URBAN LANDMARK DETECTION USING COMPUTER VISION

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea Politehnica Timişoara
în domeniul
INGINERIE ELECTRONICĂ, TELECOMUNICAȚII ȘI TEHNOLOGII INFORMAȚIONALE
de către

Ing. Ciprian-Constantin Orhei

Preşedintele comisiei:	prof.univ.dr.ing. Dan Lascu
Conducător ştiinţific:	prof.univ.dr.ing. Radu Vasiu
Referenţi ştiinţifici:	prof.univ.dr.ing. Gabriel-Miro Muntean
	prof.univ.dr.ing. Corneliu Burileanu
	prof.univ.dr.ing. Corneliu Rusu

Ziua susţinerii tezei: Timișoara, iunie, 2022

Seriile Teze de doctorat ale UPT sunt:

1. Automatică
2. Chimie
3. Energetică
4. Inginerie Chimică
5. Inginerie Civilă
6. Inginerie Electrică
7. Inginerie Electronică şi Telecomunicaţii
8. Inginerie Industrială
9. Inginerie mecanică
10. Ştiinţa Calculatoarelor
11. Ştiinţa şi Ingineria Materialelor
12. Ingineria Sistemelor
13. Inginerie Energetică
14. Calculatoare şi Tehnologia Informaţiei
15. Ingineria Materialelor
16. Inginerie şi Management
17. Arhitectură
18. Inginerie Civilă și Instalații
19. Inginerie Electronică, Telecomunicații și Tehnologii Informaționale

Universitatea Politehnica Timişoara a iniţiat seriile de mai sus în scopul diseminării expertizei, cunoştinţelor şi rezultatelor cercetărilor întreprinse în cadrul Şcolii doctorale a universităţii. Seriile conţin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susţinute în universitate începând cu 1 octombrie 2006.

România, 300223 Timişoara, Bd. Vasile Pârvan 2B
Tel./fax 0256 404677
e-mail: editura@upt.ro

Foreword

This thesis was written during my doctoral studies, which took place within the Department of Communications from the Politehnica University of Timisoara. The research tackles aspects regarding landmark detection in urban scenarios, using Computer Vision algorithms.

The research topic chosen combines my continuous growing interest in the Computer Vision domain and the new solution of Artificial Intelligence with the effect that technologies can bring to day-to-day activities. I hope that the work presented in this thesis could and will be used in cultural and tourist activities done to promote the city of Timişoara in the future.

I would like to extend my thanks to the team of the Multimedia Research Centre, with whom I worked and from whom I learned a lot during the years spent at the university. Special thanks go to my colleagues Silviu Vert and Muguraș Mocofan, whose expertise and willingness helped me overcome blocking points in my research.

Parts of the work for this thesis was done in collaboration with academia colleagues Cosmin Bonchiș and Victor Bogdan, from West University of Timişoara. I am thankful for their advice and support.

I am grateful to my PhD scientific coordinator Professor Radu Vasiu for his patience and valuable guidance during the years spent in this endeavor. His continuous involvement assured me that the research I did would not lose its scientific relevance.

I would like to extend my gratitude to Professor Marius Lobază who helped me find an appropriate track to follow in a difficult time in my life, a moment that indirectly made this thesis possible.

I am deeply grateful to my parents, Cornelia and Toma, and to my sister, Loredana, for their continuous push and support from the beginning of this, or any other, venture. Lastly, I would like to acknowledge and extend my deepest gratitude to my wife, Oana, for her love, encouragement, sacrifice of countless moments of quality time, and patience, all in order to make this "final push" and finish the thesis.

Timişoara, March 2022 Ciprian-Constantin Orhei

For my family.

ORHEI, Ciprian-Constantin

Urban landmark detection using Computer Vision

Teze de doctorat ale UPT, Seria X, Nr. YY, Editura Politehnica, 2022, 134 pagini, 80 figuri, 14 tabele.

ISSN:
ISBN:

Cuvinte cheie
Vederea artificială, Prelucrarea imaginilor, Detectarea reperelor urbane, recunoașterea reperelor urbane

Rezumat

Teza de față abordează tematica detecției și recunoașterii clădirilor reper într-un scenariu urban, folosind tehnici din domeniul vederii artificiale. Întrucât clădirile și monumentele reper servesc ca un magnet spațial în domeniul cultural al oricărei urbe, această teză se va focusa pe mediul urban timișorean. Detecția automată a unei clădiri nu este o sarcină facilă și adesea are nevoie de surse de informație multiplă. Această sarcină devine și mai dificilă într-un mediu urban unde densitatea de clădiri este mai mare și numărul de distractori este considerabil.

Studiul critic prezentat în cadrul acestei teze acoperă metodele și tehnologiile folosite până în prezent pentru rezolvarea problemei complexe de a localiza și recunoaște o clădire corect. Pentru ca studiul să fie cuprinzător, acesta tratează și problema bazelor de date, respectiv a posibilităților de simulare a unui astfel de sistem local.

Autorul propune un sistem de detecție și descrie implementarea unui prototip de aplicație mobilă care detectează și localizează clădiri repere din cadrul spațiului urban timișorean. Pentru îndeplinirea acestui scop este propusă și o metodă nouă de îmbunătățire a calității imaginii înainte de procesare, respectiv o bază de date cu repere din Timișoara. Sistemul propus este evaluat și folosind baze de date cu repere din alte zone urbane.

Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ADC	Asymmetric distance computation
A-KAZE	Accelerated KAZE features
AKM	Approximate k-Means
AOS	Additive Operator Splitting
AP	Average Precision
AR	Augmented Reality
AVS	Adaptive Vision Studio
B_S	BOF cluster size
BBF	Best Bin First
BOF	Bag of Features
BoVW	Bag of Visual Words
BoW	Bag of Words
BPBR	Biologically-plausible building recognition
BSDS500	Berkeley Segmentation Data Set and Benchmarks 500
CBIR	Content-Based Image Retrieval
CMP	The Center for Machine Perception Façade Database
CMYK	Cyan-Magenta-Yellow, Black
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CV	Computer Vision
D_GPS	Distance in m between GPS tags
D_NL	A-KAZE number of layers
D_NO	A-KAZE number of octaves
D_S	A-KAZE descriptor size
D_THR	A-KAZE threshold
DL	Deep Learning
EC50K	European Cities 50K Dataset
ECP	Ecole Centrale Paris Façade Dataset
EECVF	End-to-End Computer Vision Framework
EM	Expectation maximization
ER	Classification Error Rate
eTRIMS	eTraining for Interpreting Images of Man-Made Scenes
F1	F-measure
FED	Fast Explicit Diffusion
FLANN	Fast Library for Approximate Nearest Neighbors
FN	False Negative
FoM	Figure of Merit
FP	False Positive
GLD	Google Landmarks Dataset
GPS	Global Positioning System
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HBR	Hierarchical building recognition
HCA	Hierarchical agglomerative clustering
HE	Histogram Equalization
HIS	Hue Saturation Intensity Value
HP	High Pass Filter

HPAT	Hyper-polyhedron with adaptive threshold
HSV	Hue Saturation Lightness Value
HV	Human Vision
INRIA	The Paris Art Deco Façade Dataset
ISP	Image Signal Processing
KNN	K nearest neighbors
LDA	Linear discriminant analysis
locVLAD	Location aware Vector of Locally Aggregated Descriptors
LoG	Laplacian of Gaussian Filter
mAP	Mean of the Average Precision
MIoU	Mean Intersection over Union
ML	Machine Learning
M-LDB	Modified-Local Difference Binary
M-R R-MAC	Multi-Resolution R-MAC
MSERs	Multi-scale maximal stable extremal regions
NN	Nearest Neighbor
NNDRS	Nearest Neighbor Distance Ratio Scoring
OBD	The Oxford Buildings Dataset
P	Precision
PBD	The Paris Building Dataset
PCM	Pixel Corresponding Metric
PDE	Partial Differential Equations
PFIP	Parallel Image Processing Framework
PKUBench	Peking University Benchmark
R	Recall
RATTLE	The R Analytical Tool To Learn Easily
ResNet	Residual Network
RFBR	Relevance feedback-based building recognition
RGB	Red-Green-Blue
R-MAC	Regional Maximum activations of convolutions
SBID	Sheffield Building Image Dataset
SF	Spatial Frequency
SFBR	Steerable filter-based building recognition
SFoM	Symmetric Figure of Merit
SIFT	Scale-Invariant Feature Transform
SIMD	Single Instruction Multiple Data
SURF	Speeded-Up Robust Features
SVM	Support vector machine
SVT	Scalable Vocabulary Trees
THR	Threshold for distance for feature matching
TMBuD	The Timişoara Building Dataset
TN	True Negative
TP	True Positive
UM	Unsharp Masking
UM_D	UM smoothing dilated filter size
UM_S	UM strength of smoothing
VLAD	Vector of Locally Aggregated Descriptors
VSM	Vector-space mode
YUV	Luma component, blue projection, red projection
ZuBuD	Zurich Building Database

1. MOTIVATION

The first chapter is a presentation of the motivation that led me in choosing this topic for the thesis. Accordingly, it will highlight the relevance and period for the explored topic. This presentation will naturally expose the assumptions made within the thesis. The chapter also lists the scientific works that I published during my research activity, and it concludes by explaining the structure of the thesis.

1.1. General overview of the selected subject

Landmarks are typically defined from two perspectives: one as an object or structure that is easy to view and to recognize, and the second as a building or place that has an important historical importance. An urban landmark has been defined as an object that provides "external points of orientation, usually an easily identifiable physical object in the urban landscape" [1].

For several centuries humans created landmarks, buildings or monumental structures, to serve cities to promote their cultural status or in the direction of global metropolis [2]. Urban landmarks have a prerequisite of "special character" or "aesthetic interest" and do not have to be huge in size to be considered as such by the local inhabitants [3].

Landmarks in an urban area serve as "spatial magnet" in which cultural, civic, or economical activities take place. In this sense, they have become an important aspect in multiple domains related to tourism and culture. In all the mentioned domains, multimedia technology continuously enhances the experience of the participants and so the interest for urban landmarks has slowly grown in the scholar communities.

Identifying and locating an urban landmark is an activity that naturally blends several research domains like image signal processing (ISP), computer vision (CV), augmented reality (AR). This blending of multiple domains was the first trigger that determined me to choose this research topic for the thesis.

Urban landmark detection is not a facile task. Depending on the technology that is being used, urban landmark detection can face different challenges. One of the first important inputs for such an activity is the global positioning system (GPS), but this approach reaches its limitation because of the physical positioning of the landmarks (multiple interest points cluster in proximity). With the evolution of the imaging domain, using image cues proves to be a natural step forward in this direction. But, of course, with new opportunities appeared new challenges. Using satellite images for this task renders it hard to be used in mobile applications. On the other hand, using street view images renders it prone to visual impairment or obstruction of the field of view.

With all the challenges from the last years, this domain has evolved substantially and by fusing together multiple technologies, it offers suitable solutions that become the backbone of multiple AR tourism applications.

From a CV perspective, the detection of urban landmarks can fall in the category of object detection or image classification. This specific task is not a facile

one, as stated before, having the initial requirement to tackle at the same time two challenges: one is offering a robust object detection algorithm and the second to offer a reduced computational complex system. This complex problem is blended with the system's need to offer a scalable solution that can be used on multiple mobile devices.

The CV challenges that this domain has to offer was one of the most important aspects when deciding on this topic. The research in this direction felt one worth tackling and through this thesis I aim to offer some solutions to advance this vast and complex topic.

In my opinion, one of the most interesting cultural applications that appeared due to urban landmark detection is urban digital storytelling. Digital Storytelling is a "successful combination between the ancient and time-proven art of telling a story and the possibilities of technology today" [4]. In the case of urban environments, digital storytelling can address or define the relationship between us and the city environment [5].

The human interaction between landmarks and the ecosystem of the cities has become an interesting topic for me because of two events: one is the granting of the 2023 European Capital of Culture[1] award to my hometown, Timişoara, and the other one consists of my interactions with the project Spotlight Heritage Timişoara[2].

The European Capital of Culture award was initiated by the European Union with the aim that the elected city would generate considerable benefits in the cultural, social, and economic domain. The award is annual and has the scope of triggering urban regeneration for the awarded city. Timişoara was awarded this first for the year 2021, but it was postponed due to COVID situation for 2023.

Spotlight Heritage Timişoara is a digital cultural initiative of the eLearning Centre and the Multimedia Centre from the Politehnica University of Timişoara. The scope of this project is to reveal, by digital storytelling, the city of Timişoara through stories of cultural and historical heritage. The aim is to bring stories of neighbourhoods, communities, and inhabitants of yesterday in the spotlight using the technologic advantage that today has to offer.

As my professional experience is one focused on software development and CV algorithm development, I was clearly attracted by the challenges that urban landmark detection has to offer. The panoply of challenges for this domain can be as following: finding detection system solutions that can run on a mobile device, enhancing images to be able to utilize it in unfriendly weather conditions, creating a suitable region of interest from the image to optimize the detection or just tackle the concept of creating an offline simulation system suitable for this task.

As a result of this thesis, I wish to offer an urban landmark detection solution, from a street view perspective, that can be utilized in a mobile solution for an AR tourism application. This direction desires to exploit the continuous development of user applications aimed for Timişoara European Capital of Culture 2023.

In this thesis I will attempt to answer the following research questions:

1. What is the state of the art in urban landmark detection using mobile cameras imaging?
2. What should a simulation framework offer to be considered as a suitable solution for processing systems of this nature?
3. What image signal processing algorithms enhance the image to obtain a better detection in this case?

[1] https://en.wikipedia.org/wiki/European_Capital_of_Culture
[2] https://spotlight-timisoara.eu/

4. What are the challenges in creating an urban landmark detection solution tailored for the Timişoara use-case?

1.2. List of published articles

During my research on this subject, I presented several papers to international conferences and published scientific articles in journals. They helped me to validate my work and my research hypotheses. The following list of published articles, both as author and co-author, has been produced in connection and during the work on my thesis:

1. Silviu Vert, Oana Rotaru, Diana Andone, Miruna Antonica, Adina Borobar, **Ciprian Orhei** and Victor Holotescu. "Towards User Experience Guidelines for Mobile Augmented Reality Storytelling with Historical Images in Urban Cultural Heritage" [Extended Abstract], In 7th International XR Conference, Lisbon, Portugal, 27-29 April 2022.
2. Sirbu, Cristina Laura, Cristian Tomoiu, Szilvia Fancsali-Boldizsar, and **Ciprian Orhei**. "Real-time line matching based speed bump detection algorithm." In 2021 IEEE 27th International Symposium for Design and Technology in Electronic Packaging (SIITME), pp. 246-249. IEEE, 2021.
3. **Ciprian Orhei**, Lucian Radu, Muguraș Mocofan, Silviu Vert, Radu Vasiu. "CBIR for urban building using A-KAZE features." In 2021 IEEE 27th International Symposium for Design and Technology in Electronic Packaging (SIITME), pp. 218-221. IEEE, 2021.
4. **Ciprian Orhei**, Victor Bogdan, Cosmin Bonchiș, and Radu Vasiu. "Dilated Filters for Edge-Detection Algorithms." Applied Sciences 11, no. 22 (2021): 10716, WOS:000725536600001 (Q2 journal).
5. Silviu Vert, Diana Andone, Andrei Ternauciuc, Vlad Mihăescu, Oana Rotaru, Muguraș Mocofan, **Ciprian Orhei**, and Radu Vasiu. "User Evaluation of a Multi-Platform Digital Storytelling Concept for Cultural Heritage." Mathematics 9, no. 21 (2021): 2678, WOS:000750692000001 (Q1 journal).
6. **Ciprian Orhei**, Silviu Vert, Muguraș Mocofan, and Radu Vasiu. "TMBuD: A dataset for urban scene building detection." In International Conference on Information and Software Technologies, pp. 251-262. Springer, Cham, 2021.
7. **Ciprian Orhei**, Muguraș Mocofan, Silviu Vert, and Radu Vasiu. "An analysis of ED line algorithm in urban street-view dataset." In International Conference on Information and Software Technologies, pp. 123-135. Springer, Cham, 2021.
8. **Ciprian Orhei**, Muguraș Mocofan, Silviu Vert, and Radu Vasiu. "An automated threshold Edge Drawing algorithm." In 2021 44th International Conference on Telecommunications and Signal Processing (TSP), pp. 292-295. IEEE, 2021, WOS:000701604600063.
9. **Ciprian Orhei**, Silviu Vert, Muguraș Mocofan, and Radu Vasiu. "End-To-End Computer Vision Framework: An Open-Source Platform for Research and Education." Sensors 21, no. 11 (2021): 3691, WOS:000660684600001 (Q1 journal).
10. **Ciprian Orhei**, Muguraș Mocofan, Silviu Vert, and Radu Vasiu. "End-to-End Computer Vision Framework" In 2020 International Symposium on Electronics

and Telecommunications (ISETC), pp. 1-4. IEEE, 2020, WOS:000612681000017.

11. **Ciprian Orhei**, Silviu Vert, and Radu Vasiu. "A Novel Edge Detection Operator for Identifying Buildings in Augmented Reality Applications." In International Conference on Information and Software Technologies, pp. 208-219. Springer, Cham, 2020.
12. **Ciprian Orhei**, Victor Bogdan, and Cosmin Bonchiș. "Edge map response of dilated and reconstructed classical filters." In 2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pp. 187-194. IEEE, 2020, WOS:000674702000028.
13. Bogdan, Victor, Cosmin Bonchiș, and **Ciprian Orhei**. "Custom dilated edge detection filters." In Journal of WSCG 2020, 28, 161–168.

1.3. Structure of the PhD thesis

The thesis is structured in several chapters that are described below.

Chapter 1 is an exposition of my motivation towards choosing the subject of this thesis. With the brief introduction I wish to explain the interconnections of multiple domains that formed the foundation of my decision to choose this research topic. The chapter includes a list of published papers for which I was author or a co-author during the period of research for the thesis. The last subchapter is this short presentation of the structure of the thesis.

Chapter 2 offers an overview of the urban landmark detection domain, from general aspects of this field to specific subdomains of content-based image retrieval (CBIR) system. The chapter focuses on presenting the domain ecosystem with all the challenges and solutions that literature has to offer. Besides the critical review of existing solutions, this chapter will offer a presentation and comparison of available benchmarking datasets.

Chapter 3 aims to present my proposed simulation system. The capability of simulating a system offline is an important one with considerable benefits in development. End-to-End Computer Vision Framework (EECVF) [6], [7] is an open source, Python based framework with the goal to offer a flexible and dynamic tool for researching. To justify the decision, the chapter contains a brief evaluation of other similar frameworks.

Chapter 4 presents a proposed image sharpening algorithm that has low computational cost and is based on dilated filters [8], [9]. The proposed algorithm is evaluated on several use-cases that can appear in landmark detection systems, to better understand the benefits of it. The introduction to the chapter describes the concept of dilated filters and the literature results that triggered this idea.

Chapter 5 presents the proposed landmark detection algorithm with a deep dive in each constructing block of it. I tried, for each architectural decision inside the algorithm, to explain and justify it in our given use-case context. The evaluation of the proposed landmark detection algorithm was performed using several popular datasets, presented in Chapter 2, plus the Timişoara specific dataset that was created for this scope. The algorithm is benchmarked with a series of unique objects (buildings) from Timişoara, which are comprised of images taken from different angles, different weather conditions and even different times of the day (night and day).

Chapter 6 is the concluding part of the thesis. I start with some general conclusions regarding the research that I have done. Afterwards, I continue with enumerating theoretical and practical contributions that this thesis brings in the scientific world. In the end, I present some future research directions that were discovered when constructing this thesis.

2. URBAN LANDMARK DETECTION

2.1. Introduction

Urban scenarios understanding and their reconstruction is a research area in which CV and AR meet with consistent benefits. Successful exploits of AR in this domain focus on the culture and tourism domain [4], [10]–[13].

Building or landmark recognition in urban environments aims to distinguish between different unique classes in a large-scale image dataset [14]. This blend of technologies with the scope of landmark recognition is used in other several domains nowadays like computer gaming, urban planning, entertainment industry, movie making or digital mapping for orientation.

An interesting domain that uses building detection and urban scene understanding is Urban Reconstruction. According to [15] for this complex task of urban reconstruction four inputs are possible: airborne imagery, airborne LiDAR, ground imagery or ground LiDAR. My scope falls clearly in ground imaginary and aims to produce results that can be future used for image-based modelling, façade decomposition or modelling or even ground reconstruction of landmarks or cities.

With the advances of cameras and mobile devices, the generation of digital visual information has become abundant and facile. Today, nomadic pedestrians are accompanied by the mobile phone through everyday life in its urban environment. CV is the key in utilizing billions of images as a cues for context and object awareness, positioning, and environment understanding [16].

In general landmark recognition is a challenging task in the CV domain. Several challenges can occur when trying to fit several images from the same place as changes in illumination conditions and viewpoint, or the presence of distractors such as trees, people, or signs. In order to mitigate these challenges, the existing approaches rely on feature description with a certain degree of invariance to scale, orientation and illumination [17].

To better understand the challenges at hand I divided them into smaller problems that I treated separately: experimental datasets for landmark recognition, datasets for understanding the environment, detection solutions and specific challenges, benchmarking the detection.

In the field of landmark recognition, it is mandatory to use public datasets, with the clear advantage to have a fair and immediate comparison with competitive approaches. Different datasets offer different perspectives and views from different cities which can bring forward different problematics. The available dataset that focuses only on landmarks are described in chapter 2.2.

In chapter 2.3 I present a literature review regarding datasets that offer a semantic understanding of the urban environment. The understanding of the environment is an important topic for many applications that need to tackle the urban scenario.

The existing solutions that I could find in the literature are described in chapter 2.5. This section has the scope of describing the domain in which the landmark detection solution proposed coexists.

The need of evaluating landmark recognition is an important aspect before we can approach the algorithm review. In the same scope in chapter 2.4 evaluation schemes will be presented, benchmarks that would utilize the dataset to quantify the quality of the detection.

2.2. Building recognition datasets

Benchmarking a building detection system is not a trivial thing to do. This is a complex task due to differences and variety in both sides, detection algorithm and benchmark scheme. Constructing an overall fitting benchmark is close to impossible due to different cues used in the detection pipeline or unicity of the landmarks used in the scope. For example, if we consider a detection system that uses visual cues together with GPS cues to benchmark it on a dataset only with visual data will result in evaluating only half of the systems capabilities.

In this chapter I conducted a literature review of the existing benchmarking solutions with the scope of highlighting what is unique for each case. In Table 2.1 I present an overview of the benchmark datasets focusing on the following: year of appearance, the number of images provided(scale), the number of unique landmarks offered (classes), the number of images for each landmark and if the landmarks offer the GPS information or not. The last aspect that I was interested in is if the view of the dataset is strictly from a street-view perspective(view).

For my review I included only datasets that contain a considerable number of landmarks as unique classes and have the focus on image cues. In this direction, I will eliminate popular datasets like: Geotagged StreetView [18] which does not offer possibilities to evaluate the landmarks; Rome 16k [19] which focuses more on the capability of reconstructing the 3D view of an landmark; San Francisco [20] that consist of images plus geo-location but does not offer an evaluation set or landmark grouping; Landmarks-PointCloud [21] due the fact that is non-deterministic hence the evaluation dataset is random generated and focuses more on point cloud data.

Zurich Building Database (ZuBuD) dataset contains 1005 images for 201 unique landmarks representing buildings from Zurich City. For each landmark five arbitrary angles were selected, with the resolution of each image of 640 x 480. For evaluation a smaller dataset of 115 images is offered with different angles or scales for landmarks than the 1005 offered for training [22]. An extension of this dataset, ZuBuD+, has extended the query images to 1055 so each class would have approximative five test images associated [17].

The Oxford Buildings Dataset (OBD) consist of images from Oxford collected from Flicker[3] with different conditions (day and lighting). The dataset is composed of 5063 images from 17 different landmarks. Because the collection of images is a result of queries from Flicker for each landmark images that are not buildings are present and called distractors. For evaluation 11 different landmarks are annotated with 5 possible queries: (i) Good; (ii) OK; (iii) Junk; and (iv) Absent. These annotations depend on the quality of images and whether the object/building is present or not [23].

The Paris Building Dataset (PBD) is similar to OBD, containing 6300 images of 12 landmarks from the city of Paris collected from Flicker [24]. The OBS and PBS dataset were extended to reach 70 evaluation images and one million images respecting the same principles of creation [25].

[3] https://www.flickr.com/

Table 2.1 Overview of building detection benchmark solutions

Dataset	Year	Scale	Classes	Scale/ class	GPS	View
ZuBuD [22]	2003	1120	201	5	No	Yes
OBD [23]	2007	5063	17	7-220	No	No
PBD [24]	2008	6300	12	7-220	No	No
Holidays [26]	2008	1491	500	2-3	No	No
SBID [27]	2009	3192	40	100-400	No	Yes
EC50K [28]	2010	50778	20	30-110	Yes	No
EC1M [29], [30]	2010	909940	35	30-110	Yes	No
PKUBench [31]	2011	13179	198	66	Yes	Yes
Singapore-40 [32]	2012	13538	40	100-300	No	No
24/7 Tokyo [33]	2015	1125	125	4-9	Yes	Yes
Paris500k [34], [35]	2015	501k	3k	>1k	No	No
GLD [36]	2017	1.1M	30k	>1k	Yes	No
ZuBuD+ [17]	2017	2010	201	5	No	Yes
OBD1M [25]	2018	1M	11	>1k	No	No
PBD1M [25]	2018	1M	11	>1k	No	No
GLDv2 [37]	2019	762k	200k	>1k	Yes	No

The Holiday database is collected from personal images from holidays of the authors. The database consists of 1491 images comprising 500 unique classes. The dataset consists of high resolution images of a variety of scenes (natural, man-made, water and so on) [26].

Sheffield Building Image Dataset (SBID) consists of 3192 images taken from 40 buildings, which include a variety of modern buildings, exhibition halls, office buildings or churches. All the images are collected using digital cameras and tend to cover different times of the day and different viewpoints, resulting in highly variable lighting conditions and perspective sets. In the end images for each building vary from three to nine individual views [27].

European Cities 50K database (EC50K) consists of 50767 images from 14 European cities. From the dataset 778 images from 9 cities are annotated into 20 groups of images depicting the landmark or scene. The remaining 49989 images from the other 5 cities are the distractors [28]. An extension of EC50K is European Cities

1M dataset that consists of 909k geo-tagged images from 22 cities, again collected from Flicker and respecting the same principals [29], [30].

The PKUBench dataset contains 13179 images with 198 distinct landmarks near the Peking University campus. Each landmark is captured from multiple angles and viewpoints and using mobile phones or digital cameras. The images tend to cover a 360 degree from the frontal view of the landmark. For each image, the following information is available: GPS tag, shot size, view point, camera type, capture timestamp [31].

Singapore-40 dataset consists of 12338 training images and 1200 testing images of 40 landmarks from the city of Singapore. The database was built using public internet sources such as Google Image (40%), Flickr (40%), Photobucket (5%) and the rest were manually taken (15%). The dataset contains images with different camera settings, capturing time, illumination, weather, viewpoints, scale, and occlusion [32].

The 24/7 Tokyo dataset is a collection of images acquired by mobile phone containing 125 unique landmarks from Tokyo. Each landmark is captured from three different locations and three different times of the day. Each element of the dataset has the GPS location tag added. To evaluate 315 query images are selected from an area of the city [33].

Paris500k dataset focuses on images from the city of Paris, collected from Flickr and Panoramio[4] using geographic bounding regions for searching. For evaluation 3k Flickr images, disjointed from the training 500k, were selected. But the focus of the evaluation is image classification in categories such as landmark buildings, panoramas, sculptures, murals and so on [35].

Google-Landmarks dataset (GLD) contains 1060k images for 12894 classes from various locations in the word and with a GPS tag associated. For evaluation the dataset offers 111k images. Due to copyright restrictions this dataset is not stable: it shrinks over time as images get deleted by the users who uploaded them [36].

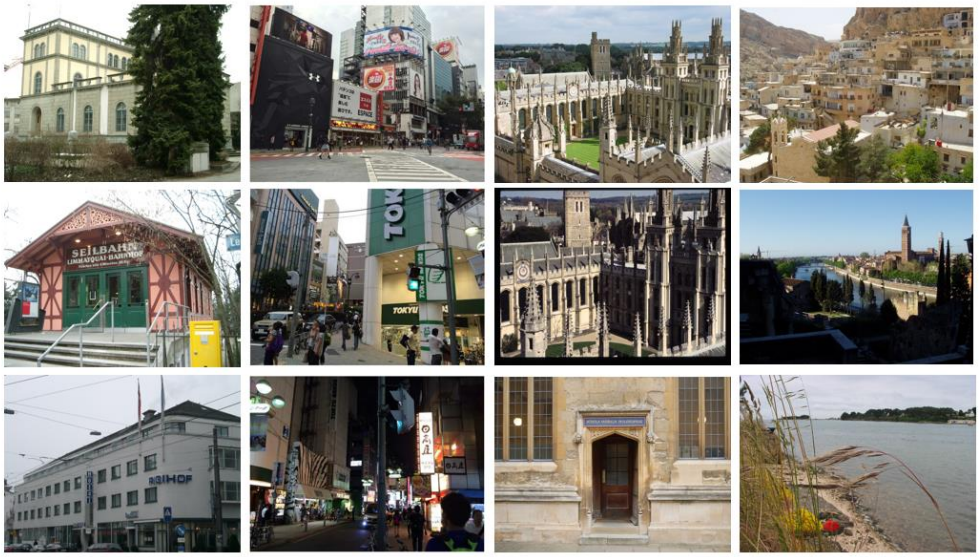


Figure 2.1 Example of images from the datasets (per column): ZuBuD, 24/7 Tokyo, Oxford, Holidays.

[4] https://www.panoramio.com

Google Landmarks Dataset v2 (GLDv2), a new large-scale dataset for instance-level recognition includes over 5M images of over 200k human-made and natural landmarks. The images that comprise the dataset were contributed to Wikimedia Commons[5] by local experts. The test set consists of 118k query images with ground truth labels [37].

Looking at Table 2.1 we can create another useful division of the presented datasets: worldwide dataset like Holidays, EC50k, GLDv1, GLDv2 or city perspective like OBD, PBD, ZuBuD, Paris500k, 24/7 Tokyo.

Another important aspect to highlight is the fact that some of the datasets are manually annotated and refined or corrected. This is an important difference from the ones that are created by querying image sharing platforms like Flickr, Panoramio. The manual annotated ones are ZuBuD, SBID, PKUBench, 24/7 Tokyo, ZuBuD+.

In Figure 2.1 we can observe examples of images that are present in the datasets. Another interesting aspect is presented in the figure such as the difference in perspective, the first two columns are from a street-view perspective the others are not.

The perspective of the images that create the dataset is important for many detection systems. Some detection schemes rely on the fact that the input images are from a street-view perspective. This assumption permits the designers of the system to employ specific calculations or pipeline decisions.

In this chapter, an overview of the existing landmark datasets was presented. Of course, because of the variety of use cases the flavours and focus of each one of them is different. From Table 2.1 we can observe that each new specific use-case brings with it a new tailored dataset and so probably the number of datasets will increase proportional with the AR applications specific with each urban environment it targets.

2.3. Urban environment understanding datasets

Automatic urban scene object recognition aims to classify and segment the image of an urban environment into categories like buildings, car, people, vegetation, and so on. This action is performed using classical approaches or training networks models for classifying scene components [38], [39].

Visual recognition and understanding of buildings are not a trivial task. This process can be affected by several challenges like obstacles in the line of sight or image distortions, saturation, and other transformations. To assume that local shape structures are sufficient to recognize objects and scenes is largely invalid in practice since objects may have a similar shape [40].

Nowadays, due to the evolution of CV algorithms, typically scene understanding is done via semantic segmentation. Semantic segmentation is the action in which each pixel of a given image is clustered into a predefined category.

In this section I will present semantic segmentation datasets that focus on urban scenarios found in literature. Selecting an appropriate dataset for a task can be a challenging step in constructing a modern CV pipeline [41].

[5] https://commons.wikimedia.org/wiki/Main_Page

An overview of found datasets in literature is presented in Table 2.2. The overview contains information about the dataset like the year of release, number of images and main resolution and the classes that are offered. Examples of images from the dataset and corresponding label are presented in Figure 2.2.

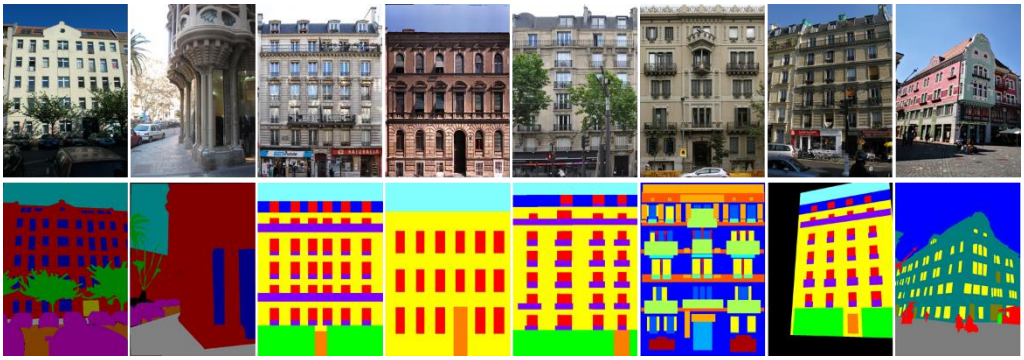


Figure 2.2 Image and corresponding labelled image from existing datasets.
Columns: eTRIMS, LabelMeFacade, ECP, ICG Graz5, INRIA, CPM, VarCity, TMBuD

Table 2.2 Overview of semantic segmentation annotated datasets

Dataset	Year	Size	Classes
eTrims [42]	2009	60	building, car, door, pavement, road, sky, vegetation, window
LabelMeFacade [43], [44]	2010	945	buildings, car, door, pavement, road, sky, vegetation, window
ECP [45], [46]	2010	109	balcony, chimney, door, outlier, roof, shop, sky, wall, window
ICG Graz50 [47]	2012	50	balcony, door, roof, shop, sky, wall, window
CMP [48]	2013	374	background, balcony, blind, cornice, deco, door, façade, moulding, pillar, shop, sill, window
VarCity [49], [50]	2014	700	background, balcony, door, roof, shop, sky, wall, window
INRIA [51]	2015	80	balcony, door, roof, shop, sky, wall, window
TMBuD [39]	2021	160	background, building, door, ground, noise, sky, vegetation, window

The eTraining for Interpreting Images of Man-Made Scenes (eTRIMS) is a collection of annotated images from European cities captured from a street scenes perspective. The dataset offers users two versions: the 4-Class eTRIMS Dataset with 4 annotated object classes and the 8-Class eTRIMS Dataset with 8 annotated object classes [42].

LabelMeFacade Database comprises 945 images with labelled polygons to describe the classes. The labelled images contain a limited number, no more than

20% percent, of unlabeled pixels. The pixelwise labelled catalogue is similar to the one utilized by eTRIMS and they are offered in a heuristic simple depth order [43].

Ecole Centrale Paris Facades Database (ECP) contains 109 manually rectified images of Paris facades with annotations. This dataset focuses more on the façade of the buildings than the urban environment [45].

The Paris Art Deco Facades dataset (INRIA) consists of 80 images of rectified facades of the Art Deco style. The dataset labelling scheme consists of 6 annotation classes. Occlusions of the facade are ignored but the occlusion reasoning is offered by the dataset [51].

The Center for Machine Perception (CMP) Facade Database consists of 606 façade images that are manually annotated from various cities of the world. Annotation scheme is defined as a set of rectangles with assigned class labels that can overlap if needed [48].

VarCity 3D Dataset comprises images along a street annotated for facade details. The dataset provides images, labels, and indexes to the 3D surface together with evaluation source code for comparing different tasks. In total the dataset offers 700 unique scenes [49].

Timişoara Building Detection Dataset (TMBuD) is created with 160 images of buildings in Timişoara with annotations for salient edges, for semantic segmentation and the GPS coordinates. Each building is presented from several perspectives and lighting conditions [39].

As we can observe in Figure 2.2, each dataset presented offers a unique collection of information regarding the urban environment. Datasets like ECP, ICG Graz50m INRIA or CPM focus more on façade details where in the case of eTRIMS, LabelMeFacade or TMBud focuses on understanding the elements near the buildings. A special way of describing the environment is presented in VarCity where the region of interest is limited to the main building in the image.

In this section I presented a review of existing semantic annotated dataset that can be used for landmark detection because of the unique street-view perspective, see Figure 2.2.

2.4. Building detection evaluation

Evaluating urban landmark detection in the image domain is similar to evaluating a CBIR system. This overlap is normal hence the main detection path is done based on features generated from images. The topic of using relevant retrieval metrics was extensively presented in [52] and later in comprehensive reviews of feature detection like [53], [54].

If we have a dataset $\{x_1, \dots x_n, \dots x_N\}$ a set of images represented by featured and a query image q, we will compare the query image with each x_n dataset using an appropriate distance function $d(q, x_n)$. The distance can be a normalized value if we would use a combination of different features.

The first metric proposed was the precision (P) and recall (R) which are detailed Equation (2.1) and (2.2). In this case P represents the percent of relevant retrieved in data for a query while R represents the successful relevant queries.

$$\boldsymbol{P} = \frac{\boldsymbol{No.relevant\ images\ retrived}}{\boldsymbol{Total\ number\ of\ images\ retrieved}} \tag{2.1}$$

$$\boldsymbol{R} = \frac{\boldsymbol{No.of\ relevant\ images\ retrieved}}{\boldsymbol{Total\ number\ of\ relevant\ images}} \tag{2.2}$$

The most common way to summarize this graph into one value is the mean average precision (mAP). The average P (AP) for a query q is the mean over the P score after each retrieved relevant item. AP is presented in Equation (2.3), where R_n is the R after the *n*th relevant image was retrieved and N_R is the total number of relevant images for the query. In Equation (2.4) the mAP is presented where Q is the set of queries q [52].

$$\boldsymbol{AP(q)} = \frac{\boldsymbol{1}}{\boldsymbol{N_R}} \sum_{\boldsymbol{n=1}}^{\boldsymbol{N_R}} \boldsymbol{P_q(R_n)} \tag{2.3}$$

$$\boldsymbol{mAP} = \frac{\boldsymbol{1}}{\boldsymbol{|Q|}} \sum_{\boldsymbol{q \in Q}} \boldsymbol{AP(q)} \tag{2.4}$$

Another interesting metric is the classification error rate (ER), if we consider a query image to be classified correctly, if the first retrieved image is relevant, see Equation (2.5).

$$\boldsymbol{ER} = \frac{\boldsymbol{1}}{\boldsymbol{|Q|}} \sum_{\boldsymbol{q \in Q}} \begin{cases} \boldsymbol{0}, & \boldsymbol{if\ most\ similar\ image\ is\ relevant} \\ \boldsymbol{1}, & \boldsymbol{otherwise} \end{cases} \tag{2.5}$$

Ranking based metrics are present in literature and are growing more popular. In a sense, if we consider that the detection of classes (landmarks in our case) is not pure image-based problems the ranking metric is more appropriate. The first metric we are exposed to is the $Rank_1$ where rank is the position of the first relevant image retrieved. An extension of it is the normalized average rank of relevant images, $\widetilde{Rank}$. $\widetilde{Rank}$ is presented in Equation (2.6), where R_i is the rank at which the i-th relevant image is retrieved (0 for perfect performance).

$$\widetilde{\boldsymbol{Rank}} = \frac{\boldsymbol{1}}{\boldsymbol{NN_R}} \left(\sum_{\boldsymbol{i=1}}^{\boldsymbol{N_R}} \boldsymbol{R_i} - \frac{\boldsymbol{N_R(N_R - 1)}}{\boldsymbol{2}} \right) \tag{2.6}$$

A metric that concerns highlighting the positive success of detection is the $TopK$, presented in Equation (2.7). In a sense $TopK$ is in opposition with ER metric and can be similar with $Rank_1$ in some cases.

$$\boldsymbol{TopK} = \frac{\boldsymbol{1}}{\boldsymbol{|Q|}} \sum_{\boldsymbol{q \in Q}} \boldsymbol{Relevant\ image\ retrivied\ in\ first\ K\ images} \tag{2.7}$$

In this chapter I have summarized the most popular metrics found in literature reviews or proposed solutions. The need for a standardized evaluation measure is clear, and we can observe that the trend is to shift towards a ranking based measuring scheme. This shift in benchmarking is normal hence the new systems utilize more than just images to perform the detection.

2.5. Building detection solutions

Landmark recognition is a problem related to finding most similar images of a building or place starting from a particular dataset. This CV task is a sub-task of the CBIR problem, and it is one of the first problems addressed in the CV domain of information retrieval.

As stated before, landmark recognition applications are an important tool for tourists who visit new places. A tourist can use a smartphone application in order to find out information about a place, such as the names of landmarks, the history, events that are currently taking place [4], [55].

The landmark or building retrieval and recognition is an open and challenging problem in the CV domain and it is extensively discussed in literature reviews like [14], [54], [56]–[59]. Even if this topic is researched for several decades now, the topic is far from being finished, because of the continuous new scenarios and situations that are generated by AR tourist applications.

In Figure 2.3 I present the generic scheme for a landmark detection. As we can observe, the system will always include two phases: the offline phase, where we prepare the features vectors for training the classifier for classification model in case of convolutional neural networks (CNN) based solutions, and the online phase where we use one image to inquiry the classifier to obtain a detection. Some of the blocks detailed in Figure 2.3 are optional (marked with dashed lines) and they are decisions that are made by the design of the system.

