Po získaní snímok, ktoré sa majú spájať, je nevyhnutné identifikovať kľúčové body v každom obrázku (obrázok 4.4). Táto úloha sa realizuje pomocou algoritmu SIFT. Algoritmus SIFT identifikuje kľúčové body v obraze a následne pre každý kľúčový bod generuje deskriptor, ktorý zachytáva jeho lokálne charakteristiky obrazu. Tieto deskriptory sú invariantné voči

Obr. 4.3: Nasledujúci obrázok znázorňuje príklady obrázkov získaných extrakciou toho istého videa. Aby bol tento proces úspešný, je potrebné zabezpečiť, aby sa extrahované obrazy prekrývali aspoň o $15 - 30\%$. Táto implementácia predpokladá aspoň 40% prekrytie.

mierke, orientácii a afinnému skresleniu obrazu, čo z nich robí robustné nástroje na porovnávanie rôznych obrazov tej istej scény. Argumenty pre SIFT boli nastavené takto: počet prvkov bol stanovený na 10 000, zatiaľ čo prah kontrastu bol nastavený na 0.05.



Obr. 4.4: Obrázok znázorňujúci kľúčové body (zelené kruhy) na oboch fotografiách pri voľbe 10 000 príznakov a zvolení prahu kontrastu 0.05.

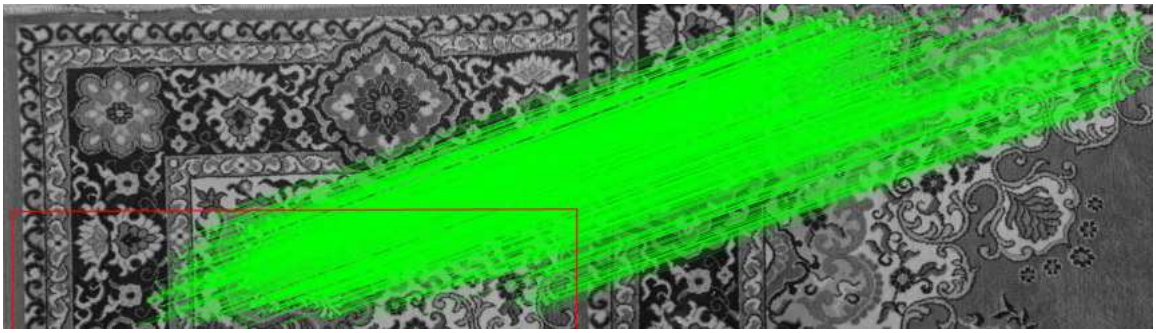
4.2.3 Párovanie príznakov pomocou FLANN

Po identifikácii príznakov na každom obrázku nasleduje párovanie týchto príznakov medzi obrázkami (obrázok 4.5). Tento proces sa realizuje pomocou algoritmu približného vyhľadávania najbližších susedov, známeho ako FLANN, s počtom stromov nastaveným na 5. Výsledkom tohto kroku je množina zhodných bodov medzi každou dvojicou obrazov. Tieto zhodné body označujú miesta, kde bol na oboch obrázkoch zistený rovnaký príznak.

4.2.4 Zarovnanie obrazov pomocou RANSAC

Po spárovaní prvkov nasleduje zarovnanie obrazov na základe spárovaných bodov. Táto úloha sa realizuje pomocou robustnej metódy na odhad geometrickej transformácie medzi každým párom obrazov, známej ako RANSAC. RANSAC funguje na základe iteračného odhadu transformácie pomocou podmnožiny zhodných bodov a vyhodnotenia kvality odhadu na základe všetkých zhodných bodov. Odhadnutá homografia mapuje body na jednom

obraze na zodpovedajúce body na druhom obraze. Maximálna povolená chyba reprojekcie, aby sa dvojica bodov považovala za inlier je nastavená implementačne na hodnotu 5.



Obr. 4.5: Obrázok znázorňuje porovnanie bodov medzi dvoma snímkami videa. Zelené čiary predstavujú zhody medzi snímkami. Priestor vymedzený červeným štvoruholníkom znázorňuje umiestnenie druhej snímky, ktorú sa pokúšame umiestniť.

4.2.5 Skreslenie obrazu

Po získaní homografickej matice sa každý pixel v obraze transformuje podľa tejto matice. Tento proces sa nazýva skreslenie obrazu. Skreslenie mení polohu pixlov v obraze tak, aby sa zodpovedali perspektíve iného obrazu. V praxi sa to dosahuje tak, že sa pre každý pixel v cieľovom obraze vypočíta jeho zodpovedajúca poloha v zdrojovom obraze pomocou inverznej homografickej matice. Hodnota pixelu v cielovom obraze sa potom nastaví na hodnotu zodpovedajúceho pixelu v zdrojovom obraze. Ak zodpovedajúca poloha nespadá presne na pixel, použije sa interpolácia (obrázok 4.6).

Obr. 4.6: Obrázok znázorňuje spojenie obrázkov, ktoré znázorňuje obrázok extrakcie 4.4, a obrázok párovania 4.5. Pri spájaní častokrát vznikajú čierne oblasti, ktoré je možné odstrániť vyrezaním najväčšieho vnútorného obdĺžnika, prípadne metódou *inpainting*.

4.2.6 Spájanie obrazov pomocou *multi-band blending*

Posledným krokom v procese je prelínanie deformovaných obrazov, aby sa vytvorila súvislá panoráma. Toto sa vykonáva pomocou viacpásmového prelínania, ktoré spája obrazy dokopy vo frekvenčnej oblasti, aby sa znížili viditeľné švy. Výsledkom je veľký obraz, v ktorom sú pôvodné obrazy plynule integrované.

Kapitola 5

Riešenie problému lokalizácie obrázku v obrovskom obrázku

Druhá časť tohto riešenia sa zaoberala lokalizáciou vybranej menšej fotografie (označenej ako *query*) vo väčšej fotografii (označenej ako *mapa*). Tento problém je možné klasifikovať ako podúlohu vyhľadávania obrázkov na základe obsahu (viď kapitola 3.4), avšak naším cieľom nie je filtrovanie informácií, ale určenie lokality. Hoci ide o veľmi podobné problémy, existujú určité špecifiká, ktoré sú relevantné len pre problém lokalizácie. Táto kapitola je štruktúrovaná podľa procesu, ktorý je nutné vykonať na získanie požadovanej lokality. Kapitola 5.1 pojednáva o tvorbe datasetu pre tento účel, kapitola 5.3 sa venuje učeniu pomocou klasifikácie, kapitola 5.4 je venovaná metóde triplet, kapitola 5.5 popisuje tvorbu *embedding* databázy a kapitola 5.6 sa zaoberá samotnou lokalizáciou *query* obrázku v rozsiahlej mape.

5.1 Tvorba datasetu

Malý počet obrázkov venujúcim sa povrchom, či materiálom ma donútil vytvoriť niekoľko datasetov. Pre účely práce som vytvoril rozsiahlu zbierku fotografií zobrazujúcich rôzne povrchy v interiéri. Najčastejším a najviac používaným datasetom je dataset koberec. Tento dataset je považovaný za referenčný vzhľadom na 11 rôznych svetelných podmienok (údaje boli získané v rôznych časoch počas dňa, prípadne použitím umelého osvetlenia). K snímaniu bola použitá kamera mobilného telefónu s parametrami zhrnutými v nasledujúcej tabuľke (tabuľka 5.1):

Špecifiká snímacieho zariadenia						
Výška	**Širka**	**Senzor**	**Video kodek**	**Audio kodek**	**FPS**	**Bitová Hĺbka**
1920	1080	108MP	H.264	AAC	60	24

Tabuľka 5.1: Snímacie špecifiká kamery pre tvorbu datasetu.

Pri tvorbe datasetu som určil niekoľko predpokladov, ktoré výrazne uľahčili jeho vytvorenie, vrátane:

- Kamera snímajúca povrch je naň rovnobežná.
- Rýchlosť snímania kamery je konštantná.

Obr. 5.1: Vozík ktorým boli snímané povrchy a materiály. Obsahuje kryt do ktorého sa uchytí telefón a kameruje sa podlaha. Aktuálna verzia umožňuje zmeniť výšku snímania.

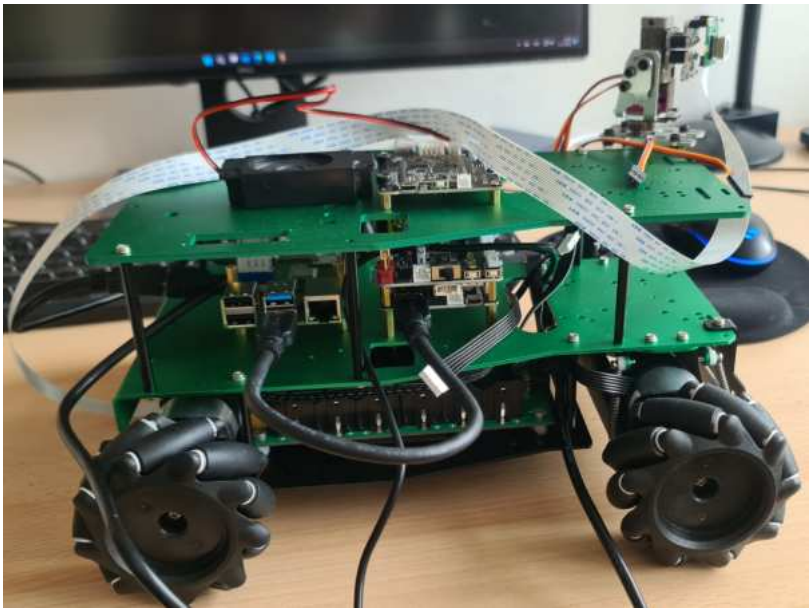
- Snímaný povrch by sa počas snímania nemal meniť (napríklad pohybom objektov v scéne).
- Diverzita datasetu je vytvorená primárne pomocou zmeny svetelných podmienok.

Proces snímania požadovaného materiálu alebo objektu zahŕňa použitie na mieru vyrobeného vozíka (obrázok 5.1, 5.2), ktorý je navrhnutý tak, aby udržiaval kameru v polohe rovnobežnej so zemou. Snímanie kamerou sa musí niekoľkokrát opakovať, aby sa objekt snímania zachytil v rôznych svetelných prostrediach. To umožňuje, aby sa dataset zovšeobecnil, a teda aby ho menej ovplyvňovali zmeny osvetlenia. Po získaní videozáznamu povrchu alebo materiálu dochádza k vzorkovaniu pri zvolenej vzorkovacej frekvencii, aby sa vytvorili snímky (*frames*). Proces tvorby datasetu sa často delí na tri časti. Je to preto, že proces vytvárania datasetu je ako celok časovo náročná operácia. Každá z týchto častí pracuje nezávisle po prijatí potrebných údajov z predchádzajúcej časti, čo umožňuje paralelné vykonávanie jednotlivých fáz v prípade, že sú k dispozícii údaje z niektorého z predchádzajúcich behov. Táto architektúra tiež umožňuje rozdeliť proces vytvárania, len na vykonanie niektorých častí. Po jedinej lokalizácií snímok videa je teda možné vykonať niekoľko priechodov vyhľadávania fragmentov[1], pričom môžeme zmeniť ich veľkosť, mieru prekrývania a iné parametre bez nutnosti opätovnej lokalizácie snímok. V ďalších častiach tejto kapitoly sa o týchto častiach hovorí podrobnejšie.

5.1.1 Proces získania lokalizovaného snímku

V tejto časti sa budeme podrobne zaoberať procesom získavania lokalizovaného snímku. Jedná sa o prvý krok spracovania získaných snímok pri tvorbe datasetu. Tento proces pozostáva z nasledujúcich krokov (viac o jednotlivých krokoch z teoretického hľadiska v kapitole 2.2):

[1] **fragment** – malý *patch* obrázku, ktorý je vstupom neurónovej siete



Obr. 5.2: Automatické zariadenie snímajúce povrchy a materiály. Vozík bude snímať podlahu samostatne bez nutnosti zásahu človeka. V súčasnosti nie je ešte plne funkčný.

1. **Načítanie dát** – načíta sa obrovský obrázok a zoznam ciest k obrázkom, ktoré sa v ňom majú hľadať.

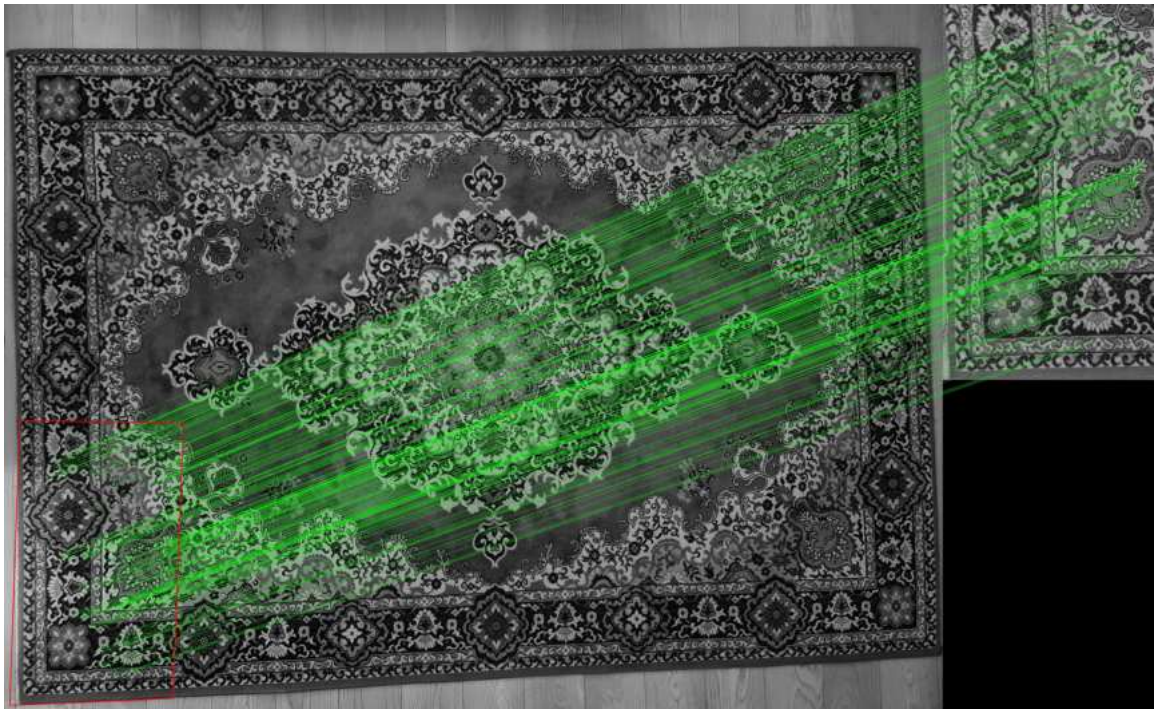
2. **Inicializácia detektora** – použije sa detektor SIFT na nájdenie kľúčových bodov a deskriptorov v obrázkoch. Tieto kľúčové body a deskriptory sa potom používajú na nájdenie zhôd medzi obrovským obrázkom a hľadanými obrázkami.