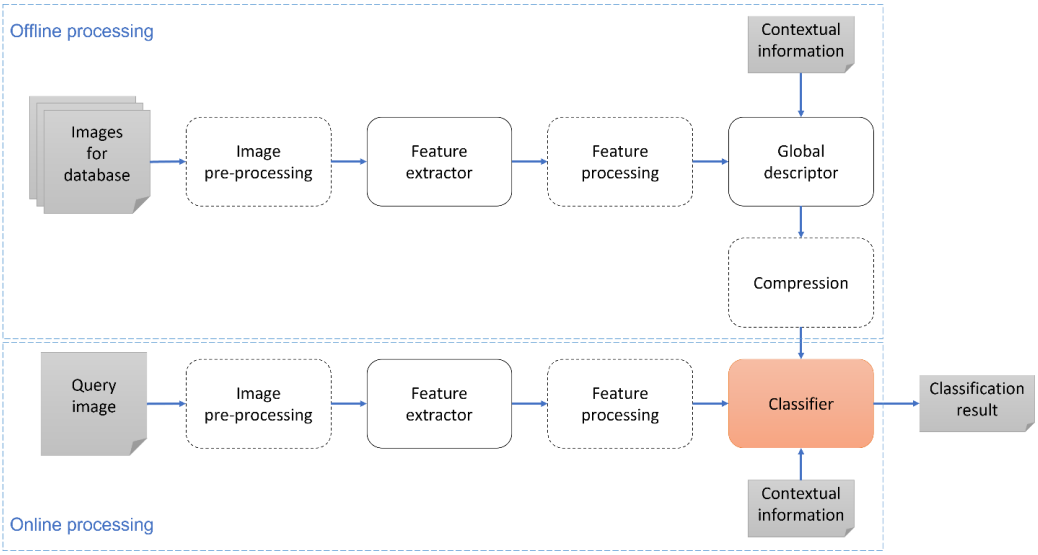


Figure 2.3 Generic landmark detection scheme

The systems will face several problems and difficulties regardless of the design and actions it consists of. The first difficult problem it would face is choosing the correct feature and descriptor for the task, different features bring more weight to different regions of pixels. The second endeavour to tackle is the discrimination of useful features from the ones that are generated by distractors like vegetation, people, signs that can appear in line of sight. Aggregating descriptors, local or global, is the next problem we would face. This activity is not trivial and multiple solutions

exist to reach a vector that uniquely represents the whole image. The last phase is one of the most important ones and it refers to the matching of new query images to the most similar vector aggregated [14], [59].

A new trend in the landmark detection domain, and in the CBIR domain, are applications that transmit query feature descriptors from a mobile device over network to a remote server hosting the database. Compressions using codecs of the query features are necessary for a low-latency retrieval in the system. Application that tackle this problem can be found in [16], [60]–[64] and have the general processing pipeline as presented in Figure 2.4.

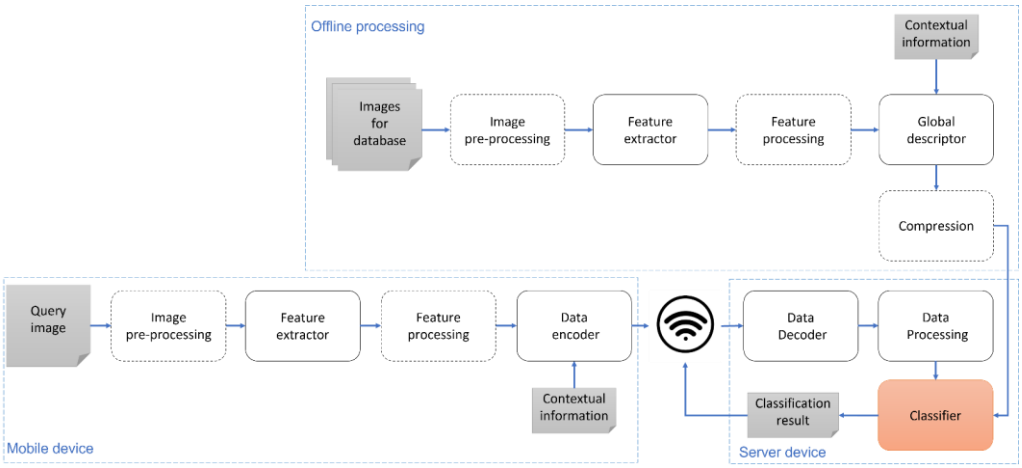


Figure 2.4 Generic scheme for landmark detection using mobile devices

In Table 2.3 an overview of multiple landmark recognition solutions is offered. The analysis focuses on the design of the solution in terms of what features are used, clustering mechanism and matching scheme. To complete the analysis, we inquire if the solution uses rich context information as GPS and on what dataset it was validated on. The scope of this analysis should bring a global image of the most common design for CBIR systems like this.

Table 2.3 Overview of landmark detection solutions

	Year	GPS tag	Features	Clustering	Matching	Tested
[65]	1999	No	Primitive image features	Grouping per features	Bayesian framework	Custom
[66]	2003	No	Affine invariant regions [67]	-	HPAT [66]	ZuBuD
[68]	2004	No	Affine invariant regions	-	KDT-ANN	ZuBuD
[69]	2006	No	Affine invariant regions	Linkage clustering [69]	Normalized RGB histogram [70]	ZuBuD
[63]	2006	Yes	i-SIFT [71]	Posterior NNA [63]	MAP [63]	ZuBuD
[23]	2007	No	Affine invariant regions + SIFT descriptor	BOF AKM [72]	VSM [73]	OBD
[74]	2007	No	Vanishing Point Localized color Histogram SIFT	-	Normalized RGB histogram	ZuBuD

[75]	2008	No	SIFT [76]	-	KNN [77]	Custom
[78]	2008	No	SIFT	KD-tree BBF [79]	NNDRS	ZuBuD
[60]	2009	No	SURF [80]	SVT [81]	Mutual dissimilarity score [60]	ZuBuD
[27]	2009	No	Gist features [82]	-	KNN	SBID
[83]	2009	Yes	LoG [84] + Gabor wavelet texture features	HAC [85]	Single linkage inter-cluster distance	Custom
[86]	2009	No	MSER [87]	K-Means [72]	Spectral graph matching	ZuBuD
[88]	2010	No	Hessian-Affine [89] + SIFT descriptor	VLAD [88]	ADC [90]	Holiday
[91]	2010	No	Gist features	-	SVM [92]	SBID
[31]	2011	No	SIFT	SVT	-	PKUBench
[31]	2011	Yes	SIFT	SVT	-	PKUBench
[32]	2012	No	SIFT	SVT	Similarity score [32]	Singapore-40
[93]	2013	No	Steerable filters [94]	-	SVM	SBID
[95]	2013	No	Harris Detector + DAISY descriptor [96]	Sparse Coding [95]	Max Pooling	Holiday
[95]	2013	No	LOG detector + SIFT descriptor	Sparse Coding	Max Pooling	Holiday
[95]	2013	No	Micro features [97]	Sparse Coding	Max Pooling	Holiday
[98]	2015	No	SURF	Sparse Coding	Modified Sum Pooling [98]	ZuBuD Holiday
[98]	2015	No	SURF + opponent color feature	Sparse Coding	Modified Sum Pooling	ZuBuD Holiday
[99]	2016	No	-	-	R-MAC	OBD, PBD, OBD1M, PBD1M
[100]	2016	No	-	-	M-R-MAC	OBD, PBD, OBD1M, PBD1M Holiday
[101]	2017	No	-	-	Custom M-R-MAC	OBD, PBD, OBD1M, PBD1M Holiday
[17]	2017	No	RootSIFT [102]	locVLAD [17]	ADC [90]	ZuBuD, ZuBuD+, Holiday
[103]	2017	No	-	-	Custom M-R-MAC	OBD, PBD, OBD1M, PBD1M Holiday
[104]	2017	No	-	-	Custom M-R-MAC	OBD, PBD, Holiday
[55]	2018	No	-	-	NU-LiteNet	Singapore-40 PBD
[105]	2020	Yes	-	-	Custom CNN [105]	Custom
[106]	2021	Yes	SURF features + FREAK [107]	-	KNN	Custom
[108]	2021	No	A-KAZE[109]	BOF KNN[72]	ANN [110]	ZuBuD

Perceptual grouping was proposed in [65], an approach that aimed to explore the semantic interrelationships among different types of primitive image features. Features like edges, junctions and parallel lines were grouped hierarchically into intermediate-level structures accordingly to shape, length and orientation. Bayesian framework is used to analyse this structure and determine the presence of a building.

Hyper-polyhedron with adaptive threshold (HPAT) indexing was proposed in [66] in order to reduce the number of feature vectors in searching for the nearest neighbours (KNN). With this new approach the authors lead to feature matching in a higher dimensional space. In the end it was proven that HPAT can improve the efficiency in localizing the KNN and reduce computational time.

A fast wide baseline matching algorithm was presented in [68]. Affine invariant column segments descriptor vectors based on geometrical, colour and intensity information are extracted for each image. The column segments are repetitive and so they are grouped into clusters, each of which is represented by a prototype column segment associated with the average descriptor vector.

Similar to [68], in [69] we are presented with a building recognition method based on intensity-based region detection that uses Principal Component Analysis (PCA) to compress the feature vectors. The vectors are then grouped using a linkage clustering scheme according to the maximal distance between the instances and characterize each cluster by its centroid. Classification is done by means of the chi-square distance of the normalized RGB histogram.

In [63] we find an application that proposes the use of Informative Descriptor Approach to the classical SIFT in order to be able to filter out irrelevant features. A decision tree is trained to rapidly and efficiently estimate a SIFT's posterior entropy value. This approach has the advantage of needing only retaining those keys for thorough analysis and voting with high information content.

Hierarchical building recognition (HBR) system is proposed in [74] and it is based on vanishing point detection and localized colour histograms. Line segments are clustered by the dominant vanishing directions. In order to find the vanishing points, these are estimated by the expectation maximization (EM) algorithm. Localized colour histograms are only computed on a limited number of pixels suggested by the indexing vectors which are matched using the chi-square distance. The detection is enhanced by extracting SIFT features and applying a simple probabilistic model to integrate the evidence from individual matches.

In [23] a ranking scheme is proposed for searching buildings in a large corpus. The descriptors are clustered into a forest of 8 randomized kd-trees visual vocabulary by approximate k-means (BOF AKM). The matching is done using tf–idf weighting by the means of a vector-space model (VSM) of visual word that calculates the similarity between the query vector and each image vector.

In [75] the authors propose a multiple building detection scheme in the image domain. The algorithm recognizes parts of a building like windows, doors or facets that are extracted based on online segments and vanishing point detection. Based on the parts detected a wall colour histogram is computed. SIFT features are utilized to describe each building after discrimination and nearest neighbour (NN) rule is used to select the closest model from the dataset.

A new voting system for KD-tree with the Best Bin First (BBF) indexing called Nearest Neighbour Distance Ratio Scoring (NNDRS) is proposed in [78]. NNDRS assigns a weight depending on the distance ratio of two nearest neighbours. For every matched feature it sums up all the weights of the matched features from the same database image till it reaches its aggregate score. The system uses SIFT features extractor to create the feature vector.

Focusing on mobile image matching application, [58] proposes an CBIR system based on SVT and by choosing the closest match by calculating the mutual dissimilarity score for weighted tree histograms of each image. The paper offers an evaluation of different encoding configurations for the Tree Histograms.

In [27] we are exposed to a biologically-plausible building recognition (BPBR) solution. The algorithm uses biologically inspired features that are extracted using a saliency model and a gist model. The gist model is constructed by dividing each feature map into sub-regions and afterwards linear discriminant analysis (LDA) is used to reduce the dimensionality of the feature vectors. For matching and classification, the NN rule is applied. In [91] the authors continue the work by proposing a relevance feedback (RF)-based building recognition (RFBR) scheme by performing a SVM-based RF after dimensionality reduction by LDA.

In [83] a clustered-based landmark recognition method from two sources is proposed. One source being the GPS-tagged photos shared on the web and another being the travel guide articles from websites. Hierarchical clustering on GPS coordinates is done to obtain dense geo-clusters which are refined based on the semantic clues embedded in the articles. Afterwards, interest points are detected by Laplacian-of-Gaussian filters (LoG) and each local region is described by a 118-dimensional Gabor wavelet texture feature. PCA filtered features are then clustered using hierarchical agglomerative clustering (HCA) to discover regions of the same or similar landmark. Matching is done by utilizing the single linkage inter-cluster distance to parse the distance between two regions.

Sketch-based representations to find major structural components of a building with the aim of recognition is presented in [86]. The algorithm detects multi-scale maximal stable extremal regions (MSERs) and describes the normalized MSER patches using histogram of oriented gradients. Clustering is done using k-Means in order to group the local patches into different structural components and matching is done using spectral graph matching.

A change in the classical clustering approach appeared in [88] where the authors propose to change BOW with a so-called vector of locally aggregated descriptors (VLAD). VLAD encodes the residual of a feature instead of the values of the features detected in the images. This new approach brought forward good results, but it had one main disadvantage that distractors from the image have a considerable weight in the resulting vocabulary.

Saliency-Aware Scalable Vocabulary Tree is proposed in [32], where the authors propose to incorporate saliency information in SVT-based mobile landmark recognition. The saliency information is incorporated in local feature extraction, vocabulary tree construction and image histogram representation. In the online recognition phase for a query image, the saliency information is only used when extracting local features and calculating the histogram.

Steerable filter-based building recognition (SFBR) model is proposed in [93]. Max-pooling is used to preserve discriminative information of local patches by searching the max value of the steerable responses. Linear discriminant analysis (LDA) is used in order to reduce the dimension of the feature vector. In the end a support vector machine (SVM) is used for classifying.

In [95] we are presented with the investigation of the usage of sparse coding for image retrieval. The authors experiment with different pooling systems and different features (individual or combined): Harris Detector + DAISY descriptor, LOG detector + SIFT descriptor and Micro features. The evaluation resulted that sparse coding obtains better results than VLAD or BOW when using all three variants of features combined.

Sparse coding was used in combination with SURF features in [98]. Sparse coding schemes can encode feature descriptors from an image into a fixed size vector. The authors proposed a new pooling method to obtain good results on the task at hand.

Regional Maximum activations of convolutions (R-MAC) was introduced in [99]. Images are used in the VGG16 [111] without any change of aspect ratio and the features are extracted from the last convolutional layer. Square regions are defined on the CNN response maps at L different scales. The features obtained in each region are L2-normalized and PCA is applied. In the end regional feature vectors are combined into a single image vector by sum-pooling and L2-normalization.

In [100] the authors propose to modify the proposed network from [99] by using as input to the network three resolutions of the same image. Then, the three computed R-MAC descriptors are sum-pooled in a final R-MAC descriptor (M-R R-MAC that stands for Multi-Resolution R-MAC). This idea is further explored in [101] where the authors used a grater descriptor dimension M-R R-MAC . This is due to the use of ResNet101 instead of VGG16.

In [103] the authors propose to execute a sum pooling calculation before on the R-MAC descriptors. Similar to [100], where three image resolutions are used, the output feature maps of the images are rescaled and summed up into one single 3D feature tensor that encodes activations at different scales.

Another adaptation of M-R R-MAC is found in [104] using a modified version of R-MAC descriptors, based on spatial attention. The authors proposed the introduction of a region of interest which suppresses the encoding of information from regions outside.

Location aware Vlad, presented in [17], were the authors propose to include a certain degree of information on the location of the features. This should help mitigate the negative effects of distractors found on the borders of the image. This is done by computing the mean of the two global descriptors: the VLAD executed on the entire original image, and the one computed on a cropped image [17].

NU-LiteNet, which adopts the development idea of SqueezeNet [112] to improve the network structure is presented in [55]. The improvement consists of the inclusion of a convolution layer with 5×5 and 7×7 filters to enable the Expand block to cope with the analysis of complex image content [55].

A particular CNN network for landmark recognition was proposed in [105] were the authors modified the popular SqueezeNet [112] by eliminating layers in between and tested it for discriminating between 2 landmarks.

A landmark recognition system based on SURF features and FREAK descriptors is proposed in [106]. The system uses the low resource consumption of the feature combination in order to match GPS filtered landmarks to query images.

In [108] the authors approach the problem with a classical view of combining a Bag of Features with K- Nearest Neighbour (BOF KNN) clustering but trust in two elements: the power of A-KAZE features and the idea of creating vocabulary clusters from all images from the dataset corresponding to one landmark.

In general, by looking over Table 2.3 we see that solutions proposed are diverse from the feature point of view to the clustering and matching solutions. In the last few years, we can observe a clear CNN based trend in the solutions proposed. From the feature used point of view the range of solutions proposed in literature vary from classical, so-called low-level features, such as edge, junctions to global features, local features and CNN based solutions.

In Table 2.4 and Table 2.5 an overview of evaluation metrics is presented on algorithms found in literature. Recognition efficiency is measured using different

metrics and dataset, so a direct comparison is not possible on the entire data presented in the table.

In Table 2.4 the Top1 metric is presented, and the entries are ranked according to results on ZuBuD dataset, most benchmarked with the respective metric. As stated in chapter 2.4 Top1 shows the capability of a system to correctly predict for a query image the correct landmark as first one retrieved.

Table 2.4 Top1 results on different dataset found in literature

Method	ZuBuD	ZuBuD+	SBID	Singapore-40
[17]	100.00%	100.00%	-	-
[88]	99.00%	99.00%	-	-
[60]	98.00%	-	-	-
[74]	95.00%	-	-	-
[69]	94.00%	-	-	-
[68]	92.00%	-	-	-
[63]	91.00%	-	-	-
[86]	81.00%	-	-	-
[66]	77.30%	-	-	-
[93]	-	-	94.70%	-
[91]	-	-	93.00%	-
[27]	-	-	85.25%	-
[32]	-	-	-	89.60%
[55]	-	-	-	81.15%

In Table 2.5 we present the mAP metric, which measures the quality of overall retrieval for each query image. In this case we observe that the most common dataset used is the Holiday dataset, which we used for ranking the results.

Table 2.5 mAP results on different dataset found in literature

Method	Holiday	OBD	PBD	OBD1M	PBD1M
[103]	94.00%	72.27%	87.10%	-	-
[17]	90.46%	61.46%	71.88%	-	-
[101]	86.10%	94.50%	90.30%	82.80%	90.60%
[99]	85.20%	66.90%	83.00%	61.60%	75.70%
[17]	80.89%	-	-	-	-
[98]	79.00%	-	-	-	-
[60]	78.79%	-	-	-	-
[95]	76.70%	-	-	-	-
[100]	66.90%	83.00%	85.20%	78.60%	79.70%
[88]	55.60%	37.80%	38.60%	-	-
[23]	-	95.30%	-	-	-
[104]	-	90.20%	95.80%	-	-

For CNN based solutions, we observe that OBD1M and PBD1M datasets are more commonly used. These datasets are called in literature as large-scale datasets because of their size.

2.6. Conclusion

As stated in the beginning of the chapter, landmark detection, mobile or not, faces several challenges that have been intensively researched over three decades now. To support the statement that the topic is not exhausted is the number of papers that we can still find published in recent years. Another important aspect supporting this statement is the annual Google Landmark Recognition [113] and Retrieval competition [114].

First real difficulty is the GPS location accuracy of the data that can drift in densely populated areas. The design of retrieval systems should handle and compensate for the error prone GPS tag system, even if geo-tagging has advanced considerably in the last years [2], [115]. Another aspect that considerably lowers the value of geo-tagging information is the proximity of the landmark at hand. In many cities the important landmarks are clustered together in markets or squares or enclosed neighbourhoods that causes real difficulties when discriminating based on GPS information.

Second challenge is the information that is captured by the images. In urban scenarios the quantity of distractors is considerable, and in this scope multiple solutions are already proposed for mobile landmark recognition systems.

The third problem systems must tackle is the trade-off between recognition accuracy and real-time requirements constraints. Adding to the run-time and resource challenge is the bandwidth limitation of current mobile devices in telecommunication networks [13], [57], [115].

Landmark recognition datasets, as presented in chapter 2.2, are an important piece in the developing chain. They offer the designers of the system the chance to evaluate and fine tune the algorithm to a certain specific use case. Similar to other domains of CV, creating a universal landmark recognition system is close to impossible. In that direction multiple datasets appeared and will appear in accordance with the needs of a certain city or region, as we can summaries from Table 2.1.

Regarding landmark datasets, new challenges were brought forward by [37], [116] to create a fair representativeness of landmarks from the world in the datasets. So, in [105] they aimed to reduce the bias of crowdsourced data and define the fair relevance of a landmark with respect to the world population

In chapter 2.3 I presented datasets that aim to help the design of a landmark detection system that can handle distractors in a better manner. The datasets presented can be used for evaluation of training different semantic aware algorithms in order to enhance the landmark detection systems.

Advances in the CV domain have enhanced techniques for building recognition and the results as well have improved as presented in Table 2.4 and Table 2.5.

Considering the positive results obtained in recent years, we are confident that this task, of landmark detection, can, and will be used, in various CV applications like automatic target detection in surveillance, real-time robot localization and visual navigation, architectural design, and 3D city reconstruction [14].

3. END-TO-END COMPUTER VISION FRAMEWORK

3.1. Introduction

CV is an interdisciplinary domain which tackles understanding the environment through digital images or videos as closely as possible, or even better, than a human can do [117]. As we can observe in Figure 3.1, there is a fundamental difference between the interpretation of visual information between the human vision (HV) and CV.

HV bases the understanding of the environment on the ability to perceive differences between light composed at different wavelengths independently of the intensity. This mechanism is mimicked by the CV using pixels, which represent a numerical value associated to one point in a picture, see Figure 3.1. A pixel can store information using a scalar form for gray scale images or a form vector for colour or multispectral ones.

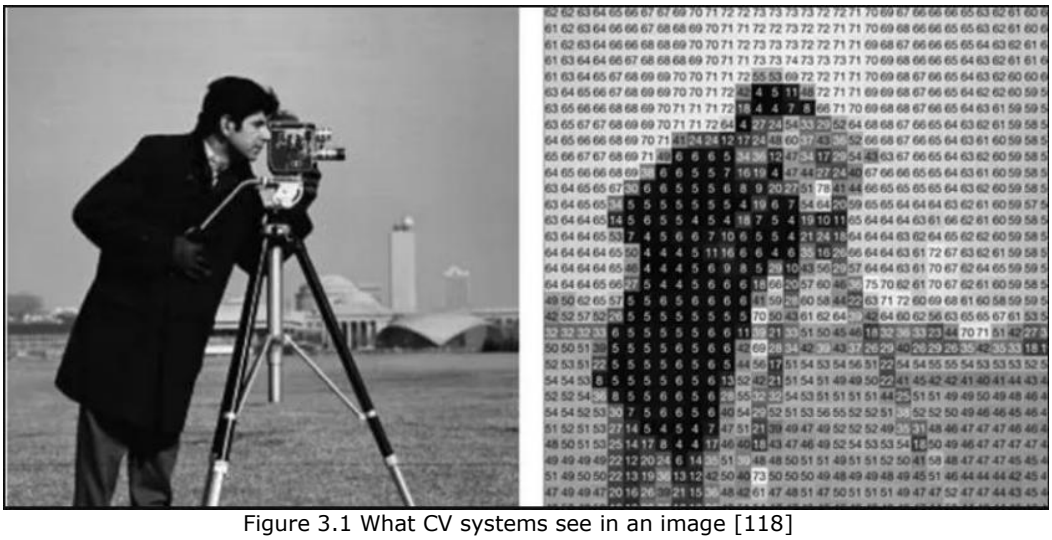


Figure 3.1 What CV systems see in an image [118]

CV application architecture complexity has evolved directly with the complexity of the topics that were addressed over time. In Figure 3.2 I present a brief overview of the challenges faced by the CV domain over the years.

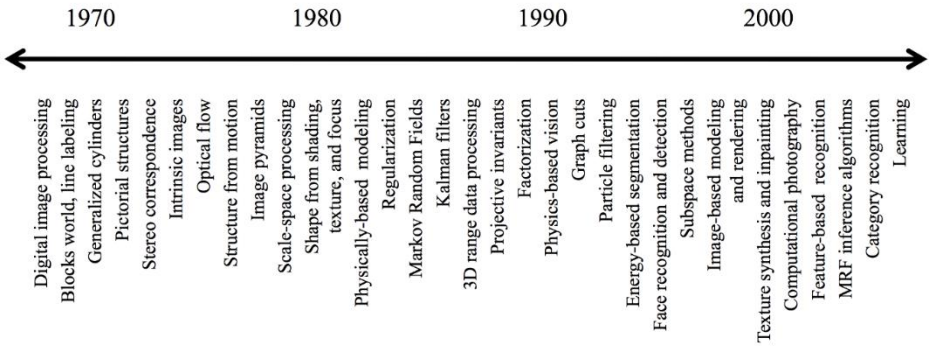


Figure 3.2 Evolution of CV topics [119]

CV pipelines include multiple steps starting from image acquisition from sensors, processing steps to enhance the image, transformation in order to reduce noise, selection of region of interest, segmentation of the image; different levels of feature extraction, high-level processing relevant to the application, and decision-making such as classifying an object [117], [120].

Nowadays CV tasks are solved using more complex AI solutions techniques using Machine Learning (ML), Neural Networks (CNN), or Deep Learning (DL) models. These techniques tackle the problems by learning model structures weighted parameters for patterns from a huge amount of data [121], [122].

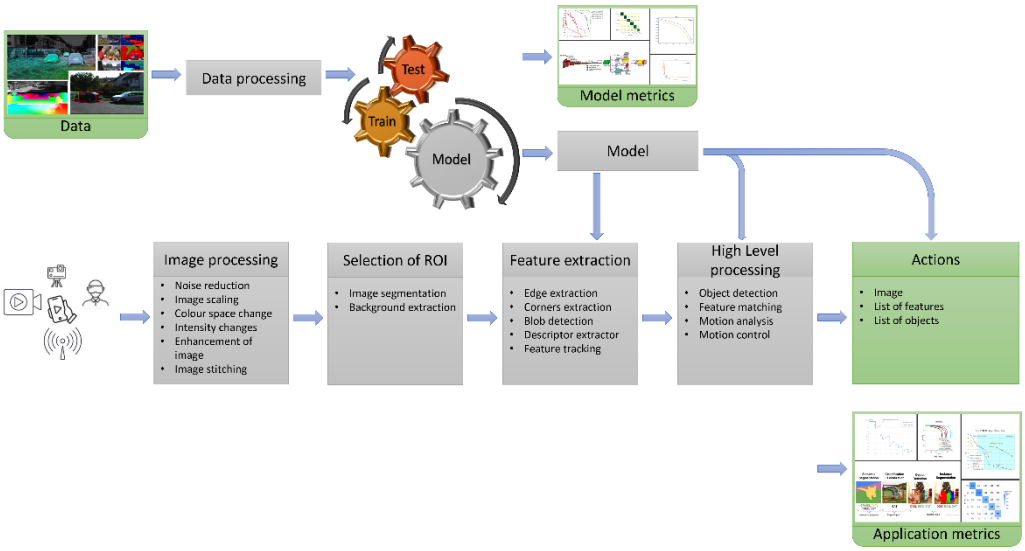


Figure 3.3 Framework of a modern CV pipeline

Given the evolution of CV systems and the continuous increasing number of applications, multiple implementation solutions for a desired pipeline exists. However, in any case, the modern CV pipeline is a mixed system that includes AI elements, such as ML or DL, and classic image processing activities, as we can see in Figure 3.3.

In modern CV literature we can clearly see that AI is strongly interconnected to the CV domain.

Even if the programming language is only a tool for implementing the means of the image processing algorithm, the selection of the most suitable language is not a trivial task. Usually, this choice is determined by the environmental conditions: the hardware the application is running on, the purpose of the application, the sensors it must interact with, and so on.

To be able to simulate the proposed system I used the python-based CV framework EECVF [6], [7], [123]. The motivation of using this framework was based on several elements: the versatility of the programming language used, the capability of extending existing CV algorithms, the capability of debug information outputting.

In subchapter 3.2 I present an overview of frameworks found in literature with a brief presentation of each entry. Followed in subchapter 3.3 where I tend to explain the justification for using a certain framework for our simulations. In subchapter 3.4 the EECVF internal structure and block is presented and in subchapter 3.6 the conclusion is presented.

3.2. Overview of existing frameworks

In this subchapter, frameworks that were found in literature are presented. The analysis of the frameworks focuses on highlighting main points of each framework and in which programming language they were developed. To be able to create an objective overview we excluded frameworks that focus on solving a particular use-case or tailored to certain architecture.

Libraries, open-source or not, are not in the focus of this analysis because we consider that they are the backbone of the majority of the frameworks presented. This on its own is an important topic and I would like to enumerate several popular ones: OpenCV [124]–[126] (C++, Java, Python), MatWorks [127] (Matlab or C++), TensorFlow (C++, Python) [128]–[131], CUDA (C++, Python) [132], SimpleCV (Python) [133], [134], CImg [135] (C++), OpenAI[136] (for Python), MatlabFns [137] (for Matlab or Octave).

Waikato Environment for Knowledge Analysis (WEKA) is a set of tools bound together in a program with a common user interface, based on C/C++ programming language [138]. Later WEKA3 appeared with no scope change from usability perspective but rather a change in the programming language from C/C++ to Java. WEKA3 is used in various application areas, in particular for educational purposes and research [139].

Parallel Image Processing Framework (PFIP) has the scope to offer scholars the possibility to execute parallel processing of image processing algorithms. The framework consists of a master (PFIPMaster) and N clients (PFIPSlave). PFIP has a programming interface intuitive and easy to understand [140].

JavaVis is an open source, Java based, CV teaching framework. It consists of a series of CV algorithms handling image domain and a series of algorithms handling the 3D domain. JavaVis functions can be launched both from command line or the JavaVis GUI. JavaVis third module consists of a visual desktop where different CV algorithms can be easily placed and connected [141].

QVision is a software framework constructed to reduce the programming effort for researching prototypes in CV or ISP. The framework offers functions from libraries

like Intel's Integrated Performance Primitives, BLAS, LAPACK, or GSL. QVision includes a simple but powerful approach to multithreaded programming, aimed for non-expertise parallel programmers, and a versatile GUI that is designed with the scope of programming ease [142].

PyCVF is an open-source framework for CV and video-mining. The framework is facile for rapid development of applications and offers a suite of CV algorithms build-in. The framework can interact with other frameworks like WEKA or toolboxes like OpenCV or Django [143].

The R Analytical Tool To Learn Easily (RATTLE) is a graphical data mining application residing in the R programming language. The framework goal is to ease the transition from basic to sophisticated data mining and analyses. Rattle offers an user interface for a facile introduction to data mining tools [144].

DARWIN framework is a two-part system based on the C++ programming language. The aims of it are to provide students and researchers with an infrastructure for researching and tuning ML methods. They provide data management, logging, and configuration infrastructure with a consistent and well-documented interface [145].

Fiji is a Java based open-source software framework that focuses on image analysis particularly in biology. The framework has the aim to combine powerful software libraries with several scripting languages to enable rapid prototyping. At the core of the Fiji framework is the popular Java library ImageJ [146].

EMZed is a framework designed for mass spectrometry users based on Python. The aim of this framework is to enable the users to create custom workflows for liquid chromatography and data analysis. The framework is aimed specifically at inexperienced programmers that want to create a comprehensive list of core functionalities. [147]

Adaptiv Vision Studio (AVS) is a software framework based on C++ with the goal of image processing and image analysis. To prove the usage of this framework a group of postgraduate students from Automatic Control and Biotechnology at the Silesian Technical University tested it as part of an CV course. AVS has proven to be a powerful environment with ready-to-use image analysis filters for CV experts and beginners [148].

CloudCV, is a comprehensive framework which aims to provide access to CV algorithms as a cloud service using APIs. The framework backbone is comprised of virtual machines running on Amazon Web Services on which users can run a considerable number of tasks in a distributed and parallel manner. CloudCV offers algorithms based on C/C++, Python or .NET for experts or beginners users [149].

HetroCV is a framework for image processing applications that auto-tunes in order to focus on run-time on heterogeneous CPU-MIC platforms. Users can configure processing details or let the framework auto-tune. The main advantage of using this framework is that statistics are extracted from the CPU and a prediction for optimal tuning parameters is done in parallel [150].

MicMac framework has been developed by the National Institute of Geographic and Forestry Information and the National School of Geographic Sciences since 2003. MicMac is a free, open-source photogrammetric software for 3D reconstruction. The framework offers predefined CV algorithms with default parameters, but developers and scientists can use MicMac to implement their own algorithms [151].

The DeepDIVA framework aims to enable a quick and intuitive setup for CV experiments based on Python language. The framework offers services to track experiments, hyperparameter optimization, and visualization of DL or ML algorithms. Another important benefit of this application is the capability to use Web Services [152].

Table 3.1 Analysis of different CV frameworks found in literature

Name	Year	Lang	Open	GUI	Main benefits
WEKA [138]	1994	C / C++	Yes	Yes	Focused on ML and Data Mining algorithms.
PFIP [140]	2004	Java	Yes	No	Focus on parallel processing CV algorithms. Uses JPVM platform.
JavaVis [141], [154]	2007	Java	Yes	Yes	CV algorithms.
QVision [142]	2008	C++	Yes	Yes	Wrapper for Intel IPP library.
WEKA3 [139]	2009	Java	Yes	Yes	Focused on ML and Data Mining algorithms.
PyCVF [143]	2010	Python	Yes	Yes	CV and video mining algorithms.
Rattle [144], [155]	2011	R		Yes	Focused on ML and Data Mining algorithms.
DARWIN [145]	2012	C++		Yes	CV and ML algorithms. Python wrapper.
FIJI [146]	2012	Java	Yes	Yes	Biological-image analysis specific. Plug-in to Matlab, ITK and WEKA.
eMZed [147]	2013	Python		Yes	Specialized for chromatography LC/MS data.
AVS [148]	2015	C / C++	Yes	Yes	CV algorithms + presentation for teaching.
CloudCV [149]	2015	Python			Distributed CV algorithm as a cloud service. Python, Matlab APIs.
HetroCV [150]	2016	C / C++	No		CV and ML algorithms on CPU-MIC platform. Focused on run-time on HW optimizations.
MicMac [151]	2017	C++		Yes	Photogrammetric algorithms available. Specialized for 3D reconstruction.
DeepDiva [152]	2018	Python	Yes		Accessible as Web Service. DL and ML algorithms.
Chainer [153]	2019	Python	Yes		DL for accelerating the research. CV and ML algorithms.
EECVF [6], [7]	2020	Python	Yes		CV, ML and benchmarking algorithms. Facile incorporation of other toolboxes.

Chainer is a framework that provides acceleration using GPU with familiar Python libraries for CV application. Researchers and practitioners can implement a full range of CV and DL models using this framework [153].

EECVF is a python-based framework for developing CV, ML or DL models with a built-in logging mechanism to ease the debugging. Several CV jobs are provided by default, but users can add new functionality in the framework easily. The framework has a separate block that handles AI activities where users can find the existing models or add theirs [7].

In

Table **3.1** a brief overview of the analysis is presented. To offer this overview it was considered the year of publication (announcement) of the framework; base programming language used; the abbreviation or name of the framework; the main benefits each framework offers to the users.

From the overview presented in

Table **3.1** we can observe that most of the frameworks found in literature are open source, which represents a clear benefit for future expansion.

Another interesting aspect I can point out is the main programming language used in the development of the frameworks. We can clearly see a correlation between the maturity level of Python language and the usage of it in constructing frameworks. This aspect is natural, from my opinion, as python offers built-in functions in order to interconnect several services provided by operating systems or HW accelerators.

3.3. Framework selection

Choosing a framework for my simulation endeavours is not a facile task. As presented in

Table **3.1** multiple variants for this task are available, based on various programming languages.

Naturally the familiarity of EECVF framework was an important factor but not singular. To select the framework to use for the system I chose the following criteria to be taken into consideration:

1. Programming language
2. Capability of simulating an entire chain of processing
3. Capability of outputting intermediate and debug data
4. Facile integration of new CV algorithms

The programming language is an important aspect. Python, which is an interpreted, high-level programming language has become the de facto standard for exploratory and computation-driven scientific research. This fact is clearly visible in the overview presented [7] and in

Table **3.1** and there for was taken in consideration as a limiting factor for the selection.

The capability of simulating in one place all the desired chain of actions needed, from ISP pre-processing tuning to high-level decision, is important when trying to construct a system for detecting urban landmarks. The facile experience from the past to integrate new CV algorithms together with new benchmark algorithms into EECVF was a factor.

The framework can handle the cycle of creating - training - evaluating an AI model, executing a CV application using the model and finally evaluating the results

and displaying them without user intervention. From the capability of handling this chain the name of "End-to-End".

The assumption that EECVF does is that tuning the model, running the CV application, and evaluating the results do not happen at the same time, which is true in our case.

"EECVF is an easy to use, modular, and flexible framework designed for researching and testing CV concepts. The framework does not require the user to handle the interconnections throughout the system" [7]. This of course is not particular to this framework, but the trivial way in which new CV algorithms can be plugged in the EECVF is an important aspect in the decision.

One disadvantage of the EECVF is the lack of a GUI for basic users but in our use case this is not the case. Furthermore, my interest was more in developing and expanding new CV algorithms into the framework than just using existing ones.

3.4. EECVF presentation

EECVF is constructed in a modular manner with the clear scope that a user can use one or several blocks of the framework. The users are not required to handle the interconnections throughout the system.

""All the components, high-level blocks, or low-level jobs, respect the concept of data coupling. EECVF aims to have a loose coupling between software elements so the dependencies would be minimal and the data flow slim. This aspect of construction helped the functionality of the EECVF to be scalable. New features can be easily integrated in the framework by users, without any need for refactoring, changing or adapting any existing features or concepts inside" [7].

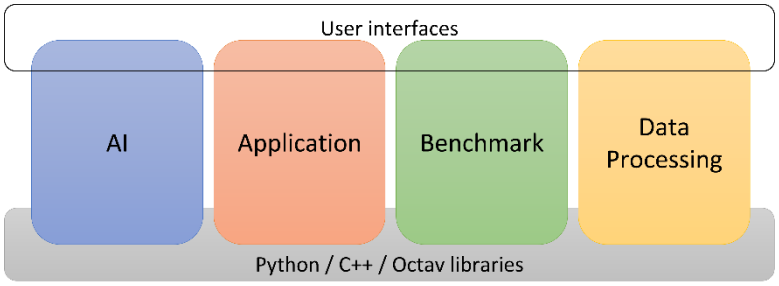


Figure 3.4 EECVF construction blocks

As stated before, one of the main goals of the framework is to unify different stages of the CV vast domain. To do so EECVF is divided into four modules as shown in Figure 3.4. Because the system can consider each block as an individual component it can employ mechanisms in order to eliminate duplication of data or interfaces inside the system [7].

As we can observe in Figure 3.4 the *User Interface block* is on top of the following four blocks: *AI*, *Application, Benchmark* and *Data Processing.*

EECVF is constructed following the Façade design principle, where the *User Interface block* acts like a façade for the entire system. Facade principle states that a complex subsystem needs to provide an interface that limits interactions with lower layers of architecture. By doing this, the system offers a controlled channel of

communications between user and the system [156]. Every task that EECVF offers provides a method to the user with the scope of abstracting the internal functionality.

To understand how EECVF inner architecture works, we need to define the terms [7]:

- a *job* is defined as a task that ultimately adds value for the CV or AI processing pipeline.
- a *service* is an operation that handles internal framework functionality and assures proper configuration and information flow.
- a *port* is defined as a channel for the information to be passed between *jobs* of the pipeline or blocks of the framework.
- a *wave* is a term used for the entire processing block done for one frame by the system.
- a *block* is defined as a subsystem that handles one specific functionality of the EECVF.

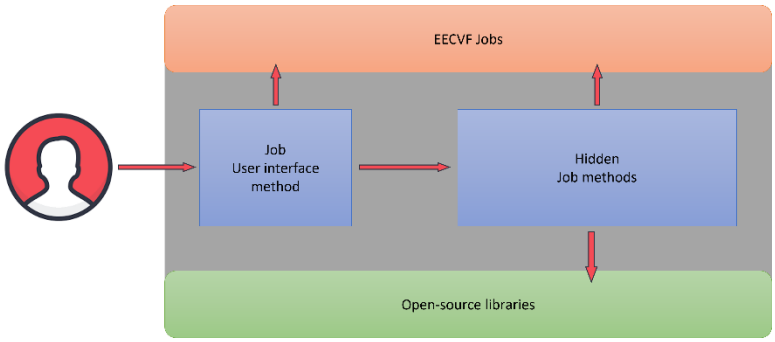


Figure 3.5 Job structure overview

Every *job* has a public interface, present in the *User Interface block*, from which it can be configured via parameters. This public method should handle all the necessary changes inside the blocks and even trigger other jobs, from the same or different block, if necessary. In Figure 3.5 the internal structure of a *job* is presented.

One can choose to use only one of the blocks or for several of EECVF. For example, a user can only use the *Application block* to run a simple CV pipeline, or the *Application block* and the *Benchmark block* to run and evaluate a novel algorithm proposal.

3.4.1. AI block

The AI block handles the customization of AI models (ML, CNN, or DL) and afterwards the training and hyper-parameter tuning of it. EECVF desires to isolate these specific activities from the rest. The first reason this isolation is applied is to respect the modular principle it is built on. This separation is also needed because these sets of activities are made prior to any usage of the model in any system. An overview of this block can be seen in Figure 3.6.

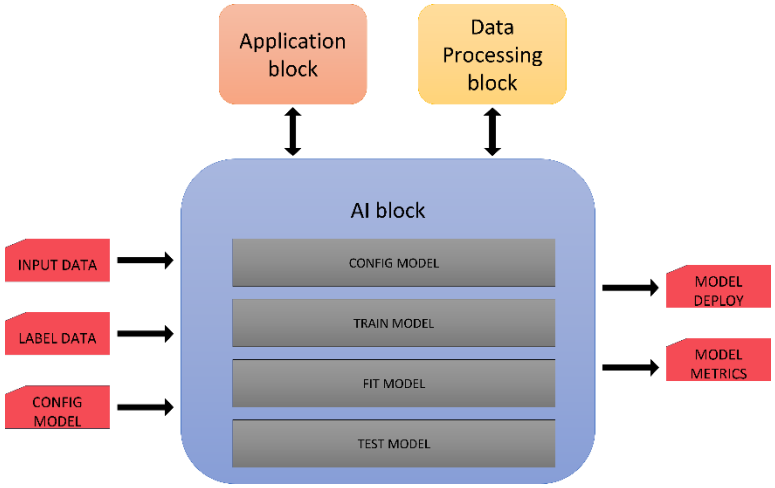


Figure 3.6 AI block overview

EECVF offers the AI block as a block in which users can implement their own models or they can use third-party models. The framework does offer build-in models, but usually every application desires to implement new ones.

In order to augment the data for training we can employ third-party functions or use the *Application block.* As stated, EECVF is not dependent on any library to be used.

3.4.2. Application block

The *Application block* is designed to handle the CV pipeline. "This block configures the actual order of execution for all triggered *jobs* and *ports* that the user describes in the attempt to simulate a use case" [7].

In some sense we can consider this block the "main block" of EECVF. In Figure 3.7 the overview for the block is presented.

To set up a processing pipeline one needs to define all the desired *jobs* with input ports and output ports, the input data source for the pipeline and if they desire a different schedule than standard.

"The Application block is designed to be modular, scalable, and flexible. To enforce this concept, it has processes that attempt to remove duplicate operations, schedule Jobs, and execute algorithms only when needed" [7].

The Proxy design pattern states that a resource in a system is offered only when it is needed by a service or functionality [156]. The Application block is constructed in the manner to respect this approach and when executing *jobs* and *services* this just in time approach is respected [7].

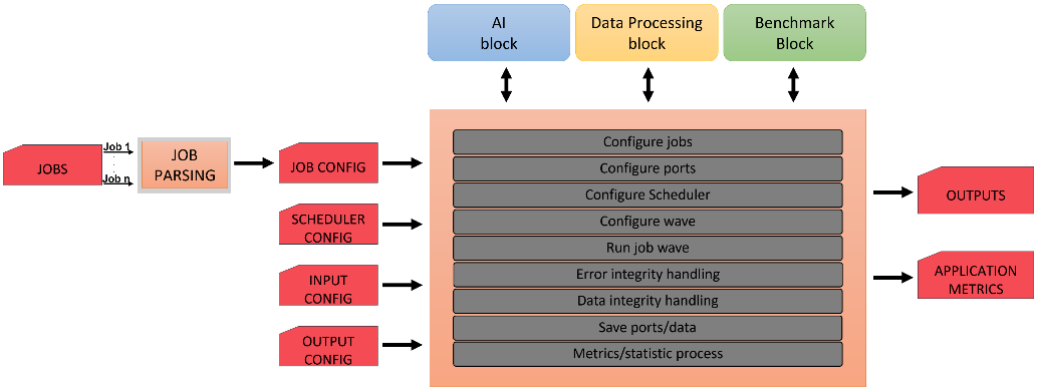


Figure 3.7 Application block overview

Job parsing algorithm is an intermediate important step for this block and has the aim to avoid duplicate *jobs* and offer a parsed list of jobs to the scheduler. The slim list of *jobs* and *ports* that is outputted by the Job parsing algorithm is a JSON file. In this phase the pipeline is defined by respecting the priority of jobs and availability of data as input [7]. The job parsing algorithm is described in Figure 3.8.

Algorithm Job Parsing Algorithm

```
/* Job with no input, only output ports                                    */
Find input job
/* jobs which process input data have process_lvl = 0                      */
Process_lvl = 1
while job to process do
    Cluster processed jobs
    Cluster unprocessed jobs
    for job unprocessed cluster do
        if input_ports of job found in output_ports from processed cluster then
            Add Process_lvl to job
        end
    end
    Increment Process_lvl
end
Set unprocessed jobs as inactive
Sort jobs by Process_lvl ascending
for Process_lvl in jobs do
    /* job name is not considered in this verification                     */
    if duplicate job then
        Set job as inactive
    end
end
Write active jobs to JSON file
```

Figure 3.8 Job Parsing algorithm

The job parsing algorithm aims to sort all the jobs in the pipeline, so no *job* is executed before the input *port* is present and populated. At the end of the algorithm by clustering continue the processed *jobs*, the ones with input *port* allocated, different from the unprocessed *jobs*, the ones without input port, *jobs* with missing ports will be set as inactive [7].

All the *jobs* from the *Application block* are composed from three phases: initialization, run and terminate. The initialization phase should handle the initial configuration for each *job* and is called only once in the pipeline. Another important

aspect that is triggered in the initialization phase is the creation of output ports. The run phase is the part of the job that we desire to execute for each frame injected into the pipeline. This phase is responsible for populating the *ports* and setting the valid flag or invalid as necessary. The last phase of each *job* is the termination part where the actions that need to be done in the last step are configured.

```
Algorithm Application Algorithm
Start timers, loggers
if input flow exists then
    Configure Application accordingly to input source
end
if config_json exists then
    for job in job_list do
        Create job for port of Job do
            if ports not exists then
                Create port
            end
            Link Job to Port
        end
    end
    for job in job_list do
        Run Job.Init()
    end
    while wave do
        Scheduler(job_list)
        Save debug data, ports
    end
    for job in job_list do
        Run Job.Terminate()
    end
end
```

Figure 3.9 Application algorithm

In Figure 3.9 I describe the generic Application algorithm. The main function of this block clearly focuses on the execution environment and not the functionality it is executing. This focus enforces the flexibility attribute of EECVF.

"The error handling responsibility inside the Application block is divided between the hidden methods of a job, that should protect against exceptions that occur in processing, and the block frame together with the scheduler. There are several mechanisms to protect against the execution failure of the Application block. Every port has a validity attribute that is set by jobs when they are filled after processing; if the attribute is false, this will cause all the jobs that consider the port as input to not execute the current wave. Every job checks, after initialization, that the necessary data and input ports exist; if this is not the case, the job will be eliminated from the pipeline and the depending on jobs will not execute. If an error occurs in the run phase of a job, this will be considered corrupted and will be skipped and logged accordingly" [7].

Statistics and debug data from *Application block* is generated automatically for each wave using *Data Processing Block*. The minimum data saved for each job are the output image port, if that is an image, and run time analysis. Naturally the user can downsize or complicate the quantity of data needed to be exported in this phase.

3.4.3. Benchmark block

The Benchmark block is responsible for evaluation, validation or integrity checking of data from the EECVF. In this block, one can opt from a number of benchmark algorithms to apply on their results or implement new metrics needed.

The construction of this block does not support a parallel design by nature, this is intentionally because the evaluation is not done at the same time as running an application. The block can run multiple evaluations for one application. In Figure 3.10 the overview for this block is presented.

"A limitation of using this block that we consider mentioning is the fact that the jobs inside this block cannot be integrated with Application block jobs. This limitation exists purposefully because the EECVF environment considers that benchmark is done post-CV pipeline execution" [7].

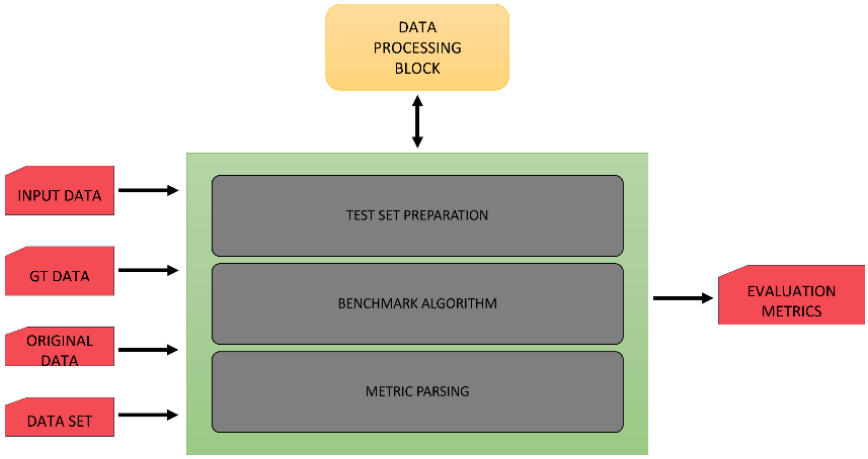


Figure 3.10 Benchmark block overview

3.4.4. Data processing block

"The Data Processing block handles the "communication" between users and EECVF. By "communication", we understand the exchange of data between the framework and the user. This can be exemplified as saved images, tables, plots, or other metrics" [7].

In this block users can configure one or several data manipulation *services* that can create plots, metrics, statistics, or any other data related manipulation desired. Like stated before by default, a limited series of data is saved, and they can be found in the Logs folder.

3.5. Example of adding into EECVF

"Adding new functionalities to the framework is a facile task because of the architecture chosen when we developed it. All functionalities of the EECVF are to be found under the so-called generic term of job" [7]. In this section I will attempt to explain how to add a new job into EECVF.

To respect the principles of this framework, and general programming principles, we should encapsulate the functionality inside one Python module. The new *job* should respect the template offered (*Application/Jobs/job_template.py*) and expose three public methods: a user-interface, an initialization and a cyclic run.

The user interface method should configure the transition between user and the Application block and describe the configurations. The new methods should be configurable using a series of parameters. Some parameters are mandatory like name of input port with respective wave and pyramid level; name for one or several output ports and parameters that are needed for the actual function [7].

Like presented in the template module the users should create inside the job user interface method a list of ports, input, and output, and to specify the format of the data. In this interface one should not forget to point the initialization function together with needed parameters and the run function, again with the needed parameters. The integrity of the ports and data flow should be handled from the low level of *init_function* and *main_function* for jobs.

After the public methods are done in order to be considered by EECVF they need to add the interface of the Application block (Application/__init__.py) and the new module to the package init module (Application/Jobs/__init__.py) [7].

To better expose how a new job is added to the framework, we will present what changes that were needed for a simple job, like the sharpening (high-pass) filter, which is presented later on in section 4.4.

Figure 3.11 Job interface example

First, we created the job interface inside the module *sharpening.py*, a module already existing in EECVF. As we can observe in Figure 3.11 a user must configure the parameters it desires beside the standard *port_input_name* and *port_output_name*. Important aspects were configured in this method like job name, input ports, output ports and name of inner method to call.

Next, we need to add the user interface of the job in the *Application/__init__.py* module, like we observe in Figure 3.12. This step will create the necessary visibility state for the user to be able to access the new job.

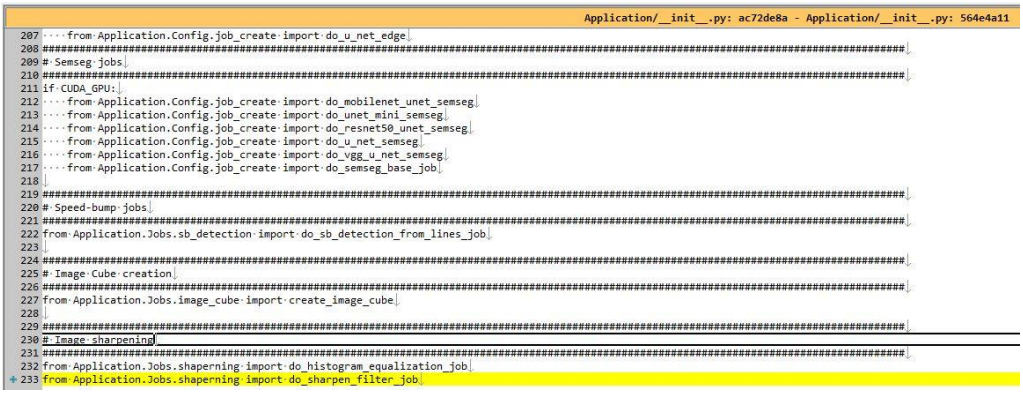


Figure 3.12 Job interface added to EECVF interface list

The final step is to add this job to the framework as presented in Figure 3.13, where we create the *main_function* method for our new job. In this case, like stated before, we created a simple job to be easy to follow. Being a relatively simple job, we only need to use OpenCV library to create the desired functionality.

Figure 3.13 Example of *main_function* of a job inside EECVF

This example is a trivial one, but it has the scope of exemplifying how facile it is to add new jobs to the existing framework. Being an open-source framework and being stored on GitHub [123] users have access to other more complex jobs to take as example if needed.

3.6. Conclusion

EECVF is an open-source Python-based framework that aims to assist researchers and teachers on day-to-day activities in the CV domain [7]. In this case I have used this framework to simulate and evaluate the proposed algorithm as a whole or as parts of it.

"EECVF is a easy to use, modular and flexible software framework that can handle complicated CV pipelines while offering all the necessary information to the user for understanding the effect of each block in the chain. Combining the robust design of our framework with the advantages of Python programming language has proven to be beneficial to the outcome. EECVF can run in multiple operating systems with minimal changes" [7].

In [7] we are presented with another benefit that EECVF can offer, the fact that it can be used in educational scope. The authors have identified that EECVF can be used in the following areas with clear benefits: usage in core courses for CV, effective and flexible software tool for students, tool for teachers to design assignments, usage for research in education.

To reproduce all the steps of the proposed algorithm inside EECVF one can run the module *main_TMBuD_detection.py.* Similarly, if we wish to reproduce the sharpening experiments, we can run *main_sharpening_dilated.py*.

4. DILATED FILTERS

4.1. Introduction

There are two main categories of image processing actions: filtering and warping. Image filtering changes the range of the image intensities, so the colours of the image are modified but the pixel positions remain intact. Image warping changes the domain of an image, in other words the pixel position changes without changing the intensity.

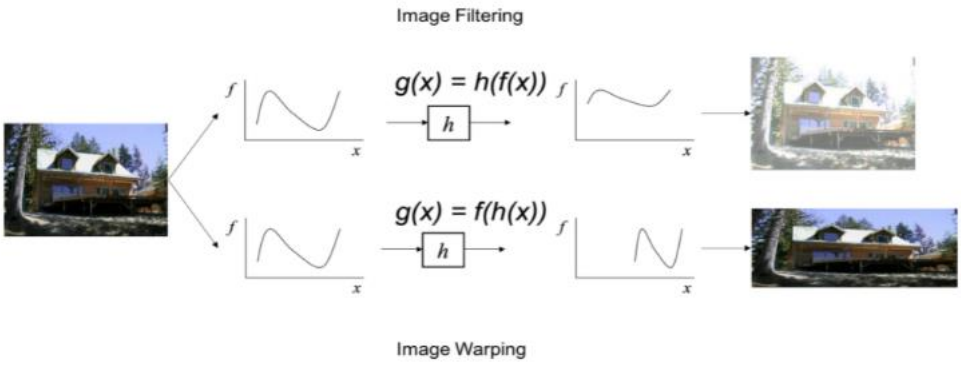


Figure 4.1 Difference between image processing [157]

Filtering is a technique for altering or transforming an image. Because of the regional scope of this it can be considered a neighbourhood operation. By filtering the value of any output pixel is calculated by an algorithm that considers the pixels in the neighbourhood of the corresponding input pixel. The region or neighbourhood taken in consideration in each case can be different from filter to filter.

The first mention of dilation in the image processing domain appears in the mathematical morphology field. The dilation operation uses a filter element to probe and expand the shapes contained in an image [158], [159].

The second mention of the idea of dilated kernels appeared in the ML domain in recent years. It is a technique in which one expands the kernels by inserting holes between the consecutive elements [160]–[164].

The third mention of dilated filters in literature is the similar idea from ML, but applied to classical filter-based convolutions outside of any AI model. The experiments were conducted on the edge detection kernels with good results [8], [9], [165].

In this chapter I will present the idea of dilated filters in the classical filter domain, see sub-chapter 4.2, followed by the presentation of the results obtained on edge detection algorithms, in section 4.3. The results from the edge detection experiments encouraged the pursuit of applying the idea on sharpening filters, which is analysed in sub-chapter 4.4.

4.2. Dilated filters

"To benefit from a higher neighbourhood of a pixel to obtain a pixel edge we define a dilated filter as in Definition 1. When the kernels are dilated, the newly added positions are considered as gaps and we ignore them by setting zeros" [8].

Definition 1. "A dilated filter is obtained by expanding the original filter by a dilation factor" [8].

The main objective of dilating is to cover more information from the input in the output obtained with every convolution operation, as we can observe in Figure 4.2. By applying this operation, it results in a wider field of view at the same computational cost.

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \quad \begin{bmatrix} a & 0 & b & 0 & c \\ 0 & 0 & 0 & 0 & 0 \\ d & 0 & e & 0 & f \\ 0 & 0 & 0 & 0 & 0 \\ g & 0 & h & 0 & i \end{bmatrix} \quad \begin{bmatrix} a & 0 & 0 & b & 0 & 0 & c \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ d & 0 & 0 & e & 0 & 0 & f \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ g & 0 & 0 & h & 0 & 0 & i \end{bmatrix}$$

$$factor = 0 \qquad factor = 1 \qquad factor = 2$$

Figure 4.2 Generic kernel dilated

The hypothesis behind is that by dilating filters, rather than expanding filters, the region covered by the transformation is bigger in terms of pixel distance and not pixel density. This means that we take into consideration more information and not more pixels.

In a sense, we can consider that by using dilated filters we have an operation that is similar to applying the original filter to a lower scaled image. This experiment is researched [165] with interesting results.

The validation of the benefits, in terms of computational resources, is proven in [8] where the authors calculated the average run time for several classical edge detection kernels: Sobel [166] and Prewitt [167]. The results are presented in Figure 4.3 and we can observe that the average run-time on 200 frames is equal between the classical $3x3$ kernel, dilated $5x5$ or dilated $7x7$.

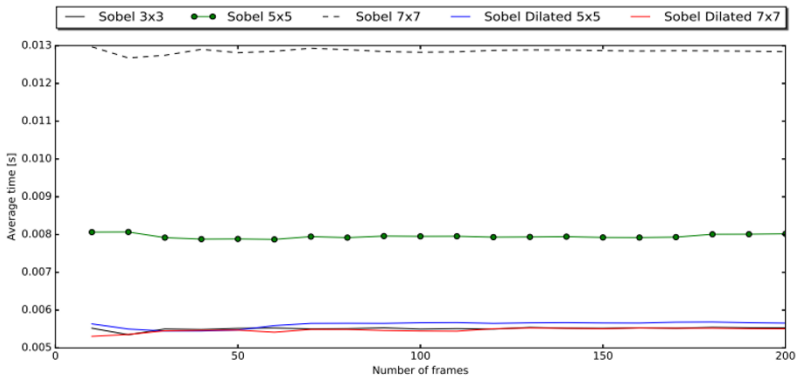


Figure 4.3 Run-time analysis of edge detection filters [8]

4.3. Dilated filters in edge detection

Edge detection is one of the basic features in CV and aims to localize variations on the image colour domain that correspond to a physical change of shapes. Edge detection, being a basic feature by definition, must be an efficient and reliable computational process with concrete results [168]–[170].

In this section I will attempt to present the positive results that were obtained by using dilated filters in several edge detection algorithms, work that was done in the past in [9]. In Figure 4.4 visual results are presented when using one of the most popular classical edge detection algorithms: Canny [171]. As we can observe in the images, each level of dilation can bring with it extra edge points.

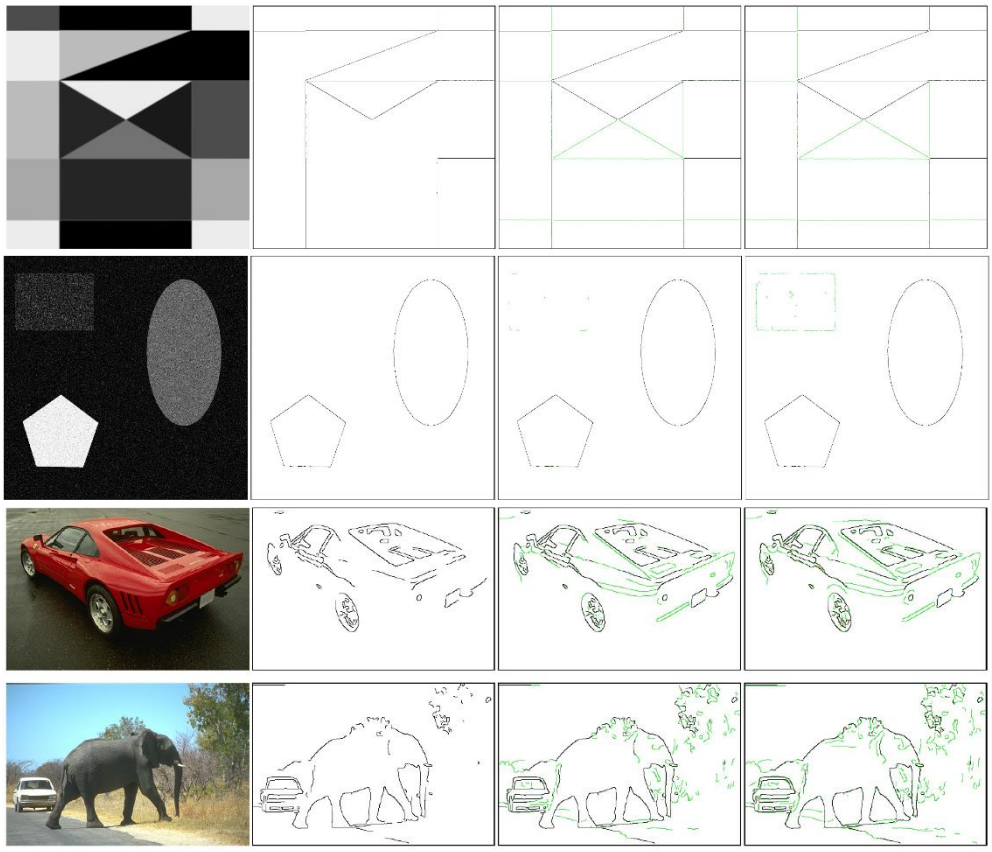


Figure 4.4 Edge map resulted using Canny algorithm with the same parameters. Columns are original image, edge map using $3x3$ kernel, edge map using $5x5$ dilated kernel, edge map using $7x7$ dilated kernel. Source of image is [9].

To prove the hypothesis evaluation was done using several edge detection techniques. In [9] details for each edge detection algorithm can be found together with steps executed for each of the following:

- First-Order Derivative Orthogonal Gradient Operators [172],
- First-Order Derivative Compass Gradient Operators [119], [173],
- Frei–Chen Operator [174], [175],

- Laplacian Edge Operator [176],
- Laplacian of Gaussian Operator [172],
- Marr–Hildreth algorithm [177],
- Canny algorithm,
- Shen–Castan Algorithm [178],
- Edge-Drawing Algorithm [179], [180].

To properly benchmark the results the authors have used two different benchmarking algorithms on two different datasets. The scope of this dual evaluation was to investigate different aspects of edge map usage.

The first benchmark dataset used was the popular Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) [181] as an expert judgement dataset. BSDS500 is a commonly used dataset for benchmarking boundary detection, a common use case of edge-detection. The dataset comprises natural images that have been manually segmented. Examples of images found in dataset are presented in Figure 4.5.

Figure 4.5 Example of images from BSDS500 with equivalent ground truth [9]

For edge detection evaluation the Pixel Correspondence Metric (PCM) algorithm [182] was used on the BSDS500 dataset. "This metric is reliable for correlating similarities because it searches for the optimal matching of the pixels between the edge images and then estimates the error produced by this matching" [9].

In Equation (4.1) PCM is defined between two images f and g, where the cost of an optimal match is represented as $C\left(M_{opt}(f,g)\right)$, the maximum localization error as η and the total number of pixels that are not zero as $|f \cup g|$.

$$\boldsymbol{PCM_{\eta}(f,g) = 100 \cdot \left(1 - \frac{C\left(M_{opt}(f,g)\right)}{(|f \cup g|)}\right)} \tag{4.1}$$

For each image precision (P), recall (R) and F-measure (F1) [183] is calculated. The highest possible value of an F-score (Equation (4.4)) is 1.0, indicating perfect P and R. P (Equation (4.2)) is the probability that a resulting edge/boundary pixel was labelled as a true edge/boundary pixel. R (Equation (4.3)) is the probability that a true edge/boundary pixel was detected [9].

$$\boldsymbol{P = \frac{TP}{TP+FP}} \tag{4.2}$$

$$\boldsymbol{R} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \tag{4.3}$$

$$\boldsymbol{F1} = \boldsymbol{2} \cdot \frac{\boldsymbol{P \cdot R}}{\boldsymbol{P+R}} = \frac{\boldsymbol{2 \cdot TP}}{\boldsymbol{2 \cdot TP+FP+FN}} \tag{4.4}$$

The second benchmark dataset used was a synthetic dataset from [165]. "The gray-scale dataset contains various edge types (such as step edges, ramp and roof edge), widely varying angles, various junctions and brightness changes" [9]. An example of the images present in this dataset are presented in Figure 4.6.

To evaluate the results on the second dataset the Symmetric Figure of Merit (SFoM) [184], [185] metric is used. This metric is a deviation of Figure of Merit (FoM) [186] where in order to avoid the computation of the distance of FPs only in one direction we would consider a combination of $FoM(f,g)$ and $FoM(g,f)$, as we can see in Equation (4.6).

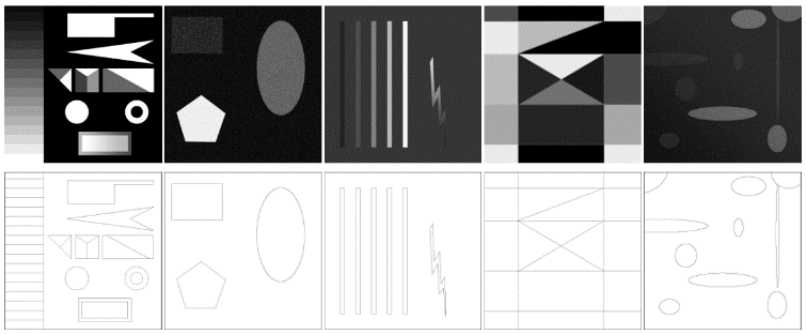


Figure 4.6 Example of images from the synthetic dataset [9]

FoM measures the discrepancy between the edge map and the reference edge image. This measure, which is mostly a distance measurement, ranges from 0 to 1 [9]. In Equation (4.5) the FOM is presented with the following notation f for the predicted image, g for the ground truth image, k as a constant equal to 19 and $d_g(p)$ as the minimal Euclidean distance between pixel p from f and g.

$$\boldsymbol{FoM(f,g)} = \frac{\boldsymbol{1}}{\boldsymbol{max(|f|,|g|)}} \cdot \boldsymbol{\sum_{p \in g} \left(1 + k \cdot d_g^2(p)\right)} \tag{4.5}$$

$$\boldsymbol{SFoM(f,g)} = \boldsymbol{\frac{1}{2} \cdot FoM(f,g) + \frac{1}{2} \cdot FoM(g,f)} \tag{4.6}$$

The mentioned edge detection algorithms were evaluated on the two different datasets using the specified benchmark algorithms to test the hypothesis that dilating the filters obtains better results than the classical ones or extended versions.

In Table 4.1 the overview of the result evaluation is presented from [9]. As we can observe in most of the cases the best results were obtained when using the dilated version of the kernels.

As we can observe by dilation an increase in F1-score is obtained. The simple structure of the dilated filters, they are also a good choice when the runtime matters. The other classical filter extensions from [187]–[192] increase the targeted region of the filter but the number of operations too without a significant improvement in order of metric obtained [9].

Table 4.1 Overall best results for each edge operator

Operator	PCM				SFoM	
	Kernel	P	R	F1	Kernel	SFOM
First Order Orthogonal	7 × 7(D)	0.571	0.679	0.621	7 × 7(D)	0.754
First Order Compass	5 × 5(D)	0.565	0.687	0.620	7 × 7(D)	0.753
Frei–Chen	5 × 5(D)	0.554	0.673	0.608	7 × 7(D)	0.481
Laplace	7 × 7(D)	0.309	0.794	0.445	5 × 5(D)	0.478
LoG	7 × 7(D)	0.490	0.659	0.562	7 × 7(D)	0.491
Marr–Hildreth	7 × 7(D)	0.450	0.744	0.561	5 × 5(D)	0.411
Canny	7 × 7(D)	0.478	0.813	0.602	7 × 7(D)	0.679
Shen–Castan	7 × 7(D)	0.483	0.711	0.576	5 × 5(D)	0.442
ED	7 × 7(D)	0.556	0.704	0.621	3 × 3	0.673

Another interesting experiment focusing on dilated filters in edge detection was done in [165]. The authors analysed and compared the dilation of filters with reconstruction approaches. In the paper an experimental proof of the hypothesis that dilating a $3x3$ a filter with a factor of 1 is similar to applying the same filter in the immediately lower scale pyramid level is found.

The experiment was designed using two popular algorithms like First Order Derivative Orthogonal and Canny algorithm evaluated on BSDS500. In one part the two-edge detection algorithm would use the dilated $5x5$ and $7x7$ kernels, and in the other part different reconstructing algorithms were used to bring back the edge map to initial pyramid level. Details on the steps done for each reconstructing algorithm can be found in [165].

The experiment was scaled in order to be tested using different filters than the classical Sobel [166] like Prewitt [167], Kirsch [193], Scharr [194], Kayyali [195], Kroon [196]. The test concludes that in most cases, using dilated filters produces similar results as processing at lower scale level and expanding it.

Naturally benefits exist when extracting features from a lower level in terms of noise reduction. But when doing so one should take in consideration the additional computational steps that appear when we desire to bring edge maps back to the initial level.

4.4. Dilated filters in image sharpening

4.4.1. Introduction

Image enhancement of contrast is a concern related to the sharpening of certain features like edges, object boundaries or textures. The main scope of this activity is to improve the visual appearance of the image.

Various approaches have been advanced for contrast enhancement. One of the most common methods used is the histogram equalization. This method is based on the mapping of grey level is a matter that is uniformly distributed. This method has disadvantages like attenuation of low contrast or high computation cost [176], [197], [198].

Another popular solution for sharpening is the high-pass filter (HP). Filters like the low pass, band reject or HP are identified as ideal filters. An ideal filter has the property that it cuts all frequencies above (or below) a certain threshold frequency. In Equation (4.7) and (4.8) the ideal filter is presented, where (M, N) is the filter sizes and D_0 is the cut off distance. Other HP filters have evolved over time like the Butterworth filter, see Equation (4.9), and Gaussian, see Equation (4.10) [176], [199]–[201].

$$\boldsymbol{H(u,v)} = \begin{cases} \boldsymbol{1, if\ D(u,v) > D_0} \\ \boldsymbol{0, if\ D(u,v) \leq D_0} \end{cases} \tag{4.7}$$

$$\boldsymbol{D(u,v)} = \left[\left(\boldsymbol{u} - \frac{\boldsymbol{M}}{\boldsymbol{2}}\right)^{\boldsymbol{2}} + \left(\boldsymbol{v} - \frac{\boldsymbol{N}}{\boldsymbol{2}}\right)^{\boldsymbol{2}}\right]^{\frac{\boldsymbol{1}}{\boldsymbol{2}}} \tag{4.8}$$

$$\boldsymbol{H(u,v)} = \frac{\boldsymbol{1}}{\boldsymbol{1} + [\boldsymbol{D_0}/\boldsymbol{D(u,v)}]^{\boldsymbol{2n}}} \tag{4.9}$$

$$\boldsymbol{H(u,v)} = \boldsymbol{1} - \boldsymbol{e}^{-\frac{\boldsymbol{D^2(u,v)}}{\boldsymbol{2D_0^2}}} \tag{4.10}$$

A downside of the HP is the fact that they can amplify noise on an image. High-pass filtering can exaggerate a small, faint detail so while HP can improve an image by sharpening detail, overdoing it will degrade the quality considerably.

$$\begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} - \begin{bmatrix} \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} \\ \boldsymbol{1} & \boldsymbol{-4} & \boldsymbol{1} \\ \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{-1} & \boldsymbol{0} \\ \boldsymbol{-1} & \boldsymbol{5} & \boldsymbol{-1} \\ \boldsymbol{0} & \boldsymbol{-1} & \boldsymbol{0} \end{bmatrix} \tag{4.11}$$

$$\begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} - \begin{bmatrix} \boldsymbol{1} & \boldsymbol{1} & \boldsymbol{1} \\ \boldsymbol{1} & \boldsymbol{-8} & \boldsymbol{1} \\ \boldsymbol{1} & \boldsymbol{1} & \boldsymbol{1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{-1} & \boldsymbol{-1} & \boldsymbol{-1} \\ \boldsymbol{-1} & \boldsymbol{9} & \boldsymbol{-1} \\ \boldsymbol{-1} & \boldsymbol{-1} & \boldsymbol{-1} \end{bmatrix} \tag{4.12}$$

As we can observe in Equation (4.11) and (4.12) examples of discrete HP in 2D. One important aspect to specify is if the kernel sum is 0 the results would be an edge detection, not an image enhancement.

Another popular solution for image enhancement is the unsharp masking (UM) technique, this technique appeared in photography with the aim to improve the quality of pictures by masking their details. In this variant we would consider subtracting a blurred copy of the image from the original image itself. Even if the solution is a simple one it comes with several downsides like high sensitivity to noise and possible enhancements to high-contrast areas [202]–[205].

$$\boldsymbol{f_{sharp}(x,y)} = \boldsymbol{f(x,y)} + \boldsymbol{k} \cdot \boldsymbol{g(x,y)} \tag{4.13}$$

The complete UM operator is presented in Figure 4.7, and we can summarize it in Equation (4.13), where k is a scaling constant. The k value constant can typically vary between 0.2 and 0.7, the higher the value the stronger the sharpening.

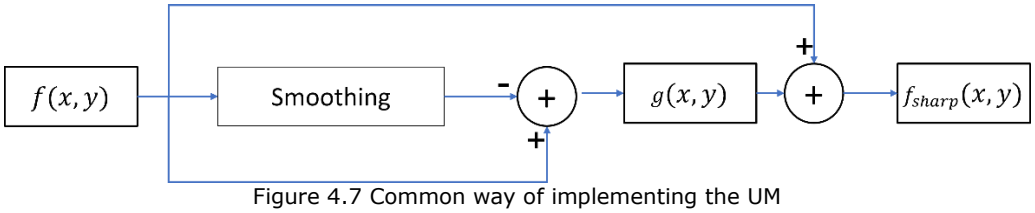


Figure 4.7 Common way of implementing the UM

In order to highlight what each algorithm presented brings in terms of image sharpness enhancement we chose an image with low visible conditions from TMBuD dataset [39]. The results and the corresponding pixel histogram for the RGB image are presented in Figure 4.8 and similar for the grayscale images in Figure 4.9. For the UM algorithm, it used a strength of 0.7.

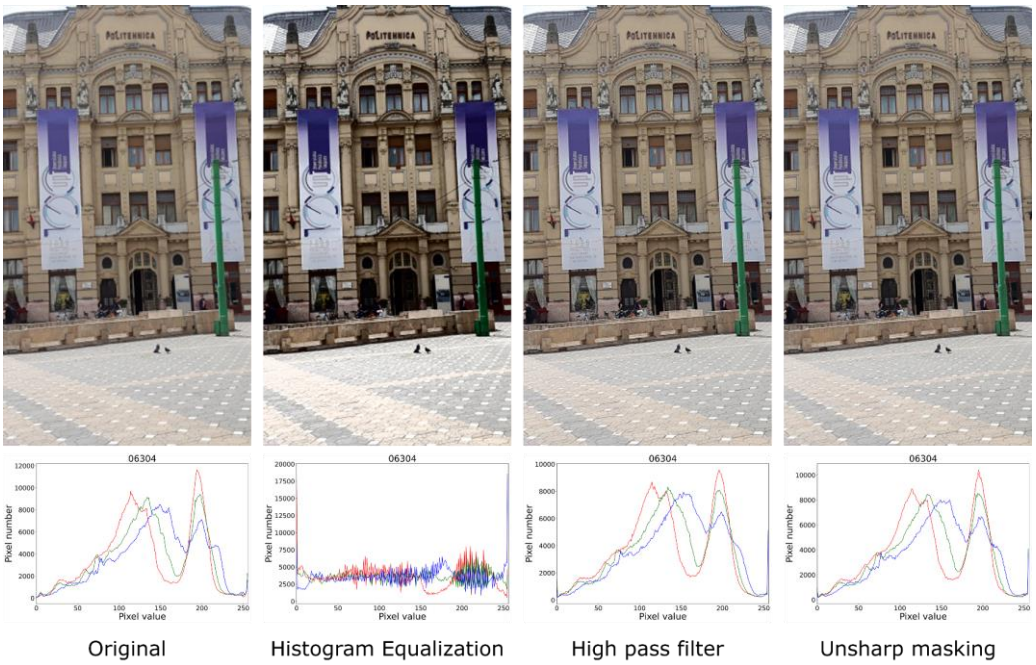


Figure 4.8 Image sharpening effect example on colour mobile phone image

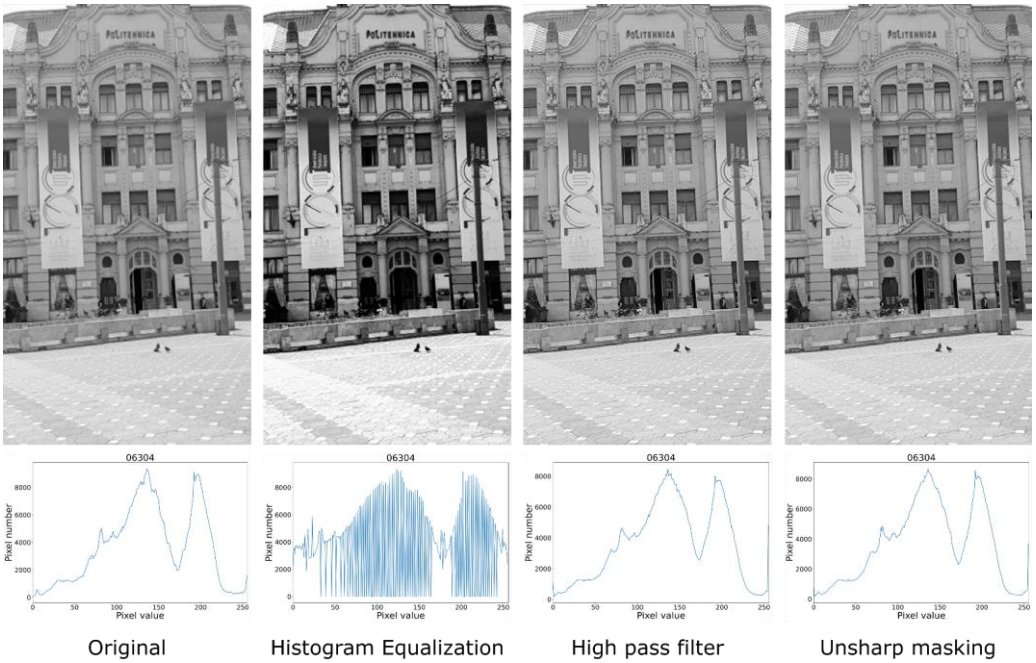


Figure 4.9 Image sharpening effect example on grayscale mobile phone image

As we can observe, the sharpening algorithm work in both colour spectrums with decent results. Of course, in literature we are presented with more complex sharpening algorithms like in [206]–[209] but for this chapter we are concern in presenting the basic solutions.

An aspect worth mentioning when looking over the histograms is that the histogram equalization algorithm makes the most alterations on the histogram, compared with the HP or UM.

When comparing the histograms from Figure 4.8 and Figure 4.9 we can observe that even if the image spectrum is changed, the trend of the pixel distribution remains the same.

4.4.2. Dilated high-pass filter and UM

From section 4.3 we concluded that dilated filters bring benefits to the edge detection process. Now I would like to investigate how this affects the sharpening process.

In this scope we would dilate the following Laplace kernels found in literature, see Figure 4.10. By doing this we would like to take in consideration when applying the sharpening of a wider neighbourhood. To correct assess if this is a benefit, we need to take in consideration the $5x5$ extended kernels.

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 1 & 2 & -17 & 2 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -4 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -18 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -8 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

3×3 5×5 5×5 dilated

Figure 4.10 Laplace kernels: $\mathbf{3x3}$, $\mathbf{5x5}$ and $\mathbf{5x5}$ dilated

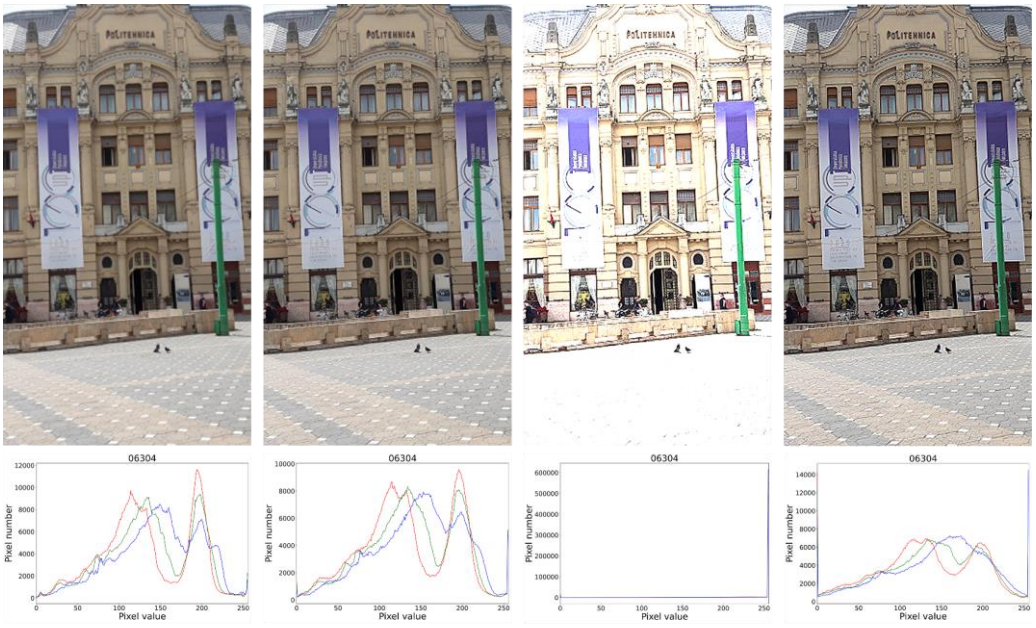
As stated before, with the gaping that occurs with dilating filers, we would expect to be more resistant to noise impact and take in consideration a wider region of interest when sharpening.