3. **Nájdenie zhôd** – algoritmus párovania príznakov FLANN sa použije medzi obrovským obrázkom a hľadanými obrázkami (obrázok 5.3). Zhody sa hľadajú obojsmerne – z obrázka obrovského obrázka do hľadaného obrázka a naopak.

4. **Aplikácia testov** – Na zhody sa aplikuje niekoľko testov, aby sa zabezpečila spoľahlivosť výsledkov. Používajú sa najmä tieto testy:

   - **Cross-check** – test kontroluje, či zhoda nájdená z obrázka A do obrázka B existuje aj v opačnom smere, teda z obrázka B do obrázka A. Týmto spôsobom sa zabezpečí, že zhoda je vzájomná a nie jednostranná.
   - **Symetrický test** – test je podobný cross-checku, ale kontroluje, či sú zhody symetrické. To znamená, že ak bod X na obrázku A zodpovedá bodu Y na obrázku B, potom by mal bod Y na obrázku B zodpovedať bodu X na obrázku A.
   - **Loweov pomerový test** [45] – test je založený na pomere vzdialeností medzi najlepšou zhodou a druhou najlepšou zhodou. Ak je tento pomer menší ako určitý prah (často sa používa hodnota 0,7), potom sa zhoda považuje za dobrú. Tento test pomáha eliminovať menej spoľahlivé zhody.

5. **Výpočet homografie** – Ak je počet dobrých zhôd väčší ako minimálny počet zhôd, vypočíta sa homografia medzi zdrojovým obrázkom a hľadaným obrázkom.



Obr. 5.3: Obrázok zobrazujúci spárované príznaky medzi obrovským obrázkom a hľadaným obrázkom (*frame*), pričom párovacie (*matching*) čiary sú znázornené zelenou farbou. Červený štvoruholník znázorňuje lokalizovanú polohu hľadaného obrázku v obrovskom obrázku.

6. **Uloženie výsledkov** – Každá úspešná nájdená homografia sa uloží. Tento proces sa opakuje pre všetky obrázky v adresári s hľadanými obrázkami. Všetky homografie obrázkov s validnou homografiou uloží do CSV súboru.

5.1.2 Proces získania lokalizovaných fragmentov (*small image patches*)

Po získaní homografie každej snímky je potrebné určiť, ktoré snímky budú obsahovať ktoré fragmenty. Táto operácia výrazne urýchli proces určovania homografie pre fragmenty, pretože bude možné vyhľadávať fragment v rámci snímky (obrázok 5.5), a nie v celom obrovskom obrázku. Okrem toho táto metodika umožňuje meniť nastavenia veľkosti fragmentov bez nutnosti opätovného získavania homografie obrazov (preskočenie procesu získania lokalizovaného snímku, teda kapitoly 5.1.1). Ak fragmenty ešte neboli vytvorené vytvoria sa teraz. Presné kroky tejto operácie sú nasledovné:

1. **Vytvorenie fragmentov** – Obrovský obrázok sa pomocou mriežky rozdelí na malé (ne)prekrývajúce sa fragmenty na základe zvolenej veľkosti (obrázok 5.4). Fragmenty sú uložené ako obrázky a ich poloha vo formáte JSON.

2. **Načítanie homografií** – homografie, ktoré mapujú polohu snímok v rámci obrovského obrázku, sú načítané z CSV súboru.

3. **Určenie príslušnosti fragmentov** – pre každú homografiu sa vypočíta štvoruholník, ktorý zodpovedá polohe snímku v obrovskej fotografii. Potom sa prechádzajú všetky fragmenty v rámci homografie a kontroluje sa, či daný fragment patrí do snímky

Obr. 5.4: Obrázok znázorňujúci fragmenty pri zvolení prekryvu 49 %.

porovnaním súradníc. Ak áno, pridá sa do zoznamu fragmentov obsiahnutých v danom snímku.

4. **Uloženie výsledkov** – nakoniec sa výsledky uložia do JSON súboru. Meno snímku tvorí kľúč a hodnotou je zoznam fragmentov obsiahnutých v danom snímku.

5.1.3 Proces vytvorenia datasetu

Posledná časť v ktorej sa vytvára finálny dataset. Na základe získaných informácií o príslušnosti fragmentov ku snímkam, sa tieto v nich vyhľadávajú a ukladajú. Tento proces sa dá zhrnúť takto:

1. **Prechod snímok** – Cyklus prechádza cez všetky snímky na základe informácií o príslušnosti fragmentov ku snímkam. Tento krok zahŕňa načítanie každého snímku jednotlivo do pamäte.

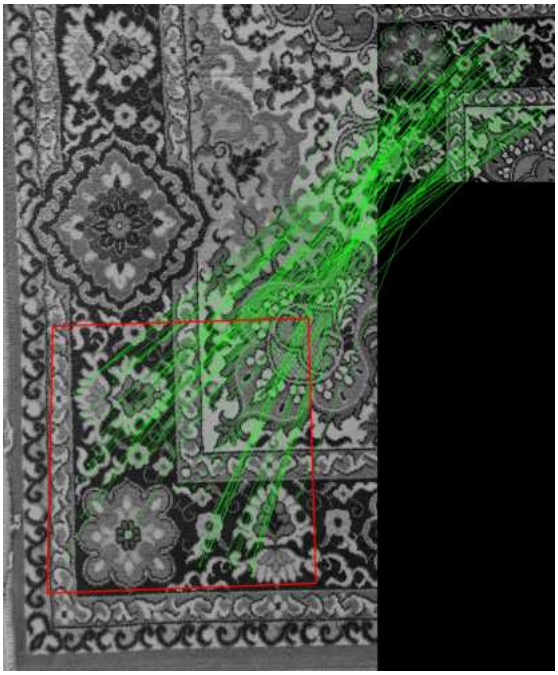
2. **Určenie homografie** – Pre každý fragment v danom snímku sa vypočíta homografia. Tento krok zahŕňa načítanie každého fragmentu jednotlivo do pamäte.

3. **Kontrola** – Skontroluje sa validita vypočítanej homografie. Overí sa, či všetky súradnice fragmentu po perspektívnej projekcii sú validné a nachádzajú sa v rámci rozmerov snímku.

4. **Uloženie** – Ak je fragment platný, uloží sa do príslušnej triedy v datasete. Trieda je určená na základe súradníc stredného bodu fragmentu v pôvodnom obrázku (obrázok 5.6 zobrazuje príklad časti triedy).

5.1.4 Vytvorené datasety a ich využitie

Súbor údajov koberec obsahuje 21457 / 68961 / 142489 obrázkov v závislosti od výstupnej veľkosti použitého obrázka. Môže sa použiť na testovanie nezávislosti na symetriu odrazu v oboch smeroch osi. Ďalšie súbory údajov zahŕňajú: laminát1, laminát2, koberec2, podlaha1,



Obr. 5.5: Obrázok zobrazuje spárované príznaky medzi snímkou a fragmentom. Červený štvoruholník znázorňuje lokalizovanú polohu hľadaného fragmentu v snímke.

podlaha2, hrdzavý plech a drevo. Tieto sú menšie (3500 – 10000 obrázkov) a pre tréning neboli využité pre nedostatok rôznorodosti, prípadne pre nemožnosť lokalizácie snímok pre nízky počet kvalitných príznakov.

5.2 Návrh neurónovej siete pre tvorbu *embeddings* (*encoder*)

Prvým krokom pri tvorbe neurónovej siete bola voľba architektúry. Pre tréning bola zvolená architektúra ResNet. Konkrétne sa využíva architektúra Resnet50 [28], ktorá je v oblasti počítačového videnia pre strojové učenie veľmi často používaná. Hlavnými stavebnými blokmi tejto architektúry sú konvolučné vrstvy, nasledované pooling vrstvami s aktivačnými funkciami ReLU (viac v kapitole 3.3.1). Pre implementáciu bol využitý programovací jazyk Python, pričom pre prácu s obrázkami bola využitá knižnica OpenCV (kapitola 2.6.1) a PIL[2], pre hlboké učenie knižnica Pytorch (kapitola 2.6.2). Pre vizualizáciu sa využívajú knižnice matplotlib[3] a seaborn[4]. Pytorch poskytuje využité modely prostredníctvom modulu `torchvision.models`. Pre tvorbu *embeddings* boli zvolené dva prístupy:

- **Učenie pomocou klasifikácie** – tréning prebieha klasifikačným spôsobom, pričom *embedding* je iba vedľajším produktom.
- **Učenie pomocou učenia vzdialeností** – tréning prebieha tak na základe vzdialeností *embeddings* pomocou triplet stratovej funckie.

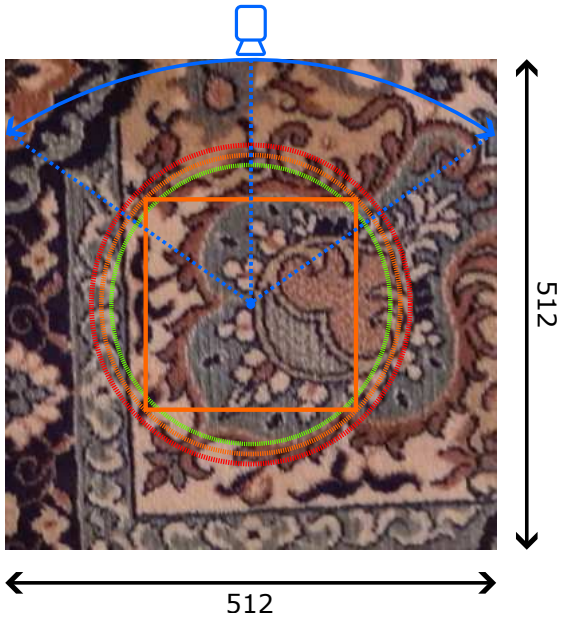
[2] https://python-pillow.org/
[3] https://matplotlib.org/
[4] https://seaborn.pydata.org/

Obr. 5.6: Na obrázku sú znázornené obrázky patriace do jednej triedy po ich vytvorení. Obrázky boli vytvorené za rôznych svetelných podmienok s cieľom zvýšiť robustnosť datasetu.

Tieto modely využívajú rôzne spôsoby, získavania *embedding*, preto sú tieto prístupy rozdelené do kapitol 5.3 pre klasifikačný prístup a 5.4 pre tréning na základe triplet loss. V kapitole 5.2.1 je spomenutý spôsob augmentovania dát pri tréningu.

5.2.1 Augmentácia dát

Pre tréning boli vytvorené špeciálne augmentácie, ktoré znižujú vplyv otáčania kamery (obrázok 5.7). Vzhľadom na to, že reálne situácie sú veľmi vzdialené od ideálnych podmienok, ktoré sú k dispozícii pri vytváraní súboru údajov, je potrebné vykonať kroky, ktoré umožňujú simuláciu náhodného otáčania kamery. Toto otáčanie sa dosiahne tak, že sa najprv vypočíta stredový bod obrazu. Potom sa vygeneruje náhodný uhol pre homografickú transformáciu, ktorý je daný určitým intervalom. Nakoniec sa vypočíta matica rotácie pre tento uhol. Uvedená matica sa potom použije na realizáciu homografickej transformácie v obraze. Následne po homografickej transformácii sa na obraz aplikuje ďalšia transformácia. Ide o stochastické operácie, ktoré si tiež vyžadujú generovanie náhodných čísel zo zadaného intervalu, zvlášť pre uhly otočenia a posunutia. Uvedené transformácie sa potom aplikujú na obraz. Nakoniec sa vypočíta nový stredový bod obrazu a určia sa súradnice výrezu obrazu. Súradnice sa určia na základe stredového bodu obrazu a polovice veľkosti výrezu. Obrázok sa potom oreže na tieto súradnice a výsledný orez sa vráti ako výstup. Tento proces zabezpečuje, že model je schopný zovšeobecniť a správne klasifikovať obrazy, ktoré boli otočené alebo posunuté, alebo sa zmenila poloha snímacej kamery, čím sa zvyšuje jeho odolnosť voči takýmto transformáciám. Bol tiež zavedený termín `augment_factor`, ktorý hovorí, koľkokrát sa má daná sada obrázkov využiť. Keďže sa fragmenty vytvárajú plne nedeterministicky, môžeme ich považovať za jedinečné.



Obr. 5.7: Znázornenie možnosti pôsobenia augmentácií na obrázok. Oranžový štvorec znázorňuje fragment výrezu prostej strednej časti. Kruhy znázorňujú možnosť rotácie. Ide o rotáciu v intervale $< -180°, 180° >$. Priestor medzi zeleným a červeným kruhom je posun. Modrá farba približuje uhol možnosti rotácie kamery.

5.3 Učenie pomocou klasifikácie

Klasifikačný prístup je jednoduchšou metódou ako triplet prístup. Sieť sa trénuje ako typický klasifikátor so stratovou funkciou krížovej entropie a optimalizátorom Adam s mierou učenia 0,001. Okrem toho sa počas trénovania používa mechanizmus skorého zastavenia, pričom trénovanie sa ukončí, ak sa hodnota validačnej stratovej funkcie nezlepší do piatich epoch. Okrem toho sa využíva princíp klesajúcej rýchlosti učenia, pri ktorom sa sieť v každej ďalšej epoche učí pomalším tempom. Vstupom modelu sú malé obrázky (fragmenty) s dimenziou $h \times w \times c$, kde h je výška obrázka, w je šírka obrázka a c je počet farebných kanálov. V tomto prípade boli konkrétne hodnoty $h = w = 224$ a $c = 3$. Obrázky zo súboru údajov boli vytvorené v zadanej veľkosti 512 (popis ich transformácie na obrázky určenej veľkosti je popísaný v kapitole 5.2.1).

E	BS	LR	OPT	DR	IS	ANGLE	H.ANGLE	OFFSET	AF
20	256	0.001	Adam	0.95	224	<-180;180>	<-55;55>	<-15;15>	14

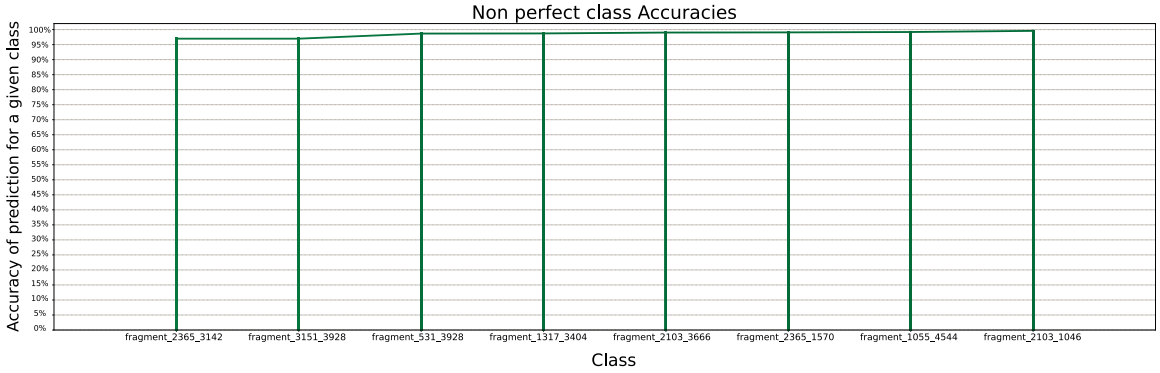
Tabuľka 5.2: Predvolené hyperparametre tréningu, kde `E` je počet epoch, `BS` je batch size, `LR` je rýchlosť učenia, `OPT` je optimalizátor, `DR` je rýchlosť úpadku, `IS` je veľkosť obrázku (výška aj šírka), `ANGLE` je interval rotačnej augmentácie, `H.ANGLE` je interval natočenia kamery (homografická rotácia), `OFFSET` je interval translačného posunu od stredu a AF je miera augmentácie.

Charakteristickým znakom tohto prístupu je, že na rozdiel od typického klasifikátora využíva ako klasifikačné triedy výrezy (fragmenty, viď obrázok 5.8) konkrétnej identickej časti povrchu. V tomto prípade je trieda určená stredom fragmentu, čo sa odráža v názve triedy. Ilustračným príkladom názvu triedy je `fragment_x_y`, kde x predstavuje hodnotu

Obr. 5.8: Obrázky fragmentov patriacich jednej konkrétnej triede (v tomto prípade `fragment_793_4190`). Jedná sa konkrétne o obrázky vygenerované na konkrétnom mieste datasetu koberec, ktoré je zobrazené na obrázku vľavo. Obrázok znázorňuje využitie rôzneho času snímania prípadne použitie umelého svetla.

súradnice osi x a y predstavuje hodnotu súradnice osi y v karteziánskom súradnicovom systéme obrovského obrazu. Model je inicializovaný počtom tried povrchov prítomných v datasete o kobercoch, ktorý je 293. Model ResNet sa načíta s predtrénovanými váhami a posledná vrstva sa odstráni, aby sa pridala vlastná klasifikačná vrstva. Ide o plne prepojenú vrstvu, ktorá mapuje výstup zo siete ResNet na počet cieľových tried. Vstupný obrázok sa potom privedie cez neurónovú sieť, kde sa extrahujú jeho príznaky. Následne sa tieto príznaky transformujú na jednorozmerný tenzor (s využitím operácie *flatten*) a ten sa kopíruje pre účely generovania výstupných *embeddings*. Na základe príznakov sa sieť rozhodne pre jednu z tried materiálu. Cieľom tohto prístupu je umožniť sieti klasifikovať triedy povrchov s vysokou mierou presnosti, pričom predpokladáme, že výsledné *embeddings*, ktoré sú len vedľajším produktom procesu klasifikácie, budú dostatočne reprezentatívne.

Tento prístup má však určité obmedzenia. Počet tried, ktoré je možné pomocou kvalifikátoru predpovedať, je obmedzený množstvom dát, ktoré sú k dispozícii pre každú triedu, a schopnosťou rozlišovať jednotlivé triedy (obrázok 5.9). V prípade, že existuje nadmerný počet tried a nedostatok údajov pre každú z nich, môže model naraziť na problémy pri učení sa rozlišovať medzi jednotlivými príznakmi tried. To však predstavuje v tomto prístupe značnú výzvu, pretože zväčšenie fotografie značných rozmerov bude mať za následok nárast počtu tried. Miera nárastu závisí od mnoho faktorov, ale predovšetkým na miere povoleného prekrývania sa fragmentov. Dokonca aj súbory údajov značnej veľkosti, ako napríklad ImageNet, sú obmedzené na 1 000 tried. V dôsledku toho by pri určitej veľkosti už nebolo možné použiť túto metódu (snímka by bola taká veľká, že vytvorených tried by bolo príliš mnoho). Okrem toho problém symetrie povrchu môže tiež spôsobiť ťažkosti pri klasifikácii. Ako už bolo uvedené v kapitole 5.2.1 o augmentácii, zvolená rotácia môže byť ľubovoľná. To však predstavuje problém pri klasifikácii symetrických povrchov, pretože je možné, že dve triedy môžu vykazovať obsah, ktorý je takmer identický. To bude mať za následok, že klasifikátor nebude schopný spoľahlivo určiť, do ktorej triedy fragment patrí.



Obr. 5.9: Graf znázorňuje pravdepodobnosť správnej klasifikácie obrázka z testovacieho súboru údajov. Zobrazuje osem tried, v ktorých bol aspoň jeden obrázok klasifikovaný nesprávne. Triedy s dokonalou úspešnosťou (223 tried) na všetkých obrázkoch z testovacieho súboru údajov nie sú na obrázku zahrnuté, z dôvodu zachovania prehľadnosti.

Jedným z možných riešení tohto problému je výber polohy fragmentov spôsobom, ktorý zabezpečí, že triedy budú dostatočne dištinktívne.

5.3.1 Budúci rozvoj učenia na základe klasifikácie

Zlepšenie tohto prístupu môže byť predmetom následnej práce. Príkladmi takýchto vylepšení môžu byť:

- **Využitie rozdielnej baseline architektúry** – Tréning bol vykonaný nad reziduálnou konvolučnou sieťou Resnet18 a ResNet50. Vylepšenie výsledkov môže priniesť využitie hlbších architektúr (ResNet101), architektúry EfficientNet alebo novogeneračných architektúr reziduálnych architektúr ResNext, prípadne *vision transformers*.
- **Použitie techník pre zvýšenie robustnosti modelu** – pridanie viac typov augmentácií, vloženie šumu a prípadne generovanie adverzných príkladov môžu byť spôsobmi riešenia zvýšenia robustnosti.

5.4 Učenie pomocou učenia vzdialeností (*diatance training*)

Druhým spôsobom učenia je učenie, pri ktorom sa využíva triplet stratová funkcia. Tento spôsob sa využíva najmä pre úlohy ako sú vyhľadávanie obrázkov, podobnosť textu, či rozpoznávanie tvárí. Cieľom trojitej stratovej funkcie je naučiť sa také *embeddings*, aby vzdialenosť medzi podobnými obrázkami (*anchor* a pozitív) bola menšia ako vzdialenosť medzi rozdielnymi obrázkami (*anchor* a negatív), pričom:

- ***anchor*** – Ide o referenčný obrázok, ktorý porovnávame s inými obrázkami.
- **pozitívny** – Ide o obrázok, s rovnakou triedou ako *anchor*.
- **negatívny** – Ide o obrázok, s rozdielnou triedou ako *anchor*.

Cieľom triplet stratovej funkcie je minimalizovať vzdialenosť medzi *anchor* a pozitívnym obrázkom a zároveň maximalizovať vzdialenosť medzi *anchor* a negatívnym obrázkom. Vzo-

E	BS	LR	OPT	LRAT	IS	ANGLE	H.ANGLE	OFFSET	ED
18	4	0.0001	Adam	0.3 | 0.7	224	<-180;180>	<-55;55>	<-15;15>	512

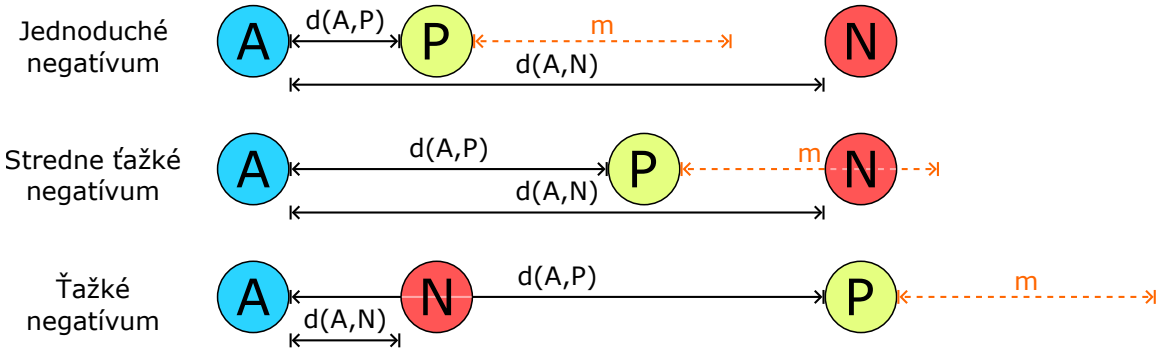
Tabuľka 5.3: Predvolené hyperparametre tréningu, kde `E` je počet epoch, `BS` je batch size, `LR` je rýchlosť učenia, `OPT` je optimalizátor, `LossW` je váha pridelená pre CE a triplet, `IS` je veľkosť obrázku (výška aj šírka), `ANGLE` je interval rotačnej augmentácie, `H.ANGLE` je interval natočenia kamery (homografická rotácia), `OFFSET` je interval translačného posunu od stredu a AF je dimenzia *embedding*-u

rec na výpočet triplet stratovej funkcie je zobrazený vzorcom (3.10). Pri implementácii boli vyskúšané rôzne spôsoby výberu jednotlivých obrázkov (tiež znázornené na obrázku 5.10):

- **Náhodný výber** – jeden zo spôsobov výpočtu triplet stratovej funkcie. Na základe obmedzení na triednu príslušnosť sa vyberú 3 náhodné obrázky. Tento spôsob môže dospieť k výsledku, problémom je, že nemusí byť najúčinnejší ani najefektívnejší. Keďže náhodný výber považuje všetky prvky za rovnako náročné, bez ohľadu na to aká je ich reálna náročnosť. Preto dochádza k plytvaniu zdrojov pri učení pretože model vynakladá rovnaný čas na náročné aj na ľahké príklady. Avšak ľahké negatívne príklady poskytujú modelu veľmi málo nových učených informácií a naopak ťažké negatívne príklady, sú bohatým zdrojom nových učených informácií.
- **Dolovanie ťažkých negatív (*hard-negative mining*)** – technika, ktorá sa zameriava na ťažké negatíva. Týmto spôsobom nútia model naučiť sa viac diskriminačných vlastností a zlepšujú jeho schopnosť rozlišovať medzi triedami. To môže viesť k robustnejšiemu modelu, ktorý dosahuje lepšie výsledky na nevidených dátach. Tiež umožňuje zefektívniť proces trénovania. Zameraním sa na najinformatívnejšie príklady sa môže model naučiť viac informácií z menšieho počtu príkladov, čo môže znížiť potrebný čas, či výpočtové zdroje potrebné na trénovanie.
- **Dolovanie stredne ťažkých negatív (*semi-hard-negative mining*)** – technika, ktorá sa zameriava na stredne ťažké negatíva. Vyberá dvojicu *anchor*-negatív, ktorá má väčšiu vzdialenosť ako dvojica *anchor*-pozitív, ale stále v rámci vzdialenosti danou odsadením (*margin*).