Because of the conclusions from the last section, where I obtained similar results on grayscale and coloured images, I will present only the results on colour images for visual comparison.

Visual results for the sharpening HP with different Laplace kernel versions can be observed in Figure 4.12 and in Figure 4.11. First remark in this case is that when using the $5x5$ kernel, the results are not promising. We clearly see a burn effect on the image in case of the expanded kernels.

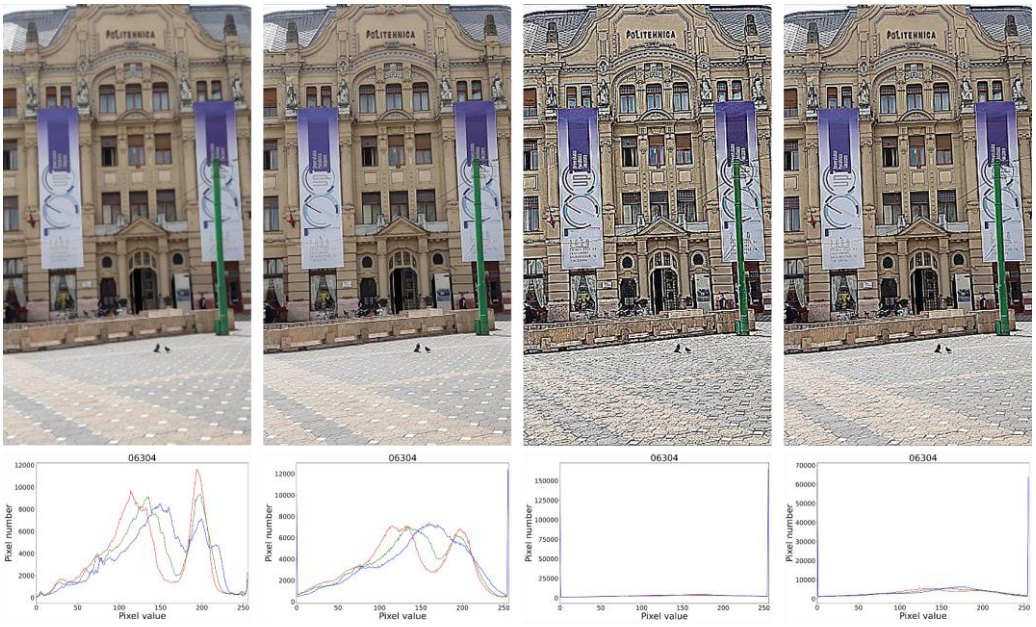
Typically, when applying a sharpen filter we wish to enhance the image for future processing but avoid over saturation or crisping effect on edges. In the case of $5x5$ the effect is present and renders the images close to unusable for must applications.

In both cases presented in Figure 4.12 and Figure 4.11 we observe better results for $5x5$ dilated filters than the classical $3x3$ kernel.



Original High pass filter 3x3 High pass filter 5x5 High pass filter 5x5(d)

Figure 4.12 Results of HP using Laplace V1 kernel



Original High pass filter 3x3 High pass filter 5x5 High pass filter 5x5(d)

Figure 4.11 Results of HP using Laplace V2 kernel

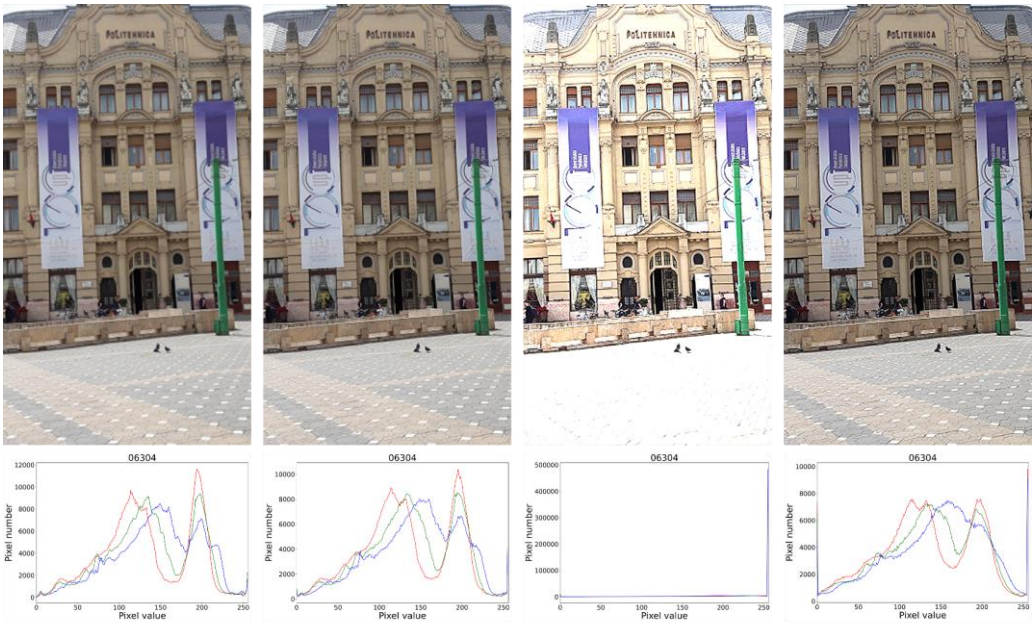


Figure 4.14 Results of UM using Laplace V1 kernel

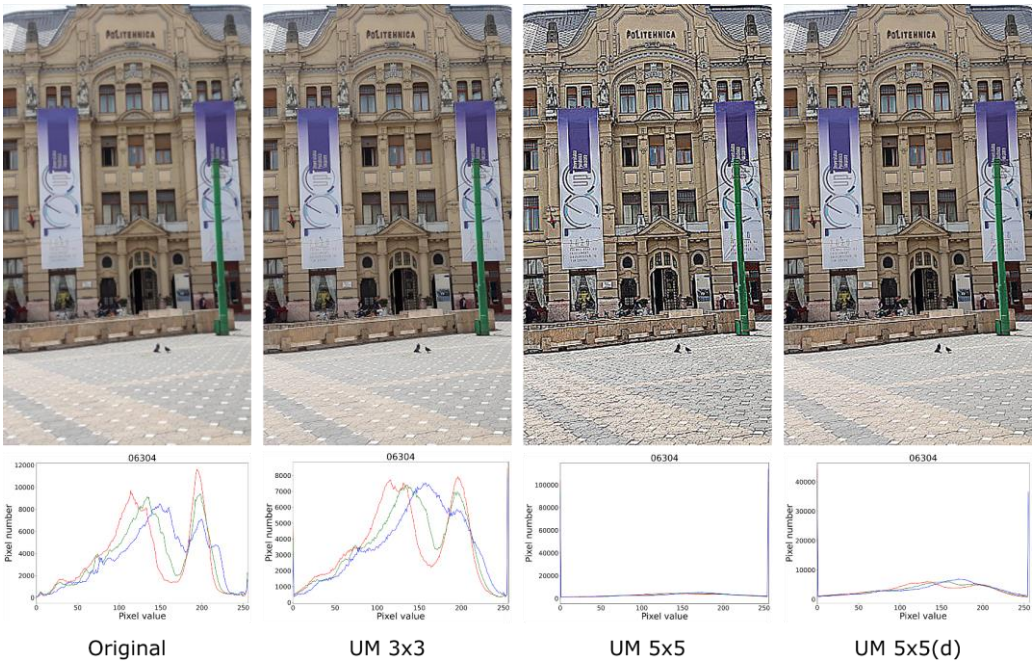


Figure 4.13 Results of UM using Laplace V2 kernel

In Figure 4.14 and Figure 4.13 the results of UM are presented using standard $3x3$, extended $5x5$ and dilated $5x5$ Laplace kernels V1, respectively V2.

As in the HP case we see a clear over sharpening effect in case of the $5x5$ extended version of the kernels. But when using a $5x5$ dilated version we can observe, visually, an improvement of the sharpness.

These initial visual results are encouraging enough in order to engage into a statistical evaluation of this proposed sharpening solution. Like in case of edge detection, I hope to obtain better metrics using a wider neighbourhood with the same computation resources as using $3x3$.

4.4.3. Benchmarking the sharpness

In this section evaluation criterions for sharpening are presented. The evaluation of an enhanced image is a difficult task for two reasons: there is no specific index that determines the optimal level of sharpness and the fact that any sharpening algorithm we would use has several parameters that can change the output [210].

From literature we concluded that the following metrics are used to evaluate and judge the effect of sharpening: entropy of an image, the spatial factor and mean contrast [211]–[213].

Entropy, in information theory, is the average level of information inherent or the amount of uncertainty of an event associated with a given probability distribution. Entropy can and is used to measure the 'disorder' level. As the level of disorder is higher, the entropy value increases, and the expected event becomes less predictable. The formula for it is presented in Equation (4.14) [173].

$$\boldsymbol{H(s_m) = -\sum_{n=0}^{255} p_n(s_n) \cdot \log_2\left(p_n(s_m)\right)} \quad (4.14)$$

Spatial Frequency (SF) is the measure of the overall activity level in an image [214]. For an MxN image block F, with grey values we can define SF as in Equation (4.15), where RF is the row frequency, described in Equation (4.16) and CF the column frequency, described in Equation (4.17).

$$\boldsymbol{SF = \sqrt{(RF)^2 + (CF)^2}} \quad (4.15)$$

$$\boldsymbol{RF = \sqrt{\frac{1}{MN}\sum_{m=1}^{M}\sum_{n=2}^{N}[F(m,n) - F(m,n-1)]^2}} \quad (4.16)$$

$$\boldsymbol{CF = \sqrt{\frac{1}{MN}\sum_{n=1}^{N}\sum_{m=2}^{M}[F(m,n) - F(m-1,n)]^2}} \quad (4.17)$$

The last metric I will use is the mean contrast of an image, or mean pixel value, see Equation (4.18). Even if this is not a complex metric it will show how the contrast changed and by doing so if an object becomes more distinguishable.

$$C = \left(\sum_{n=1}^{N}\sum_{m=1}^{M} F(m,n)\right) \Big/ (M \cdot N) \tag{4.18}$$

To better benchmark my results, the evaluation will be done on a series of images, presented in Figure 4.15. The dataset comprises different situations like strongly blurred, slightly blurred, low visibility, darked regions, low resolution, and high-resolution images. The images were chosen from the TMBuD dataset and will be used in the detection phase too.

The good resolution image is included in the evaluation dataset with the scope of permitting us to observe if the sharpening deteriorates substantially the "good" images.

Figure 4.15 Images from the benchmark dataset

4.4.4. Evaluation results

In this evaluation I included the results of HP and UM, presented in section 4.1, using $3x3$, $5x5$ extended and $5x5$ dilated filters of Laplace V1 and V2, presented in Figure 4.10. The metrics and dataset used are the ones presented in section 4.4.3. For this evaluation I would consider only grayscale images.

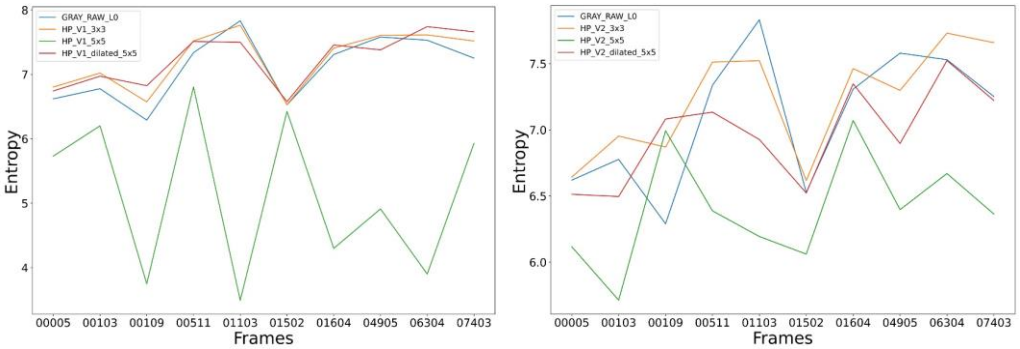


Figure 4.18 Entropy metric for HP for each image from dataset

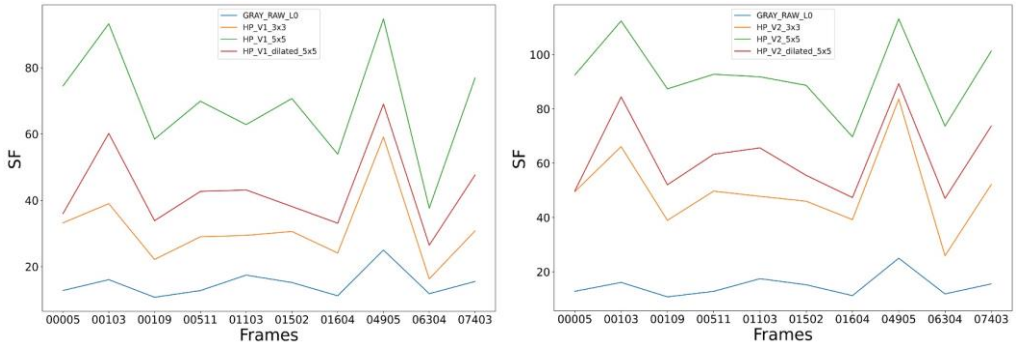


Figure 4.17 Mean contrast metric for high pass filter for each image from dataset

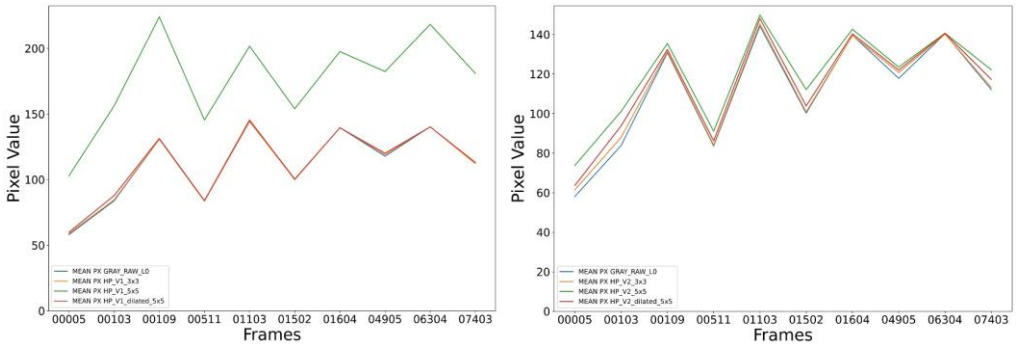


Figure 4.16 SF metric for HP for each image from dataset

In Figure 4.18 I present the Entropy metric results for each of the 10 images from the dataset. As presented, a higher value of Entropy is associated with a sharper image so we can observe that when using V1 Laplace, Figure 4.18(left), we obtain in most cases higher values. When using the V2 kernel, Figure 4.18(right), we clearly

observe a non-deterministic trend of Entropy values, in some cases the entropy falls beneath the original image even.

In Figure 4.16 we are presented with the SF matric results per image. As we can observe the highest values for this metric is obtain when using the $5x5$ extended kernel. But as stated before, none of these metrics are deterministic directly with the quality of the sharpness of the image. In this case, we see that SF value that high is equivalent to burning effect, as we saw in the visual results.

In Figure 4.17 we are presented with the mean contrast value of the images where we observe and can conclude that none of the filters produce systematic changes in the intensity distribution of the image. This is an important aspect when evaluating image manipulation of this sort.

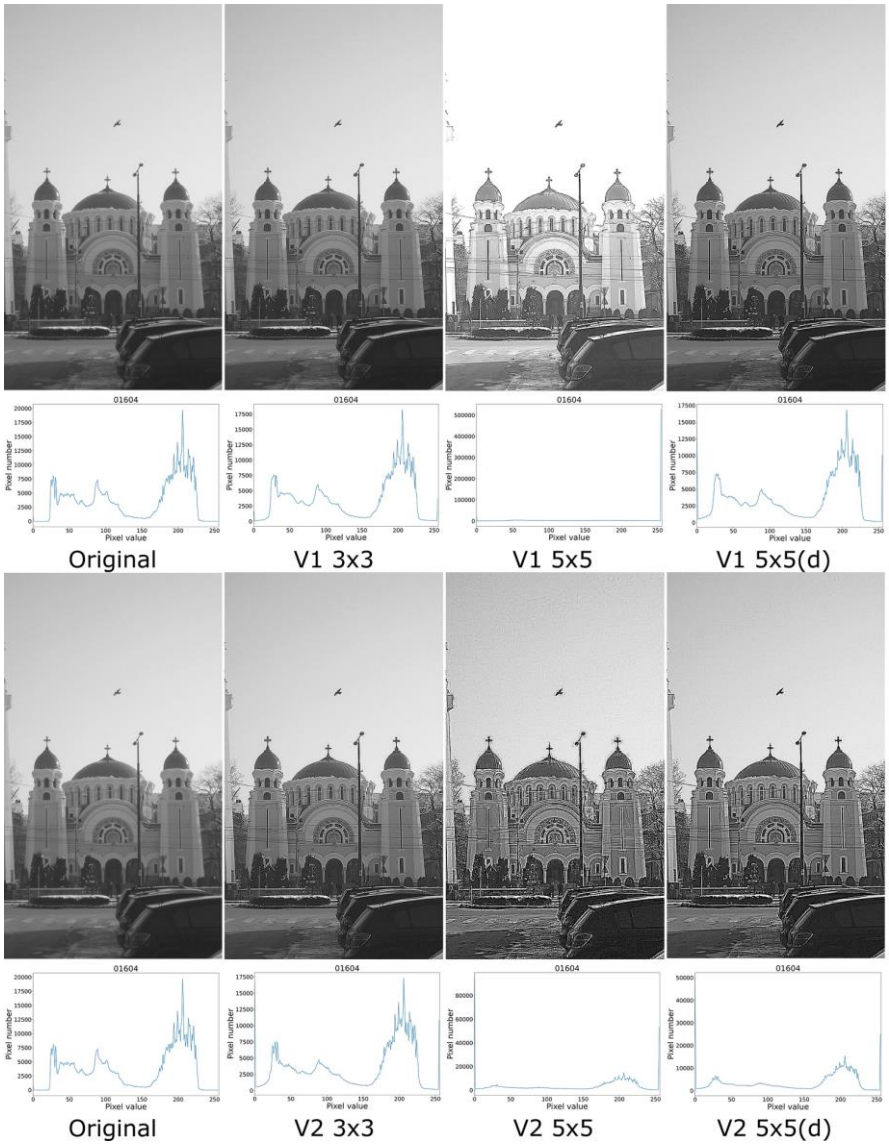


Figure 4.19 Visual results of high pass filter using V1 and V2

In Figure 4.19 we are presented with a visual example of the results we obtained on image "01604" from the dataset. As we can observe when using the dilated filters, the images are enhanced better than with the classical $3x3$ but lower than when using $5x5$. From the histograms we can conclude that using V1 kernels does not flatten that much the histogram as in case of the V2 kernels.

Table 4.2 Average results per entire dataset for high pass filter

	Entropy	**SF**
Original image	7.105	14.881
HP V1 $3x3$	7.234	31.383
HP V1 $5x5$	5.143	69.286
HP V1 $5x5$ dilated	7.236	43.032
HP V2 $3x3$	7.226	49.896
HP V2 $5x5$	6.396	92.332
HP V2 $5x5$ dilated	6.966	62.801

The average results from the metrics used, Entropy and SF, are centralized in Table 4.2 and we conclude that the trend we saw in Figure 4.18 and Figure 4.16 are respected across the entire dataset. The Entropy is lower when using the $5x5$ kernels but higher than when using the $5x5$ dilated. In case of SF we see that when using the dilated version of the kernels the values are moderate higher than when using the classical $3x3$.

Now, after investigating the high pass filter approach we will investigate the UM approach to see if the results are consistent to those found till now.

In Figure 4.22 we can observe the entropy metric results when using the UM algorithm for sharpening with $3x3$, $5x5$ and $5x5$ dilated kernels. Different from the high pass results in this case we obtain a smaller value for Entropy when using the $5x5$ extended. But when using the $5x5$ dilated version we observe a slight increase in the metric, which is an improvement that can indicate clearly that we avoid over sharpening the image.

Similar to before, in case of SF metric, see Figure 4.21, we obtain the highest values when using the $5x5$ extended, followed by the UM when using $5x5$ dilated. In this case we can conclude that using $5x5$ dilated offer more sharpness but not to the limit of "burning" the edge of an image.

An interesting aspect is observed in Figure 4.20 where the mean contrast is plotted. The mean value of contrast in each image does not change radically when applying the UM with either kernel, exception being V1 $5x5$ extended. This may be an important aspect because it suggests that even if the pixel values are changed the distribution of them is not.

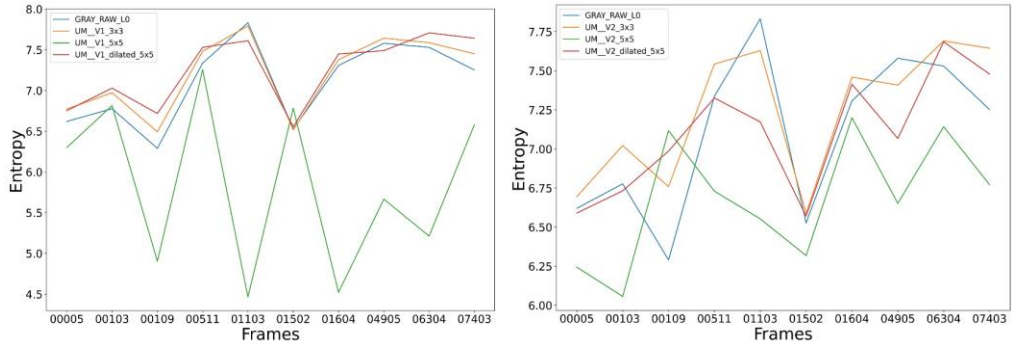


Figure 4.22 Entropy metric for UM for each image from dataset

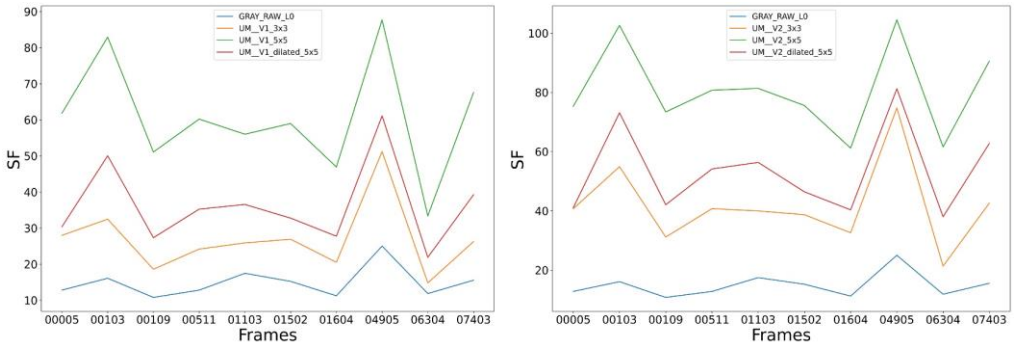


Figure 4.21 SF metric for UM for each image from dataset

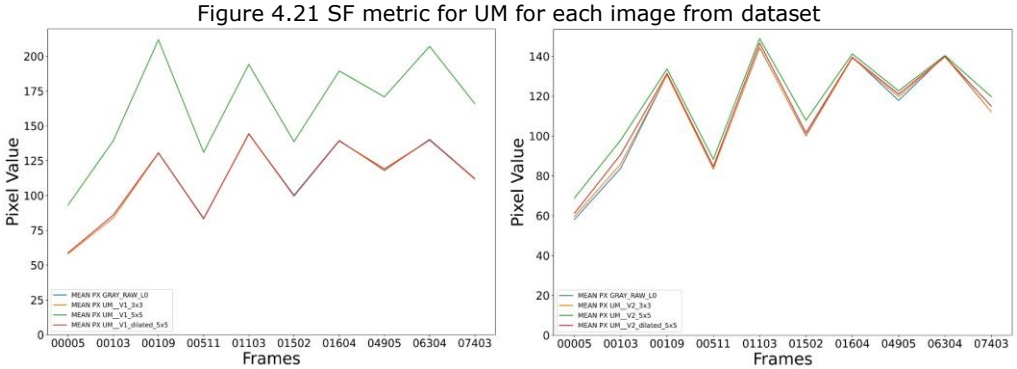


Figure 4.20 Mean contrast metric for UM for each image from dataset

In Figure 4.23 visual results are presented for this case and in Table 4.3 the average results of the metrics calculated. As before we can observe that using $5x5$ dilate kernels bring forward better results than the classical $3x3$. Another aspect worth mentioning is that using an $5x5$ extended filter does not seem to be a good solution for this activity.

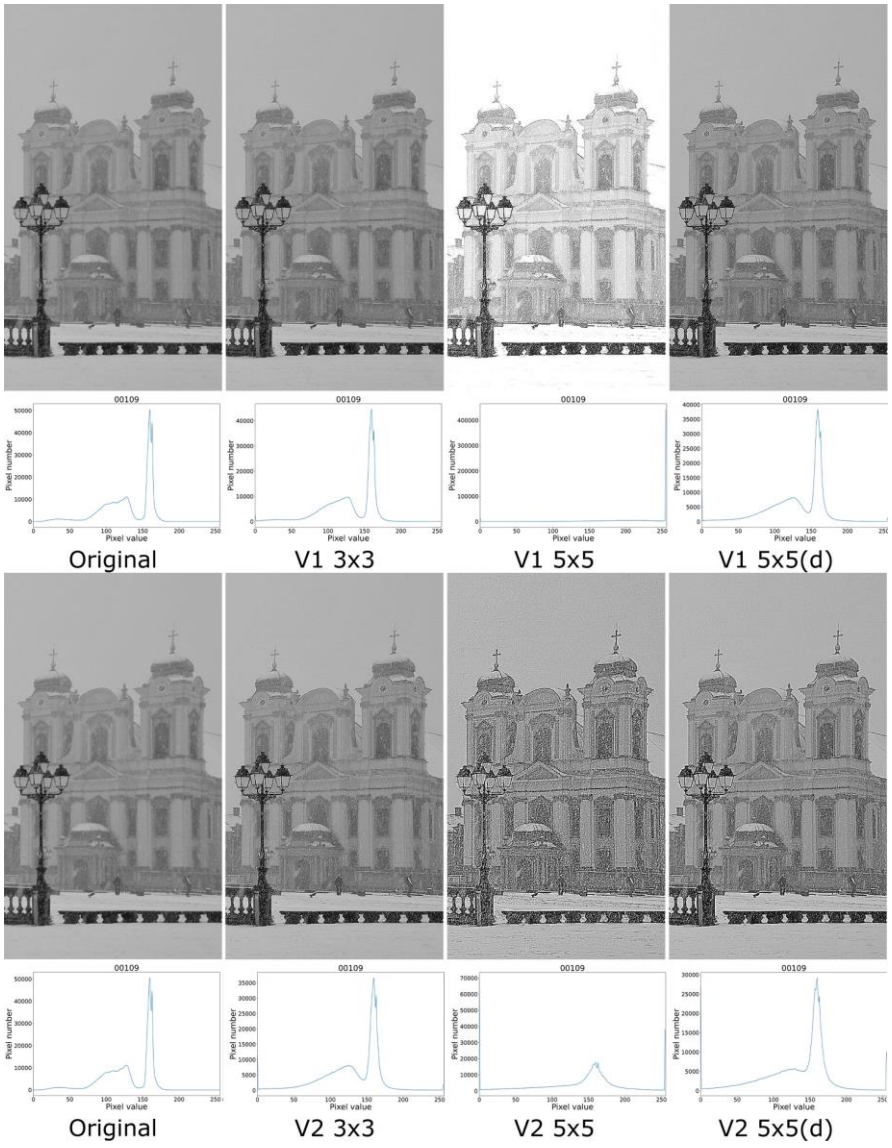


Figure 4.23 Visual results of UM using V1 and V2

From Table 4.2 and Table 4.3 we can conclude that using dilated filters for sharpening an image is an option to consider. In general, it brings better results than the classical $3x3$ ones in terms of image enhancement.

Table 4.3 Average results per entire dataset for UM

	Entropy	**SF**
Original image	7.105	14.881
UM V1 $3x3$	7.208	26.875
UM V1 $5x5$	5.850	60.652
UM V1 $5x5$ dilated	7.248	36.220
UM V2 $3x3$	7.243	41.752
UM V2 $5x5$	6.678	80.673
UM V2 $5x5$ dilated	7.102	53.548

4.5. Conclusion

In this chapter I presented the concept of dilated filters. This action of dilation even if simple, just gaping with zeros the new formed position, brings forward good results in simple or complex algorithms that use convolutions.

Firstly, I presented from the literature the effects that dilated filters have upon edge detection algorithms, see chapter 4.3. From Table 4.1 I concluded that using dilated kernels instead of classical ones has benefits.

Secondly, encouraged by the results obtained for edge detection, I investigated the possibility of using dilated filters for image sharpening. The concept of image sharpening is presented in chapter 4.4.1 and the evaluation in chapter 4.4.4.

From Table 4.2 and Table 4.3 I concluded that using dilated filters for image sharpening is a path worth pursuing. In our scope we only analysed the feasibility of the concept using standard basic classical sharpening solutions in order to prove this. This scope limitation is driven by a practical factor too, factor being that in the proposed landmark detection algorithm a fast computational solution is needed.

This investigation has generated the idea that dilated filters can be used in more advanced sharpening algorithms like [205], [208], [210]. This aspect would be researched in future work.

Another aspect that would be beneficial to research would be the possibility of measuring the level of sharpness quality of an image. This topic was approached in literature by scholars over the years in [215], [216].

5. PROPOSED LANDMARK DETECTION SYSTEM

5.1. Introduction

In this chapter I will present the proposed system for detection of urban landmarks specific for mobile device street-view perspective. The proposed system or algorithm can be used in any urban scenario but, in this case, it will be tuned for the Timişoara scenario.

Landmark (building) recognition is deemed to be an object detection or content-based image retrieval problem for a specific scope. Compared to general object recognition tasks, this specific one brings more challenges because most urban images contain both human-made objects and natural ones.

"Images taken of the same building could demonstrate a wide range of variability – they may be taken from different viewpoints, under different lighting conditions, or suffer from partial occlusions from trees, moving vehicles, other buildings, or themselves. Therefore, an ideal building recognition technique should be sensitive enough to identify an individual building while robust to different geometric and photometric image transformations" [14].

In Figure 2.3 the general scheme for a landmark detection system is presented. The proposed system is similar in certain parts with [23], [78], [106] and it is a continuous of our previous work done in [108].

In the beginning, in chapter 5.2 I will present the benchmark dataset that will be used for evaluating the entire algorithm and fine-tune certain parts. The TMBuD dataset presented is an extension of the one presented in chapter 2.3.

In chapter 5.3 the proposed landmark detection algorithm is presented in detail. The system will use ground (street-view) images and GPS contextual data to recognize human made landmarks from Timişoara.

The proposed algorithm is capable of handling difficult situations or scenarios when multiple landmarks of interest are clustered into a small arial like city squares or tourist neighbourhoods. These situations are the most interesting ones from the applications perspective as tourism applications usually have difficulties in this kind of situation.

The system is benchmarked on publicly available dataset and Timişoara dataset. These duality in evaluation should offer the necessary confidence upon the system to be considered in future tourism applications. The results for this evaluation are presented in chapter 5.4.

Beside the evaluation based on image datasets I will present an evaluation upon a recorded movie from a mobile phone with scenarios from Timişoara. These certain evaluation use cases should simulate the future use in mobile tourism applications. In this use case the system should be able to handle motion and continuous distractors in the detection. Initial results have revealed positive and encouraging results upon evaluation.

5.2. TMBuD -detection dataset

In this chapter the TMBuD building detection dataset is presented, this is an extension of the TMBuD dataset [39] presented in chapter 2.3. This dataset aims to support a quantitative evaluation of mobile visual search supported by GPS context, of a subset of landmarks in the Timişoara area.

The main novelty of this dataset is the clear targeted scope of containing landmarks only from Timişoara city, which is the main arial targeted by our landmark detection system.

The TMBuD building dataset contains 1097 images with the resolution of 768x1024 pixels taken using a mobile phone from a street view perspective. The images are organized into 125 district landmarks located in several tourist parts of Timişoara. The dataset can be accessed from the TMBuD repository [217] using the release branch *TMBuD_Building_Detection_v1*.

Figure 5.1 Example of one unique landmark inside TMBuD dataset

An important aspect to specify is that the benchmark comprises mobile photos of urban landmarks that aim to include variable quality, blurring, lighting changes, occlusions, and various viewing angles. This aspect is presented in Figure 5.1 where several images were selected from one landmark. The intention was to capture photos of different shot sizes (close, medium, long) and different angles of direction to be able to mimic real life scenarios.

The proposed benchmark dataset has different perspectives and conditions, as stated before, but it offers the possibility to evaluate a system in two unique conditions: low resolution images captured and night-time conditions. As presented in Figure 5.2 the dataset offers a limited number of images from landmarks in a low resolution, 7.66% of them, and images taken at night, 14.9% of them.

Low resolution images can occur in a mobile detection system from multiple sources like bad signal, low end mobile device and so on. This fact triggers the idea that a benchmark dataset should incorporate this kind of images inside of it to observe how a system will react.

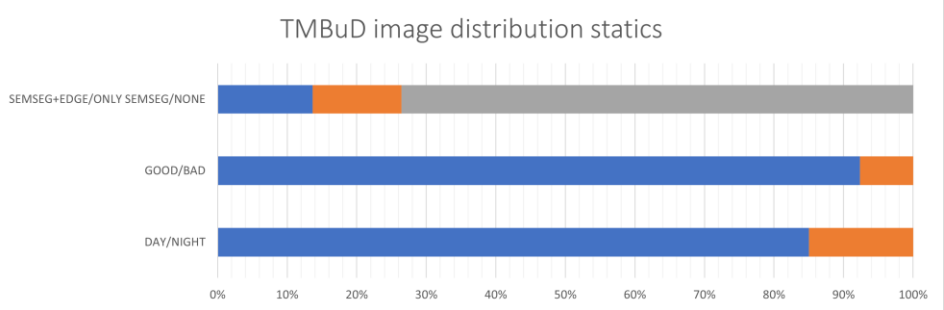


Figure 5.2 Distribution of images from TMBuD

For each image from the dataset a series of information is available that are described in Table 5.1. All the information is stored inside a csv file (*DATA SPLIT.csv*) which is used for creating the desired datasets.

Table 5.1 Information for each image in TMBuD dataset

Field of csv file	**Explication**
Picture name	Naming format is *xxxyy*, where the *xxx* represents the landmark number and *yy* the image number
Landmark name	The name of the landmark represented in the image.
Coordinates Landmark	The coordinates of the landmark in WGS 84 Web Mercator system, taken from google maps[6].
Coordinates image	The coordinates of the landmark in WGS 84 Web Mercator system, taken from google maps[6].
GT salient edges	If the image has a ground truth of salient edges equivalence.
GT labels	If the image has a ground truth of semantic segmentation equivalence.
Condition	If the image was taken at NIGHT or DAY condition.
Quality	A human visual evaluation of the resolution quality of the image.
Dataset STANDARD	Subcategory of the "STANDARD" dataset the image is used (TRAIN/VALIDATE/TEST)
Dataset 3_2	Subcategory of the "Building detection 3 TRAIN 2 TEST" dataset the image is used (TRAIN/TEST/NONE)
Dataset 3_5_NIGHT	Subcategory of the "Building detection 3 TRAIN 5 TEST" dataset the image is used (TRAIN/TEST/NONE)
Dataset 3_N	Subcategory of the "Building detection 3 TRAIN N TEST" dataset the image is used (TRAIN/TEST/NONE)

[6] https://www.google.com/maps

Modern mobile systems do not have the limitation of recording or capturing good quality images at night. This fact triggered again the need of incorporating in the benchmark images at night. The presence of these images in the evaluation dataset cause the systems to be evaluated in these conditions.

Different from generalized visual search, mobile visual search scenarios can benefit from rich contextual information like GPS. This contextual help of the GPS data can be a helpful tool if we are talking about an isolated landmark, but it can become problematic for cases where landmarks are geographically close one from another.

In Figure 5.3 we can observe the distribution of landmarks and position from which the images were taken in case of two squares of Timişoara. In cases like this we can clearly observe that a system cannot rely on GPS data alone to discriminate, from the same relative coordinates one can "target" different landmarks.

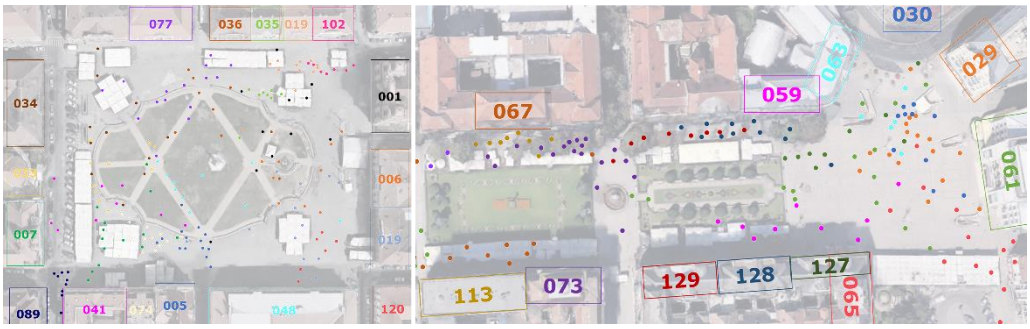


Figure 5.3 Landmark photo distribution by overlaying the location point of each collected photos on the Google map of "Unirii" Square Timișoara (left) and "Victory" Square Timișoara (right)

The two examples shown in Figure 5.3 clearly represent the main difficulty landmark detections system can and face in real life applications. If we look in the left image of Unirii Square, we can clearly see that from one point in space (one GPS tag) we can observe no less than 17 registered landmarks and many more buildings that are not of interest to us.

This situation becomes even more interesting when we consider that in this kind of scenario the number of distractors is constant and high. This fact is known and expected in urban areas like this where population density is always high.

To measure the photography diversity for each landmark we use the (lossless) JPEG compression size of the average images, similar to [31], [218]. This measure is done by computing the average image of each landmark and creating the lossless JPG file. The size of the file reveals the amount of information available. The theory behind this is that a diverse image will result in a blurrier average, compared with one that has a little diversity will result in a more structured, sharper one. Therefore, we can expect to see a smaller JPG file size of the average image for a landmark that offers more diverse images.

In Figure 5.5 two landmarks are presented with the average image results. The two landmarks chosen to be represented are the outliners of the dataset, the smallest and largest lossless JPG size.

To better understand the diversity of the landmarks offered by TMBuD we can see in Figure 5.4 the distribution of images and lossless JPG size of average image per landmark. On average, there are 7 different perspectives per landmark and lossless size of approximately 1.4 Mb.

Figure 5.5 Example of average images for one landmark. First row is the landmark with the highest lossless JPG size and in the second row the one with the smallest.

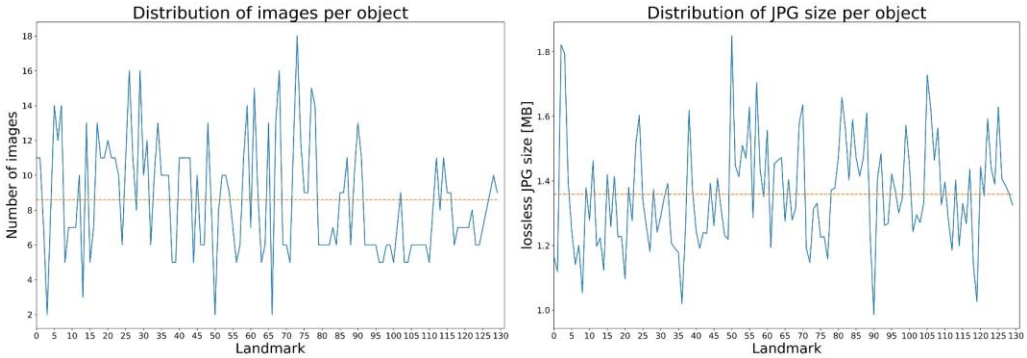


Figure 5.4 Number of images (left) and lossless JPG size (right) for each landmark from the dataset

To construct a certain dataset, one should use the offered python module (*parse_database.py*) with the instruction offered in the README file. For building detection benchmarking one can use the following variants:

i. BUILDING_DET_3 - which will offer 125 unique landmarks with 3 training images and 2 testing or inquiry images. For an easy integration the format the images are offered is similar to ZuBuD dataset [22]. Total number of images will be: 375 for training and 251 for testing.

ii. BUILDING_DET_3_NIGHT - which will offer 56 unique landmarks with 3 images for training and for testing 3-day time images and minimum 2 night time images. The format will remain the same as the previous variant. Total number of images will be: 168 for training and 281 for testing.

i. BUILDING_DET_3_N – which will offer 125 unique landmarks with 3 images for training and a variable number of testing images. Total number of images will be: 380 for training and 717 for testing.

The dataset provides as diverse landmark appearances as possible to simulate the real-world search challenges. Hence, the volunteers are encouraged to capture queries and ground truth photos without intent to avoid the intruding foreground, called noise in the semantic labelling.

In comparison with the other benchmark datasets, presented in Table 2.1, from chapter 2.2, the TMBuD benchmark offers a smaller amount of images per objective but tackles the problem of low resolution cases plus the night conditions.

5.3. Proposed detection system

Landmark urban detection problems can be assimilated as a CBIR problem. In many ways the same problems are tackled to obtain a good detection and classification as in a general CBIR system.

In Figure 5.6 I present the overall processing pipeline proposed for my use case. The proposed flow is presented in detail in the future subchapter and evaluated afterwards.

If we want to summarize all the steps needed for this action, we can enumerate the following: (i) image and metadata gathering; (ii) image pre-processing (task needed to enhance the final classification); (iii) feature extractor; (iv) code book generation; (v) classifier.

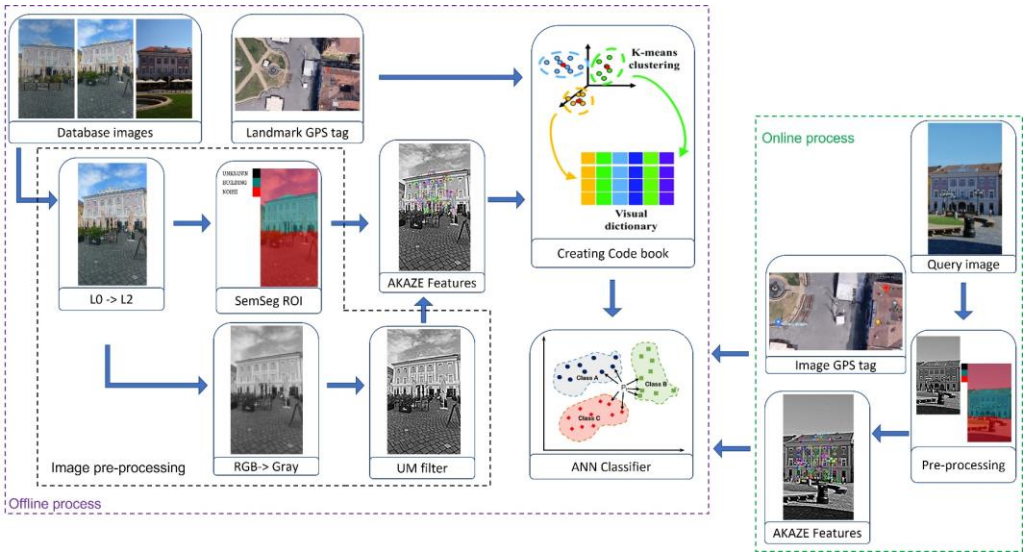


Figure 5.6 Proposed pipeline for landmark detection

The proposed system in the offline path takes the dataset images and generates a vector of A-KAZE features that is filtered using a region of interest. This action compared to other solutions does not add a smaller weight to the features outside the region but ignores them all together. The assumption is that they are generated from distractors in the image. Afterwards, the features vectors are grouped together in a vector for each landmark which are clustered into BOF structure using the KNN clustering algorithm.

In the on-line phase we mirror the pre-processing part in order to filter out the features that are generated from distractors and using ANN classifier we find the closest similar landmark vector to the inquiry image.

To enhance the detection, we use GPS tags to limit our search range within the clusters. By doing so we gain benefits in the direction of detection accuracy and run-time.

In order to generate better features, in terms of localization and quantity, we enhance the images using dilated sharpening UM algorithm. The configuration of the

UM is carefully chosen so the low-quality images get enhanced but the good quality images do not get over-sharpen.

5.3.1. Image pre-processing

Pre-processing is a common naming used for operations done on the lowest level of abstraction upon an image. The aim of these operations is to improve the image data and to suppress unwilling distortions or enhance some image features important for further processing [219].

In our scope the aim of this action is to enhance the image in order to obtain better features extracted on one hand and to obtain a faster processing time for the whole chain on the other hand.

5.3.1.1. Grayscale transformation

In CV applications multiple colour spaces are used to describe a scene, either for human use or machine use. A colour space is a method in which we can specify, create and visualize colours [119].

Different colour spaces are suited for different applications or acquisition equipment. Most common colour spaces are: RGB (Red-Green-Blue), CMYK (Cyan-Magenta-Yellow, Black), HSV (Hue Saturation Lightness Value), HIS (Hue Saturation Intensity Value), YUV (luma component, blue projection, red projection), grayscale (amount of light).

In our system the raw image input is in RGB colour model format, which is a unification of 3 different colour matrixes, one for each colour channel. For a faster processing pipeline, we would like to process our images in grayscale format, which has only one-color channel.

The grayscale representation of an image is a matrix of pixels that are represented as 8-bit integers with range from 0-255, representing intensity changes.

There are multiple formulas to change the colour space between RGB and grayscale. In our algorithm we use Equation (5.1) to convert between the colour spaces, where Y is the gray level, R the red value, G the green value and B the blue value of a pixel.

$$\boldsymbol{Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B} \tag{5.1}$$

In Figure 5.7 the popular 'Lena' image is transformed from RGB to grayscale image as an example.

Figure 5.7 Image representation of RGB channels and grayscale

5.3.1.2. Pyramid level transformation

Pyramid representation is a type of multi-scale image representation specific to the CV domain. The image is subject to repeated smoothing and subsampling in order to obtain a scale-space representation. The name of the image pyramid is because if we arrange the images in decreasing order of their resolution, we would obtain a pyramid shape with a square base, see Figure 5.8 from [220].

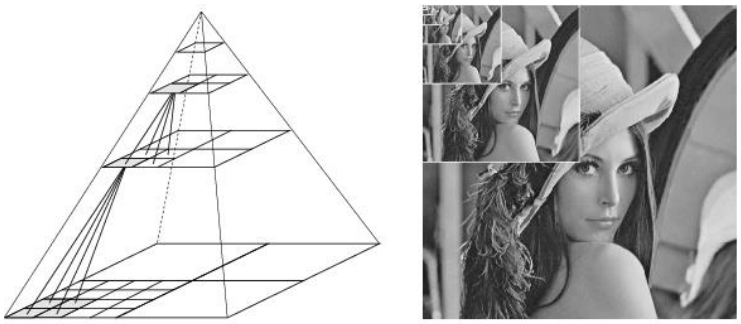


Figure 5.8 Example of image pyramid transformation [214]

Operations regarding scaling between levels of the image pyramids can be found in [221]. The main two operations are: REDUCE, see Equation (5.2), and EXPAND, see Equation (5.3). In the equations G_l represents the image at level l and $G_{l,k}$ represents the image obtained by expanding G_l k times. In case of REDUCE operation $w(m,n)$ is the weighting function that is used.

$$\boldsymbol{G_l(i,j) = \sum_m \sum_n w(m,n) \cdot G_{l-1}(i,j) \cdot (2i+m, 2j+n)} \tag{5.2}$$

$$\boldsymbol{G_{l,k}(i,j) = 4 \sum_m \sum_n G_{l,k-1}\left(\frac{2i+m}{2}, \frac{2j+n}{2}\right)} \tag{5.3}$$

The Laplacian pyramid, presented in Equation (5.4), is considered the resulting bypass, typically details like edges, from the REDUCTION process. This is obtained by the image transformation through scales of the pyramid, results that are usually lost in the process.

$$\boldsymbol{L_l = G_l - EXPAND(G_l)} \tag{5.4}$$

The RECONSTRUCT process of an image is presented in Equation (5.5). In order for this process we need the preserved details stored in the LAPLACE PYRAMID [221].

$$\boldsymbol{RECONSTRUCTED(G_l) = L_l + EXPAND(G_l)} \tag{5.5}$$

Even if the processing steps presented here are basic and fundamental, they represent important steps in each processing pipeline for CV.

5.3.1.3. Image sharpening

In order to enhance the image properties for better feature detection, I applied a sharpening transformation on the grayscale image. The scope of this activity is to enhance certain features like edges, object boundaries or textures as described in chapter 4.4.1.

The sharpening mechanism chosen is the UM filter using dilated filters presented in chapter 4.4.2. To be able to evaluate the benefits of this task I use the same dataset as in chapter 4.4.3, presented in Figure 4.15. The choice of evaluating a limited number of images and not the entire TMBuD dataset is a practical one. The dataset was created with the aim of containing all scenarios hence we can be assured that the evaluation is objective.

The aim of this evaluation is to choose which level of dilatation would be the optimal one for our proposed algorithm. For an overview of the effects of dilated sharpening we would plot the number of A-KAZE features, described in chapter 5.3.3.2, the entropy of the images and SF, described in chapter 4.4.3. This activity would be done on three different pyramid levels in order for us to observe if the trend of the results remain the same if we scale down the images.

From Figure 4.23 and Table 4.3 we concluded that using expanded $5x5$ Laplace kernels for UM sharpening produce false artifacts in the images. This is something that is not desired for our use case because it can cause creation of false features afterwards. In this direction I will not include the $5x5$ kernels in this analysis.

The evaluation will consist of using UM with V1 and V2 classical and dilated kernels variants, see Figure 4.10. We will evaluate kernels with dilation factor of 1 $(5x5)$ and 2 $(7x7)$. As presented in chapter 4 the hypothesis is that this will produce a sharper image and by doing so more features will be generated.

First, we can observe the visual results of this transformation in Figure 5.9 when using V1 kernel and in Figure 5.11 when using V2 kernel.

First thing we observe is that directly proportional with the dilation factor used, the number of features increases. This aspect, if taken into consideration alone, can be a positive thing. But one must be precautious about this fact as at a closer look we observe that the increased number is caused by the strong variations in intensity caused by the "burning" effect of over sharpening.

If we visually compare the results obtained using V2, see Figure 5.10, we can observe a huge improvement in the features number but a clear over sharpening transformation.

An important positive aspect that we observe by visual analysis is that the hypothesis that dilated kernels bring benefit results is sustainable.

In Figure 5.9 we are presented with the results for the entire dataset used. We can observe that in every case sharpening the image resulted in a bigger number of features detected. Moreover, when using dilated filters, the number increased even more.

Looking over the results presented we preliminary conclusion appears that using Laplace V1 kernel with a dilation factor of 2, so $7x7$, seems the correct choice in this use case. If we look upon the number of features produced as an effect of the transformation, the number increased but without causing a clear over sharpening effect. This aspect is sustained by the Entropy and SF values for each image presented in Figure 5.9. The values are typically higher than the original image but lower than the ones produced when using the V2 kernel.

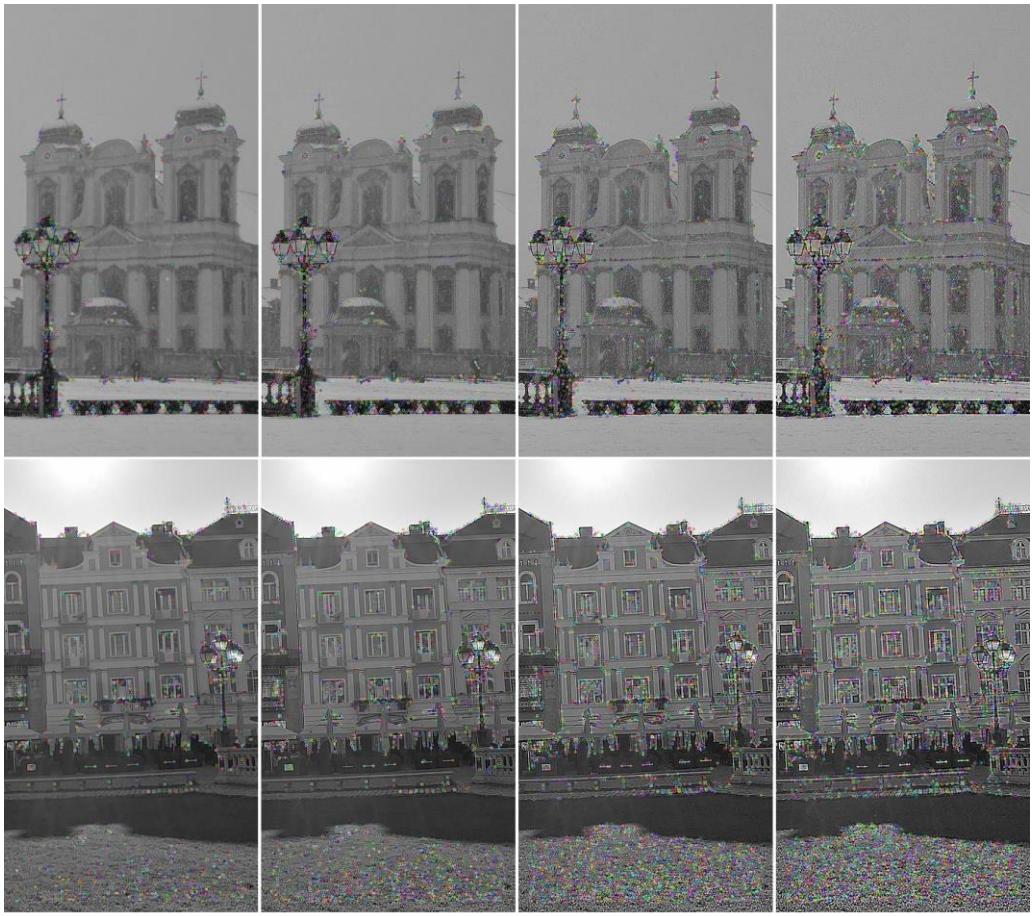


Figure 5.10 A-KAZE features detected on several images (00109, 07403) on pyramid level 0 enhanced using UM filters with kernel V1

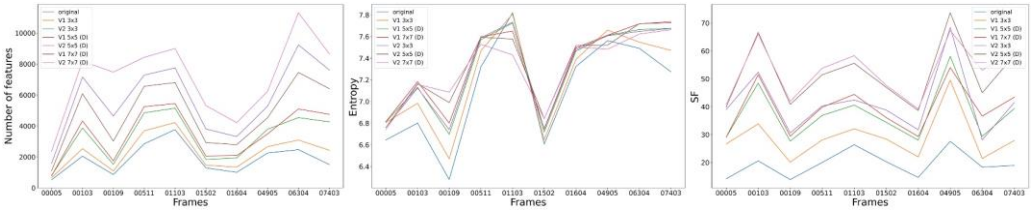


Figure 5.9 Number of features and image properties for pyramid level 0

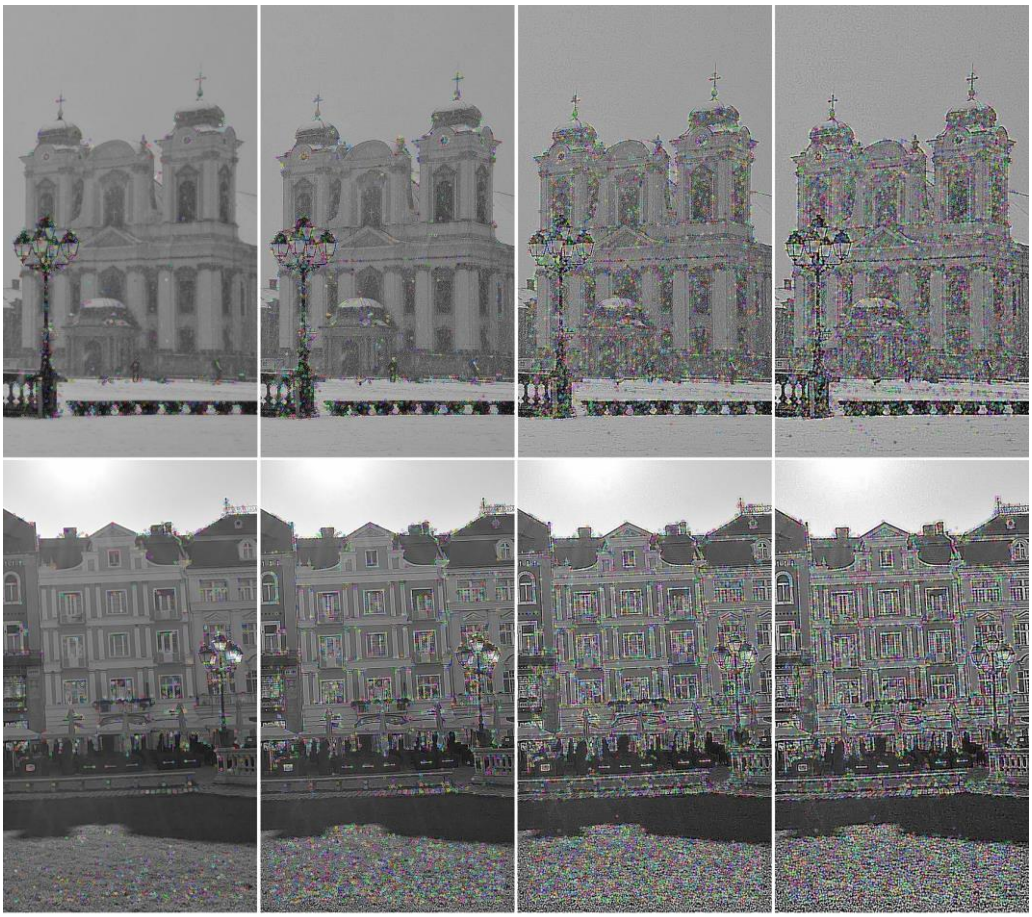


Figure 5.11 A-KAZE features detected on several images (00109, 07403) on pyramid level 0 enhanced using UM filters with kernel V2

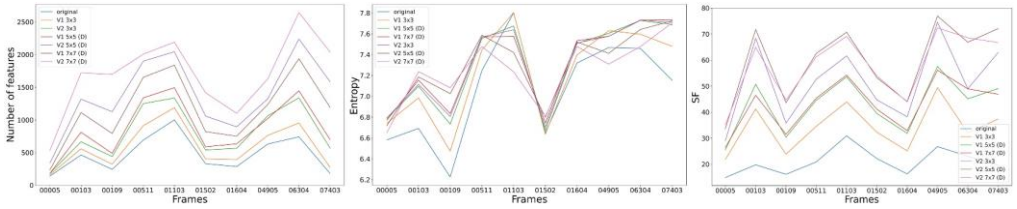


Figure 5.12 Number of features and image properties for pyramid level 1

Second part of our evaluation, I desire to understand how the sharpening effect transfers from one pyramid level to another in a number of features. To do so, I plotted the same information as for level 0 for level 1 and 2 in Figure 5.12, respective Figure 5.13.

Concerning the number of features obtained, the number is lower, as expected, from the resolution decrease caused by the lower scaling. But an interesting

aspect is observed if we concern ourselves by the image characteristics after sharpening, Entropy and SF. The values obtained remain in the same range even if the image is downscaled.

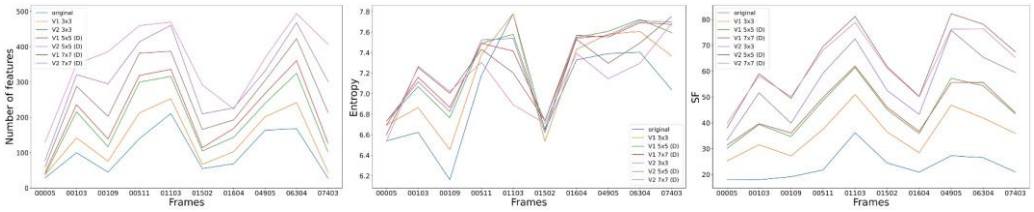


Figure 5.13 Number of features and image properties for pyramid level 2

From the statistical data presented, I can conclude that the benefits of sharpening the image using a dilated filter before the feature extraction step brings further benefits regardless of the pyramid level that is processed. To support my findings, I present an example of image 00005 processed in several pyramid levels, see Figure 5.14.

Figure 5.14 Features detected on image 00005 on different pyramid levels

From the visual and statistical data presented in this chapter the hypothesis that using dilated filter and UM sharpening produces an enhancement upon the feature detection is proven. In our proposed algorithm further, I will use the dilated $7x7$ V1 Laplace kernel for the UM algorithm. By using this I desire to increase the number of features we can process even for complicated use cases, like low visibility, low resolution images, glaring and so on.

5.3.2. Semantic Segmentation

Semantic segmentation is a technique used to cluster parts of the image together that belong to the same object or environment detail. This task is done on a pixel level and aims to group semantically the entire pixel map of an image.

Semantic Segmentation is a challenging task in the CV domain and because of its difficult nature it is still highly researched in the field. Multiple literature reviews like [222]–[226] aim to offer a better understanding of the wide methods and solutions for certain use cases.

Segmentation of the images can be achieved through "classical" methods like: thresholding, region based, edge based, watershed or clustering but in modern literature most of the solutions are solved using DL or CNN methods [117], [224].

Image segmentation tasks can be classified into three groups based on the amount of information they can offer. The classical semantic segmentation refers to the classification of each pixel into a class. The first extension is the instance segmentation, where the aim is to categories based on instance and not only as a category. The news approach is called panoptic segmentation where the aim is to offer the classification and instance in an image. An example is presented in Figure 5.15 [227].

Figure 5.15 Example of segmentation task [227]

For our system I investigated an DL based method for choosing our region of interest, so we will aim to achieve a classical semantic segmentation. In order to achieve this task, we can define several key phases, which will be detailed in this chapter. The minimum list of activities needed for this task are data preparation, data augmentation, choosing the proper model, training, and evaluating the trained model.

For our system I chose Residual Networks with 50 layers (ResNet-50) as base model [228] of our network with SegNet [229] as a segmentation model. In this chapter all steps are explained and detailed for each phase.

5.3.2.1. Preparing data

Data preparation is one of the most important steps when talking about AI CV methods, like in this case. The accuracy of the model is directly proportional on the quality of the training set of data.

There are three terms that will be helpful for this chapter to define in the beginning: training data, which refers to the data used for training and fitting the algorithm, validation data, data that is used to continuous validate the training process, and testing data, which represents a set of new images, but not radically different on which we evaluate the resulting model.

For our training, I will use datasets presented in chapter 2.3 and not a general semantic segmentation dataset like PASCAL [230], KITTY [231] or CityScape [232].

As stated before, the main concern of our application is street view perspective. This perspective should be strongly considered when training the DL model.

By analysing Table 2.2 and Figure 2.2, where we compared available semantic datasets, I can easily conclude that for our use case we would only consider the following dataset: eTRIMS, LabelMeFacade, TMBuD. Concerning ECP, ICG Graz5 and CPM from our perspective are too focused on the building façade and that will cause deformation of the understanding inside the model. Another interesting view is the one from VarCity where only the main building is labelled, and the rest is marked as unknown. But, by using the dataset, we will create confusion about the other building from the scene when training (3 dataset label them and one not).

As presented in chapter 5.2, TMBuD was extended to contain more labelled images and to be similar with scenarios that can occur in real-life use cases in the future. These aspects have determined us to split part of this dataset for the test dataset. So, in the SEMSEG_EVAL_FULL variant of TMBuD, the existing data will be divided into 250 images for training and 50 images for testing.

In order to be better tuned, the semantic segmentation for the application on hand I opted to correlate the labels to obtain a sort of binarization: building and background (that I call noise). In Table 5.2 the correlation between original labels and new labels is presented. As we can observe, we grouped under the BUILDING label all the elements that compose a building and marked all the rest as NOISE. We reserved the third class, UNKNOWN, for pixels that the model cannot decide on.

Table 5.2 Class corelation of used dataset for training

Dataset	UNKNOWN	BUILDING	NOISE
eTrims	-	BUILDING, DOOR, WINDOW	VARIOUS, CAR, PAVEMENT, ROAD, SKY, VEGETATION
LabelMeFacade		BUILDING, DOOR, WINDOW	VARIOUS, CAR, PAVEMENT, ROAD, SKY, VEGETATION
TMBuD	UNKNOWN	BUILDING, DOOR, WINDOW	SKY, VEGETATION, GROUND, NOISE

In Figure 5.16 we are presented with visual results of new label ground truth for the datasets.

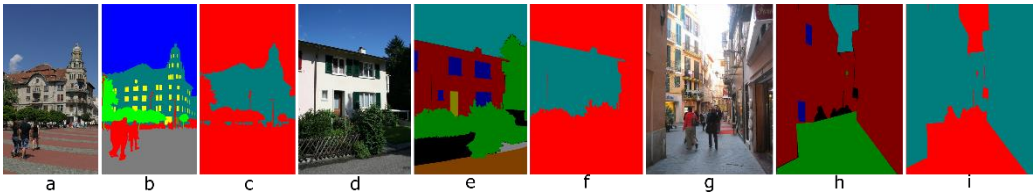


Figure 5.16 Example of dataset label correlation.
a) TMBuD original image; b) TMBuD original label; c) TMBuD new label; d) eTrims original image; e) eTrims original label; f) eTrims new label; g) LabelMeFacade original image; h) LabelMeFacade original label; i) LabelMeFacade new label.

With the rest of the images, I divided them equally between training and validation dataset. The validation dataset will result in 622 images, composed from 30 images out of eTrims, 468 out of LabelMe and 124 out of TMBuD.

To enhance the training results of the model, typically, on training dataset a process of data augmentation is done. Data augmentation is a process in which we

create copies of the images slightly transformed by colour or position modification. In our case, 26 transformations were done for each image consisting of zooming, sharpening, blurring, flipping, pixelating, rotating, or changing colour intensities. In Figure 5.17 some resulting transformations are presented. The effect of all this transformation is to expose the network when training with multiple different scenarios for an image.

Figure 5.17 Example of augmentation done for image 00703 from the training dataset

In the end we will have 3 datasets as follows: training dataset composed of 18039 images, validation dataset composed of 622 and testing dataset composed of 50 images. Even if that distribution does not correspond with the classical distribution, it is sufficient for our training.

Another transformation that was applied across all the images is a resize to 512 pixels height by 320 pixels width. This size transformation was done in order that all the images used can be broken apart in 32 by 32 pixels blocks to fit the requirements of training the network.

5.3.2.2. ResNet50 SegNet

Choosing the right model to use for an DL task is not a trivial action. One needs to take into consideration several correlations between the desired task and the model to use. The models are different according to the data format one has, desired activity, resources at hand and so on.

From the beginning it would be good that the following operations, commonly used in the networks, will be defined:

- **Convolution** – is the mathematical operation that combines signal a and b or filtering input a with kernel b. It is a process to overlay 'b' on 'a', multiply the numbers and sum the products and move.
- **Max pooling** – is the down sample of an input by locally summarising the data where the local maxima of the filtered region are carried forward.
- **Average pooling** – is the down sample of an input by locally summarising the data where the average of the filtered region is carried forward.

- **Activation function** – defines the output for a given input. Combining linear functions yields a linear function; however, in order to compute more in-depth features, nonlinearity is required [233].
- **ReLu** – is an activation linear function that outputs the input directly if is positive i.e. > 0, otherwise, it will output zero [233].
- **Softmax classifier** – is an activation function which imparts probabilities of each input belonging to each output when there are more than two outputs.
- **Batch Normalisation** – is a layer that normalises the hidden nodes for future use.
- **Downsampling** – is the operation that reduces the size of the image, example of this is Max pooling.
- **Upsampling** – is the operation opposite of down sampling where the resulting image is expanded. The most common way of unpooling is using the Nearest Neighbour.

ResNet or Residual Network is one of the popular used neural networks for semantic segmentation and it was introduced in 2016 by [228]. Originally, the network was hard to train because of the large number of layers and depth but that was solved by introducing shortcut connection between the layers. ResNet50 is the deviation of the original model that contains only 50 layers [224], [228].

In order to understand the inner workings of the ResNet50 model we have found a great representation in [234] and it is presented in Figure 5.18. In the network, two types of shortcuts are used. The first one is the identity block which eliminates the convolution layers at shortcut. The second one is the convolution block as a convolution layer at shortcut. In the second case, the input dimensions are smaller than the output dimensions.

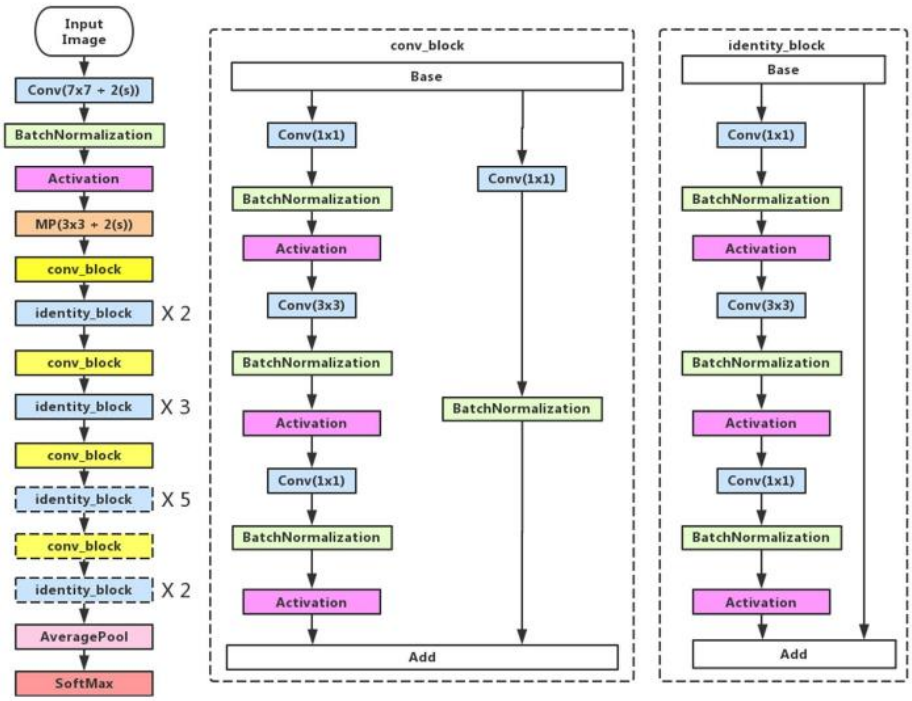


Figure 5.18 ResNet50 architecture details [224]

SegNet model appeared in 2017 presented by [229] and it is one of the most popular models for pixel-wise semantic segmentation. The trainable segmentation model consists of an encoder with a corresponding decoder followed by a pixel-wise classification. In case of SegNet the encoder architecture is topologically similar to the 13 convolutional layers of VGG16 network [111].

The role of the decoder network is to map the low-resolution feature maps to full input resolution feature maps for each pixel. The novelty of SegNEt is the way the decoder samples the lower resolution feature maps [229]. The structure of encoder-decoder network is presented in Figure 5.19.

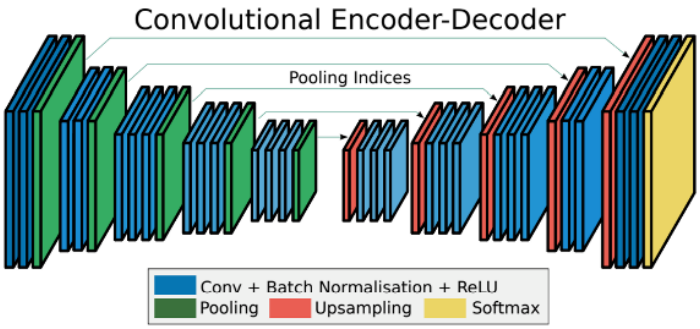


Figure 5.19 Illustration of the SegNet architecture [218]

One of the most important features of SegNet is that information is transferred directly instead of convolving it. This new approach renders this encoder-decoder network as one of the best models to use for pixel wise semantic segmentation [224].

For our use case I consider that using the ResNet50 SegNet combination, SegNet decoder and ResNet50 encoder, is the best fit after reviewing the literature and other applications.

5.3.2.3. Training results

Training a network typically consists of two phases: forward phase and backward phase. In the forward phase the inputs are passed through the network and in the following phase, backwards, the gradients are back propagated and weights are updated. The training process was done using the EECVF, described in chapter 3, using Tenserflow [129] and Keras.

For the training I used the following parameters: 300 epochs to train, a batch size of 6, train images per epoch of 10 for each batch, validation images per epoch of 3, input resolution of 320x512 pixels, optimizer to use Adam.

I calculated the loss between the labels given and the training labels for each epoch using the cross-entropy loss [235] described in Equation (5.6). In Equation (5.6) N is the total number of pixels, n is the total number of classes, x is the training input label and $\hat{x}$ is the output predicted label. The cross-entropy loss is commonly used for classification applications.

$$\boldsymbol{Loss} = -\frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{N}}\sum_{\boldsymbol{j}=\mathbf{1}}^{\boldsymbol{n}}\left(\widehat{\boldsymbol{x}_{\boldsymbol{i}}}\log \boldsymbol{x}_{\boldsymbol{i},\boldsymbol{j}}\right) + \left(\mathbf{1}-\widehat{\boldsymbol{x}_{\boldsymbol{i},\boldsymbol{j}}}\right)\log\left(\mathbf{1}-\boldsymbol{x}_{\boldsymbol{i},\boldsymbol{j}}\right) \tag{5.6}$$

Another metric that I used for each epoch is the accuracy, which is the ratio between the amount of correctly classified pixels and the total number of them. The formula for this metric can be found in Equation (5.7), N_{cls} be the number of classes, N_{xy} is the number of pixels which belong to class x and were labelled as class y.

$$\boldsymbol{Accuracy} = \frac{\sum_{x=1}^{N_{cls}} \boldsymbol{N_{xx}} + \sum_{x=1}^{N_{cls}} \boldsymbol{N_{yy}}}{\sum_{x=1}^{N_{cls}} \boldsymbol{N_{xx}} + \sum_{x=1}^{N_{cls}} \boldsymbol{N_{yy}} + \sum_{x=1}^{N_{cls}} \boldsymbol{N_{xy}} + \sum_{x=1}^{N_{cls}} \boldsymbol{N_{yx}}} \tag{5.7}$$

In Figure 5.20 I present the plotted values for Accuracy and Loss for the training process. The two graphs show the sanity and correctness of the training process. The first graph tracks the accuracy, and it offers a valuable insight on the amount of overfitting that is happening in the model. If the gap between the training and validation graph is high then the model is strongly overfitted. The second graph tracks the training loss, as evaluation on the individual batches is done during the forward pass. The loss can become more linear with a low learning rate. With high learning rates they will start to look more exponential.

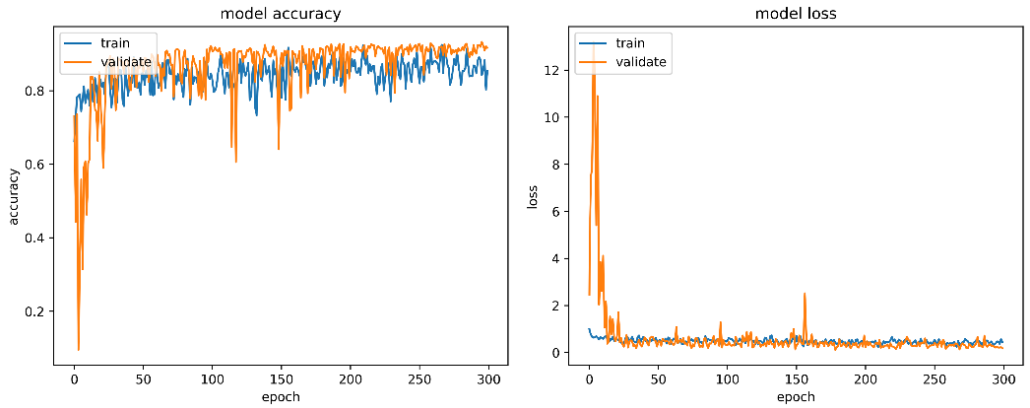


Figure 5.20 ResNet50-SegNet training model metrics

In our case I can observe that the model is slightly overfitted, but not to a degree that should cause problems. When talking about the learning rate we can observe a relatively good plot appearing. In both cases, the amount of "wiggle" is dependent on the batch size, which is dependent on the amount of data we have at hand.

For evaluating the prediction of the model, I used the test dataset, which contains images that the network has never seen before. The metric I used is the Mean Intersection over Union (MIoU) that is presented in Equation (5.8). MIoU can be defined as the average of the ratio between the numbers of true positives N_{xx} or intersection, over the sum of true positives N_{xx}, false negatives N_{yx}, false positives N_{xy} or union, for each class.

$$\boldsymbol{MIoU} = \frac{\boldsymbol{1}}{\boldsymbol{N_{cls}}} \sum_{x=1}^{N_{cls}} \frac{\boldsymbol{N_{xx}}}{\sum_{y=1}^{N_{cls}} \boldsymbol{N_{xy}} + \sum_{y=1}^{N_{cls}} \boldsymbol{N_{yx}} - \boldsymbol{N_{xx}}} \tag{5.8}$$

As in the case of any other network training there is a point in the training of the model when we need to stop. The effect of not doing that is the over-fitting of the model. Overfitting is the concept that the model is statistically fitted exactly against the given dataset resulting in a very bad performance on unseen data.

To try to avoid this effect as much as possible, I conducted the approach of considering the epochs as a parameter to tune and trained several models with the scope of finding the best fit.

In Table 5.3 the MIoU metric results are presented for different configurations of the model according to the number of epochs it was trained.

Table 5.3 MIoU results for different training configurations of ResNet50-SegNet

Number of epochs	UNKNOWN	BUILDING	NOISE	AVERAGE
50	0.000%	0.855%	0.888%	0.872%
100	0.009%	0.854%	0.905%	0.879%
150	0.000%	0.825%	0.852%	0.839%
200	0.155%	0.880%	0.649%	0.764%
250	0.004%	0.862%	0.894%	0.878%
275	0.000%	0.863%	0.893%	0.878%
300	0.000%	0.881%	0.917%	0.899%

In Figure 5.21 prediction results of the model are presented. For a better understanding of the results, I opted to present an overlay view of the original image and semantic segmented labels predicted.

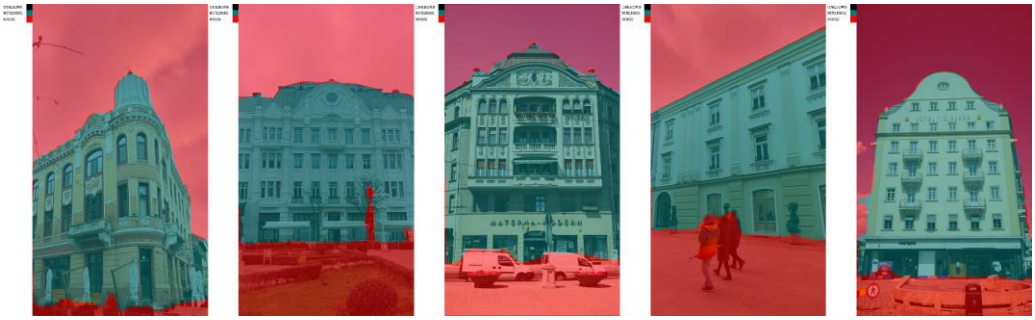


Figure 5.21 Example of predicted images with the trained model. Overlay colours: red is noise class, green is building, black is unknown.

In the end, it is important to state that from the obtained results, visual and statically, it seems that the model manages to predict good results in terms of differentiating our useful region of interest in the image.

5.3.3. Features detector

A feature detector is defined as an algorithm that highlights feature-points or also called interest-points or key-points in an image. Features in CV are detected usually as corners, blobs, edges, joints, lines, or other clustering structures. The main goal of a feature detector is to describe different ways of unique patterns in a pixel neighbourhood. This process is identified as feature description as for each feature a distinctive identity is associated. The identity will enable the effective recognition for later matching [236]–[238].

Feature detectors desire to have high repeatability and unique distinctiveness against several image transformations like viewpoint, blurring, noise, rotation. Most popular feature detectors are SIFT (blob based) [76], SURF (blob based) [80], KAZE (blob based) [239], A-KAZE (blob based) [109], ORB (corner based) [240], and BRISK (corner based) [241].

A-KAZE features was published in 2012 and it exploits non-linear scale space through non-linear diffusion filtering. The name comes from the Japanese word kaze which translates in wind and refers to the flow of air ruled by nonlinear processes on a large scale [239].

Accelerated-KAZE (A-KAZE) feature detector appeared in 2013 which is also based on nonlinear diffusion filtering like KAZE but its non-linear scale spaces are constructed using a computationally efficient framework called Fast Explicit Diffusion (FED) [109].

5.3.3.1. KAZE features

Before presenting KAZE or A-KAZE features we need to present the nonlinear partial differential equations (PDEs). "Nonlinear diffusion describes the evolution of the luminance of an image through increasing scale levels as the divergence of a certain flow function that controls the diffusion process" [239].

In Equation (5.9) the classical nonlinear diffusion formula is presented, where div represents the divergence operator and ∇ represents the gradient operator. The adaptive diffusion to a local image structure is possible because of the conductivity function c. The function c depends on the local image differential structure, and this function can be either a scalar or a tensor. The time t is the scale parameter [239].