Optimálne výsledky sa dosiahli využitím dolovanie stredne ťažkých negatív pri tréningu. Hoci si trénovanie neurónovej siete vyžiadalo o niečo dlhší tréning, ako pri použití dolovania ťažkých negatív, výsledné *embeddings* pre lokalizáciu vykazovali lepšie výsledky. Druhou najlepšou metódou bolo ťaženie ťažkých negatívov. Trénovanie touto metódou bolo najkratšie, avšak kvalita *embeddings* zaostávala za najefektívnejším prístupom. Trénovanie náhodným výberom si vyžadovalo značný čas a výsledné *embeddings* ani po dlhom trénovaní nedosahovali kvalitu najlepšej metódy. V porovnaní s klasifikáciou je možné zhrnúť niektoré kľúčové rozdiely učenia na základe vzdialenosti:

+ Učenie na základe vzdialeností je menej závislé na počte tried, čo môže byť výhodné pri práci s veľkými obrázkami alebo datasetmi s veľkým počtom tried.
+ Tento prístup môže byť robustnejší voči problémom so symetriou, pretože sa sústredí na vzdialenosti medzi obrázkami, a nie na ich konkrétne triedy.
– Metóda dolovania ťažkých / stredne ťažkých negatív môže byť výpočtovo veľmi náročná.



Obr. 5.10: Typy negatív rozlíšené na základe vzdialenosti. Jednoduché negatíva znázornené vo vrchnej časti obrázka, majú pomalú konvergenciu. Sú to také trojice, pri ktorých platí, že $d(A,P) + m < d(A,N)$. Ďalším typom negatív sú stredné ťažké negatíva, kde naopak platí, že $d(A,P) < d(A,N) < d(A,P) + m$. Posledným typom negatív sú ťažké negatíva, pre ktoré platí že $d(A,N) < d(A,P)$. Predloha obrázku pochádza z [67].

– Implementácia tohto prístupu je v porovnaní s klasifikáciou značne komplexnejšia.

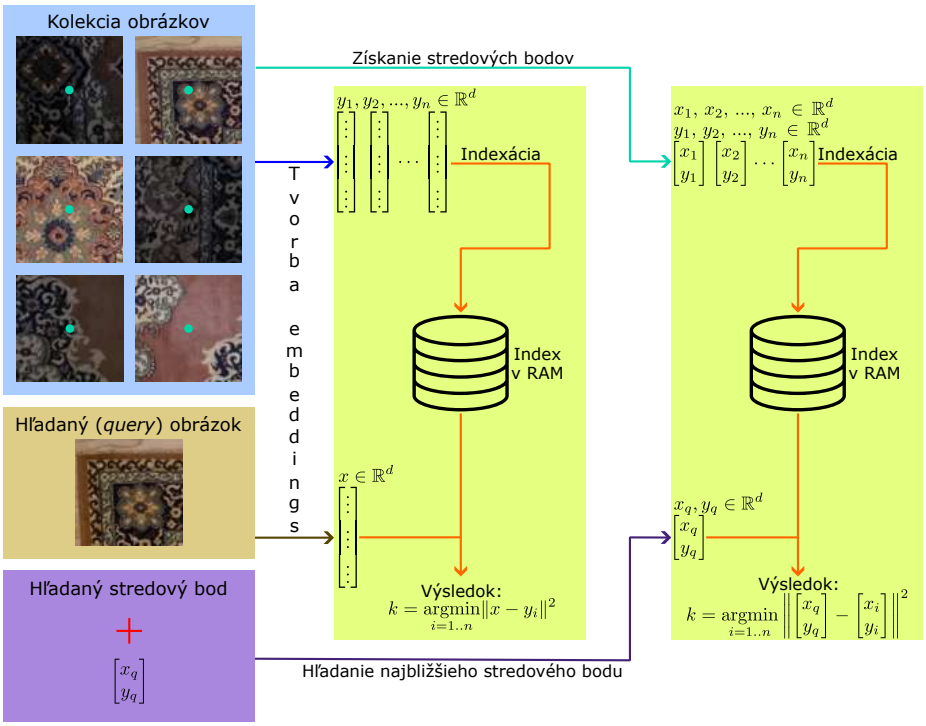
5.4.1 Budúci rozvoj učenia na základe vzdialeností

Zlepšenie tohto prístupu môže byť predmetom následnej práce. Príkladmi takýchto vylepšení môžu byť:

- **Využitie *quadriplet* stratovej funkcie** – *quadriplet* stratová funkcia je jednoduchým zovšeobecnením *triplet* stratovej funkcie pričom namiesto troch využíva štyri obrázky, konkrétne: *anchor*, pozitívny obrázok a dva negatívne obrázky. Táto zmena by mala zabezpečiť menšiu vnútro-triednu variáciu a väčšiu medzitriednu variáciu v *embedding* priestore, čo by malo viesť k lepšiemu výsledku v tomto riešení (triedy, ktoré sú veľmi podobné môžu byť odlíšené).
- **Vylepšenie výberu jednotlivých obrázkov** – Doteraz sa vyskúšali tri metódy vytvárania trojíc. Existujú však aj ďalšie experimentálne metódy, ktoré ešte neboli vyskúšané. Využitie niektorej z týchto metód by mohlo priniesť vylepšenie tréningu.
- **Použitie alternatívnych metrík** – pre hodnotenie epoch a záverečné hodnotenie je možné vyžiť rôzne metriky. Zatiaľ čo stratová funkcia môže byť príliš abstraktná pre hodnotenie modelu, plánujem experimentovať s inými metrikami, ako je skóre F1, krivka ROC a iné, ktoré by mohli poskytnúť komplexnejší pohľad na výsledky modelu.
- **Využitie siamských neurónových sietí** – siamské neurónové siete sú špecifickým typom neurónových sietí určených na spracovanie párových dát. Mám v úmysle vykonať experimenty s týmto typom sietí a porovnať ich výkon so súčasným riešením.

5.5 Vytvorenie databázy *embedings*

Vytvorenie databázy *embeddings* z veľkého obrazu je kľúčovým aspektom procesu lokalizácie (obrázok 5.11). Cieľom tohto postupu je transformovať obraz na sadu *embeddings*, ktoré



Obr. 5.11: Schéma znázorňujúca postup vytvárania indexu pre *queries* kolekcie obrázkov. Prvá časť tohto procesu zahŕňa konverziu mediálnych súborov do *embeddings*, ktorý sa následne uloží do indexu. Na základe tohto indexu sa spracúvajú *queries* s cieľom efektívne získať relevantné výsledky. Následne sa nájdu najbližší susedia ako odpoveď na *query*. V druhej časti sa vyhľadáva stredový bod na základe zadaného stredového bodu. Rovnakým spôsobom sa najskôr vytvorí index stredových bodov. Výsledkom druhej časti je najbližší stredový bod. Obrázok je vytvorený na základe obrázku v článku [29].

obsahujú základné vizuálne príznaky rôznych oblastí v rámci obrazu. Tieto *embeddings* sa následne môžu použiť pre vyhľadávanie najbližšieho obrázku na základe iného vyhľadávaného obrázku (viac o CBIR v kapitole 3.4). Toto umožňuje efektívne lokalizovať najbližší *embedding* v databáze. Na základe hľadaného (*query*) obrázku, ktorý je fragmentom (*patch*) je databáza prehľadaná s cieľom nájsť jemu najpodobnejší fragment (*patch*). Docieli sa to porovnaním *embedding* fragmentu s *embeddings* uloženými v databáze. Obrázky, ktoré sú si podobné, sa v tomto priestore nachádzajú v tesnej blízkosti, zatiaľ čo obrázky, ktoré si nie sú podobné, sú od seba značne vzdialené. Proces tvorby databázy je možné rozdeliť na niekoľko krokov:

- Z obrovského obrázku sa extrahujú fragmenty využitím pravidelného intervalu extrakcie.
- Získajú sa *embeddings* využitím neurónovej siete a to buď klasifikačným alebo dištančným modelom.
- Embeddings všetkých získaných extrakcií sa uložia do indexu FAISS[5].

[5]**index FAISS** – dátová štruktúra navrhnutá pre efektívne vyhľadávanie približného najbližšieho suseda a pre zhlukovanie hustých vektorov

- Okrem *embeddings* sa tiež ukladajú do rozdielneho FAISS indexu aj súradnice stredu každého *embedding* v obrovskom obrázku.
- Tiež sa uložia metadáta k dátam vo FAISS indexe, konkrétne názov fragmentu a jeho súradnice v obrovskom obrázku.

5.6 Program pre lokalizáciu query obrázku

Poslednou a najdôležitejšou fázou realizácie tejto práce bola lokalizácia hľadaného obrazu v obrovskom obrázku pomocou rozsiahlej *embedding* databázy, ktorá bola vytvorená v predchádzajúcom kroku. Fotografia sa najskôr rozdelí na menšie štvorcové výseky a transformuje sa na tenzor (kapitola 5.6.1), pričom *embedding* každého výseku sa určí pomocou natrénovaného modelu neurónovej siete. Následne sa pre každú inštanciu *subquery embedding* identifikuje K najbližších susedov v databáze *embeddings* vytvorenej z obrovskej fotografie (kapitola 5.6.2). Nakoniec sa vytvorí zadaný určený počet náhodných 4-kombinácií nájdených najbližších výskytov približného vyhľadávania na vytvorenie hypotéz homografie (kapitola 5.6.3). Vypočíta sa celková vzdialenosť pre každú kandidátnu homografiu (kapitola 5.6.4) a výsledkom je tá s najnižšou celkovou vzdialenosťou. Výsledná homografia sa potom podľa možnosti spresní pomocou mapovania šablón (kapitola 5.6.5).

5.6.1 Tvorba *subqueries* fragmentov

V tejto fáze lokalizácie sa query obrázok, rozdelí na menšie segmenty (obrázok 5.12) označované ako fragmenty (*patches*). Tento proces je možné rozdeliť na niekoľko krokov:

1. Vypočíta sa počet fragmentov, ktoré možno z obrazu extrahovať na základe rozmerov *query* obrazu a požadovanej veľkosti fragmentu. Tiež sa vypočíta sa plán delenia priestoru rovnomerným spôsobom.
2. Extrakcia políčok sa potom vykonáva systematicky prechádzaním obrázku po mriežke. Okrem samotného obrázku transformovaného na tenzor sa tiež ukladajú súradnice každého fragmentu v *query* a stredový bod každého fragmentu.
3. Proces vracia fragmenty vo forme tenzorov a iné informácie získané v tomto procese (súradnice obrázku v *query*, stredový bod obrázka a iné).

5.6.2 Nájdenie K najbližších *embeddings* pre *subqueries*

Po získaní tenzorov fragmentov (*patches*) je nevyhnutné identifikovať najpodobnejšie *embeddings* (K-nearest) zodpovedajúce istej množine embeddingov fragmentov v rámci databázy embeddingov vytvorenej z obrazu s vysokým rozlíšením. Tento proces je základnou operáciou v systémoch vyhľadávania obrazu na základe obsahu (CBIR). Tento proces pozostáva z nasledujúcich krokov:

1. Pre tenzory fragmentov sa vytvoria *embeddings*, reprezentujúce charakteristické vlastnosti *query* obrazu v kompaktnej forme. Tento krok zahŕňa transformáciu nesprocovaných obrazových dát do menej rozmerného priestoru, ktorý zachytáva jeho základné charakteristiky.

Obr. 5.12: Príklad obrázku rozdeleného na *subqueries*. Vzdialenosti medzi obrázkami sú unifikované. Každý fragment (*subquery*) má pridelenú jednoznačnú farbu. Veľkosť týchto fragmentov je v tomto prípade 224×224.

2. Následne sa vykoná vyhľadávanie s cieľom nájsť K-najbližších susedov pre každý *embedding* fragmentu. Táto operácia vyhľadávania sa vykonáva v priestore *embedding* indexu FAISS, ktorý je vopred vytvorený počas vytvárania databázy z obrazu s vysokým rozlíšením (kapitola 5.5). Cieľom je identifikovať také embeddings, ktoré sú najviac podobné s *embeddings* fragmentov *query* obrázku. Tieto identifikované *embeddings* sa potom uložia na ďalšie použitie.

5.6.3 Náhodné vzorkovanie na základe *subqueries*

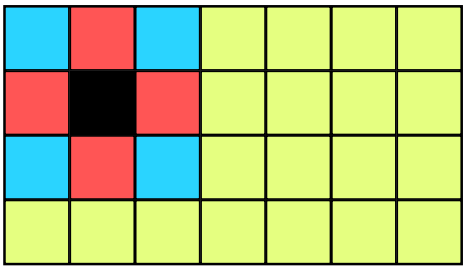
V ďalšej fáze sa vygenerujú náhodné kombinácie indexov, ktoré sú využité na predpovedanie homografie (viac o homografii v kapitole 2.2) na základe stredových bodov malých fragmentov. Index je v tomto prípade poradie fragmentu v rámci všetkých ostatných fragmentov, pričom indexácia prebieha po riadkoch. Pri implementácií tohto prístupu vzniklo niekoľko problémov, ktoré je nutné adresovať:

- Pri výbere štyroch susedných fragmentov vznikali najmä invalidné homografie.
- Pri výbere štyroch fragmentov, ktoré susedné nie sú, dochádzalo k zahadzovaniu mnoho potencionálne validných riešení.

Kompromis, ktorý navrhuje táto implementácia je zobrazený na obrázku 5.13. Užívateľ má možnosť si vybrať z dvoch možnosti susedstva:

- **štvorčlenné susedstvo** – označuje štyri pixely, ktoré sú umiestnené priamo horizontálne a vertikálne k pixelu p.
- **osemčlenné susedstvo** – patria sem susedia **štvorčlenného susedstva** a tiež štyri pixely, ktoré diagonálne susedia s p.

Užívateľ má taktiež možnosť ovplyvniť, koľko susedstiev je v rámci hypotézy povolených. Každá kombinácia indexov tvoriacich hypotézu neobsahuje žiadnych susedov (prípadne obsahuje povolený počet susedstiev). Výsledkom je množina štvoríc spĺňajúca obmedzujúce podmienky.