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{t}} = \boldsymbol{div}(\boldsymbol{c}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{t}) \cdot \boldsymbol{\nabla L}) \tag{5.9}$$

The function c can depend on the gradient magnitude to reduce the diffusion of edges, and by doing so to encourage smoothing within a region instead of smoothing across boundaries. The conductivity function can be represented as in Equation (5.10), where luminance function ∇L_σ is the gradient of a Gaussian smoothed version of the original image L. Perona and Malik described two different formulations for the conductivity function g as in Equations (5.11) and (5.12), where k is the contrast factor that controls the level of diffusion [242].

$$\boldsymbol{c}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{t}) = \boldsymbol{g}(|\boldsymbol{\nabla L_\sigma}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{t})|) \tag{5.10}$$

$$\boldsymbol{g_1} = \boldsymbol{exp}\left(-\frac{|\boldsymbol{\nabla L_\sigma}|^2}{\boldsymbol{k^2}}\right) \tag{5.11}$$

$$g_2 = \frac{1}{1 + \frac{|\nabla L_\sigma|^2}{k^2}} \tag{5.12}$$

But in [243] a different diffusion function for rapidly decreasing diffusivities, where smoothing on both sides of an edge is much stronger than smoothing across it, is presented in Equation (5.13). In [244] we can find a different variant of the diffusion function that is presented in Equation (5.14).

$$g_3 = \begin{cases} 1, & |\nabla L_\sigma|^2 = 0 \\ 1 - \exp\left(-\frac{3.315}{(|\nabla L_\sigma|/k)^8}\right), & |\nabla L_\sigma|^2 > 0 \end{cases} \tag{5.13}$$

$$g_c = \frac{1}{\sqrt{1 + \frac{|\nabla L_\sigma|^2}{k^2}}} \tag{5.14}$$

As there is no analytical solution to PDEs, the Additive Operator Splitting (AOS) technique is used. One possible discretization of the diffusion equation is the so-called linear implicit or semi-implicit scheme. The vector-matrix solution is presented in Equation (5.15), where A_l is a matrix that encodes the image conductivities for each dimension. In the scheme it is necessary to solve a linear system of equations, where the system matrix is tridiagonal and diagonally dominant. Such systems can be solved by means of the Thomas algorithm in an efficient manner [239], [243].

$$L^{i+1} = \left(I - \tau \sum_{l=1}^{m} A_l(L^i)\right)^{-1} \cdot L^i \tag{5.15}$$

KAZE use similar approach to SIFT by discretizing the scale space in logarithmic steps arranged in a series of O octaves and S sub-levels. This is represented in Equation (5.16) where σ_0 is the base scale level and N is the total number of filtered images. Because we work in Gaussian scale space, convolution of an image with a Gaussian of standard deviation σ (in pixels) is equivalent to filtering the image for some time $t = \frac{\sigma^2}{2}$. Now by transforming the scale space $\sigma_i(o, s)$ to time units by means of the following mapping $\sigma_i \to t_i$ we obtain Equation (5.17).

$$\sigma_i(o, s) = \sigma_0 2^{o+s/S}, o \in [0 \dots O-1], s \in [0 \dots S-1], i \in [0 \dots N] \tag{5.16}$$

$$t_i = \frac{1}{2}\sigma_i^2, i = \{0 \dots N\} \tag{5.17}$$

Considering Equation (5.15) the AOS scheme becomes the one in Equation (5.18), which is stable for any step of the feature.

$$L^{i+1} = \left(I - (t_{i+1} - t_i) \cdot \sum_{l=1}^{m} A_l(L^i)\right)^{-1} \cdot L^i \tag{5.18}$$

For detecting points of interest, the response of scale-normalized determinant of the Hessian at multiple scale levels is computed. See Equation (5.19) where (L_{xx}, L_{yy}) are the second order horizontal and vertical derivatives respectively, and L_{xy} is the second order cross derivative [239], [245].

$$\boldsymbol{L_{Hessian} = \sigma^2\left(L_{xx}L_{yy} - L_{xy}^2\right)} \tag{5.19}$$

In order to speed-up the search for extrema, the responses over a window of size 3×3 pixels are checked, in order to quickly discard non-maxima responses. In the end the position of the keypoint is estimated with sub-pixel accuracy using the method proposed in [246].

"Similar to SURF, KAZE finds the dominant orientation in a circular area of radius $6\sigma_i$. For each sample in the circular area, first order derivatives L_x and L_y are weighted with a Gaussian centered at the interest point. Then, the derivative responses are represented as points in vector space and the dominant orientation is found by summing the responses within a sliding circle segment covering an angle of $\pi/3$. From the longest vector the dominant orientation is obtained" [239].

"For building the descriptor M-SURF [247] is adapted to the non-linear scale. For a detected feature at scale σ_i, first order derivatives L_x and L_y of size σ_i are computed over a $24\sigma_i x 24\sigma_i$ rectangular grid. This grid is divided into 4×4 subregions of size $9\sigma_i x 9\sigma_i$with an overlap of $2\sigma_i$. The derivative responses in each subregion are weighted with a Gaussian centered on the subregion center and summed into a descriptor vector $d_v = \left(\sum L_x, \sum L_y, \sum|L_x|, \sum|L_y|\right)$. When considering the dominant orientation of the keypoint, each of the samples in the rectangular grid is rotated according to the dominant orientation" [239].

In Figure 5.22 we exemplify the differences in features detected if we use different diffusion functions.

Figure 5.22 Example of KAZE features on an image from TMBuD using same parameters but with different diffusion functions. From left to right the diffusion functions used are described by the Equation (5.11), (5.12), (5.13) and (5.14).

5.3.3.2. A-KAZE features

A-KAZE features aim to speed up the processes by using FED [248], [249] numerical schemes embedded in a pyramidal framework. Additionally, they changed the descriptor to a Modified-Local Difference Binary (M-LDB) [250] which is more efficient regarding rotation invariants and storage [109].

Building the nonlinear space is changed by using FED which is more accurate and easier to implement than AOS. The main idea is to perform M cycles of n explicit diffusion steps with varying step sizes τ_j that originate from the factorization of a box filter, as we can see in Equation (5.20) [109].

$$\boldsymbol{\tau_j = \frac{\tau_{max}}{2\cos^2\left(\pi\frac{2j+1}{4n+2}\right)}} \tag{5.20}$$

The determinant of the Hessian for each of the filtered images in the nonlinear scale space is changed to the one presented in Equation (5.21).

$$\boldsymbol{L_{Hessian} = \sigma^2\left(L_{xx}L_{yy} - L_{xy}^2\right)} \tag{5.21}$$

"The last modification is using M-LDB as feature descriptor which follows the same principal as BRIEF [251] but using binary tests between the average of areas instead of single pixels for additional robustness. In addition to the intensity values, the mean of the horizontal and vertical derivatives in the areas being compared is used, resulting in 3 bits per comparison. Rotation invariance is obtained by estimating the main orientation of the key point as in KAZE, and the grid of LDB rotated accordingly. Instead of using the average of all pixels inside each subdivision of the grid, we subsample the grids in steps that are a function of the scale σ of the feature" [109].

In Figure 5.23 I exemplify the differences in features detected if we use different diffusion functions.

Figure 5.23 Example of A-KAZE features on an image from TMBuD using same parameters but with different diffusion functions. From left to right the diffusion functions used are described by the Equation (5.11), (5.12), (5.13) and (5.14).

From the evaluation done in [236], [237], [252]–[257] we can observe that A-KAZE features spends less time to detect feature points but on the down side the number of feature points is small and the repetition rate is low. One important positive aspect is that A-KAZE features have proven to be more rotation invariant than other compared feature detectors.

Even if KAZE or SIFT features seem to be more robust when considering the scale and affine variants the fact that A-KAZE is more invariant to rotation, which in our use case is important, and the fact that from the speed perspective it scored better for detecting feature time, matching feature time and correspondence time.

5.3.4. Bag of features

Bag of Features (BoF), also named as Bag of Visual Words (BoVW) or Bag of Keypoints, is a method of representing and clustering features from an image that appeared in [72].

BoF is an extension of the popular Bag of Words (BoW) that is used for text categorization [258], [259]. For CV BoF is used in several contexts but one of the most common is for CBIR systems [108].

Typically for constructing a BoF several standard steps are involved: (1) detection key-points and description of images, (2) Assigning patch descriptors to a set of predetermined clusters using a vector quantization algorithm, (3) constructing a bag of features, which counts the number of patches assigned, (4) applying a classifier by treating the bag of features as the features vector [72], [260], [261]. In Figure 5.24 we present a good representation of a generic BoF construction scheme found in [262].

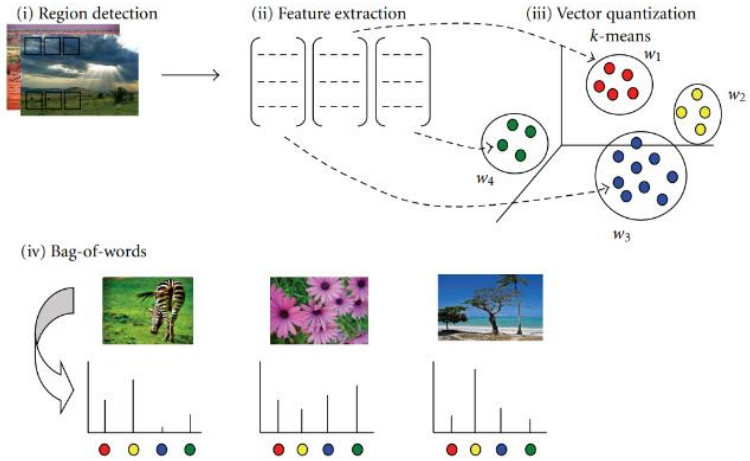


Figure 5.24 Four steps for constructing a BoF [255]

Points or regions of interest are detected in the images and a visual codebook is constructed using a technique called quantization. Quantization refers to grouping in a vector similar feature together. Each of the groups is represented by centre clusters called visual word or codeword. Forward on each new keypoint is added to the closest codeword in the codebook, or new ones are created [260].

Algorithm Building of BoF algorithm

```
/* Extract interest points or regions from the images                    */
for each image do
    keypoints, descriptors = create_features(image)
    Add keypoints, descriptors to correct group
end
/* Using K-Means clustering algorithm create codebook                    */
codebook = quantizePts(groups)
/* BoF feature vector generation                                         */
for each group do
    BoF = computeHistogram(groups, codebook)
end
```

Figure 5.25 Pseudo-code for BOF steps

The steps for BoF creation are described as pseudocode in Figure 5.25. The first step is creating features from images and by doing that we create the called training dataset. In literature we can find multiple examples of feature detectors used for this step like SIFT, SURF, A-KAZE and so on. In our case the feature detector is detailed in chapter 5.3.3.2.

The second step is important for the future clustering and classification using BoF. This step consists of adding to the same vector or group all the descriptors resulting from images that convey the same object or class.

In the third step the clusters are created inside the BoF where all the features descriptor are clustered in k clusters using the K-Means clustering algorithm [81], [263]. As stated, this step is also known as the vector quantization step. In this step we will create k clusters with an associated centroid c. These centroids represent the main features that are present in the whole training dataset.

The K-means clustering algorithm computes centroids and repeats the steps until the optimal centroid is found. This approach is also known as the flat clustering algorithm, and it is described in Figure 5.26.

Algorithm K-Means algorithm

```
/* As input the algorithm expects a dataset of points P = p_1, ..., p_n and number of cluster k   */
Choose k initial centers C = c_1, ..., c_k
while any C_i changes do
    Do assignment step:
    for i = 1, ...N do
        Find closest center c_k ∈ C to instance p_i
        Assign instance p_i to set C_k
    end
    Do update step:
    for i = 1, ...k do
        Set c_i to be the center of mass of all points in C_i
    end
end
```

Figure 5.26 Pseudo-code for K-Means algorithm

There are some difficulties when using the K-means algorithm like choosing the correct value for K and the computational cost of clustering when the dataset is large [260]. Being one of the simplest unsupervised clustering algorithms it is widely used in application and extensions of the algorithm are present in literature that aim to mitigate the shortcomings [58], [72].

The BoF representation is notable because of its relatively simple and strong performance in several vision tasks. Even if the BoF approach may have degraded results than other techniques for object detection and localization research is still very much an active field. Like in other sub-domains of CV it is a continuous trade-off between performances and scalability of proposed solution [264].

5.3.5. Feature matching

One of the most important and complex problems in the CV domain remains the matching of features across an image or multiple. Feature matching consists of the attempt of finding corresponding features from two different dataset based on a search distant algorithm.

In order to solve this problem, which my system also encountered, to match new features with the vocabulary clusters and to find the correct association, the nearest neighbour search algorithm (NN) is chosen. This problem of NN is one of importance in several other applications like image recognition, data compression, pattern recognition and classification.

The nearest neighbour search problem can be resumed as follows: given a set of points $P = \{p_1, \dots, p_n\}$ in a vector space X, these points must be processed in a way that when given a new query point $q \in X$ it will be able to find the points in P that are nearest to q efficiently [110].

Solving this problem in the high dimensional space with better resource consumptions and results than the brute force search is hard to achieve. In literature, solutions that have proven to be good enough are presented like: Approximate Nearest Neighbour kd-tree (ANN-KDT) [265], [266], Approximate Nearest Neighbour k-means (ANN-KM) [267], Approximate Nearest Neighbour Hierarchical k-means(ANN-HKM) [81] or Approximate Nearest Neighbour Local Sensitive Hashing (ANN-LSH) [268].

For the proposed system we chose to use ANN-KDT with the implementation offered by the Fast Library for Approximate Nearest Neighbours (FLANN) [110] that offers an image matching algorithm for a fast ANN search in high dimensional spaces.

By quantitative studies done in [77], [269] resulted that simple linear search or brute force outperform space partitioning for high dimensionality (>10 dimensions) at the a high cost of run-time resources.

ANN-KDT is a multi-dimensional search algorithm to find a data point in a k-dimensional set of data, inspired by the binary search algorithm. From a complexity point of view ANN-KDT achieves $O(\log n)$ search time for sorted data, and $O(\mathrm{nlog}\, n)$ for unsorted data.

The idea of the ANN-KDT algorithm is to partition the space using hyperplanes orthogonal to the coordinate axes. Each leaf node contains a cluster with a number of vectors, the other nodes in the binary kd-tree consist of a splitting dimension d and a splitting value v. When a query happens, it has to look at one dimension at each node to decide into which subtree to descend. Then the process is repeated for the vectors

x in the bucket for the closest one [77]. A useful representation of the search algorithm is showed in Figure 5.27 [7].

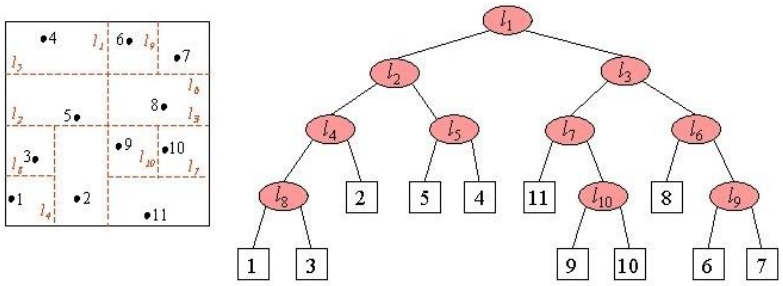


Figure 5.27 Example of representation for ANN-KDT search algorithm[7]

Even if the dimensions of our current dataset would not cause problems in term of resources to use the brute for variant, we have chosen to incorporate from the begging a more advance search algorithm.

5.3.6. GPS filtering

Geo-tagging is fast emerging in the domain of digital photography. The presence of geographical relevant information as metadata of images and videos has opened new possibilities of processing in the CV domain. Geotagging or geo-referencing is the action of adding geographical identification metadata to various media [270], [271].

In our case both reference images or inquiry images have geo-tagged metadata information with the scope of enabling us to filter out or refine our search. As we can observe in Figure 5.28, where we highlighted available images and landmarks, this information can be used to focus on images from a certain radius or even filter out from the search images that clearly are not in the scope. In the case presented, if we would search for images of landmark *001,* we can directly eliminate images farther than 150m/200m, as they cannot contain the landmark.

The shortest path between two points on the earth, if we consider earth as being an ellipsoid of revolution, is called a geodesic. We would consider the popular Vicenty's modified indirect problem to calculate the distance between two points on the earth [272].

In order to present the used formulas of the inverse problem, as it is presented in [272], we need to define the following: a as the length of semi-major axis of the ellipsoid; f as the flattening of the ellipsoid, $b = (1-f)a$ as the length of semi-minor axis of the ellipsoid; ϕ as latitude of a point; $U = \left((1-f)\ tan\ tan\ \phi\right)$ as the reduce latitude; L as longitude; λ as the difference in longitude of the points on the auxiliary sphere; α as azimuth of a point; s as ellipsoidal distance between the two points; σ as angular separation between points.

[7] https://slidetodoc.com/nearest-neighbor-search-problem-whats-the-nearestaurant-to/

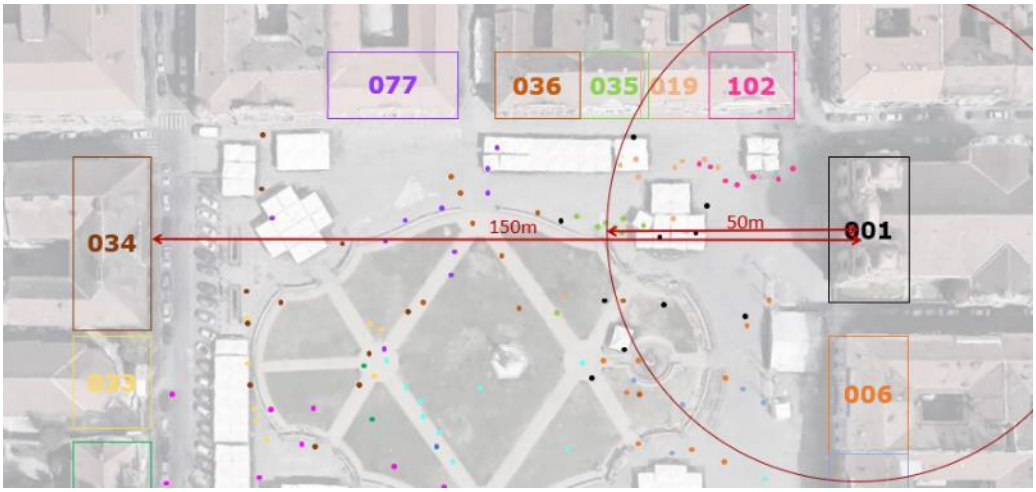


Figure 5.28 Geo-tag photograph filtering

Given two points on the map (ϕ_1, L_1) and (ϕ_2, L_2) the inverse problem consists in finding the azimuths α_1 and α_2 together with the ellipsoidal distance s. To do that we need to iteratively evaluate equation (5.22) till (5.28) until λ converges.

$$\sin\sigma = \sqrt{(\cos U_2 \sin\lambda)^2 + (\cos U_1 \sin U_2 - \sin U_1 \cos U_2 \cos\lambda)^2} \tag{5.22}$$

$$\cos\sigma = \sin U_1 \sin U_2 + \cos U_1 \cos U_2 \cos\lambda \tag{5.23}$$

$$\sigma = \tan^{-1} 2(\sin\sigma, \cos\sigma) \tag{5.24}$$

$$\sin\alpha = \frac{\cos U_1 \cos U_2 \sin\lambda}{\sin\sigma} \tag{5.25}$$

$$\cos(2\sigma_m) = \cos\sigma - \frac{2\sin U_1 \sin U_2}{\cos^2\alpha} = \cos\sigma - \frac{2\sin U_1 \sin U_2}{1 - sin^2\alpha} \tag{5.26}$$

$$C = \frac{f}{16}\cos^2\alpha\left[4 + f(4 - 3\cos^2\alpha)\right] \tag{5.27}$$

$$\lambda = L + (1 - C)f \sin\alpha\{\sigma + C\sin\sigma[\cos(2\sigma_m) + C\cos\sigma(-1 + 2\cos^2(2\sigma_m))]\} \tag{5.28}$$

When λ has converged to the degree of accuracy desired we can evaluate Equations (5.29) till (5.35).

$$u^2 = \cos^2\alpha\left(\frac{a^2 - b^2}{b^2}\right) \tag{5.29}$$

$$A = 1 + \frac{u^2}{1024}(4096 + u^2[-768 + u^2(320 - 175u^2)]) \tag{5.30}$$

$$B = \frac{u^2}{1024}(256 + u^2[-128 + u^2(74 - 47u^2)]) \tag{5.31}$$

$$\Delta\sigma = B\sin\sigma\left\{\cos(2\sigma_m) + \frac{1}{4}B\left(\cos\sigma[-1+2\cos^2(\sigma_m)] - \frac{B}{6}\cos[2\sigma_m][-3+4\sin^2\sigma][-3+4\cos^2(\sigma_m)]\right)\right\} \quad (5.32)$$

$$s = bA(\sigma - \Delta\sigma) \quad (5.33)$$

$$\alpha_1 = \tan^{-1}2(\cos U_2 \sin\lambda, \cos U_1 \sin U_2 - \sin U_1 \cos U_2 \cos\lambda) \quad (5.34)$$

$$\alpha_2 = \tan^{-1}2(\cos U_1 \sin\lambda, -\sin U_1 \cos U_2 + \cos U_1 \sin U_2 \cos\lambda) \quad (5.35)$$

A modification occurred later in this process of calculation when Vincety proposed new formulas for A and B, that are presented in Equation (5.36) and (5.37).

$$A = \frac{1 + \frac{1}{4}\left(\frac{\sqrt{1+u^2}-1}{\sqrt{1+u^2}+1}\right)^2}{1 - \frac{\sqrt{1+u^2}-1}{\sqrt{1+u^2}+1}} \quad (5.36)$$

$$B = \frac{\sqrt{1+u^2}-1}{\sqrt{1+u^2}+1}\left(1 - \frac{3}{8}\left(\frac{\sqrt{1+u^2}-1}{\sqrt{1+u^2}+1}\right)^2\right) \quad (5.37)$$

5.4. Benchmark and evaluation

In this chapter, the benchmarking of the proposed algorithm will be presented. First I will benchmark the system on the ZuBuD dataset, a popular dataset that focuses on street view images. Afterwards we will benchmark on the proposed TMBuD dataset. In the end the proposed system will be evaluated on mobile recorded data in order to show a real-life scenario behaviour.

For this chapter, I will use the following notations: A-KAZE descriptor size (D_S), A-KAZE number of octaves (D_NO), A-KAZE number of layers (D_NL), A-KAZE threshold (D_THR), smoothing dilated filter size (UM_D), UM strength of smoothing (UM_S), BOF cluster size (B_S), threshold for distance for feature matching (THR), distance in m between GPS tags (D_GPS).

As stated before, all the simulations done in this section can be reproduced using the EECVF, presented in chapter 3. The experiments were done using a desktop device AMD Ryzen 5 3600 6-Core Processor 3.95 Ghz, 16GB RAM, GeForce RTX2060.

5.4.1. Benchmark on public datasets

First, I would like to evaluate the performances of the proposed algorithm on public datasets to have a comparison with other existing algorithms. From the analysis done in chapter 2.2, on datasets, and the summary done in Table 2.4 I choose to evaluate our proposed system on two popular datasets: ZuBuD and ZuBuD+.

As a metric for our evaluation, I have chosen to use the Top1 metric, presented in Equation (2.7). If we consider the landmark detection system, more than CBIR this metric makes more sense than mAP.

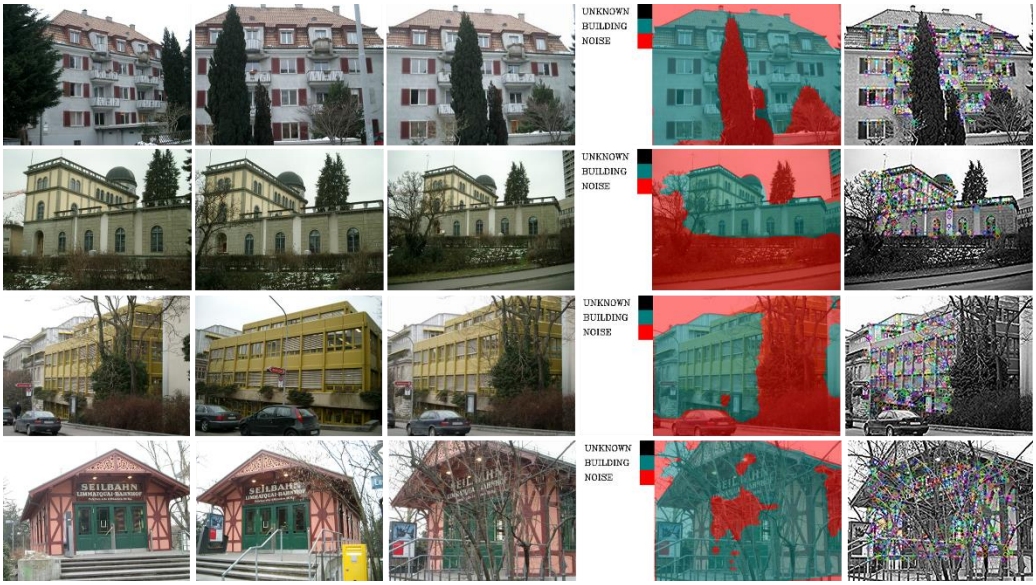


Figure 5.29 ZuBuD detection visual results.
Rows are as follows: object 113, 001,003,006.
Columns are left to right: 2 out of 5 train images, test image, ROI selection, feature map obtained.

For this experiment I scaled all the ZuBuD images to 240x320 size. This transformation of the input images was done by other solutions found in literature. The scope of this scaling is to ease the processing.

In Figure 5.29 I extracted images to create an idea about how the system works in these situations. The first aspect that I conclude is that the ROI selection works well considering that no data from the dataset was used in the training of the semantic segmentation. Visually we can observe that the assumptions made about the effect of restricting the region in which features should be extracted is valid. As we observe features are only created mainly on the buildings targeted.

As we observe in Table 5.4 we obtain good results on ZubuD and ZuBuD+ datasets with our proposed algorithm. For this evaluation of the proposed system the GPS tag processing is disabled because the dataset used does not have that information. But we experimented with different configurations of pyramid level and ROI selection.

The first experiments were conducted by switching on and off the ROI selection. As we can see in Table 5.4 the results get slightly better when using ROI. In column Config we have detailed the configuration used for the results.

The scope of experimenting without the semantic segmentation is to eliminate its contribution, which is not ideal. This effect is because of the lack of training data provided for this scenario. But as we can observe in Figure 5.29 the model predicts rather good the image in given circumstances.

Table 5.4 Top1 results on datasets.

Solution	Config	ZuBuD	ZuBuD+
[17]	-	100.00%	100.00%
Proposed no ROI on L0	D_S=8, D_NO=5, D_NL=6, D_THR=0.0012, UM_D=7, UM_S=0.1, B_S=300, THR=0.82	99.13%	99.80%
[88]	-	99.00%	99.00%
Proposed on L0	D_S=8, D_NO=5, D_NL=6, D_THR=0.001, UM_D=7, UM_S=0.5, B_S=150, THR=0.80	98.26%	99.60%
[60]	-	98.00%	-
[74]	-	95.00%	-
[69]	-	94.00%	-
Proposed on L1	D_S=8, D_NO=4, D_NL=6, D_THR=0.001, UM_D=7, UM_S=0.5, B_S=45, THR=0.80	93.91%	98.60%
Proposed no ROI on L1	D_S=8, D_NO=4, D_NL=6, D_THR=0.001, UM_D=5, UM_S=0.9, B_S=30, THR=0.82	92.17%	98.20%
[68]	-	92.00%	-
[63]	-	91.00%	-
[86]	-	81.00%	-
[66]	-	77.30%	-

On the other hand we can conclude that going down on the pyramid scales does have benefits on the run-time and resource consumption but affect the

recognition results. For L2 the results are very bad and somehow expected, if we consider the image to be 60x80 pixels in dimension.

As a preliminary conclusion, we can state that the proposed algorithm, even if tuned for the Timişoara scenario, obtains positive results on a public dataset. The results are expected as the proposed algorithm is an extension of the algorithm [108].

Further, we wish to investigate the effect of different descriptor sizes upon the results. This is an important aspect when thinking about resource consumption. In Figure 5.35 we present this analysis, to obtain the results we used the same configuration as presented in Table 5.4 for pyramid level 0 (original size) and no ROI selection done via semantic segmentation. We observe a positive result regarding this aspect, the fact that the results remain the same either for an 8bit size descriptor or a 1024-bit size descriptor.

The second aspect that has an important contribution to resource consumption is the cluster size for each landmark in the dictionary. In Figure 5.35 I present this analysis, for the results we used the same configuration as presented in Table 5.4 for pyramid level 0 (original size) and no ROI selection done via semantic segmentation. In this case we had to limit our analysis to a cluster size of 300 because above that threshold not all landmarks generated enough unique features. In conclusion, we clearly observed that a bigger cluster size brought better results.

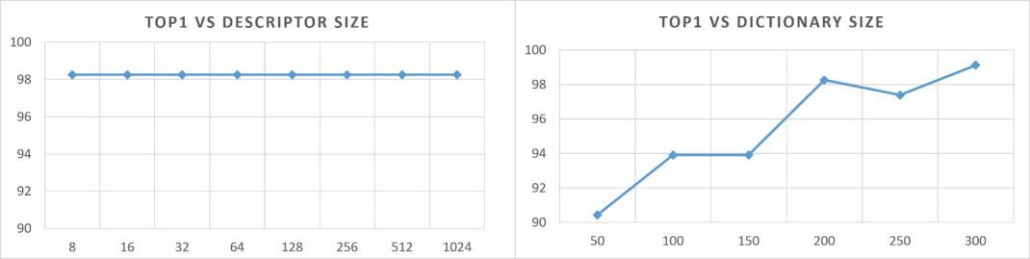


Figure 5.30 Evolution of Top1 metric concerning the descriptor size used (left) and BOF dictionary size per cluster (right)

For the lower scale pyramid level, I did not continue the investigation because the results will scale in the same trend, similar to the Top1 results.

Future work in this direction can be done in the direction of benchmarking the proposed algorithm in other scenarios by using other public datasets.

5.4.2. Benchmark on proposed datasets

In this section I want to evaluate the proposed algorithm on the novel TMBuD dataset, which is presented in chapter 5.2. In order to have a better understanding I use all three configurations available: Dataset 3_2 (3 views for training and 2 for evaluating), Dataset 3_5_Night (3 views for training and 5 for evaluating that contain night scenarios images), Dataset 3_N (unbalanced number of test images).

In Figure 5.31 I present visual results of the processing done when running the proposed algorithm on Dataset 3_N dataset. As we can observe the algorithm manages to handle selecting the correct ROI in both circumstances, day and night.

Another aspect that we can observe is the fact the proposed system can handle low quality images. For example the test image for landmark 001 is taken in

a dense fog and snow situation or in case of landmark 047 the test images are a night condition with low lighting.

Figure 5.31 TMBuD detection visual results.
Rows are as follows: object 001, 018, 047.
Columns are left to right: 3 train images, test image, ROI selection, feature map obtained

To test the algorithm, I did the evaluation on all three provided variants of the proposed dataset with two configurations: with GPS tag or without GPS tag. By using difference configuration, we would get the chance to observe the effect they have on the result. Each version of the dataset will bring forward new challenges in which the system needs to handle.

In Table 5.5 I present all the configurations and Top1 metrics obtained when evaluating the proposed dataset on different variants of it. As a first preliminary conclusion, I can observe that when introducing the GPS tag results are improved for any pyramid level. The second preliminary conclusion is the fact that using L3 pyramid level does not obtain good results overall even if the resource consumption is reduced dramatically. The lack of results on L3 is not something that we did not expect as the input image is only 90x160 pixels in size.

I can observe that the best result is obtained using GPS on L1 pyramid level. The results are sorted upon the metrics obtained in the 3_N dataset which I consider being the most complex of the three provided. A positive thing is that the difference

between the best result obtained, and the desired L2 with GPS is only 2% which gives us confidence to use the configuration further.

Table 5.5 Summary of experiments done with Top1 results

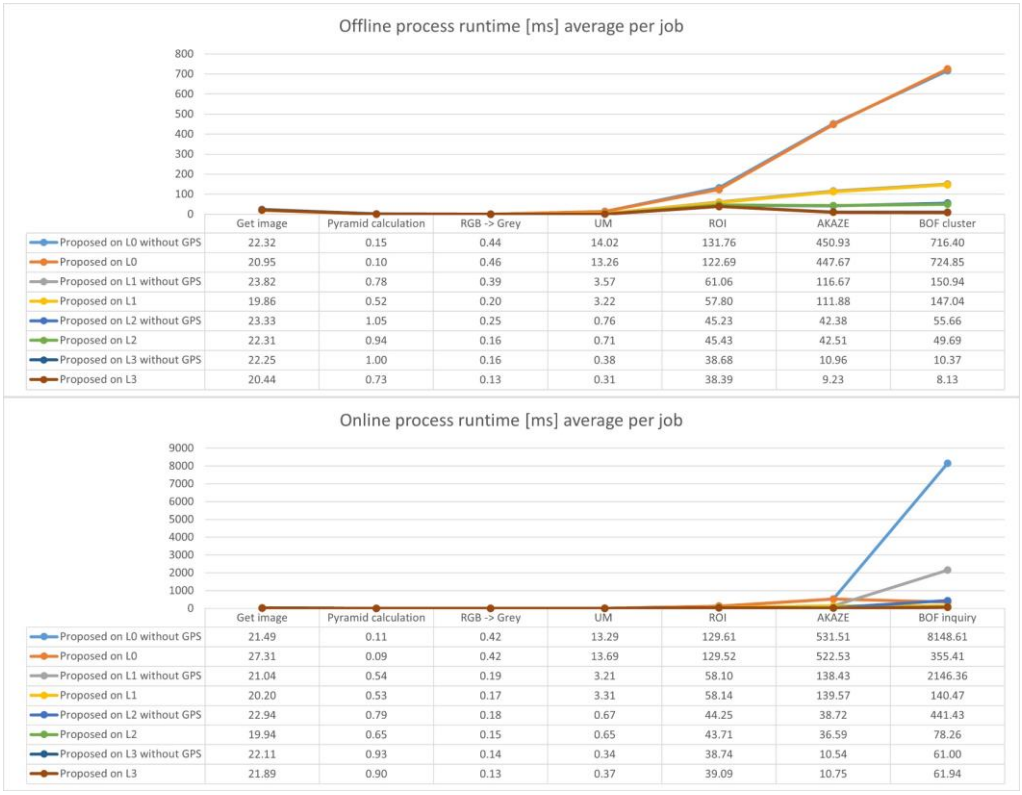
Configuration	Lvl	GPS	3_2	3_5_Night	3_N
D_S=8, D_NO=6, D_NL=3, D_THR=0.001, UM_D=7, UM_S=0.9, B_S=400, THR=0.8, D_GPS=100	L1	Yes	93.62%	93.59%	92.05%
D_S=8, D_NO=6, D_NL=3, D_THR=0.001, UM_D=7, UM_S=0.9, B_S=400, THR=0.8, D_GPS=100	L0	Yes	91.91%	91.81%	91.91%
D_S=8, D_NO=6, D_NL=3, D_THR=0.001, UM_D=7, UM_S=0.9, B_S=350, THR=0.8, D_GPS=100	L2	Yes	92.82%	92.17%	90.10%
D_S=8, D_NO=6, D_NL=3, D_THR=0.001, UM_D=7, UM_S=0.9, B_S=400, THR=0.8	L1	No	88.44%	87.90%	85.65%
D_S=8, D_NO=6, D_NL=3, D_THR=0.001, UM_D=7, UM_S=0.9, B_S=400, THR=0.8	L0	No	82.86%	87.90%	84.66%
D_S=8, D_NO=6, D_NL=3, D_THR=0.001, UM_D=7, UM_S=0.9, B_S=350, THR=0.8	L2	No	86.85%	83.27%	79.78%
D_S=8, D_NO=6, D_NL=3, D_THR=0.001, UM_D=7, UM_S=0.9, B_S=30, THR=0.8, D_GPS=75	L3	Yes	74.90%	72.60%	69.59%
D_S=8, D_NO=6, D_NL=3, D_THR=0.001, UM_D=7, UM_S=0.9, B_S=30, THR=0.8	L3	No	48.61%	43.77%	41.42%

Of course, we could further tune and refine the search for the best parameters combination. But this is something that is desired but rather hard to accomplish if taken into consideration that beside the 13 parameters that we varied we need to add the fixed parameters we had for the BOF generation: trees (set at 8) and leafs(set at 50).

Figure 5.32 Miss detected images for L2 with GPS tag

In Figure 5.32 we present an example of miss matched images. As we can observe most of the missing examples have the same root cause: low key-point matching or ROI selection is not accurate enough. Both of those problems can be fixed in a real word application in the late fine-tuning phase. Of course, if we consider that the end use case is a real-life application one needs to be aware that not all cases can be covered.

In Figure 5.33 I present with the run-time analysis for each configuration evaluated in Table 5.5. When talking about a detection system most of the time the decision comes down to a trade-off between resources and performance. First thing that we can observe is the fact that for bigger resolution images, like L0 and L1, the GPS tag filtering has a visible effect on overall online runtime. If we look at L2, which would be our target as we consider a good balance between performance and resources. Another aspect that we need to mention is that only by filtering using GPS we reduced the BOF inquiry runtime by 70%.



	Get image	Pyramid calculation	RGB -> Grey	UM	ROI	AKAZE	BOF cluster
Proposed on L0 without GPS	22.32	0.15	0.44	14.02	131.76	450.93	716.40
Proposed on L0	20.95	0.10	0.46	13.26	122.69	447.67	724.85
Proposed on L1 without GPS	23.82	0.78	0.39	3.57	61.06	116.67	150.94
Proposed on L1	19.86	0.52	0.20	3.22	57.80	111.88	147.04
Proposed on L2 without GPS	23.33	1.05	0.25	0.76	45.23	42.38	55.66
Proposed on L2	22.31	0.94	0.16	0.71	45.43	42.51	49.69
Proposed on L3 without GPS	22.25	1.00	0.16	0.38	38.68	10.96	10.37
Proposed on L3	20.44	0.73	0.13	0.31	38.39	9.23	8.13

	Get image	Pyramid calculation	RGB -> Grey	UM	ROI	AKAZE	BOF inquiry
Proposed on L0 without GPS	21.49	0.11	0.42	13.29	129.61	531.51	8148.61
Proposed on L0	27.31	0.09	0.42	13.69	129.52	522.53	355.41
Proposed on L1 without GPS	21.04	0.54	0.19	3.21	58.10	138.43	2146.36
Proposed on L1	20.20	0.53	0.17	3.31	58.14	139.57	140.47
Proposed on L2 without GPS	22.94	0.79	0.18	0.67	44.25	38.72	441.43
Proposed on L2	19.94	0.65	0.15	0.65	43.71	36.59	78.26
Proposed on L3 without GPS	22.11	0.93	0.14	0.34	38.74	10.54	61.00
Proposed on L3	21.89	0.90	0.13	0.37	39.09	10.75	61.94

Figure 5.33 Runtime analysis of proposed algorithm on different levels and configurations

If we look at the online processing runtime, we can observe that the system can run around 6 frames per second (total runtime is 160ms without image acquire job). We eliminated the job that obtains the image as in a real time system the process is different from the one we used in our experiment. If we consider that in this step of the development the algorithm is not optimised for real-time conditions, it is a decent frame rate to process at.

The analysis presented in Figure 5.33 is again a confirmation that choosing the L2 level for processing in the proposed system was an inspired decision. The loss in accuracy is less harmful for the overall system than the benefits we gain from runtime. Another solution in this case would be the processing at different pyramid levels of parts of the algorithm

As we experimented in the public dataset section, section 5.4.1, in Figure 5.34 we present the evolution of Top1 metric when changing the descriptor size and dictionary cluster size. The evolution of the metric is similar to previous similar experiments. The fact that the Top1 metric does not change when using a higher sized descriptor is a benefit when talking about resource consumption. In the other experiment we observe that the metric increases linear with the dictionary cluster size, which means that if needed we can use a smaller size without dramatic loss in metrics.

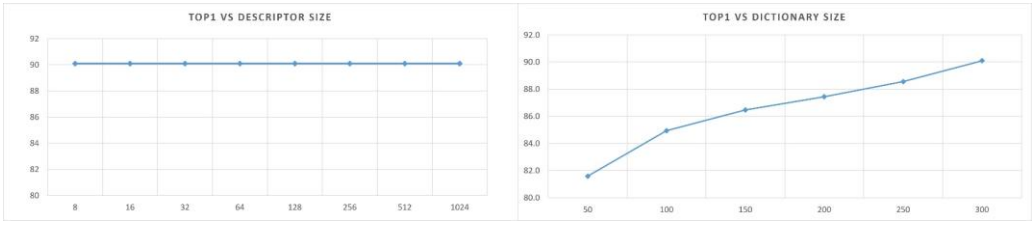


Figure 5.34 Evolution of Top1 metric concerning the descriptor size used (left) and BOF dictionary size per cluster (right)

5.4.3. Evaluation on mobile data

As stated before, I aimed to port the proposed algorithm on a mobile device. To get an idea how the system performs in a real live scenario, I captured several recordings using a mobile phone and used them as test scenarios.

If in the last chapters I have benchmarked the proposed algorithm on publicly available datasets and in our proposed dataset, in this chapter I test my algorithm on mobile recording scenarios.

I will use two scenarios, both in squares from Timişoara, where landmarks are close together and GPS data does not offer a great discrimination capability. The scenarios are presented in Figure 5.35, where user position is represented and the landmarks available in the dataset. To get a better image of the scenario I represented the distance to each landmark from the recording geographical point.

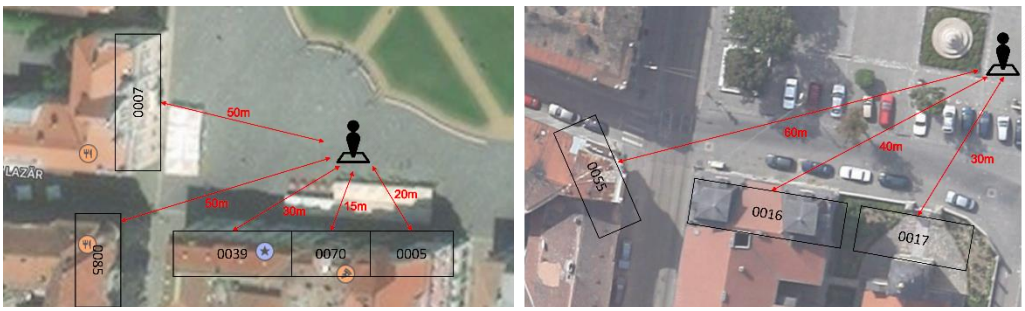


Figure 5.35 Description of capture scenarios with marking of position and distance to landmarks that are in the database. First experiment (left) is recorded in "Unirii" Square and second (right) in "Traian Square"

In order to do this experiment, a frame tracking was added to make the detection more stable. In that sense I implemented a naïve tracker in order to combat single frame flickering of detection. The algorithm is detailed in Figure 5.39.

```
Algorithm Frame detection tracking
/* Value of confidence to be added for each detection                    */
Set boost_value
/* Value of confidence to be decreased for each frame                    */
Set decay_value
/* Value of matched feature percent in which we consider the detection   */
Set feature_detection_rate
Create detection object dictionary
for each frame do
    Get object detection value
    if value > feature_detection_rate then
        if object in dictionary then
            Increase confidence with boost_value
            Check that confidence does not pass 100 for the object
            Decay confidence of rest of object with decay_value
        end
        else
            Add object to dictionary
            Increase confidence with boost_value
            Decay confidence of rest of object with decay_value
        end
    end
    else
        Decay confidence of objects with decay_value
    end
    Delete objects from dictionary with confidence 0
end
```

Figure 5.36 Detection frame tracker algorithm

The values I chose for these experiments are decay rate of 25, boost rate of 20 and detection threshold of 10 and consider a confidence of 50% to take in consideration the object. A decay rate of 25 means that each frame the object is not detected it loses 25% percent of its confidence, which translates in the fact that after 2 frames with no detection the object is discarded. A boost rate of 20 means that we need 3 consecutive detections to reach the desired confidence.

For the experiments, I have trained the algorithms with the landmarks available for TMBuD 3_2 dataset. That means the system is aware of 125 unique andmarks and it will try to classify between an object(landmark) from the dataset or 'None' if the building is not known.

In Figure 5.37 I chose 2 frames from the first experiment to show the way the system works on completely new data(images). I represented the original image, semantic segmented image, A-KAZE features detected features matched between the current frame and BOF clusters. In the last image I added the Landmark name and confidence of the detection.

Figure 5.37 Frames inside the system processing chain

As I expected, the semantic segmentation works as properly on new data. We see a relatively good pixel classification of the images, which represents a good baseline for the feature filtration we observe in the second image.

To better understand in Figure 5.38 I extracted several frames from the experiment. As we can observe in the beginning of the experiment the system detects correctly that we are looking at landmark object '005', the Bruck House. After a while the mobile phone is moved in another orientation. This causes the system to be unable to find any landmarks (which is correct), see t=242 image or t=503. But in t=257 we see the system manages to detect object '039' even if the transition is fast and, in the end, detects landmark '007' with ease.

Figure 5.38 Frame extraction from the experiment 1

In the second experiment, I chose a similar situation with 3 landmarks one side the other and numerous descriptors (cars, trees, people). In Figure 5.39 I present extractions from the experiment. As we can observe in the image corresponding to t=16 no landmark was detected when looking at buildings that are not in the dataset.

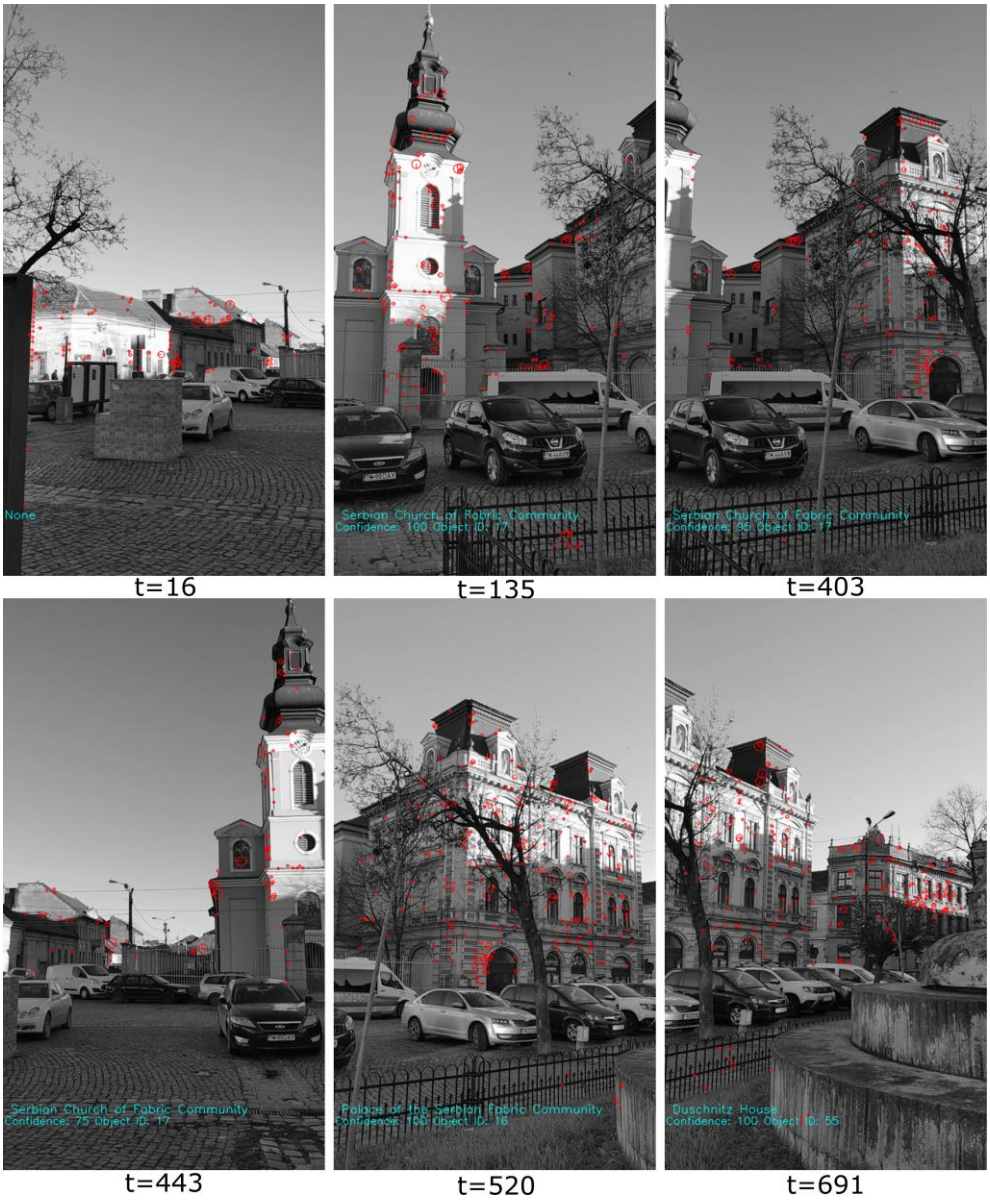


Figure 5.39 Frame extraction from the experiment 2

5.5. Conclusions

In this chapter, I presented the proposed algorithm, as seen in Figure 5.6 and detailed in chapter 5.3. The landmark detection system consists of two parts: the off-line processing part and on-line processing part.

The offline path takes the dataset images and generates a vector of A-KAZE features that is filtered using a region of interest. Afterwards, the features vectors are grouped together in a vector for each landmark which are clustered into BOF structure using the KNN clustering algorithm.

In the on-line phase we mirror the pre-processing part in order to filter out the features that are generated from distractors and using ANN classifier we find the closest similar landmark vector to the inquiry image.

In chapter 5.2 the TMBuD landmark detection dataset is proposed as a solution to benchmark and fine-tune the landmark detection system to Timişoara use case. As we can observe in Figure 5.3, Figure 5.5 and Figure 5.4 the dataset is a complex one providing challenging scenarios for benchmarking algorithm.

In chapter 5.4 we evaluate the proposed detection system on publicly available dataset, Timisoara dataset and mobile video stream in order to establish the performance. Each benchmark tends to offer a different perspective of use-cases in which the proposed system obtains positive results.

From Table 5.4 and Table 5.5 we can clearly state that the proposed algorithm has good results obtaining a value of 99.13% Top1 on ZuBuD dataset and 92.05% on TMBuD v3_N dataset.

6. CONTRIBUTIONS AND CONCLUSIONS

6.1. Conclusions

The background research for this thesis was done as part of the Multimedia Research Centre, Faculty of Electronics, Telecommunications, and Information Technologies of the Politehnica University of Timişoara. The research activity was complemented by volunteer activities and contributed to mobile AR applications serving the city of Timişoara for the title of European Capital of Culture in 2023.

My thesis can be summarized as a proposal for a landmark detection scheme tailored for Timişoara's urban environment. This complex algorithm can be integrated in a mobile application that can offer tourists the chance to better discover the urban scenario of our city.

To justify the novelty and actuality of the work done in this thesis I created a distribution of the references used over the year of publishing, which is presented in Figure 6.1. As we can observe the quantity of cited papers is denser in the last decade which is an expected distribution for this kind of manuscript. This distribution shows that the novel work done is based on actual materials from literature.

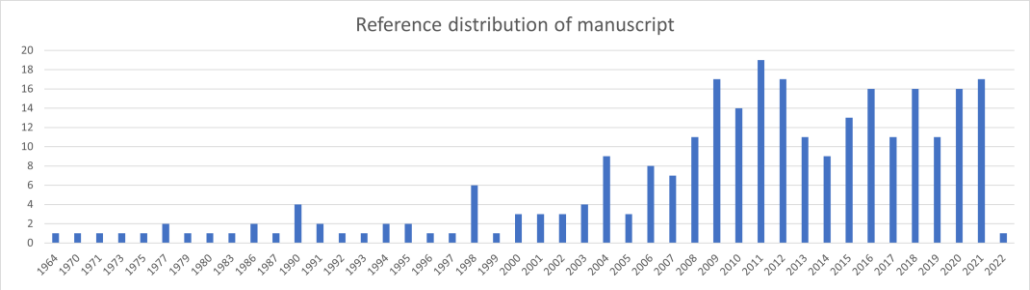


Figure 6.1 The distribution of cited works by year of publication

First, I focused on reviewing and understanding the CV landmark recognition domain in order to organize the existing solutions and challenges that this field faces. The critical review revealed that even if the domain is extensively researched in the last decades, it is far from being exhausted. Solutions were proposed that use classical low-level features such as lines or corners to more complex pixel-wise features as SIFT, SURF or A-KAZE. In the last years CNN based solutions emerged with the scope to offer solutions to existing problems.

Even if new solutions appear continuous in these directions they still cannot tackle and solve all the new datasets challenges that appear or data encoding difficulties.

We can say that at this moment we are assisting in a three-directions race for the perfect landmark recognition system. The system should tackle the image domain retrieval problem, mobile resource problem, and information band-width limitation. These are interconnected problems, newer and more powerful CV retrieval solutions

bring with them the need of more data processing, a need that stresses the limits of existing resources and bandwidth.

The continuous trade-off between performances and resources has a double role. On one hand it is the hard bottleneck of the system, but on the other hand is the catalyst that forces new solutions to appear.

Secondly, the focus moved on the simulation environment. This topic is a crucial and not trivial aspect of any system development. In order to develop any software system, one needs a stable simulation environment that offers the user capability of resimulation and logging information. In that direction an extensive review of existing software frameworks was done to understand what is available at this point.

After the analysis of existing frameworks, I chose to use EECF [7] as a framework for development of the proposed recognition system. This decision was first dictated of course by the familiarity with this solution, but secondly because it is a python-based system that offers the possibility to develop an end-to-end recognition chain inside of it.

The next research problem the thesis tackles is the enhancement of low-quality images. The diversity of image quality, illumination and resolution is a common issue in mobile applications. This context becomes even more important when we consider that landmark detection is mostly a CBIR domain problem. In this direction I investigated the usage of dilated filters in sharpening algorithms in order to enhance the feature extraction process. This step of feature creation is a fundamental step in our pursuit.

In the end the thesis focuses on the main hypothesis of the research: creating a tailored landmark recognition system for Timişoara urban environment. Information and context of the urban environment is the first thing that needs to be solved. In that sense I created a dataset containing only landmarks from Timişoara. The dataset respects and concurs easily with other similar sets found in literature. Second aspect is the proposal of a novel landmark detection system that is done in chapter 5. The proposed system uses features, A-KAZE, which have a low complexity but with fair results. This combined with the region of interest filtration and proven clustering concept resulted in a stable landmark detection system.

To correctly ensure assessment of the proposed CBIR system we evaluated on several popular landmark recognition dataset and in the Timişoara dataset that was provided. From the benchmark results we have confirmed that the proposed system obtains promising results.

Next, I attempted to answer the research questions that I raised at the beginning of the thesis, based on my research findings:

1. *What is the state of the art in urban landmark detection using mobile cameras imaging?*

   An extensive analysis and review of the landmark detection domain is provided in chapter 2. In order to be able to consider all the difficulties we can encounter, the problem was divided into smaller parts: landmark datasets and challenges, urban environment understanding datasets and landmark detection solutions.

2. *What should a simulation framework offer to be considered as a suitable solution for processing systems of this nature?*

   A review of existing framework solutions is provided in chapter 3 that focuses on highlighting main points of each framework and in which programming language they were developed. As criterion for choosing a framework I

considered the following: (1) main programming language used; (2) Capability of simulating an entire chain of processing; (3) Capability of outputting intermediate and debug data; (4) Facile integration of new CV algorithms.

3. What image signal processing algorithms enhance the image to obtain a better detection in this case?
In order to answer this question, I turned to the concept of dilated filters, that are described in chapter 4. From literature I found that using dilated filters can improve the image processing pipeline. This concept was validated from filter-based edge detection to CNN semantic segmentation. In that sense I successfully applied the dilated concept to image sharpening algorithms and obtained better results with the same processing resources needed.

4. What are the challenges in creating an urban landmark detection solution tailored for Timişoara use-case?
This is the fundamental question that was answered in this thesis, and it is tackled in chapter 5. As stated, before this is a two-part problem: the specific dataset needed for this and the tailored landmark recognition system. Starting with the information gathered and structured in chapter 2 I was able to provide a Timişoara dataset of landmarks that is easily scalable in the future and a landmark recognition system which can be easily adapted for a mobile application.

6.2. Theoretical contributions

1. Overview of existing CV software frameworks

I present a critical review on CV software frameworks offered in literature. The overview is presented in

Table **3.1** and considers the following information: the year of publication (announcement) of the framework, base programming language used (in more complex frameworks multiple were used), and the main benefits each framework offers to the users. The first contributions to this specific topic were done in previous work presented in [6], [7].

2. Analysis of dilated filters in several CV topics

In chapters 4.2 and chapter 4.3 I presented an analysis upon benefits of using dilated filters in CV topics like edge detection, line detection. The analysis is quantified in Table 4.1 and it is the basis for the decision of using dilated filters for image sharpening. Dilated filters were investigated in my previous work like [8], [9], [165].

3. Analysis on edge detection algorithm focused on urban scenarios

In previous work like [188], [273], [274], the vast domain of edge detection was tackled with the scope of analysing which algorithms are most suitable for detection of low level edge detection features in urban scenario images for future landmark detection algorithms. In [188], a new first order derivative edge detection

filter was proposed that was more suitable for detection edges on images that contain buildings.

4. Overview of Building (Landmark) recognition datasets

I present a critical review of the existing Landmark recognition dataset that is summarized in Table 2.1. The overview of the benchmarks focuses on the year of appearance, the number of images provided, the number of unique landmarks offered (classes), the number of images for each landmark, if the landmarks offer the GPS information, and if the images are from a street-view perspective(view). This work was sustained by our previous work done in[108], [188].

5. Overview of Urban environment understanding datasets

I present a critical review of the existing urban environment understanding dataset that is summarized in Table 2.2. The overview contains information about the dataset like the year of release, number of images and main resolution and the classes that are offered. Examples of each existing dataset can be found Figure 2.2. A first inside on this topic was published in our previous paper [39].

6. Overview of Landmark recognition solutions

I present an extensive overview of the existing Landmark recognition system from literature. The generic design of an landmark recognition system is presented in Figure 2.3 and Figure 2.4 while specific design decision for each reviewed systems is presented in Table 2.3. To better understand the domain at hand, the evaluation metrics for several systems are presented in Table 2.4 and Table 2.5.

7. Theoretical bases for Timişoara urban detection

In chapter 5 I proposed a solution tailored for detection of landmarks from the Timişoara urban environment. The basis of the system is presented in chapter 2 and the overall processing pipeline is presented in Figure 5.6. In order to forthfill this investigation, we used the knowledge gathered for past work from [4], [39], [108], [188].

6.3. Practical contributions

1. EECVF framework development

In my previous works [6], [7] the End-to-end Computer Vision Framework (EECVF) was proposed. The framework was developed with the main target to ease further development of myself and other researchers in the field of CV. The framework is a functional python-based solution in a modular manner with the clear scope that a user can use one or several blocks of the framework. The users are not required to handle the interconnections throughout the system.

2. Classical Edge detection algorithm

In my previous work [188] I have developed a first order derivative edge detection algorithm with the aim of revealing more edges on the 0 and 90 degrees angles. In [273] I extended the work on edge detection proposing a new Edge Drawing algorithm with an automated parameter choosing step. Following other researched was done in this domain be me like [274], [275].

3. Dilated High Pass and Unsharp Masking

In chapter 4.4 I presented the dilated filter sharpening concept. The dilated filters concept is applied to two low computational sharpening algorithms (the High Pass filter and Unsharp masking scheme). The result of this new concept is presented visually in Figure 4.19 and Figure 4.23, while statistical metrics are presented in Table 4.2 and Table 4.3. The encouraging results stood at the foundation of the decision to use this in the proposed landmark recognition solution. The dilated concept was extensively researched in my past works [8], [9], [165], [273].

4. Implementation in EECVF of the experiments done in the thesis

The analysis and benchmarking of the new dilated sharpening concept and proposed landmark recognition algorithm is implemented in the EECVF framework. By doing this anyone can re-run and redo all the experiments that were presented in chapter 4.4 or chapter 5.4. This can be done by executing the python module from EECFV: *main_sharpening_dilated* for the dilatation experiments and *main_TMBuD_detection* for the detection experiments.

5. TMBuD recognition dataset

extended the TMBuD dataset from an urban understanding dataset into a building detection dataset. The new dataset is presented in chapter 5.2. An important aspect to specify is that the benchmark comprises mobile photos of urban landmarks that aim to include variable quality, blurring, lighting changes, occlusions, and various viewing angles. In order to create the dataset, previous work done in [4], [39], [108], [188], [274] has offered important inside in the topic at hand.

6. Semantic segmentation training and evaluation for Timişoara urban environment

In chapter 5.3.2 I present the tailoring and fine tuning done for training a semantic segmentation network based on the ResNet model, that is used for the landmark recognition system. From Table 5.3 we can conclude that the process of fine-tunning the model for our use case was successful and this is supported by the visual results presented in Figure 5.21.

7. Landmark recognition system for Timişoara urban environment

In chapter 5 I proposed a tailored solution for landmarks detection from the Timişoara urban environment. In chapter 5.4 I present the results obtained on popular landmark datasets and on the proposed TMBuD one. From the results I can conclude that the research was successful in proposing a novel recognition system for Timişoara

landmarks. Implementing the solution proposal was built upon previous work done in [4], [6], [7], [39], [108], [188], [274] which created the basis for this development.

6.4. Future work

Several research directions can be derived based upon the research findings of this thesis. I will describe them briefly below.

No "bad" mobile retrieved images. One problem that appears and will continue to appear in the mobile CV domain is the quality of images delivered by the cameras. The quality can be low from the sensor itself or the user taking the image or even because of bandwidth limitations. In this concern as future work, I will concern myself with investigating the usage of dilated filters in more complex sharpening algorithms.

Better understanding of the input image. In terms of the semantic segmentation used for the proposed system we could investigate the usage of a more light-weight guided network like CGNet [276], ERFNet [277], RTSeg [278] or MobilNet [279]. The run-time was not an issue for this step of creating a prototype, but this would become more strident when creating an actual mobile application.

CNN based landmark recognition. As we concluded from the literature review, like any other CV domain, AI solutions appear in order to solve the problems at hand. In future research I would undergo the modification of the proposed system in a way of utilizing CNN models like R-MAC [99], SqueezeNet [112] or NU-LiteNet [55].

Landmark recognition using Semantic Segmentation. A new concept can change the way we tackle this problem, in one network to semantically understand the environment and detect the landmark. Similar concepts of fusing together in the same model semantic segmentation and image classification was proposed in literature in papers like [280], [281].

BIBLIOGRAPHY

[1] K. Lynch, *The Image of the City*. MIT Press, 1964.

[2] J. Yun, "A copy is (not a simple) copy: Role of urban landmarks in branding Seoul as a global city," *Front. Archit. Res.*, vol. 8, no. 1, pp. 44–54, Mar. 2019, doi: 10.1016/j.foar.2018.12.005.

[3] G. Eckbo, *Urban Landscape Design*. Accessed: Dec. 15, 2021. [Online]. Available: https://library.wur.nl/WebQuery/titel/477655

[4] S. Vert *et al.*, "User Evaluation of a Multi-Platform Digital Storytelling Concept for Cultural Heritage," *Mathematics*, vol. 9, no. 21, Art. no. 21, Jan. 2021, doi: 10.3390/math9212678, WOS:000750692000001 (Q1 journal).

[5] M. Daskolia, G. Dettori, and R. P. Lejano, "28. Urban Digital Storytelling," in *28. Urban Digital Storytelling*, Cornell University Press, 2017, pp. 271–278. doi: 10.7591/9781501712791-030.

[6] **C. Orhei**, M. Mocofan, S. Vert, and R. Vasiu, "End-to-End Computer Vision Framework," in *2020 International Symposium on Electronics and Telecommunications (ISETC)*, Nov. 2020, pp. 1–4. doi: 10.1109/ISETC50328.2020.9301078, WOS:000612681000017.

[7] **C. Orhei**, S. Vert, M. Mocofan, and R. Vasiu, "End-To-End Computer Vision Framework: An Open-Source Platform for Research and Education," *Sensors*, vol. 21, no. 11, Art. no. 11, Jan. 2021, doi: 10.3390/s21113691, WOS:000660684600001 (Q1 journal).

[8] V. Bogdan, C. Bonchis, and **C. Orhei**, *Custom Dilated Edge Detection Filters*. Václav Skala - UNION Agency, 2020. doi: 10.24132/CSRN.2020.3001.19.

[9] **C. Orhei**, V. Bogdan, C. Bonchis, and R. Vasiu, "Dilated Filters for Edge-Detection Algorithms," *Appl. Sci.*, vol. 11, no. 22, 2021, doi: 10.3390/app112210716, WOS:000725536600001 (Q2 journal).

[10] S. Vert and R. Vasiu, "Relevant Aspects for the Integration of Linked Data in Mobile Augmented Reality Applications for Tourism," in *Information and Software Technologies*, Cham, 2014, pp. 334–345. doi: 10.1007/978-3-319-11958-8_27.

[11] S. Vert and R. Vasiu, "Augmented Reality Lenses for Smart City Data: The Case of Building Permits," in *Recent Advances in Information Systems and Technologies*, Cham, 2017, pp. 521–527. doi: 10.1007/978-3-319-56535-4_53.

[12] S. Vert, D. Andone, and R. Vasiu, "Augmented and Virtual Reality for Public Space Art," *ITM Web Conf.*, vol. 29, p. 03006, 2019, doi: 10.1051/itmconf/20192903006.

[13] K. Brata and D. Liang, "Comparative study of user experience on mobile pedestrian navigation between digital map interface and location-based augmented reality," *Int. J. Electr. Comput. Eng.*, vol. 10, pp. 2037–2044, Apr. 2020, doi: 10.11591/ijece.v10i2.pp2037-2044.

[14] J. Li, W. Huang, L. Shao, and N. Allinson, "Building recognition in urban environments: A survey of state-of-the-art and future challenges," *Inf. Sci.*, vol. 277, pp. 406–420, Sep. 2014, doi: 10.1016/j.ins.2014.02.112.

[15] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. van Gool, and W. Purgathofer, "A Survey of Urban Reconstruction: A Survey of Urban Reconstruction," *Comput. Graph. Forum*, vol. 32, no. 6, pp. 146–177, Sep. 2013, doi: 10.1111/cgf.12077.

[16] L. Paletta, G. Fritz, C. Seifert, P. Luley, and A. Aimer, *A Mobile Vision Service for Multimedia Tourist Applications in Urban Environments*. 2006, p. 572. doi: 10.1109/ITSC.2006.1706801.
[17] F. Magliani, N. M. Bidgoli, and A. Prati, "A location-aware embedding technique for accurate landmark recognition," in *Proceedings of the 11th International Conference on Distributed Smart Cameras*, New York, NY, USA, Sep. 2017, pp. 9–14. doi: 10.1145/3131885.3131905.
[18] J. Knopp, J. Sivic, and T. Pajdla, "Avoiding Confusing Features in Place Recognition," in *Computer Vision – ECCV 2010*, Berlin, Heidelberg, 2010, pp. 748–761. doi: 10.1007/978-3-642-15549-9_54.
[19] S. Agarwal *et al.*, "Building Rome in a day," *Commun. ACM*, vol. 54, no. 10, pp. 105–112, Oct. 2011, doi: 10.1145/2001269.2001293.
[20] D. M. Chen *et al.*, "City-scale landmark identification on mobile devices," in *CVPR 2011*, Jun. 2011, pp. 737–744. doi: 10.1109/CVPR.2011.5995610.
[21] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua, "Worldwide Pose Estimation Using 3D Point Clouds," in *Computer Vision – ECCV 2012*, Berlin, Heidelberg, 2012, pp. 15–29. doi: 10.1007/978-3-642-33718-5_2.
[22] H. Shao, T. Svoboda, and L. Van Gool, "Zubud-Zurich buildings database for image based recognition." Technique Report No. 260, Swiss Federal Institute of Technolog, 2003.
[23] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, Minneapolis, MN, USA, Jun. 2007, pp. 1–8. doi: 10.1109/CVPR.2007.383172.
[24] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008, pp. 1–8. doi: 10.1109/CVPR.2008.4587635.
[25] F. Radenovic, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, Jun. 2018, pp. 5706–5715. doi: 10.1109/CVPR.2018.00598.
[26] H. Jegou, M. Douze, and C. Schmid, "Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search," in *Computer Vision – ECCV 2008*, Berlin, Heidelberg, 2008, pp. 304–317. doi: 10.1007/978-3-540-88682-2_24.
[27] J. Li and N. M. Allinson, "Subspace learning-based dimensionality reduction in building recognition," *Neurocomputing*, vol. 73, no. 1, pp. 324–330, Dec. 2009, doi: 10.1016/j.neucom.2009.08.016.
[28] Y. Avrithis, G. Tolias, and Y. Kalantidis, "Feature map hashing: sub-linear indexing of appearance and global geometry," in *Proceedings of the 18th ACM international conference on Multimedia*, New York, NY, USA, Oct. 2010, pp. 231–240. doi: 10.1145/1873951.1873985.
[29] Y. Avrithis, Y. Kalantidis, G. Tolias, and E. Spyrou, "Retrieving landmark and non-landmark images from community photo collections," in *Proceedings of the 18th ACM international conference on Multimedia*, New York, NY, USA, Oct. 2010, pp. 153–162. doi: 10.1145/1873951.1873973.
[30] Y. Kalantidis *et al.*, "VIRaL: Visual Image Retrieval and Localization," *Multimed. Tools Appl.*, vol. 51, no. 2, pp. 555–592, Jan. 2011, doi: 10.1007/s11042-010-0651-7.

[31] R. Ji *et al.*, "PKUBench: A context rich mobile visual search benchmark," in *2011 18th IEEE International Conference on Image Processing*, Brussels, Belgium, Sep. 2011, pp. 2545–2548. doi: 10.1109/ICIP.2011.6116182.

[32] K.-H. Yap, Z. Li, D.-J. Zhang, and Z.-K. Ng, "Efficient mobile landmark recognition based on saliency-aware scalable vocabulary tree," in *Proceedings of the 20th ACM international conference on Multimedia*, New York, NY, USA, Oct. 2012, pp. 1001–1004. doi: 10.1145/2393347.2396367.

[33] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla, "24/7 Place Recognition by View Synthesis," 2015, pp. 1808–1817. Accessed: Dec. 28, 2021. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Torii_247_Place_Recognition_2015_CVPR_paper.html

[34] T. Weyand, J. Hosang, and B. Leibe, "An Evaluation of Two Automatic Landmark Building Discovery Algorithms for City Reconstruction," in *Trends and Topics in Computer Vision*, vol. 6554, K. N. Kutulakos, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 310–323. doi: 10.1007/978-3-642-35740-4_24.

[35] T. Weyand and B. Leibe, "Visual landmark recognition from Internet photo collections: A large-scale evaluation," *Comput. Vis. Image Underst.*, vol. 135, pp. 1–15, Jun. 2015, doi: 10.1016/j.cviu.2015.02.002.

[36] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-Scale Image Retrieval With Attentive Deep Local Features," 2017, pp. 3456–3465. Accessed: Dec. 28, 2021. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2017/html/Noh_Large-Scale_Image_Retrieval_ICCV_2017_paper.html

[37] T. Weyand, A. Araujo, B. Cao, and J. Sim, "Google Landmarks Dataset v2 - A Large-Scale Benchmark for Instance-Level Recognition and Retrieval," 2020, pp. 2575–2584. Accessed: Dec. 28, 2021. [Online]. Available: https://openaccess.thecvf.com/content_CVPR_2020/html/Weyand_Google_Landmarks_Dataset_v2_-_A_Large-Scale_Benchmark_for_Instance-Level_CVPR_2020_paper.html

[38] P. Babahajiani, L. Fan, and M. Gabbouj, "Object Recognition in 3D Point Cloud of Urban Street Scene," in *Computer Vision - ACCV 2014 Workshops*, Cham, 2015, pp. 177–190. doi: 10.1007/978-3-319-16628-5_13.

[39] **C. Orhei**, S. Vert, M. Mocofan, and R. Vasiu, "TMBuD: A Dataset for Urban Scene Building Detection," in *Information and Software Technologies*, Cham, 2021, pp. 251–262. doi: 10.1007/978-3-030-88304-1_20.

[40] J. Fu, J.-K. Kämäräinen, A. G. Buch, and N. Krüger, "Indoor Objects and Outdoor Urban Scenes Recognition by 3D Visual Primitives," in *Computer Vision - ACCV 2014 Workshops*, Cham, 2015, pp. 270–285. doi: 10.1007/978-3-319-16628-5_20.

[41] A. Zlateski, R. Jaroensri, P. Sharma, and F. Durand, "On the Importance of Label Quality for Semantic Segmentation," 2018, pp. 1479–1487. Accessed: Dec. 26, 2021. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2018/html/Zlateski_On_the_Importance_CVPR_2018_paper.html

[42] F. Korc and W. Forstner, "eTRIMS Image Database for Interpreting Images of Man-Made Scenes," Dept. of Photogrammetry, University of Bonn, TR-IGG-P-2009-01, 2009. [Online]. Available: http://www.ipb.uni-bonn.de/projects/etrims_db/

[43] B. Fröhlich, E. Rodner, and J. Denzler, "A Fast Approach for Pixelwise Labeling of Facade Images," in *2010 20th International Conference on Pattern Recognition*, Aug. 2010, pp. 3029–3032. doi: 10.1109/ICPR.2010.742.

[44] C.-A. Brust, S. Sickert, M. Simon, E. Rodner, and J. Denzler, "Efficient Convolutional Patch Networks for Scene Understanding," in *CVPR Scene Understanding Workshop*, 2015.

[45] O. Teboul, L. Simon, P. Koutsourakis, and N. Paragios, "Segmentation of building facades using procedural shape priors," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2010, pp. 3105–3112. doi: 10.1109/CVPR.2010.5540068.

[46] O. Teboul, I. Kokkinos, L. Simon, P. Koutsourakis, and N. Paragios, "Shape grammar parsing via Reinforcement Learning," in *CVPR 2011*, Jun. 2011, pp. 2273–2280. doi: 10.1109/CVPR.2011.5995319.

[47] H. Riemenschneider *et al.*, "Irregular lattices for complex shape grammar facade parsing," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 1640–1647. doi: 10.1109/CVPR.2012.6247857.

[48] R. Tyleček and R. Šára, "Spatial Pattern Templates for Recognition of Objects with Regular Structure," in *Pattern Recognition*, Berlin, Heidelberg, 2013, pp. 364–374. doi: 10.1007/978-3-642-40602-7_39.

[49] H. Riemenschneider, A. Bódis-Szomorú, J. Weissenberg, and L. Van Gool, "Learning Where to Classify in Multi-view Semantic Segmentation," in *Computer Vision – ECCV 2014*, Cham, 2014, pp. 516–532. doi: 10.1007/978-3-319-10602-1_34.

[50] A. Martinović, J. Knopp, H. Riemenschneider, and L. Van Gool, "3D all the way: Semantic segmentation of urban scenes from start to end in 3D," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 4456–4465. doi: 10.1109/CVPR.2015.7299075.

[51] R. Gadde, R. Marlet, and N. Paragios, "Learning Grammars for Architecture-Specific Facade Parsing," *Int. J. Comput. Vis.*, vol. 117, no. 3, pp. 290–316, May 2016, doi: 10.1007/s11263-016-0887-4.

[52] H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun, "Performance evaluation in content-based image retrieval: overview and proposals," *Pattern Recognit. Lett.*, vol. 22, no. 5, pp. 593–601, Apr. 2001, doi: 10.1016/S0167-8655(00)00118-5.

[53] T. Deselaers, D. Keysers, and H. Ney, "Features for image retrieval: an experimental comparison," *Inf. Retr.*, vol. 11, no. 2, pp. 77–107, Apr. 2008, doi: 10.1007/s10791-007-9039-3.

[54] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, "A survey of content-based image retrieval with high-level semantics," *Pattern Recognit.*, vol. 40, no. 1, pp. 262–282, Jan. 2007, doi: 10.1016/j.patcog.2006.04.045.

[55] C. Termritthikun, S. Kanprachar, and P. Muneesawang, "NU-LiteNet: Mobile Landmark Recognition using Convolutional Neural Networks," *ArXiv181001074 Cs*, Oct. 2018, Accessed: Jan. 30, 2022. [Online]. Available: http://arxiv.org/abs/1810.01074

[56] Y.-H. Liu, A. J. T. Lee, and F. Chang, "Object recognition using discriminative parts," *Comput. Vis. Image Underst.*, vol. 116, no. 7, pp. 854–867, Jul. 2012, doi: 10.1016/j.cviu.2012.03.007.

[57] P. Bhattacharya and M. Gavrilova, "A Survey of Landmark Recognition Using the Bag-of-Words Framework," in *Studies in Computational Intelligence*, vol. 441, 2013, pp. 243–263. doi: 10.1007/978-3-642-31745-3_13.

[58] X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *Comput. Vis. Image Underst.*, vol. 150, pp. 109–125, Sep. 2016, doi: 10.1016/j.cviu.2016.03.013.

[59] F. Magliani, T. Fontanini, and A. Prati, "Landmark Recognition: From Small-Scale to Large-Scale Retrieval," in *Recent Advances in Computer Vision*, vol. 804, M. Hassaballah and K. M. Hosny, Eds. Cham: Springer International Publishing, 2019, pp. 237–259. doi: 10.1007/978-3-030-03000-1_10.

[60] D. M. Chen, S. S. Tsai, V. Ch, G. Takacs, J. Singh, and B. Girod, "Tree histogram coding for mobile image matching," 2009.

[61] T. Chen, K. Wu, K.-H. Yap, Z. Li, and F. S. Tsai, "A Survey on Mobile Landmark Recognition for Information Retrieval," in *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, Taipei, Taiwan, 2009, pp. 625–630. doi: 10.1109/MDM.2009.107.

[62] M. El Choubassi, O. Nestares, Y. Wu, I. Kozintsev, and H. Haussecker, "An Augmented Reality Tourist Guide on Your Mobile Devices," in *Advances in Multimedia Modeling*, Berlin, Heidelberg, 2010, pp. 588–602. doi: 10.1007/978-3-642-11301-7_58.

[63] G. Fritz, C. Seifert, and L. Paletta, "A Mobile Vision System for Urban Detection with Informative Local Descriptors," in *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, Jan. 2006, pp. 30–30. doi: 10.1109/ICVS.2006.5.

[64] G. Fritz, C. Seifert, M. Kumar, and L. Paletta, "Building Detection from Mobile Imagery Using Informative SIFT Descriptors," in *Image Analysis*, Berlin, Heidelberg, 2005, pp. 629–638. doi: 10.1007/11499145_64.

[65] Q. Iqbal and J. K. Aggarwall, "Applying perceptual grouping to content-based image retrieval: building images," in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, Jun. 1999, vol. 1, pp. 42-48 Vol. 1. doi: 10.1109/CVPR.1999.786915.

[66] H. Shao, T. Svoboda1, T. Tuytelaars, and L. Van Gool, "HPAT Indexing for Fast Object/Scene Recognition Based on Local Appearance," in *Image and Video Retrieval*, Berlin, Heidelberg, 2003, pp. 71–80. doi: 10.1007/3-540-45113-7_8.

[67] T. Tuytelaars and L. Van Gool, "Matching Widely Separated Views Based on Affine Invariant Regions," *Int. J. Comput. Vis.*, vol. 59, no. 1, pp. 61–85, Aug. 2004, doi: 10.1023/B:VISI.0000020671.28016.e8.

[68] T. Goedeme, T. Tuytelaars, and L. Van Gool, "Fast wide baseline matching for visual navigation," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, Jun. 2004, vol. 1, p. I–I. doi: 10.1109/CVPR.2004.1315009.

[69] N. J. C. Groeneweg, B. de Groot, A. H. R. Halma, B. R. Quiroga, M. Tromp, and F. C. A. Groen, "A Fast Offline Building Recognition Application on a Mobile Telephone," in *Advanced Concepts for Intelligent Vision Systems*, Berlin, Heidelberg, 2006, pp. 1122–1132. doi: 10.1007/11864349_102.

[70] M. J. Swain and D. H. Ballard, "Color indexing," *Int. J. Comput. Vis.*, vol. 7, no. 1, pp. 11–32, Nov. 1991, doi: 10.1007/BF00130487.

[71] G. Fritz, L. Paletta, and H. Bischof, "Object recognition using local information content," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, Aug. 2004, vol. 2, pp. 15-18 Vol.2. doi: 10.1109/ICPR.2004.1333968.

[72] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual Categorization with Bags of Keypoints," in *Workshop on statistical learning in computer vision*, Prague, 2004, vol. 1, p. 16.
[73] "Baeza-Yates: Modern information retrieval - Google Scholar." https://scholar.google.com/scholar_lookup?title=Modern%20Information%20Retrieval&author=R.%20Baeza-Yates&publication_year=1999 (accessed Jan. 30, 2022).
[74] W. Zhang and J. Košecká, "Hierarchical building recognition," *Image Vis. Comput.*, vol. 25, no. 5, pp. 704–716, May 2007, doi: 10.1016/j.imavis.2006.05.016.
[75] H.-H. Trinh, D.-N. Kim, and K.-H. Jo, "Facet-based multiple building analysis for robot intelligence," *Appl. Math. Comput.*, vol. 205, no. 2, pp. 537–549, Nov. 2008, doi: 10.1016/j.amc.2008.05.059.
[76] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004, doi: 10.1023/B:VISI.0000029664.99615.94.
[77] F. Bajramovic, F. Mattern, N. Butko, and J. Denzler, "A Comparison of Nearest Neighbor Search Algorithms for Generic Object Recognition," in *Advanced Concepts for Intelligent Vision Systems*, Berlin, Heidelberg, 2006, pp. 1186–1197. doi: 10.1007/11864349_108.
[78] X. Wangming, W. Jin, L. Xinhai, Z. Lei, and S. Gang, "Application of Image SIFT Features to the Context of CBIR," in *2008 International Conference on Computer Science and Software Engineering*, Dec. 2008, vol. 4, pp. 552–555. doi: 10.1109/CSSE.2008.1230.
[79] J. S. Beis and D. G. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 1997, pp. 1000–1006. doi: 10.1109/CVPR.1997.609451.
[80] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008, doi: 10.1016/j.cviu.2007.09.014.
[81] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Jun. 2006, vol. 2, pp. 2161–2168. doi: 10.1109/CVPR.2006.264.
[82] C. Siagian and L. Itti, "Rapid Biologically-Inspired Scene Classification Using Features Shared with Visual Attention," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 2, pp. 300–312, Feb. 2007, doi: 10.1109/TPAMI.2007.40.
[83] Y.-T. Zheng *et al.*, "Tour the World: building a web-scale landmark recognition engine," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1085--1092.
[84] K. Mikolajczyk and C. Schmid, "Scale & Affine Invariant Interest Point Detectors," *Int. J. Comput. Vis.*, vol. 60, no. 1, pp. 63–86, Oct. 2004, doi: 10.1023/B:VISI.0000027790.02288.f2.
[85] M. Svensén and C. M. Bishop, "Pattern recognition and machine learning." Springer Berlin/Heidelberg, Germany, 2007.
[86] Y.-C. Chung, T. X. Han, and Z. He, "Building recognition using sketch-based representations and spectral graph matching," in *2009 IEEE 12th International Conference on Computer Vision*, Sep. 2009, pp. 2014–2020. doi: 10.1109/ICCV.2009.5459444.