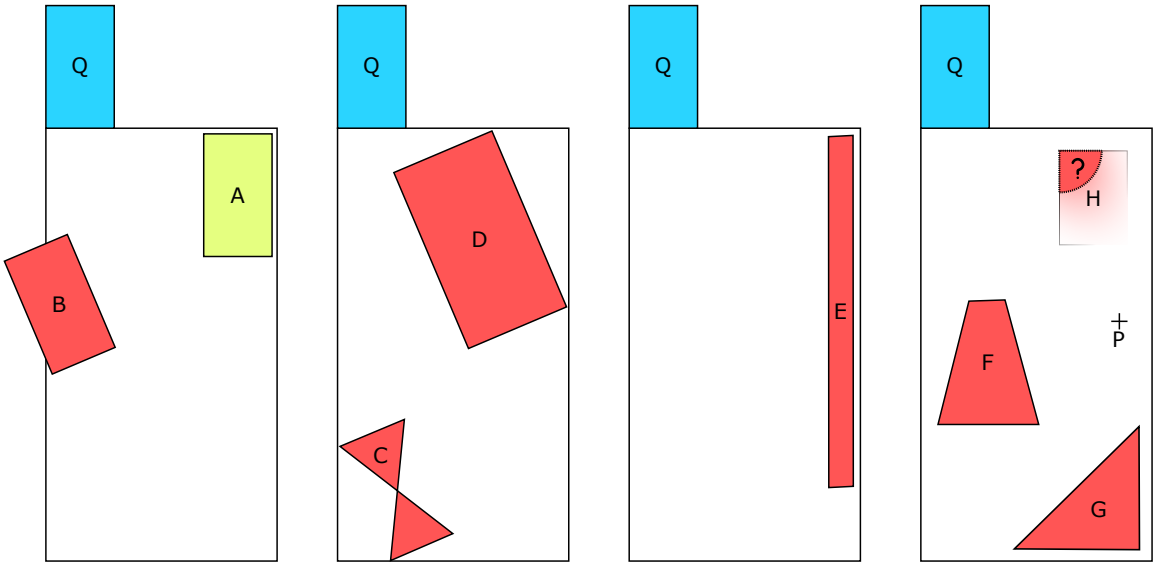
Obr. 5.13: Typy implementovaných susedstiev. Čierny štvorec znázorňuje prvý výber fragmentu. Červené štvorce znázorňujú fragmenty, ktoré nie je možné v indexoch hypotézy použiť, ak sa využíva štvorčlenné susedstvo (indexy $1, 7, 9, 15$). Pri využití osemsusedstva, nie je možné využiť červené (indexy $1, 7, 9, 15$) ani modré (indexy $0, 2, 14, 16$) štvorce. Umiestnenie ďalšieho indexu spôsobí obmedzenie ďalších štvorcov. Je nutné umiestniť 4 indexy.

5.6.4 Tvorba hypotéz homografie, ich filtrácia a ich hodnotenie

V ďalšej fáze je potrebné vyhodnotiť všetky hypotézy. Tieto hypotézy sú určené ako množina štvoríc indexov potenciálnych homografií, vytvorené v predchádzajúcom kroku. Pre každú štvoricu, ktorá bola vytvorená v predchádzajúcom kroku, sa vypočíta potenciálna homografia. Ak je daná homografia platná, vypočíta sa nová poloha (bounding box) *query* obrazu vo veľkom obraze. Nasleduje séria testov, ktoré dôsledne prefiltrujú kvalitné homografie od menej kvalitných (obrázok 5.14). Testy, ktoré sa vykonávajú, zahŕňajú:

- **Test príslušnosti do hraníc obrovského obrázka** – cieľom testovacej fázy je zistiť, či hranice ohraničenia potenciálnej homografie nepresahujú hranice určené obrovským obrazom.
- **Rozpäťový test plochy** – cieľom testovania je zistiť platnosť plochy vymedzenej novo určenou homografiou v porovnaní s plochou *query* obrázka v určenom rozsahu. Tento test slúži aj na overenie správneho poradia uhlov (neplatný štvoruholník).
- **Rozpäťový test zachovania pomeru strán** – cieľom testovania je zistiť platnosť zachovania pomeru strán stanoveného navrhovanou homografiou v súlade s *query*.
- **Testy uhlov** – cieľom testovania uhlov je zistiť, či uhly:
  - neobsahujú hodnota NaN,
  - konvergujú k pravým uhlom pri zadanej odchýlke $(90° - \alpha, 90° + \alpha)$,
  - ich súčet rovná alebo takmer rovná $360°$.

V prípade, že homografia nevyhovuje niektorému z uvedených testov, vylúči sa z ďalšieho výpočtu. Pre platné homografie sa vypočítajú potenciálne pozície ich fragmentov (*subqueries*). Následne sa na základe stredových bodov potenciálnych homografií vypočítajú najbližšie zodpovedajúce stredové body v obrovskom obraze ich vyhľadaním v indexe FAISS so



Obr. 5.14: Na obrázku sú znázornené prípady homografie, ktoré môžu viesť k vylúčeniu z ďalšieho výpočtu, ako sa určilo na základe uvedených testov. Štyri testy sú znázornené na štyroch obrázkoch, pričom červené tvary a homografia P označujú situácie, ktoré vylučujú ďalší výpočet. Zelený tvar predstavuje platnú homografiu, zatiaľ čo modrý tvar predstavuje obraz s *query*. Homografia B presahuje hranice obrovského obrazu. Plocha homografie D presahuje povolenú toleranciu pre povolené rozpätie plochy. Homografia C znázorňuje obrázok s neplatným poradím vrcholov. Obrázok zobrazujúci homografiu E nezachováva pomer strán. Homografia H obsahuje NAN alebo 0 ako jeden z uhlov. Homografia F nezachováva takmer pravé uhly (nepravé uhly sú povolené, ale len po učenú hranicu šikmosti). Homografia G nemá správny počet vrcholov (súčet vnútorných uhlov nemá 360 stupňov). Homografia P degraduje na 1 vrchol.

stredovými bodmi. Takto identifikované stredové body sa použijú na určenie potenciálnych polôh príslušných fragmentov. Vzhľadom na krok výberu vzorky pri vytváraní databázy *embeddings* však tieto polohy nie sú úplne presné. Je potrebné nájsť súradnice najbližšieho vzorkovaného obrazu, ktorý má *embedding*. Najdôležitejšou získanou informáciou je index, ktorý je totožný s indexom v databáze FAISS s *embeddings*. *Embedding* potencionálneho fragmentu je možné nepriamo získať použitím spoločného indexu v oboch databázach FAISS zo stredového bodu (stredový bod → index → embedding). Následne sa pomocou stanovenej metriky vypočíta vzdialenosť medzi potenciálnym homografickým *embedding* fragmentu a jeho skutočným náprotivkom z *query* obrázka. Táto operácia sa vykoná pre všetky vytvorené potenciálne hypotézy. Ohodnotenie celkovej vzdialenosti je vykonané pomocou harmonického priemeru, ktorý je možné vypočítať ako:

$$H = \frac{n}{\frac{1}{d_1} + \frac{1}{d_2} + \frac{1}{d_3} + \ldots + \frac{1}{d_n}}, \tag{5.1}$$

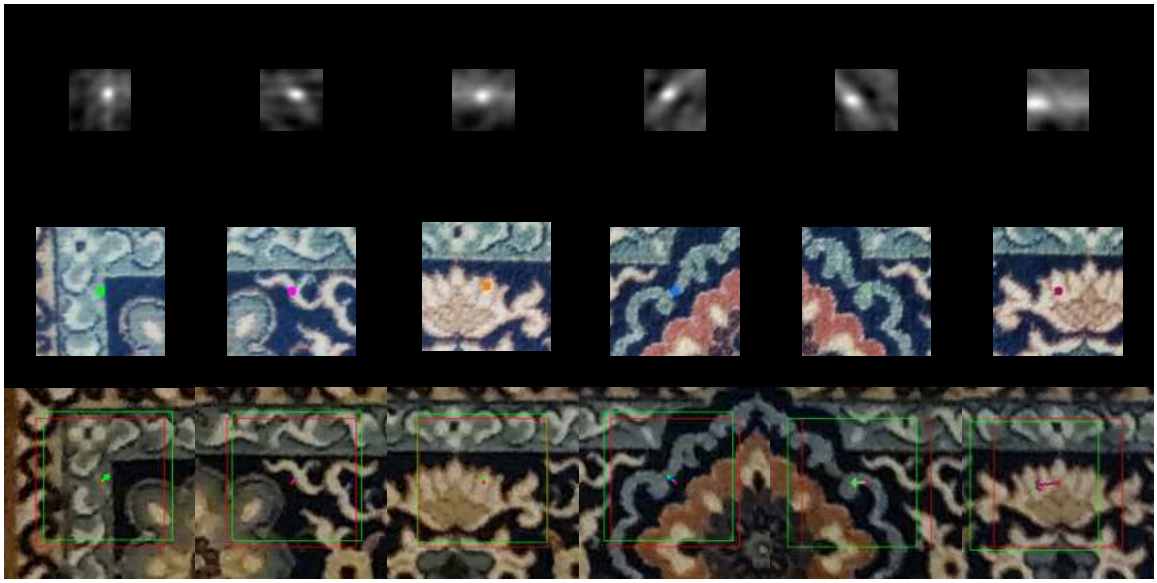
kde H je harmonický priemer, n je celkový počet indexov v jednej hypotéze (v tomto konkrétnom prípade 4), $d_1, d_2, \ldots, d_n$ sú hodnoty vzdialeností medzi potencionálnymi fragmentami a ich skutočnými náprotivkami.

Výsledkom je slovník, ktorý obsahuje všetky platné hypotézy. Hypotéza s najnižšou celkovou vzdialenosťou sa považuje za najpravdepodobnejšiu spomedzi hodnotených hypotéz.

5.6.5 Optimalizácia homografie pomocou mapovania šablón

Poslednou fázou lokalizácie je spresnenie homografie. Výsledkom poslednej fázy (kapitola 5.6.4) je najvernejšia hypotéza. Táto hypotéza však ešte nemusí mať požadované parametre presnosti, preto ju môžeme spresniť pomocou porovnávania šablón (*template matching*) metódou normalizovanej krížovej korelácie, rovnica (2.3). Na tento proces potrebujeme dva obrázky:

- obrázok fragmentu query obrázku s jeho určenou predpokladanou polohou v obrovskom obrázku,
- výrez (rovnako veľkého, prípadne väčšieho) obrázka, ktorý sa na danom mieste obrovského obrázku reálne nachádza.



Obr. 5.15: Na obrázku je znázornené fungovanie techniky porovnávania šablón. V druhom riadku obrázkov sú uvedené obrázky šablón, ktoré sa majú vyhľadať, a zobrazuje sa tiež stredový bod s jeho charakteristickou farbou. Tretí riadok obrázkov ukazuje, ako sa pozícia stredového bodu zmenila v súlade s vypočítanou energetickou mapou spojenou s daným obrázkom (táto mapa je zobrazená v prvom riadku obrázkov). Šípka označuje smer a veľkosť pohybu (vektor) stredového bodu z pôvodného predpokladu (červená bodka) na spresnený predpoklad (zelená bodka). Energetická mapa je znázornená v prvom riadku obrázkov, pričom energia je najvyššia v najjasnejšom bode (toto je prípad NCC).

Tieto dva obrázky umožňujú spresniť polohu fragmentu s presnosťou na pixel. Šablóna je fragmentom *query* obrázka, zatiaľ čo hľadaný obrázok je výrezom z obrovského obrázka. Postup je nasledovný:

1. Obrázky sa konvertujú do odtieňov šedej a aplikuje sa na ne Non-Local Means Denoising. S použitím techniky CLAHE sa použije aj adaptívne vyrovnanie histogramu. Táto technika zvyšuje kontrast obrazu, čím uľahčuje identifikáciu zhôd.
2. Šablóna sa potom posúva po obrázku a pre každú pozíciu sa vypočíta skóre zhody (obrázok 5.15, posledný riadok obrázkov). Výsledkom je matica, v ktorej každá hodnota predstavuje skóre zhody pre šablónu na príslušnej pozícii na obrázku.

3. Miesto s najvyšším skóre zhody je umiestnené v ľavom hornom bode. Potom sa určí stredový bod (vypočítaný ako priemer ľavého horného a pravého dolného bodu) a vypočíta sa posun v globálnych súradniciach.

Tento proces sa vykonáva nad *inliers*, pričom *inlier* v tomto význame znamená takú hodnotu, ktorá zo vstupného zoznamu vzdialeností a ktorá je menšia alebo rovná súčinu multiplikátora a minimálnej hodnoty v zozname. V prípade, že po získaní *inliers* je ich počet menší ako 4, tak sú pridávané najmenšie prvky zo zoznamu, kým nie sú aspoň 4 *inliers*.

V prípade, že dôjde k spresnení homografie, výsledkom je táto spresnená homografia. V prípade, že sa tak nestane, výsledná homografia bude pôvodná optimálna homografia pred spresnením. Táto technika má niekoľko výhod a nevýhod. V nasledujúcom zozname sa budeme venovať niektorým jej významnejším aspektom:

+ Jedná sa o veľmi rýchlu a efektívnu techniku.
+ V prípade žiadnej prípadne malej rotácie dokáže pozíciu upresniť v presnosti pixelov.
– Hoci je táto technika odolná voči rotácii, praktické testy ukázali, že výrazná rotácia môže viesť k problémom.
– Proces spresňovania nie je účinný, ak základná homografia nie je dostatočne presná, čo vedie k tomu, že fragment nie je možné lokalizovať v určenom okolí.
– Keďže multiplikátor zvolených *inliers* je určený empiricky, existuje možnosť, že počet *inliers* môže byť znížený až na 4, čo vedie k riešeniu, ktoré vykazuje rovnaké podmienky vzniku ako pôvodná homografia bez spresnenia. To môže viesť k riešeniu, ktoré je buď lepšie, ale aj horšie ako pôvodné.