[87] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image Vis. Comput.*, vol. 22, no. 10, pp. 761–767, Sep. 2004, doi: 10.1016/j.imavis.2004.02.006.
[88] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2010, pp. 3304–3311. doi: 10.1109/CVPR.2010.5540039.
[89] K. Mikolajczyk *et al.*, "A Comparison of Affine Region Detectors," *Int. J. Comput. Vis.*, vol. 65, no. 1, pp. 43–72, Nov. 2005, doi: 10.1007/s11263-005-3848-x.
[90] H. Jégou, M. Douze, and C. Schmid, "Product Quantization for Nearest Neighbor Search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 117–128, Jan. 2011, doi: 10.1109/TPAMI.2010.57.
[91] J. Li and N. M. Allinson, "Relevance feedback-based building recognition," in *Visual Communications and Image Processing 2010*, Jul. 2010, vol. 7744, pp. 92–100. doi: 10.1117/12.863191.
[92] G. Guo, S. Z. Li, and K. L. Chan, "Support vector machines for face recognition," *Image Vis. Comput.*, vol. 19, no. 9, pp. 631–638, Aug. 2001, doi: 10.1016/S0262-8856(01)00046-4.
[93] J. Li and N. Allinson, "Building Recognition Using Local Oriented Features," *IEEE Trans. Ind. Inform.*, vol. 9, no. 3, pp. 1697–1704, Aug. 2013, doi: 10.1109/TII.2013.2245910.
[94] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 9, pp. 891–906, 1991.
[95] T. Ge and Q. Ke, "Sparse-Coded Features for Image Retrieval," in *BMVC*, 2013, pp. 132--1.
[96] S. Winder, G. Hua, and M. Brown, "Picking the best DAISY," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 178–185. doi: 10.1109/CVPR.2009.5206839.
[97] M. Zontak and M. Irani, "Internal statistics of a single natural image," in *CVPR 2011*, Jun. 2011, pp. 977–984. doi: 10.1109/CVPR.2011.5995401.
[98] Y. Zhang, J. Chen, X. Huang, and Y. Wang, "A Probabilistic Analysis of Sparse Coded Feature Pooling and Its Application for Image Retrieval," *PLOS ONE*, vol. 10, no. 7, p. e0131721, Jul. 2015, doi: 10.1371/journal.pone.0131721.
[99] G. Tolias, R. Sicre, and H. Jégou, "Particular object retrieval with integral max-pooling of CNN activations," *ArXiv151105879 Cs*, Feb. 2016, Accessed: Jan. 31, 2022. [Online]. Available: http://arxiv.org/abs/1511.05879
[100] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, "Deep Image Retrieval: Learning Global Representations for Image Search," in *European conference on computer vision*, 2016, pp. 241--257. Accessed: Jan. 31, 2022. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-46466-4_15
[101] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, "End-to-End Learning of Deep Visual Representations for Image Retrieval," *Int. J. Comput. Vis.*, vol. 124, no. 2, pp. 237–254, Sep. 2017, doi: 10.1007/s11263-017-1016-8.
[102] R. Arandjelović and A. Zisserman, "Three things everyone should know to improve object retrieval," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 2911–2918. doi: 10.1109/CVPR.2012.6248018.
[103] O. Seddati, S. Dupont, S. Mahmoudi, and M. Parian, "Towards Good Practices for Image Retrieval Based on CNN Features," 2017, pp. 1246–1255. Accessed: Jan. 31, 2022. [Online]. Available:

https://openaccess.thecvf.com/content_ICCV_2017_workshops/w18/html/Seddati_Towards_Good_Practices_ICCV_2017_paper.html
[104] Z. Laskar and J. Kannala, "Context Aware Query Image Representation for Particular Object Retrieval," in *Image Analysis*, Cham, 2017, pp. 88–99. doi: 10.1007/978-3-319-59129-2_8.
[105] S. K. Raza, T. Rajab, S. A. Ahmed, and M. Khurram, "Visual Landmarks Recognition of Urban Structures using Convolutional Neural Network," in *2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, Sukkur, Pakistan, Jan. 2020, pp. 1–9. doi: 10.1109/iCoMET48670.2020.9073795.
[106] Y. Wu, W. Che, and B. Huang, "An Improved 3D Registration Method of Mobile Augmented Reality for Urban Built Environment," *Int. J. Comput. Games Technol.*, vol. 2021, p. e8810991, Feb. 2021, doi: 10.1155/2021/8810991.
[107] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast Retina Keypoint," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 510–517. doi: 10.1109/CVPR.2012.6247715.
[108] **C. Orhei**, L. Radu, M. Mocofan, S. Vert, and R. Vasiu, "CBIR for urban building using A-KAZE features," in *2021 IEEE 27th International Symposium for Design and Technology in Electronic Packaging (SIITME)*, Oct. 2021, pp. 218–221. doi: 10.1109/SIITME53254.2021.9663587.
[109] P. Alcantarilla, J. Nuevo, and A. Bartoli, "Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces," in *Procedings of the British Machine Vision Conference 2013*, Bristol, 2013, p. 13.1-13.11. doi: 10.5244/C.27.13.
[110] M. Muja and D. G. Lowe, "FAST APPROXIMATE NEAREST NEIGHBORS WITH AUTOMATIC ALGORITHM CONFIGURATION:," in *Proceedings of the Fourth International Conference on Computer Vision Theory and Applications*, Lisboa, Portugal, 2009, pp. 331–340. doi: 10.5220/0001787803310340.
[111] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ArXiv14091556 Cs*, Apr. 2015, Accessed: Jan. 19, 2022. [Online]. Available: http://arxiv.org/abs/1409.1556
[112] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," *ArXiv160207360 Cs*, Nov. 2016, Accessed: Jan. 30, 2022. [Online]. Available: http://arxiv.org/abs/1602.07360
[113] "Google Landmark Recognition 2021." https://kaggle.com/c/landmark-recognition-2021 (accessed Feb. 22, 2022).
[114] "Google Landmark Retrieval 2021." https://kaggle.com/c/landmark-retrieval-2021 (accessed Feb. 22, 2022).
[115] K.-H. Yap, T. Chen, Z. Li, and K. Wu, "A Comparative Study of Mobile-Based Landmark Recognition Techniques," *IEEE Intell. Syst.*, vol. 25, no. 1, pp. 48–57, Jan. 2010, doi: 10.1109/MIS.2010.12.
[116] Z. Kim *et al.*, "Towards A Fairer Landmark Recognition Dataset," *ArXiv210808874 Cs*, Aug. 2021, Accessed: Jan. 30, 2022. [Online]. Available: http://arxiv.org/abs/2108.08874
[117] R. Klette, *Concise Computer Vision*. London: Springer London, 2014. doi: 10.1007/978-1-4471-6320-6.
[118] "10 Ways Computer Vision and Data Analytics Help Your Business," *Express Analytics*, Oct. 04, 2016. https://expressanalytics.com/blog/10-things-about-computer-vision-and-data-analytics-you-may-not-know/ (accessed Feb. 22, 2022).

[119] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer Science & Business Media, 2010.
[120] S. Krig, "Vision Pipelines and Optimizations," in *Computer Vision Metrics: Textbook Edition*, S. Krig, Ed. Cham: Springer International Publishing, 2016, pp. 273–317. doi: 10.1007/978-3-319-33762-3_8.
[121] M. Buckler, S. Jayasuriya, and A. Sampson, "Reconfiguring the Imaging Pipeline for Computer Vision," 2017, pp. 975–984. Accessed: Dec. 16, 2021. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2017/html/Buckler_Reconfiguring_the_Imaging_ICCV_2017_paper.html
[122] P. Patel and A. Thakkar, "The upsurge of deep learning for computer vision applications," *Int. J. Electr. Comput. Eng. IJECE*, vol. 10, no. 1, p. 538, Feb. 2020, doi: 10.11591/ijece.v10i1.pp538-548.
[123] **C. Orhei**, *CipiOrhei/eecvf*. 2021. Accessed: Feb. 22, 2022. [Online]. Available: https://github.com/CipiOrhei/eecvf
[124] G. Bradski, "The OpenCV Library.," *Dr Dobbs J. Softw. Tools Prof. Program.*, vol. 25, no. 11, 2000, Accessed: Dec. 17, 2021. [Online]. Available: https://elibrary.ru/item.asp?id=4934581
[125] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, Inc., 2008.
[126] "OpenCV: Structured forests for fast edge detection." https://docs.opencv.org/3.4.9/d0/da5/tutorial_ximgproc_prediction.html (accessed Mar. 01, 2020).
[127] "Image Processing Toolbox." https://www.mathworks.com/products/image.html (accessed Dec. 17, 2021).
[128] P. Goldsborough, "A Tour of TensorFlow," *ArXiv161001178 Cs*, Oct. 2016, Accessed: Dec. 17, 2021. [Online]. Available: http://arxiv.org/abs/1610.01178
[129] M. Abadi, "TensorFlow: learning functions at scale," in *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, New York, NY, USA, Sep. 2016, p. 1. doi: 10.1145/2951913.2976746.
[130] M. Abadi *et al.*, "{TensorFlow}: A System for {Large-Scale} Machine Learning," 2016, pp. 265–283. Accessed: Dec. 17, 2021. [Online]. Available: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi
[131] J. V. Dillon *et al.*, "TensorFlow Distributions," *ArXiv171110604 Cs Stat*, Nov. 2017, Accessed: Dec. 17, 2021. [Online]. Available: http://arxiv.org/abs/1711.10604
[132] R. Farber, *CUDA Application Design and Development*. Elsevier, 2011.
[133] K. Demaagd, A. Oliver, N. Oostendorp, and K. Scott, *Practical Computer Vision with SimpleCV: The Simple Way to Make Technology See*. O'Reilly Media, Inc., 2012.
[134] "SimpleCV." http://simplecv.org/ (accessed Dec. 17, 2021).
[135] D. Tschumperlé, "The CImg Library," in *IPOL 2012 Meeting on Image Processing Libraries*, Cachan, France, Jun. 2012, p. 4 pp. Accessed: Dec. 17, 2021. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00927458
[136] M. Chen *et al.*, "Generative Pretraining From Pixels," in *Proceedings of the 37th International Conference on Machine Learning*, Nov. 2020, pp. 1691–1703. Accessed: Dec. 17, 2021. [Online]. Available: https://proceedings.mlr.press/v119/chen20s.html

[137] "Peter's Functions for Computer Vision." https://www.peterkovesi.com/matlabfns/ (accessed Dec. 17, 2021).
[138] G. Holmes, A. Donkin, and I. H. Witten, "WEKA: a machine learning workbench," in *Proceedings of ANZIIS '94 - Australian New Zealnd Intelligent Information Systems Conference*, Nov. 1994, pp. 357–361. doi: 10.1109/ANZIIS.1994.396988.
[139] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009, doi: 10.1145/1656274.1656278.
[140] D. Pescaru and M. Mocofan, "An easy-to-use distributed framework for image processing," *Facta Univ. - Ser. Electron. Energ.*, vol. 17, no. 3, pp. 455–466, 2004, doi: 10.2298/FUEE0403455P.
[141] J. M. Perez, B. Bonev, P. Suau, D. Viejo, and M. Cazorla, "JavaVis: open source code for computer vision subjects," in *Proceedings of the FLOSS International Conference 2007*, Cadiz, May 2007, p. 10.
[142] A. Rodríguez, P. López-de-Teruel, A. Ruiz, G. García-Mateos, and L. Fernández, *QVision, a Development Framework for Real-time Computer Vision and Image Processing Research.* 2008, p. 414.
[143] B. Nouvel and S. Satoh, "The python computer vision framework," in *Proceedings of the 18th ACM international conference on Multimedia*, New York, NY, USA, Oct. 2010, pp. 1481–1484. doi: 10.1145/1873951.1874253.
[144] G. Williams J., "Rattle: A Data Mining GUI for R," *R J.*, vol. 1, no. 2, p. 45, 2009, doi: 10.32614/RJ-2009-016.
[145] S. Gould, "DARWIN: A Framework for Machine Learning and Computer Vision Research and Development," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 3533--3537, 2012.
[146] J. Schindelin *et al.*, "Fiji: an open-source platform for biological-image analysis," *Nat. Methods*, vol. 9, no. 7, pp. 676–682, Jul. 2012, doi: 10.1038/nmeth.2019.
[147] P. Kiefer, U. Schmitt, and J. A. Vorholt, "eMZed: an open source framework in Python for rapid and interactive development of LC/MS data analysis workflows," *Bioinformatics*, vol. 29, no. 7, pp. 963–964, Apr. 2013, doi: 10.1093/bioinformatics/btt080.
[148] K. Radlak, M. Frackiewicz, M. Szczepanski, M. Kawulok, and M. Czardybon, "Adaptive Vision Studio — Educational tool for image processing learning," in *2015 IEEE Frontiers in Education Conference (FIE)*, Oct. 2015, pp. 1–8. doi: 10.1109/FIE.2015.7344309.
[149] H. Agrawal *et al.*, "CloudCV: Large-Scale Distributed Computer Vision as a Cloud Service," in *Mobile Cloud Visual Media Computing: From Interaction to Service*, G. Hua and X.-S. Hua, Eds. Cham: Springer International Publishing, 2015, pp. 265–290. doi: 10.1007/978-3-319-24702-1_11.
[150] D. Wang, D. J. Foran, X. Qi, and M. Parashar, "HetroCV: Auto-tuning Framework and Runtime for Image Processing and Computer Vision Applications on Heterogeneous Platform," in *2015 44th International Conference on Parallel Processing Workshops*, Sep. 2015, pp. 119–128. doi: 10.1109/ICPPW.2015.21.
[151] E. Rupnik, M. Daakir, and M. Pierrot Deseilligny, "MicMac – a free, open-source solution for photogrammetry," *Open Geospatial Data Softw. Stand.*, vol. 2, no. 1, p. 14, Jun. 2017, doi: 10.1186/s40965-017-0027-2.
[152] M. Alberti, V. Pondenkandath, M. Würsch, R. Ingold, and M. Liwicki, "DeepDIVA: A Highly-Functional Python Framework for Reproducible

Experiments," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Aug. 2018, pp. 423–428. doi: 10.1109/ICFHR-2018.2018.00080.
[153] S. Tokui *et al.*, "Chainer: A Deep Learning Framework for Accelerating the Research Cycle," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, Jul. 2019, pp. 2002–2011. doi: 10.1145/3292500.3330756.
[154] M. Cazorla and D. Viejo, "JavaVis: An integrated computer vision library for teaching computer vision," *Comput. Appl. Eng. Educ.*, vol. 23, no. 2, pp. 258–267, 2015, doi: 10.1002/cae.21594.
[155] G. Williams, *Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery*. Springer Science & Business Media, 2011.
[156] E. Freeman, E. Robson, B. Bates, and K. Sierra, *Head First Design Patterns*. O'Reilly Media, Inc., 2008.
[157] "Stanford Artificial Intelligence Laboratory Tutorials." https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html (accessed Feb. 22, 2022).
[158] J. S. J. Lee, R. M. Haralick, and L. G. Shapiro, "Morphologic Edge Detection," *IFAC Proc. Vol.*, vol. 19, no. 9, pp. 7–14, Jun. 1986, doi: 10.1016/S1474-6670(17)57504-7.
[159] J. Lee, R. Haralick, and L. Shapiro, "Morphologic edge detection," *IEEE J. Robot. Autom.*, vol. 3, no. 2, pp. 142–156, Apr. 1987, doi: 10.1109/JRA.1987.1087088.
[160] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs," *ArXiv14127062 Cs*, Jun. 2016, Accessed: Dec. 19, 2021. [Online]. Available: http://arxiv.org/abs/1412.7062
[161] C. S. Perone, E. Calabrese, and J. Cohen-Adad, "Spinal cord gray matter segmentation using deep dilated convolutions," *Sci. Rep.*, vol. 8, no. 1, p. 5966, Apr. 2018, doi: 10.1038/s41598-018-24304-3.
[162] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018, doi: 10.1109/TPAMI.2017.2699184.
[163] R. Hamaguchi, A. Fujita, K. Nemoto, T. Imaizumi, and S. Hikosaka, "Effective Use of Dilated Convolutions for Segmenting Small Object Instances in Remote Sensing Imagery," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Mar. 2018, pp. 1442–1450. doi: 10.1109/WACV.2018.00162.
[164] M. Gridach and I. Voiculescu, "OXENDONET: A Dilated Convolutional Neural Networks For Endoscopic Artefact Segmentation," 2020.
[165] **C. Orhei**, V. Bogdan, and C. Bonchiş, "Edge map response of dilated and reconstructed classical filters," in *2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, Sep. 2020, pp. 187–194. doi: 10.1109/SYNASC51798.2020.00039, WOS:000674702000028.
[166] I. Sobel and Feldman, "A 3×3 isotropic gradient operator for image processing," Jan. 1973.

[167] J. M. S. Prewitt, "Object enhancement and extraction," *Pict. Process. Psychopictorics B Lipkin Rosenfeld Eds N. Y. Acad.*, vol. 10, no. 1, pp. 15–19, Jan. 1970.
[168] D. Ziou and S. Tabbone, "Edge Detection Techniques - An Overview," vol. Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii 8, no. 8, pp. 537–559, Jan. 1998.
[169] G. Papari and N. Petkov, "Edge and line oriented contour detection: State of the art," *Image Vis. Comput.*, vol. 29, no. 2–3, pp. 79–103, Feb. 2011, doi: 10.1016/j.imavis.2010.08.009.
[170] H. Spontón and J. Cardelino, "AReview of Classic Edge Detectors," *Image Process. Line*, vol. 5, pp. 90–123, Jun. 2015, doi: 10.5201/ipol.2015.35.
[171] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, Nov. 1986, doi: 10.1109/TPAMI.1986.4767851.
[172] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, 1st ed. USA: Addison-Wesley Longman Publishing Co., Inc., 1992.
[173] R. C. Gonzalez, R. E. Woods, and B. R. Masters, "Digital Image Processing, Third Edition," *J. Biomed. Opt.*, vol. 14, no. 2, p. 029901, 2009, doi: 10.1117/1.3115362.
[174] Frei and C.-C. Chen, "Fast Boundary Detection: A Generalization and a New Algorithm," *IEEE Trans. Comput.*, vol. C–26, no. 10, pp. 988–998, Oct. 1977, doi: 10.1109/TC.1977.1674733.
[175] R.-H. Park, "A fourier interpretation of the Frei-Chen edge masks," *Pattern Recognit. Lett.*, vol. 11, no. 9, pp. 631–636, Sep. 1990, doi: 10.1016/0167-8655(90)90016-U.
[176] R. Jain, R. Kasturi, and B. G. Schunck, *Machine vision*. New York: McGraw-Hill, 1995.
[177] D. Marr, E. Hildreth, and S. Brenner, "Theory of edge detection," *Proc. R. Soc. Lond. B Biol. Sci.*, vol. 207, no. 1167, pp. 187–217, Feb. 1980, doi: 10.1098/rspb.1980.0020.
[178] S. Castan, J. Zhao, and J. Shen, "New edge detection methods based on exponential filter," in *10th International Conference on Pattern Recognition [1990] Proceedings*, Jun. 1990, vol. i, pp. 709–711 vol.1. doi: 10.1109/ICPR.1990.118199.
[179] C. Akinlar and C. Topal, "Edlines: Real-time line segment detection by Edge Drawing (ed)," in *2011 18th IEEE International Conference on Image Processing*, Sep. 2011, pp. 2837–2840. doi: 10.1109/ICIP.2011.6116138.
[180] C. Topal and C. Akinlar, "Edge Drawing: A combined real-time edge and segment detector," *J. Vis. Commun. Image Represent.*, vol. 23, no. 6, pp. 862–872, Aug. 2012, doi: 10.1016/j.jvcir.2012.05.004.
[181] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour Detection and Hierarchical Image Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011, doi: 10.1109/TPAMI.2010.161.
[182] M. S. Prieto and A. R. Allen, "A similarity metric for edge images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1265–1273, Oct. 2003, doi: 10.1109/TPAMI.2003.1233900.
[183] Y. Sasaki, "The truth of the F-measure," *Teach Tutor Mater*, Jan. 2007.
[184] M.-P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching," in *Proceedings of 12th International Conference on Pattern Recognition*, Oct. 1994, vol. 1, pp. 566–568 vol.1. doi: 10.1109/ICPR.1994.576361.

[185] M. Baptiste, H. Abdulrahman, and P. Montesinos, "A Review of Supervised Edge Detection Evaluation Methods and an Objective Comparison of Filtering Gradient Computations Using Hysteresis Thresholds," *J. Imaging*, vol. 4, p. 74, May 2018, doi: 10.3390/jimaging4060074.
[186] I. E. Abdou and W. K. Pratt, "Quantitative design and evaluation of enhancement/thresholding edge detectors," *Proc. IEEE*, vol. 67, no. 5, pp. 753–763, May 1979, doi: 10.1109/PROC.1979.11325.
[187] N. Bandai, S. Sanada, K. Ueki, S. Funabasama, S. Tsuduki, and T. Matsui, "Morphological Analysis for Kinetic X-ray Images of the Temporomandibular Joint," *Nihon Hoshasen Gijutsu Gakkai Zasshi*, vol. 59, pp. 951–7, Sep. 2003, doi: 10.6009/jjrt.KJ00000921840.
[188] **C. Orhei**, S. Vert, and R. Vasiu, "A Novel Edge Detection Operator for Identifying Buildings in Augmented Reality Applications," in *Information and Software Technologies*, Cham, 2020, pp. 208–219. doi: 10.1007/978-3-030-59506-7_18.
[189] M. Abdul and R. Lateef, "Expansion and Implementation of a 3x3 Sobel and Prewitt Edge Detection Filter to a 5x5 Dimension Filter," Jun. 2019.
[190] S. Gupta, S. G. Mazumdar, and M. T. Student, "Sobel Edge Detection Algorithm," *Int. J. Comput. Sci. Manag. Res. 2*, vol. 2, no. 2, pp. 1578–1583, Feb. 2013.
[191] J. Chen, D. Q. Chen, and S. H. Meng, "A Novel Region Selection Algorithm for Auto-focusing Method Based on Depth from Focus," in *Proceedings of the Fourth Euro-China Conference on Intelligent Data Analysis and Applications*, Cham, 2018, pp. 101–108. doi: 10.1007/978-3-319-68527-4_11.
[192] G. (Henry) Levkine, "Prewitt, Sobel and Scharr gradient 5x5 convolution matrices." First draft,February 2011 Second Draft, June 2012., Jun. 2012.
[193] R. A. Kirsch, "Computer determination of the constituent structure of biological images," *Comput. Biomed. Res.*, vol. 4, no. 3, pp. 315–328, Jun. 1971, doi: 10.1016/0010-4809(71)90034-6.
[194] H. Scharr, "Optimal operators in digital image processing [Elektronische Ressource] /," 2014.
[195] E. Kawalec-Latała, "Edge Detection on Images of Pseudoimpedance Section Supported by Context and Adaptive Transformation Model Images," *Stud. Geotech. Mech.*, vol. 36, no. 1, pp. 29–36, Mar. 2014, doi: 10.2478/sgem-2014-0004.
[196] D.-J. Kroon, "NUMERICAL OPTIMIZATION OF KERNEL BASED IMAGE DERIVATIVES," *Short Pap. Univ. Twente*, p. 3, 2009.
[197] W. K. Pratt, *Digital image processing*. Hoboken, N.J.: Wiley-Liss, 2002.
[198] G. (Gianni) Ramponi, N. Strobel, S. Mitra, and T.-H. Yu, "Nonlinear unsharp masking methods for image contrast enhancement," *J Electron. Imaging*, vol. 5, pp. 353–366, Jul. 1996, doi: 10.1117/12.242618.
[199] A. Dogra and P. Bhalla, "Image Sharpening By Gaussian And Butterworth High Pass Filter," *Biomed. Pharmacol. J.*, vol. 7, no. 2, pp. 707–713, May 2015.
[200] C.-C. Yang, "Image enhancement by the modified high-pass filtering approach," *Optik*, vol. 120, no. 17, pp. 886–889, Nov. 2009, doi: 10.1016/j.ijleo.2008.03.016.
[201] M. R. Metwalli, A. H. Nasr, O. S. Farag Allah, and S. El-Rabaie, "Image fusion based on principal component analysis and high-pass filter," in *2009 International Conference on Computer Engineering Systems*, Dec. 2009, pp. 63–70. doi: 10.1109/ICCES.2009.5383308.

[202] G. Ramponi and A. Polesel, "A Rational Unsharp Masking Technique," *J. Electron. Imaging*, vol. 7, pp. 333–338, 1998.
[203] A. Polesel, G. (Gianni) Ramponi, and V. J. Mathews, "Image Enhancement via Adaptive Unsharp Masking," *IEEE Trans. Image Process. Publ. IEEE Signal Process. Soc.*, vol. 9, pp. 505–10, Feb. 2000, doi: 10.1109/83.826787.
[204] Z. Al-Ameen, A. Muttar, and G. Al-Badrani, "Improving the Sharpness of Digital Image Using an Amended Unsharp Mask Filter," *Int. J. Image Graph. Signal Process.*, vol. 11, pp. 1–9, Mar. 2019, doi: 10.5815/ijigsp.2019.03.01.
[205] Guang Deng, "A Generalized Unsharp Masking Algorithm," *IEEE Trans. Image Process.*, vol. 20, no. 5, pp. 1249–1261, May 2011, doi: 10.1109/TIP.2010.2092441.
[206] R. Dian, S. Li, A. Guo, and L. Fang, "Deep Hyperspectral Image Sharpening," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5345–5355, Nov. 2018, doi: 10.1109/TNNLS.2018.2798162.
[207] H. Ibrahim and N. S. Pik Kong, "Image sharpening using sub-regions histogram equalization," *IEEE Trans. Consum. Electron.*, vol. 55, no. 2, pp. 891–895, May 2009, doi: 10.1109/TCE.2009.5174471.
[208] S. H. Kim and J. P. Allebach, "Optimal unsharp mask for image sharpening and noise removal," *J. Electron. Imaging*, vol. 14, no. 2, p. 023005, Apr. 2005, doi: 10.1117/1.1924510.
[209] J. G. M. Schavemaker, M. J. T. Reinders, J. J. Gerbrands, and E. Backer, "Image sharpening by morphological filtering," *Pattern Recognit.*, vol. 33, no. 6, pp. 997–1012, Jun. 2000, doi: 10.1016/S0031-3203(99)00160-0.
[210] I. Papamarkou, N. Papamarkos, and S. Theochari, "A novel image sharpening technique based on 2D-DWT and image fusion," in *17th International Conference on Information Fusion (FUSION)*, 2014, pp. 1--8.
[211] K. De and V. Masilamani, "Image Sharpness Measure for Blurred Images in Frequency Domain," *Procedia Eng.*, vol. 64, pp. 149–158, 2013, doi: 10.1016/j.proeng.2013.09.086.
[212] J. P. de Villiers, "A comparison of image sharpness metrics and real-time sharpening methods with GPU implementations," in *Proceedings of the 7th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, New York, NY, USA, Jun. 2010, pp. 53–62. doi: 10.1145/1811158.1811168.
[213] J. P. D. Villiers, "A comparison of image sharpness metrics and real-time sharpening methods with GPU implementations," 2010.
[214] A. M. Eskicioglu and P. S. Fisher, "Image quality measures and their performance," *IEEE Trans. Commun.*, vol. 43, no. 12, pp. 2959–2965, Dec. 1995, doi: 10.1109/26.477498.
[215] E. Peli, "Contrast in complex images," *J. Opt. Soc. Am. A*, vol. 7, no. 10, p. 2032, Oct. 1990, doi: 10.1364/JOSAA.7.002032.
[216] A. K. Tripathi, S. Mukhopadhyay, and A. K. Dhara, "Performance metrics for image contrast," in *2011 International Conference on Image Information Processing*, Nov. 2011, pp. 1–4. doi: 10.1109/ICIIP.2011.6108900.
[217] **C. Orhei**, *TMBuD Dataset*. 2022. Accessed: Feb. 22, 2022. [Online]. Available: https://github.com/CipiOrhei/TMBuD
[218] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.

[219] M. Sonka, V. Hlavac, and R. Boyle, "Image pre-processing," in *Image Processing, Analysis and Machine Vision*, M. Sonka, V. Hlavac, and R. Boyle, Eds. Boston, MA: Springer US, 1993, pp. 56–111. doi: 10.1007/978-1-4899-3216-7_4.
[220] W. Kropatsch, Y. Haxhimusa, and A. Ion, "APPLIED GRAPH THEORY IN COMPUTER VISION AND PATTERN RECOGNITION," 2006. https://www.semanticscholar.org/paper/APPLIED-GRAPH-THEORY-IN-COMPUTER-VISION-AND-PATTERN-Kropatsch-Haxhimusa/360df009ac49b231c7089f12d040a9314a08de8e (accessed Jan. 14, 2022).
[221] E. Adelson, C. Anderson, J. Bergen, P. Burt, and J. Ogden, "Pyramid Methods in Image Processing," *RCA Eng*, vol. 29, Nov. 1983.
[222] M. Thoma, "A Survey of Semantic Segmentation," *ArXiv160206541 Cs*, May 2016, Accessed: Jan. 18, 2022. [Online]. Available: http://arxiv.org/abs/1602.06541
[223] A. Milan *et al.*, "Semantic Segmentation from Limited Training Data," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1908–1915. doi: 10.1109/ICRA.2018.8461082.
[224] F. Lateef and Y. Ruichek, "Survey on semantic segmentation using deep learning techniques," *Neurocomputing*, vol. 338, pp. 321–348, Apr. 2019, doi: 10.1016/j.neucom.2019.02.003.
[225] L. Wang, X. Chen, L. Hu, and H. Li, "Overview of Image Semantic Segmentation Technology," in *2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, Dec. 2020, vol. 9, pp. 19–26. doi: 10.1109/ITAIC49862.2020.9338770.
[226] D. Feng *et al.*, "Deep Multi-modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1341–1360, Mar. 2021, doi: 10.1109/TITS.2020.2972974.
[227] "A Gentle Introduction to Image Segmentation for Machine Learning." https://www.v7labs.com/blog/image-segmentation-guide, https://www.v7labs.com/blog/image-segmentation-guide (accessed Feb. 22, 2022).
[228] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," 2016, pp. 770–778. Accessed: Jan. 18, 2022. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html
[229] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017, doi: 10.1109/TPAMI.2016.2644615.
[230] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes Challenge: A Retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jan. 2015, doi: 10.1007/s11263-014-0733-5.
[231] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013, doi: 10.1177/0278364913491297.
[232] M. Cordts *et al.*, "The Cityscapes Dataset for Semantic Urban Scene Understanding," 2016, pp. 3213–3223. Accessed: Jan. 18, 2022. [Online].

Available: https://openaccess.thecvf.com/content_cvpr_2016/html/Cordts_The_Cityscapes_Dataset_CVPR_2016_paper.html

[233] S. Sharma, S. Sharma, U. Scholar, and A. Athaiya, "□ACTIVATION FUNCTIONS IN NEURAL NETWORKS," vol. 4, no. 12, p. 7, 2020.

[234] Q. Ji, J. Huang, W. He, and Y. Sun, "Optimized Deep Convolutional Neural Networks for Identification of Macular Diseases from Optical Coherence Tomography Images," *Algorithms*, vol. 12, no. 3, Art. no. 3, Mar. 2019, doi: 10.3390/a12030051.

[235] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," 2015, pp. 3431–3440. Accessed: Jan. 19, 2022. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2015/html/Long_Fully_Convolutional_Networks_2015_CVPR_paper.html

[236] O. Andersson and S. R. Marquez, "A comparison of object detection algorithms using unmanipulated testing images: Comparing SIFT, KAZE, AKAZE and ORB," p. 31, 2016.

[237] S. A. K. Tareen and Z. Saleem, "A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK," in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, Mar. 2018, pp. 1–10. doi: 10.1109/ICOMET.2018.8346440.

[238] M. Hassaballah, A. A. Abdelmgeid, and H. A. Alshazly, "Image Features Detection, Description and Matching," in *Image Feature Detectors and Descriptors : Foundations and Applications*, A. I. Awad and M. Hassaballah, Eds. Cham: Springer International Publishing, 2016, pp. 11–45. doi: 10.1007/978-3-319-28854-3_2.

[239] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, "KAZE Features," in *Computer Vision – ECCV 2012*, Berlin, Heidelberg, 2012, pp. 214–227. doi: 10.1007/978-3-642-33783-3_16.

[240] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2564–2571. doi: 10.1109/ICCV.2011.6126544.

[241] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust invariant scalable keypoints," in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2548–2555. doi: 10.1109/ICCV.2011.6126542.

[242] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 7, pp. 629–639, Jul. 1990, doi: 10.1109/34.56205.

[243] J. Weickert, "Efficient image segmentation using partial differential equations and morphology," *Pattern Recognit.*, vol. 34, no. 9, pp. 1813–1824, Sep. 2001, doi: 10.1016/S0031-3203(00)00109-6.

[244] J. Weickert, "Applications of nonlinear diffusion in image processing and computer vision," Working paper, Feb. 2008. Accessed: Jan. 11, 2022. [Online]. Available: http://ub-madoc.bib.uni-mannheim.de/1845

[245] T. Lindeberg, "Feature Detection with Automatic Scale Selection," *Int. J. Comput. Vis.*, vol. 30, no. 2, pp. 79–116, Nov. 1998, doi: 10.1023/A:1008045108935.

[246] M. Brown and D. Lowe, "Invariant Features from Interest Point Groups," in *In British Machine Vision Conference*, 2002, pp. 656–665.

[247] M. Agrawal, K. Konolige, and M. R. Blas, "CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching," in *Computer Vision – ECCV*

2008, Berlin, Heidelberg, 2008, pp. 102–115. doi: 10.1007/978-3-540-88693-8_8.

[248] J. Weickert, S. Grewenig, C. Schroers, and A. Bruhn, "Cyclic Schemes for PDE-Based Image Analysis," *Int. J. Comput. Vis.*, vol. 118, no. 3, pp. 275–299, Jul. 2016, doi: 10.1007/s11263-015-0874-1.

[249] S. Grewenig, J. Weickert, and A. Bruhn, "From Box Filtering to Fast Explicit Diffusion," in *Pattern Recognition*, Berlin, Heidelberg, 2010, pp. 533–542. doi: 10.1007/978-3-642-15986-2_54.

[250] X. Yang and K.-T. Cheng, "LDB: An ultra-fast feature for scalable Augmented Reality on mobile devices," in *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Nov. 2012, pp. 49–57. doi: 10.1109/ISMAR.2012.6402537.

[251] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua, "BRIEF: Computing a Local Binary Descriptor Very Fast," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1281–1298, Jul. 2012, doi: 10.1109/TPAMI.2011.222.

[252] Y. Ou, Z. Cai, J. Lu, J. Dong, and Y. Ling, *Evaluation of Image Feature Detection and Matching Algorithms*. 2020, p. 224. doi: 10.1109/ICCCS49078.2020.9118480.

[253] H. Seong, H. Choi, H. Son, and C. Kim, "Image-based 3D Building Reconstruction Using A-KAZE Feature Extraction Algorithm," presented at the 34th International Symposium on Automation and Robotics in Construction, Taipei, Taiwan, Jul. 2018. doi: 10.22260/ISARC2018/0127.

[254] D. Bojanić, K. Bartol, T. Pribanić, T. Petković, Y. D. Donoso, and J. S. Mas, "On the Comparison of Classic and Deep Keypoint Detector and Descriptor Methods," in *2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA)*, Sep. 2019, pp. 64–69. doi: 10.1109/ISPA.2019.8868792.

[255] K. Bartol, D. Bojanić, T. Pribanić, T. Petković, Y. D. Donoso, and J. S. Mas, "On the Comparison of Classic and Deep Keypoint Detector and Descriptor Methods," *2019 11th Int. Symp. Image Signal Process. Anal. ISPA*, pp. 64–69, Sep. 2019, doi: 10.1109/ISPA.2019.8868792.

[256] O. Yakovleva and K. Nikolaieva, "Research of descriptor based image normalization and comparative analysis of SURF, SIFT, BRISK, ORB, KAZE, AKAZE," *Natl. Tech. Univ. Kharkiv Polytech. Inst.*, vol. 4, no. 4, pp. 89–101, 2020, doi: doi.org/10.20998/2522-9052.2020.4.13.

[257] A. Moghimi, T. Celik, A. Mohammadzadeh, and H. Kusetogullari, "Comparison of Keypoint Detectors and Descriptors for Relative Radiometric Normalization of Bitemporal Remote Sensing Images," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 14, pp. 4063–4073, 2021, doi: 10.1109/JSTARS.2021.3069919.

[258] T. Joachims, "Text categorization with Support Vector Machines: Learning with many relevant features," in *Machine Learning: ECML-98*, Berlin, Heidelberg, 1998, pp. 137–142. doi: 10.1007/BFb0026683.

[259] H. Lodhi, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text Classification using String Kernels," *J. Mach. Learn. Res.*, vol. 2, pp. 419--444, 2002.

[260] A. Ramanan and M. Niranjan, "A Review of Codebook Models in Patch-Based Visual Object Recognition," *J. Signal Process. Syst.*, vol. 68, no. 3, pp. 333–352, Sep. 2012, doi: 10.1007/s11265-011-0622-x.

[261] T. Matsukawa, K. Suzuki, and T. Kurita, "Preliminary Local Feature Selection by Support Vector Machine for Bag of Features," p. 6, 2009.

[262] C.-F. Tsai, "Bag-of-Words Representation in Image Annotation: A Review," *ISRN Artif. Intell.*, vol. 2012, pp. 1–19, Nov. 2012, doi: 10.5402/2012/376804.
[263] K. Mikolajczyk, B. Leibe, and B. Schiele, "Multiple Object Class Detection with a Generative Model," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Jun. 2006, vol. 1, pp. 26–36. doi: 10.1109/CVPR.2006.202.
[264] S. O'Hara and B. A. Draper, "Introduction to the Bag of Features Paradigm for Image Classification and Retrieval," *ArXiv11013354 Cs*, Jan. 2011, Accessed: Jan. 20, 2022. [Online]. Available: http://arxiv.org/abs/1101.3354
[265] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time," *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 209–226, Sep. 1977, doi: 10.1145/355744.355745.
[266] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *J. ACM*, vol. 45, no. 6, pp. 891–923, Nov. 1998, doi: 10.1145/293347.293348.
[267] K. Fukunaga and P. M. Narendra, "A Branch and Bound Algorithm for Computing k-Nearest Neighbors," *IEEE Trans. Comput.*, vol. C–24, no. 7, pp. 750–753, Jul. 1975, doi: 10.1109/T-C.1975.224297.
[268] A. Andoni and P. Indyk, "Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions," in *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, Oct. 2006, pp. 459–468. doi: 10.1109/FOCS.2006.49.
[269] R. Weber, H.-J. Schek, and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces," in *VLDB*, 1998, vol. 98, pp. 194--205.
[270] J. Luo, D. Joshi, J. Yu, and A. Gallagher, "Geotagging in multimedia and computer vision—a survey," *Multimed. Tools Appl.*, vol. 51, no. 1, pp. 187–211, Jan. 2011, doi: 10.1007/s11042-010-0623-y.
[271] E. Amitay, N. Har'El, R. Sivan, and A. Soffer, "Web-a-where: geotagging web content," in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, Jul. 2004, pp. 273–280. doi: 10.1145/1008992.1009040.
[272] C. F. F. Karney, "Algorithms for geodesics," *J. Geod.*, vol. 87, no. 1, pp. 43–55, Jan. 2013, doi: 10.1007/s00190-012-0578-z.
[273] **C. Orhei**, M. Mocofan, S. Vert, and R. Vasiu, "An automated threshold Edge Drawing algorithm," in *2021 44th International Conference on Telecommunications and Signal Processing (TSP)*, Jul. 2021, pp. 292–295. doi: 10.1109/TSP52935.2021.9522661, WOS:000701604600063.
[274] **C. Orhei**, M. Mocofan, S. Vert, and R. Vasiu, "An Analysis of ED Line Algorithm in Urban Street-View Dataset," in *Information and Software Technologies*, Cham, 2021, pp. 123–135. doi: 10.1007/978-3-030-88304-1_10.
[275] C. L. Sirbu, C. Tomoiu, S. Fancsali-Boldizsar, and **C. Orhei**, "Real-time line matching based speed bump detection algorithm," in *2021 IEEE 27th International Symposium for Design and Technology in Electronic Packaging (SIITME)*, Oct. 2021, pp. 246–249. doi: 10.1109/SIITME53254.2021.9663602.
[276] T. Wu, S. Tang, R. Zhang, J. Cao, and Y. Zhang, "CGNet: A Light-Weight Context Guided Network for Semantic Segmentation," *IEEE Trans. Image Process.*, vol. 30, pp. 1169–1179, 2021, doi: 10.1109/TIP.2020.3042065.
[277] E. Romera, J. M. Álvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation," *IEEE*

Trans. Intell. Transp. Syst., vol. 19, no. 1, pp. 263–272, Jan. 2018, doi: 10.1109/TITS.2017.2750080.

[278] M. Siam, M. Gamal, M. Abdel-Razek, S. Yogamani, and M. Jagersand, "RTSeg: Real-Time Semantic Segmentation Comparative Study," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, Oct. 2018, pp. 1603–1607. doi: 10.1109/ICIP.2018.8451495.

[279] Y. Sugimoto and M. Aono, "Semantic Segmentation based on Extended MobileNet with FPN," in *2021 8th International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, Sep. 2021, pp. 1–6. doi: 10.1109/ICAICTA53211.2021.9640256.

[280] S. Takahama *et al.*, "Multi-Stage Pathological Image Classification Using Semantic Segmentation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), Oct. 2019, pp. 10701–10710. doi: 10.1109/ICCV.2019.01080.

[281] W.-H. Yeo, Y.-J. Heo, Y.-J. Choi, and B.-G. Kim, "Place Classification Algorithm Based on Semantic Segmented Objects," *Appl. Sci.*, vol. 10, no. 24, Art. no. 24, Jan. 2020, doi: 10.3390/app10249069.