Kapitola 6

Experimenty

6.1 Experimenty s tréningom neurónových sietí

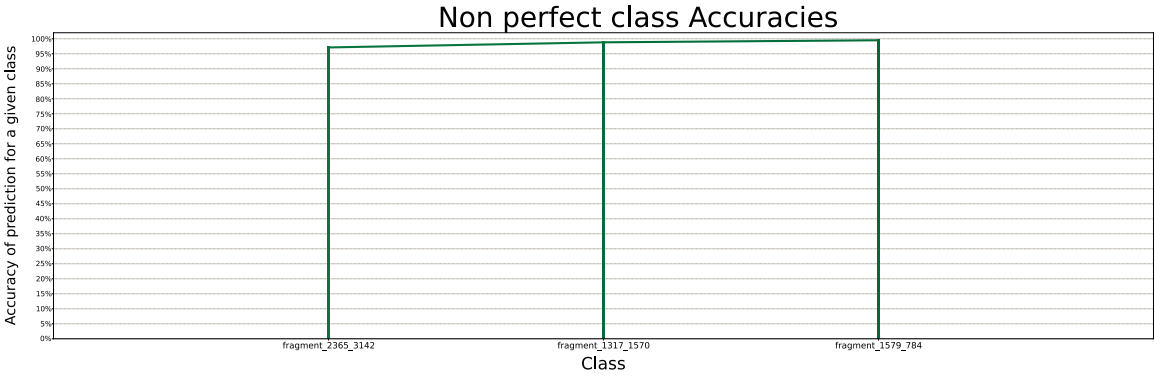
S cieľom určiť optimálnu verziu modelu sa vyhodnotilo niekoľko rôznych alternatív. Všetky experimenty boli vykonané nad datasetom carpet, z dôvodu jeho rozsahu a možnosti testovať vlastnosti, ktoré v zmiešanom datasete testovať nejdú (osová symetria (viď obrázok 6.1), vplyv svetelných podmienok na tréning (viď obrázky 6.3, 6.4) a iné). Cieľom experimentov s neurónovými sieťami bolo určiť optimálnu konfiguráciu hyperparametrov modelu, vybrať stratovú funkciu a optimalizátor s najpriaznivejšími výsledkami a upraviť rozšírenie na zlepšenie zovšeobecnenia modelu. Tréning prebiehal buď na osobnom počítači (NVIDIA RTX 4070Ti) alebo na školskom počítači Sofia (NVIDIA RTX A5500). Experimenty obsahujú parametre, ktoré sa počas tréningu nemenili vzhľadom na architektonický návrh alebo povahu súboru údajov. Tieto parametre sú podčiarknuté. Parametre zahŕňajú:

- **Veľkosť obrázka** – 224×224
- **Počet tried** – 231 (závisí od rozlíšenia a povoleného prekrývania)
- **Počet epoch** – 20
- **Pomer rozdelenia datasetu** – 70% trénovací/ 20% validačný /10% testovací.
- ***Batch size*** – 4/8/16/32/64/128/256
- **Rýchlosť učenia** – 0.0001 až 0.001
- **Optimalizátor** – Adam / SGD
- **Uhol rotačnej augmentácie** – $< -180; 180 >$ – vyjadruje uhol bežnej rotácie okolo svojej osi.
- **Uhol augmentácie homografickej rotácie** – $< -55; 55 >$ – vyjadruje uhol rotácie „s kamerou“.
- **Augmentácia posunu** – $< -15; 15 >$ pixelov
- **Augment factor** – Táto metrika predstavuje počet generovaní rôznych fragmentov z tých istých obrázkov. Používa sa na umelé zväčšenie súboru údajov.

Výsledky experimentov s trénovaním neurónových sietí zobrazujú výsledky, ktoré sa líšili od typických výsledkov a priniesli informácie, ktoré zásadne ovplyvnili postup implementácie. Na zobrazenie boli použité tieto metódy zobrazenia:

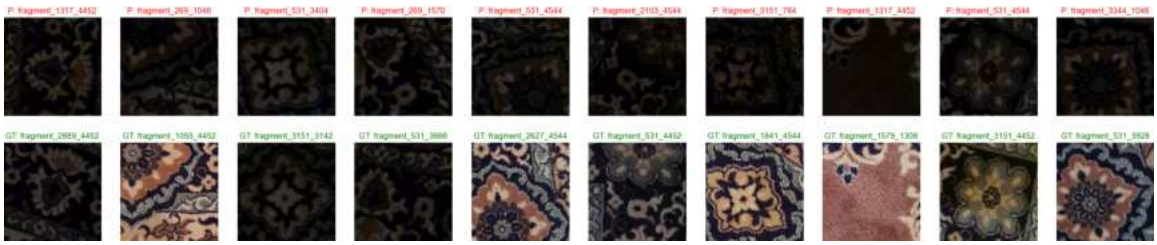
- **Výrezom z *confusion matrix*** – Graf znázorňuje percentuálny podiel obrázkov patriacich do danej triedy, ktoré boli ako také správne identifikované. V ideálnom prípade by výsledná matica mala byť diagonálna. Vzhľadom na vysoký počet tried nie je možné zobraziť celú *confusion matrix* ako obrázok, pretože by to viedlo k strate značnej vizuálnej informácie. Preto sa matica znázorňuje vyrezaním problematickej časti.
- **Graf presnosti počas epoch** – Graf znázorňuje presnosť pre každú epochu.
- **Graf priemernej hodnoty stratovej funkcie / funkcií počas epoch** – Graf znázorňuje hodnotu straty počas jednotlivých epoch. V prípade tréningu viacerými funkciami straty, sú zobrazené všetky z nich (obrázok 6.8).
- **Graf chybovosti na triede** – Znázorňuje presnosť modelu na jednotlivých triedach (obrázok 6.2).

Obr. 6.1: Na obrázku je znázornená matica zámeny (*confucion matrix*), ktorá ilustruje výskyt vysokej chybovosti v určitých triedach. Po preskúmaní týchto prípadov sa zistilo, že dané triedy vykazujú osovo symetrické charakteristiky, čo predstavuje výzvu v podobe problému symetrie materiálu koberca. Riešenie tohto problému zahŕňalo začlenenie posunutia a rotácie do rozšírení.

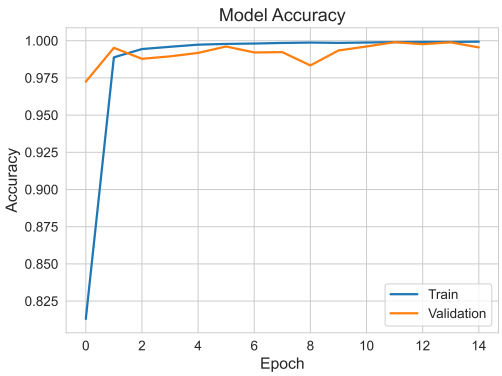


Obr. 6.2: Obrázok znázorňuje výsledok tréningu, v ktorom vznikli tri triedy s nedokonalým hodnotením tried. Po preskúmaní týchto prípadov bol identifikovaný problém, ktorý možno pozorovať na obrázku 6.3, alebo aj obrázok 6.4.

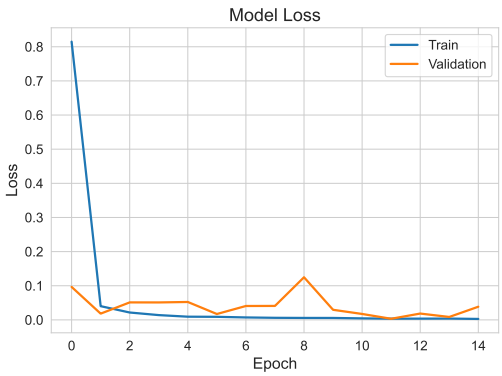
Obr. 6.3: Tento obrázok znázorňuje obrázky, ktoré boli na obrázku 6.2 nesprávne klasifikované. Všetky tri snímky sú tmavé a značnú časť plochy snímky zaberá bezpríznaková oblasť (hnedá plná plocha). Tento problém bol tiež vyriešený výraznejšou augmentáciou.



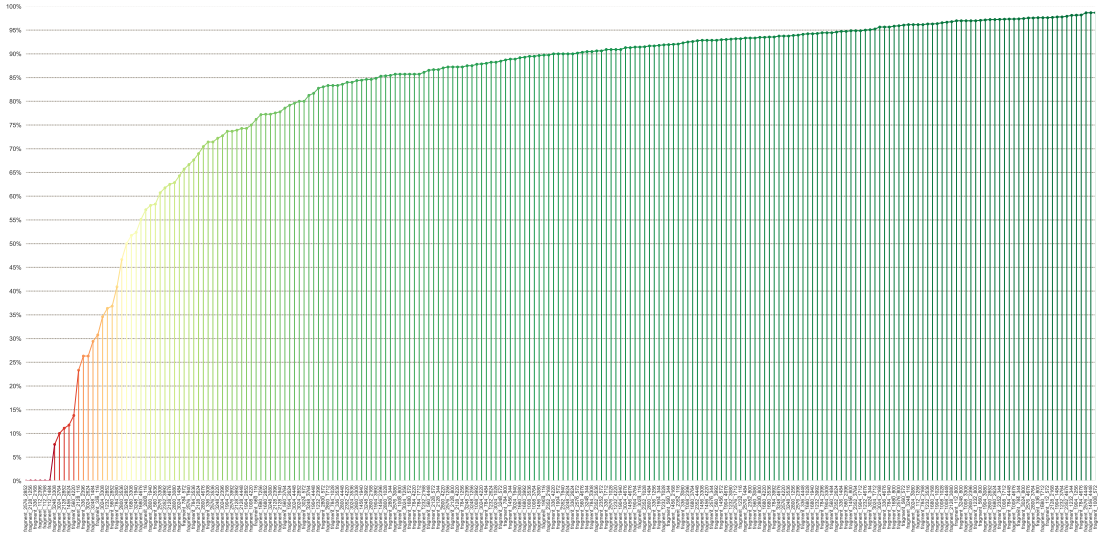
Obr. 6.4: Obrázky z behu, pri ktorom vznikolo pomerne veľa zámen a to aj v prípade svetlého obrázku za tmavý. Zámeny vznikali hlavne v prípade, že sa jednalo o symetriu v rámci koberca (rovnaký fragment na rôznom mieste).



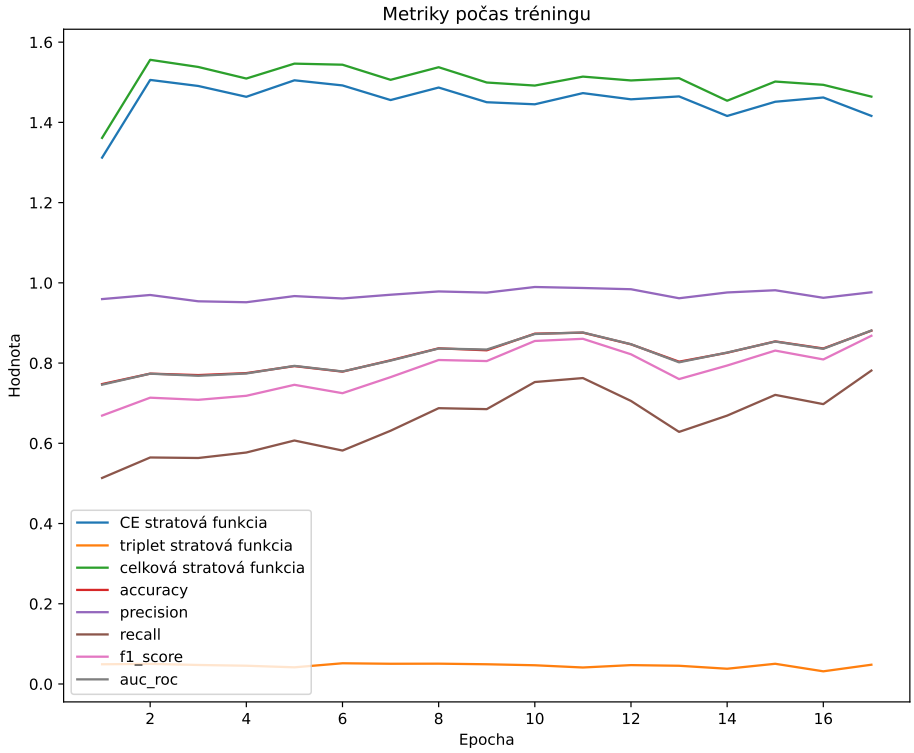
Obr. 6.5: Presnosť modelu pri tréningu 15 epoch nad datasetom s augmentáciami, pričom parametre sú prvé voľby zo zoznamu vyššie.



Obr. 6.6: Hodnota stratovej funkcie modelu pri tréningu 15 epoch nad datasetom s augmentáciami, pričom parametre sú prvé voľby zo zoznamu vyššie.



Obr. 6.7: Tento príklad ilustruje scenár tréningu, v ktorom sa nepoužilo žiadna augmentácia. V tomto prípade je zrejmý vplyv nerovnováhy tried. Niektoré triedy sú skutočne zastúpené veľmi sporadicky a absencia augmentácie spôsobuje, že tento problém je obzvlášť výrazný.



Obr. 6.8: Obrázok znázorňuje tréningové metriky pri použití duálnej stratovej funkcie. Termín „duálna funkcia“ označuje súčasné použitie CE aj *triplet* stratovej funkcie, pričom obe sú vážené.

6.2 Experimenty pri lokalizácii

S cieľom zistiť účinnosť navrhovanej metodiky lokalizácie sa vykonala séria testov s použitím rôznych fotografií koberca. Testy sa uskutočnili čisto vizuálnym spôsobom, pričom miesto, z ktorého bola každá snímka zhotovená, bolo známe. Okrem toho sa vykonalo porovnanie s tradičnými postupmi vrátane SIFT, FLANN a RANSAC. Väčšina snímok sa však pomocou SIFT nachádzala v inom rohu, čo je pravdepodobne spôsobené problémom s osovou súmernosťou (6.13).

Obr. 6.9: Obrázok znázorňuje najbližšie zhody obrázkov identifikované pre *subqueries*. V súčasnom stave bolo pre každý fragment *subquery* identifikovaných päť najbližších zhôd (viď obrázok 6.15). Spojovacie čiary označujú dvojice medzi *subquery* a ich zodpovedajúcimi projekciami vo veľkom obraze. Každý identifikovaný fragment obsahuje aj hodnotu vzdialenosti. Veľký červený štvoruholník predstavuje počiatočnú hypotézu homografie založenú na štyroch zodpovedajúcich bodoch.

Obr. 6.10: Obrázok znázorňuje možné hypotézy homografie fragmentov po projekcii. V tomto prípade existuje šesť potenciálnych hypotéz, pričom sa vyberie tá, ktorá vykazuje najnižšiu vzdialenosť hypotézy.

Obr. 6.11: Obrázok znázorňuje umiestnenie fotografie *query* na veľkú fotografiu na základe jej prvotnej homografie.

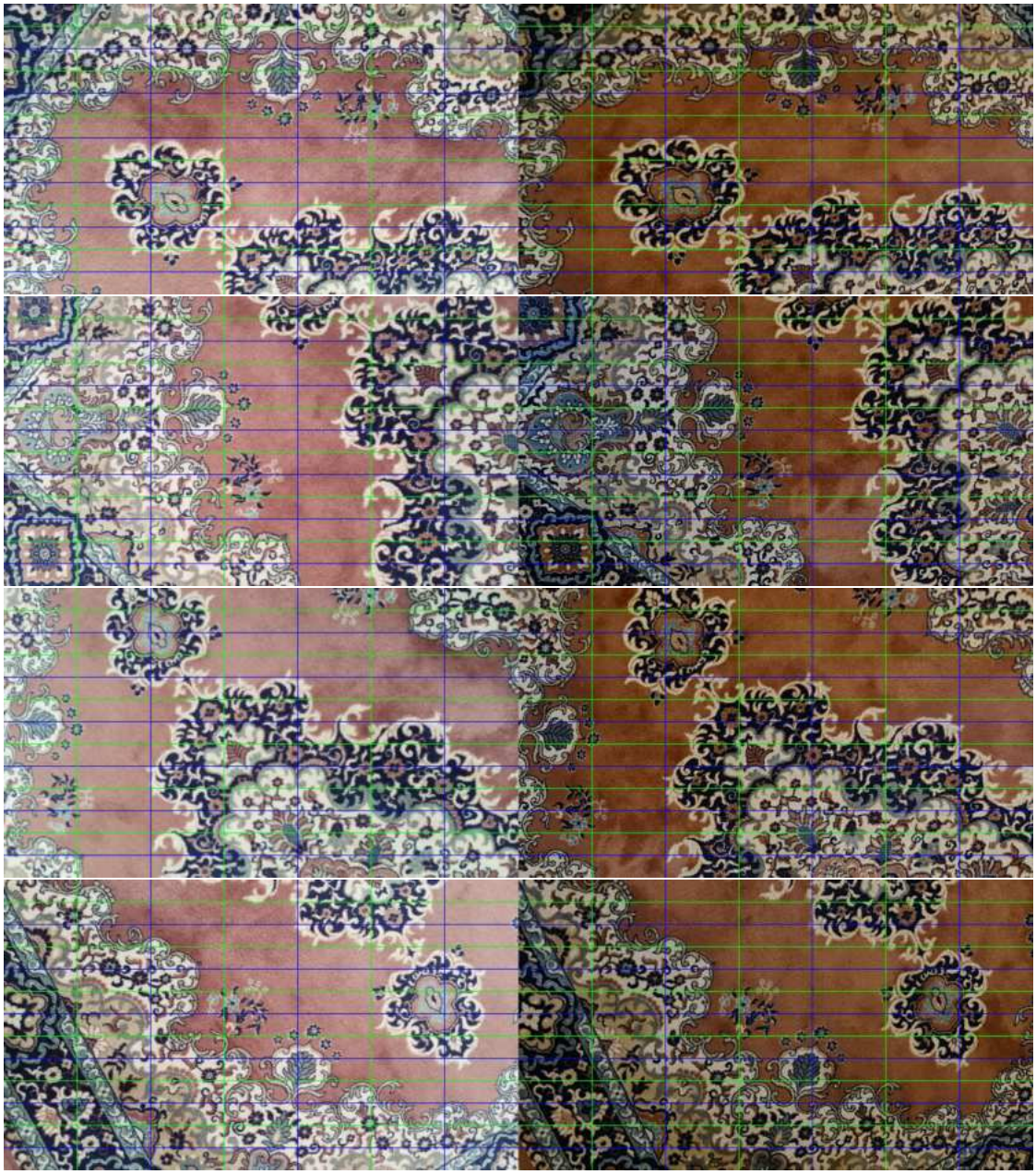
Obr. 6.12: Obrázok znázorňuje umiestnenie fotografie *query* na veľkú fotografiu na základe jej spresnenej homografie.

Obr. 6.13: Obrázok znázorňuje umiestnenie fotografie *query* na veľkú fotografiu na základe jej spresnenej homografie. Riešenie je správne na základe uloženia pozície snímania. Spresnenie odhad vylepšilo.

Obr. 6.14: Na obrázku je modrou farbou znázornený štvoruholník na predpokladanom mieste fotografie s *query*, ako ho určil algoritmus trad. *pipeline*. Tento algoritmus vybral nesprávne stredovo symetrické riešenie.

Obr. 6.15: Obrázok znázorňuje 5 najbližších fragmentov k danej *subquery*.

Obr. 6.16: Obrázok znázorňuje *query* a jej lokalizovanú homografiu v obrovskom obrázku. Na obrázku je nanesená mriežka pre prehľad spoločných miest v oboch obrázkoch.



Obr. 6.17: Obrázok ilustruje, že implementovaná metóda je účinná pri spracovaní celého spektra query obrázkov a preukazuje určitý stupeň nezávislosti od rotácie a translácie. Treba poznamenať, že metóda zatiaľ nie je schopná tolerovať výrazné zmeny zoomu, čo predstavuje potenciálnu oblasť budúceho vývoja.

Obr. 6.18: Obrázok znázorňuje výsledok procesu lokalizácie. Farebné štvorce predstavujú zhodné fragmenty v *query* a na obrovskej fotografii. Tie sú identifikované ako *inliers* alebo fragmenty s najmenšími vzdialenosťami a čiary, ktoré ich spájajú, vyjadrujú zhody *inliers*. Ružový štvoruholník označuje neupresnenú (počiatočnú) homografiu, zatiaľ čo červený štvoruholník vyjadruje homografiu po upresnení pomocou porovnávania šablón.

Kapitola 7

Záver

Cieľom tejto práce bolo získať veľkú fotografiu a následne ju prehľadávať pomocou menšej vybranej fotografie povrchu. Táto práca predstavuje témy vytvorenia veľkej fotografie pomocou počítačového videnia a vyhľadania hľadaného obrazu v nej pomocou reziduálnych neurónových sietí. Bolo preskúmaných viacero prístupov a vykonaných niekoľko experimentov, pričom tie najsľubnejšie boli aplikované v tejto práci. Práca je rozdelená do niekoľkých kapitol, ktoré postupne odhaľujú proces tvorby práce.

V úvodnej časti práce je uvedený súčasný stav a informácie relevantné pre pochopenie problematiky spájania a lokalizácie obrazu, ktoré sú objasnené v literatúre. V nasledujúcej časti sa podrobne opisuje vytvorenie veľkého obrazu zo snímok videa s využitím tradičných prístupov, konkrétne spájania obrazov. Okrem toho je vyvinutá aplikácia na uľahčenie zobrazenia homografie videosnímok po ich získaní, v ktorej môže používateľ pozorovať pokrytie čiastkových datasetov a ich príslušný prínos.

Okrem toho bolo vytvorených niekoľko datasetov obsahujúcich lokalizované fragmenty. Napríklad koberec dátovej sady obsahuje (x,y,z) snímok v závislosti od zvoleného rozlíšenia. Súbory údajov boli vytvorené s použitím rôznych svetelných podmienok, aby sa zvýšila rozmanitosť snímok rovnakého miesta snímania.

Generovanie *embeddings* sa uskutočňuje pomocou reziduálnych konvolučných neurónových sietí. Použili sa dve techniky trénovania: klasifikácia a učenie na základe vzdialenosti. Klasifikačný prístup dosiahol vysokú mieru presnosti (99,8 %) klasifikácie. Avšak ako vhodnejšie riešenie pre problém lokalizácie sa javí metóda dištančného učenia. Trénovanie len pomocou trojčlennej stratovej funkcie prinieslo uspokojivé výsledky. Na urýchlenie trénovania a dosiahnutie rýchlejšej konvergencie sa však použila krížová CE aj *triplet* stratová funkcia spolu s metódami ťažkého alebo stredne ťažkého negatívneho dolovania. Výsledné metriky preukázali presnosť 99.1%, čo znamená, že vkladanie z tejto neurónovej siete vykazovalo zvýšenú kvalitu a vhodnosť na lokalizáciu.

Proces lokalizácie prebieha v *embedding* priestore na základe hľadania najbližších K-susedov v rámci indexu FEISS. Proces lokalizácie vykazuje uspokojivé výsledky pre väčšinu obrázkov, pričom v niektorých prípadoch sa ako prospešná ukázala aj optimalizácia párovania šablón. Experimentálne dôkazy naznačujú, že metóda je schopná presne určiť polohu aj v prípadoch, keď tradičná metóda zlyhala z dôvodu osovej symetrie, či nedostatku zhôd.

Ďalšia práca bude zameraná na zlepšenie neurónovej siete s cieľom dosiahnuť vyššiu kvalitu *embeddings*. Tiež možná automatizácia tvorby datasetu pomocou robota, či pridanie funkcií do aplikácie na zobrazenie homografie, prípadne inými prospešnými zlepšeniami.

Literatúra

[1] *The OpenCV Reference Manual* [https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html]. 2.4.13.7. OpenCV, April 2014.

[2] A, B. a K, D. Analytical Study on Digital Image Processing Applications. *International Journal of Computer Science and Engineering*. Jún 2020, zv. 7, s. 4–7. DOI: 10.14445/23488387/IJCSE-V7I6P102.

[3] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al Dujaili, A., Duan, Y. et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*. Springer Science and Business Media LLC. marec 2021, zv. 8, č. 1. DOI: 10.1186/s40537-021-00444-8. ISSN 2196-1115. Dostupné z: http://dx.doi.org/10.1186/s40537-021-00444-8.

[4] Andoni, A., Indyk, P., Laarhoven, T., Razenshteyn, I. a Schmidt, L. Practical and Optimal LSH for Angular Distance. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M. a Garnett, R., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2015, sv. 28. Dostupné z: https://proceedings.neurips.cc/paper_files/paper/2015/file/2823f4797102ce1a1aec05359cc16dd9-Paper.pdf.

[5] Aumueller, M., Bernhardsson, E. a Faitfull, A. *ANN-Benchmarks*. 2024. Accessed: 2024-05-07. Dostupné z: https://ann-benchmarks.com/index.html.

[6] Ballard, D. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*. 1981, zv. 13, č. 2, s. 111–122. DOI: https://doi.org/10.1016/0031-3203(81)90009-1. ISSN 0031-3203. Dostupné z: https://www.sciencedirect.com/science/article/pii/0031320381900091.

[7] Bernhardsson, E. *Approximate Nearest Neighbors Oh Yeah* [https://github.com/spotify/annoy]. 2005. [Online; Accessed 27-04-2024].

[8] Bian, J., Yang, R., Liu, Y., Zhang, L., Cheng, M.-M. et al. *MatchBench: An Evaluation of Feature Matchers*. 2018.

[9] Burt, P. J. a Adelson, E. H. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.* New York, NY, USA: Association for Computing Machinery. oct 1983, zv. 2, č. 4, s. 217–236. DOI: 10.1145/245.247. ISSN 0730-0301. Dostupné z: https://doi.org/10.1145/245.247.

[10] Cao, S.-Y., Hu, J., Sheng, Z. a Shen, H.-L. Iterative Deep Homography Estimation. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, s. 1869–1878. DOI: 10.1109/CVPR52688.2022.00192.

[11] Cao, S.-Y., Zhang, R., Luo, L., Yu, B., Sheng, Z. et al. Recurrent Homography Estimation Using Homography-Guided Image Warping and Focus Transformer. In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, s. 9833–9842. DOI: 10.1109/CVPR52729.2023.00948.

[12] Chen, W., Chen, X., Zhang, J. a Huang, K. *Beyond triplet loss: a deep quadruplet network for person re-identification*. 2017.

[13] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X. et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021.

[14] Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G. et al. *The Faiss library*. 2024.

[15] Dubey, S. R. A Decade Survey of Content Based Image Retrieval Using Deep Learning. *IEEE Transactions on Circuits and Systems for Video Technology*. 2022, zv. 32, č. 5, s. 2687–2704. DOI: 10.1109/TCSVT.2021.3080920.

[16] Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J. et al. *D2-Net: A Trainable CNN for Joint Detection and Description of Local Features*. 2019.

[17] Elliott, D. L. A Better Activation Function for Artificial Neural Networks. In:. 1993. Dostupné z: https://api.semanticscholar.org/CorpusID:60842824.

[18] Fischler, M. A. a Bolles, R. C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*. New York, NY, USA: Association for Computing Machinery. jun 1981, zv. 24, č. 6, s. 381–395. DOI: 10.1145/358669.358692. ISSN 0001-0782. Dostupné z: https://doi.org/10.1145/358669.358692.

[19] François, C. *Deep learning with Python*. Manning Publications Co., 2018.

[20] Fukushima, K. Visual Feature Extraction by a Multilayered Network of Analog Threshold Elements. *IEEE Transactions on Systems Science and Cybernetics*. 1969, zv. 5, č. 4, s. 322–333. DOI: 10.1109/TSSC.1969.300225.

[21] Fukushima, K., Miyake, S. a Ito, T. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*. 1983, SMC-13, č. 5, s. 826–834. DOI: 10.1109/TSMC.1983.6313076.

[22] Ge, T., He, K., Ke, Q. a Sun, J. Optimized Product Quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2014, zv. 36, č. 4, s. 744–755. DOI: 10.1109/TPAMI.2013.240.

[23] Ghosh, D. K., Kaabouch, N. a Fevig, R. A. Robust Spatial-Domain Based Super-Resolution Mosaicing of CubeSat Video Frames: Algorithm and Evaluation. *Comput. Inf. Sci.* 2014, zv. 7, s. 68–81.

[24] Goodfellow, I., Bengio, Y. a Courville, A. *Deep Learning*. MIT Press, 2016. Adaptive computation and machine learning. ISBN 9780262035613. Dostupné z: https://books.google.co.in/books?id=Np9SDQAAQBAJ.

[25] Goodfellow, I. J., Pouget Abadie, J., Mirza, M., Xu, B., Warde Farley, D. et al. *Generative Adversarial Networks*. 2014.

[26] Grimson, W. E. L. Object recognition by computer - the role of geometric constraints. In:. 1991. Dostupné z: https://api.semanticscholar.org/CorpusID:1530384.

[27] Hartley, R. a Zisserman, A. *Multiple View Geometry in Computer Vision*. 2. vyd. Cambridge University Press, 2004.

[28] He, K., Zhang, X., Ren, S. a Sun, J. Deep Residual Learning for Image Recognition. *CoRR*. 2015, abs/1512.03385. Dostupné z: http://arxiv.org/abs/1512.03385.

[29] Hervé Jegou, J. J. *Faiss: A library for efficient similarity search*. Engineering at Meta, 2017. Accessed: May 5, 2024. Dostupné z: https://engineering.fb.com/2017/03/29/data-infrastructure/faiss-a-library-for-efficient-similarity-search/.

[30] Hochreiter, S. a Schmidhuber, J. Long Short-term Memory. *Neural computation*. December 1997, zv. 9, s. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[31] Hou, B., Ren, J. a Yan, W. Unsupervised Multi-Scale-Stage Content-Aware Homography Estimation. *Electronics*. 2023, zv. 12, č. 9. DOI: 10.3390/electronics12091976. ISSN 2079-9292. Dostupné z: https://www.mdpi.com/2079-9292/12/9/1976.

[32] Hough, P. V. *Method and means for recognizing complex patterns*. Google Patents, december 18 1962. US Patent 3,069,654.

[33] Huang, Q., Feng, J., Zhang, Y., Fang, Q. a Ng, W. Query-aware locality-sensitive hashing for approximate nearest neighbor search. *Proc. VLDB Endow.* VLDB Endowment. sep 2015, zv. 9, č. 1, s. 1–12. DOI: 10.14778/2850469.2850470. ISSN 2150-8097. Dostupné z: https://doi.org/10.14778/2850469.2850470.

[34] Iwasaki, M. *Neighborhood Graph and Tree for Indexing High-dimensional Data Quantized graph-based method* [https://github.com/yahoojapan/NGT]. 2015. [Online; Accessed 27-04-2024].

[35] Iwasaki, M. a Miyazaki, D. *Optimization of Indexing Based on k-Nearest Neighbor Graph for Proximity Search in High-dimensional Data*. 2018.

[36] Kingma, D. P. a Ba, J. Adam: A Method for Stochastic Optimization. *CoRR*. 2014, abs/1412.6980. Dostupné z: https://api.semanticscholar.org/CorpusID:6628106.

[37] Klambauer, G., Unterthiner, T., Mayr, A. a Hochreiter, S. Self-Normalizing Neural Networks. *CoRR*. 2017, abs/1706.02515. Dostupné z: http://arxiv.org/abs/1706.02515.

[38] Kokate, M., Wankhede, V. a Patil, R. S. Survey: Image Mosaicing Based on Feature Extraction. *International Journal of Computer Applications*. Foundation of Computer Science. 2017, zv. 165, č. 1, s. 5.

[39] Krizhevsky, A., Sutskever, I. a Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2012, sv. 25.

[40] Le, H., Liu, F., Zhang, S. a Agarwala, A. Deep Homography Estimation for Dynamic Scenes. *CoRR*. 2020, abs/2004.02132. Dostupné z: https://arxiv.org/abs/2004.02132.

[41] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E. et al. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*. 1989, zv. 1, č. 4, s. 541–551. DOI: 10.1162/neco.1989.1.4.541.

[42] Lederer, J. *Activation Functions in Artificial Neural Networks: A Systematic Overview*. 2021.

[43] Li, W., Zhang, Y., Sun, Y., Wang, W., Li, M. et al. Approximate Nearest Neighbor Search on High Dimensional Data — Experiments, Analyses, and Improvement. *IEEE Transactions on Knowledge and Data Engineering*. 2020, zv. 32, č. 8, s. 1475–1488. DOI: 10.1109/TKDE.2019.2909204.

[44] Liu, J. a Li, X. Geometrized Transformer for Self-Supervised Homography Estimation. In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2023, s. 9522–9531. DOI: 10.1109/ICCV51070.2023.00876.

[45] Lowe, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*. Hingham, MA, USA: Kluwer Academic Publishers. nov 2004, zv. 60, č. 2, s. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94. ISSN 0920-5691. Dostupné z: ^4^.

[46] Luo, Y., Wang, X., Wu, Y. a Shu, C. Detail-Aware Deep Homography Estimation for Infrared and Visible Image. *Electronics*. 2022, zv. 11, č. 24. DOI: 10.3390/electronics11244185. ISSN 2079-9292. Dostupné z: https://www.mdpi.com/2079-9292/11/24/4185.

[47] Malkov, Y. A. a Yashunin, D. A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*. IEEE. 2018, zv. 42, č. 4, s. 824–836.

[48] McCulloch, W. S. a Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*. 1990, zv. 52, č. 1, s. 99–115. ISSN 0092-8240.

[49] Mou, W., Wang, H. a Seet, G. Robust Homography Estimation Based on Nonlinear Least Squares Optimization. *Mathematical Problems in Engineering*. Hindawi Publishing Corporation. Feb 2014, zv. 2014, s. 897050. DOI: 10.1155/2014/897050. ISSN 1024-123X. Dostupné z: https://doi.org/10.1155/2014/897050.

[50] Muja, M. a Lowe, D. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. In:. Január 2009, sv. 1, s. 331–340.

[51] Park, Y., Cafarella, M. J. a Mozafari, B. Neighbor-Sensitive Hashing. *Proc. VLDB Endow.* 2015, zv. 9, s. 144–155. Dostupné z: https://api.semanticscholar.org/CorpusID:9214934.

[52] Peng, D., Gui, Z. a Wu, H. *Interpreting the Curse of Dimensionality from Distance Concentration and Manifold Effect*. 2024.

[53] Pérez, P., Gangnet, M. a Blake, A. Poisson image editing. *ACM SIGGRAPH 2003 Papers*. 2003. Dostupné z: https://api.semanticscholar.org/CorpusID:6541990.

[54] Ramachandran, P., Zoph, B. a Le, Q. V. *Searching for Activation Functions*. 2017.

[55] Rosenblatt, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*. 1958, 65 6, s. 386–408. Dostupné z: https://api.semanticscholar.org/CorpusID:12781225.

[56] Rousseeuw, P. Least Median of Squares Regression. *Journal of The American Statistical Association - J AMER STATIST ASSN*. December 1984, zv. 79, s. 871–880. DOI: 10.1080/01621459.1984.10477105.

[57] Ruder, S. An overview of gradient descent optimization algorithms. *ArXiv preprint arXiv:1609.04747*. 2017.

[58] Sarlin, P., DeTone, D., Malisiewicz, T. a Rabinovich, A. SuperGlue: Learning Feature Matching with Graph Neural Networks. *CoRR*. 2019, abs/1911.11763. Dostupné z: http://arxiv.org/abs/1911.11763.

[59] Schroff, F., Kalenichenko, D. a Philbin, J. FaceNet: A Unified Embedding for Face Recognition and Clustering. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, s. 815–823.

[60] Serengil, S. I. *Swish as Neural Networks Activation Function*. 2018. [Online; accessed 2024-04-21]. Dostupné z: https://sefiks.com/2018/08/21/swish-as-neural-networks-activation-function/.

[61] Sun, Y., Wang, W., Qin, J., Zhang, Y. a Lin, X. SRS: Solving c-Approximate Nearest Neighbor Queries in High Dimensional Euclidean Space with a Tiny Index. *Proceedings of the VLDB Endowment (PVLDB)*. Very Large Data Base Endowment Inc. september 2014, zv. 8, č. 1, s. 1–12. DOI: 10.14778/2735461.2735462. ISSN 2150-8097.

[62] Suárez, I., Sfeir, G., Buenaposada, J. M. a Baumela, L. BEBLID: Boosted efficient binary local image descriptor. *Pattern Recognition Letters*. 2020, zv. 133, s. 366–372. DOI: https://doi.org/10.1016/j.patrec.2020.04.005. ISSN 0167-8655. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0167865520301252.

[63] Szeliski, R. *Computer Vision: Algorithms and Applications*. Springer International Publishing, 2022. Texts in Computer Science. ISBN 9783030343729. Dostupné z: https://books.google.cz/books?id=QptXEAAAQBAJ.

[64] Terven, J., Cordova Esparza, D. M., Ramirez Pedraza, A. a Chavez Urbiola, E. A. *Loss Functions and Metrics in Deep Learning*. 2023.

[65] Tian, Y., Balntas, V., Ng, T., Barroso Laguna, A., Demiris, Y. et al. D2D: Keypoint Extraction with Describe to Detect Approach. In: Springer. *Asian Conference on Computer Vision*. 2020, s. 223–240.

[66] Wang, Q., Zhang, J., Yang, K., Peng, K. a Stiefelhagen, R. *MatchFormer: Interleaving Attention in Transformers for Feature Matching*. 2022.

[67] Wang, Z., Liang, X. a Wang, C. Controllable Text-to-Image Generation with Enhanced Text Encoder and Edge-Preserving Embedding. *Journal of Physics: Conference Series*. Apríl 2021, zv. 1856, s. 012003. DOI: 10.1088/1742-6596/1856/1/012003.

[68] Wang, Z. a Yang, Z. Review on image-stitching techniques. *Multimedia Systems*. 2020, zv. 26, s. 413 – 430. Dostupné z: https://api.semanticscholar.org/CorpusID:214599752.

[69] Ward, G. Hiding seams in high dynamic range panoramas. In: *International Conference on Computer Graphics and Interactive Techniques*. 2006.

[70] Wei, S.-D. a Lai, S.-H. Fast Template Matching Based on Normalized Cross Correlation With Adaptive Multilevel Winner Update. *IEEE Transactions on Image Processing*. 2008, zv. 17, s. 2227–2235.

[71] Wei, S.-D. a Lai, S.-H. Fast Template Matching Based on Normalized Cross Correlation With Adaptive Multilevel Winner Update. *IEEE Transactions on Image Processing*. 2008, zv. 17, s. 2227–2235. Dostupné z: https://api.semanticscholar.org/CorpusID:15472768.

[72] Werbos, P. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science. Thesis (Ph. D.). Appl. Math. Harvard University*. Dizertačná práca.

[73] Xiang, T., Xia, G.-S. a Zhang, L. IMAGE STITCHING WITH PERSPECTIVE-PRESERVING WARPING. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Copernicus GmbH. jún 2016, III–3, s. 287–294. DOI: 10.5194/isprs-annals-iii-3-287-2016. ISSN 2194-9050. Dostupné z: http://dx.doi.org/10.5194/isprs-annals-III-3-287-2016.

[74] Xu, S., Chen, S., Xu, R., Wang, C., Lu, P. et al. Local feature matching using deep learning: A survey. *Information Fusion*. Elsevier BV. júl 2024, zv. 107, s. 102344. DOI: 10.1016/j.inffus.2024.102344. ISSN 1566-2535. Dostupné z: http://dx.doi.org/10.1016/j.inffus.2024.102344.

[75] Yan, Q., Xu, Y., Yang, X. a Nguyen, T. HEASK: Robust homography estimation based on appearance similarity and keypoint correspondences. *Pattern Recognition*. 2014, zv. 47, č. 1, s. 368–387. DOI: https://doi.org/10.1016/j.patcog.2013.05.007. ISSN 0031-3203. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0031320313002112.

[76] Yi, K. M., Trulls, E., Lepetit, V. a Fua, P. *LIFT: Learned Invariant Feature Transform.* 2016.

[77] Zeng, B. *Towards Understanding Residual Neural Networks.* 2019. Diplomová práca. Massachusetts Institute of Technology.

[78] Zeng, H., Deng, X. a Hu, Z. A new normalized method on line-based homography estimation. *Pattern Recognit. Lett.* 2008, zv. 29, s. 1236–1244. Dostupné z: https://api.semanticscholar.org/CorpusID:11786993.

[79] Zhang, H. a Ling, Y. *HVC-Net: Unifying Homography, Visibility, and Confidence Learning for Planar Object Tracking.* 2022.

[80] Zhang, L., Wen, T. a Shi, J. *Deep Image Blending.* 2019.