



Universidad Autónoma de Querétaro
Facultad de Ingeniería
División de Investigación y Posgrado

FACADE LABELLING FOR BETTER AR REGISTRATION
ETIQUETADO DE LA FACHADA PARA MEJORAR EL
SEGUIMIENTO DE LA REALIDAD AUMENTADA

TESIS

Que como parte de los requisitos para obtener
el Doctorado de Ingeniería

Presenta:

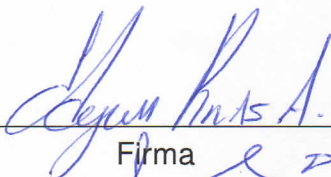
Andrés Takács

Dirigido por:

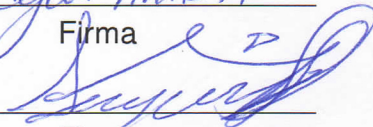
Dr. Edgar Alejandro Rivas Araiza

SINODALES


Dr. Edgar Alejandro Rivas Araiza
Presidente


Firma

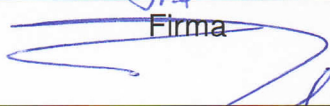
Dr. Aurelio Domínguez González
Secretario


Firma

Dr. Jesús Carlos Pedraza Ortega
Vocal

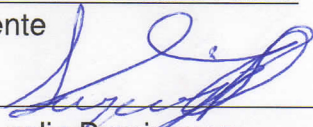

Firma

Dr. Manuel Toledano Ayala
Suplente


Firma

Dr. Alberto Pastrana Palma
Suplente


Firma


Dr. Aurelio Domínguez
Director de la Facultad de Ingeniería


Dra. Ma. Guadalupe Flavia Loarca Piña
Directora de Investigación y Posgrado

Centro Universitario
Querétaro, Qro.
Enero 2017
México

Abstract

The demand for Augmented Reality or other self-localizing computer vision applications is increasing. This implies an expansion also in the variety of usage locations which calls for higher standard visual descriptors that can be used in outdoor environments. The characteristics of these sceneries require more robustness and invariance additionally of repeatability from these visual features. The descriptors need to be invariant to light condition, and transformation changes likewise to give support for efficient classification. This research demonstrates that the use of specifically designed visual descriptors is an effective way to optimize and make more robust the outdoor Augmented Reality applications.

To achieve this, this work proposes a new self-adjusting framework based on genetic algorithms and a machine learning module to create and optimize extensible modular descriptors for specific outdoor environments. The algorithm generates descriptors, improves their efficiency and trains them for classification. It controls the image preparation and machine learning parameters and also optimizes the descriptor size through managing the active member modules and values. To show the strength of the descriptor, we compared with the most used standard descriptors—on speed, accuracy, and invariance to light condition, image resolution changes, affine transformation, scale and rotation—and the results show that it has the best classification results.

The final part of this work integrates the newly designed descriptor with a trained Random Forest module into the Parallel Tracking And Mapping system. This approach is able to classify into semantic groups and remove the unnecessary the detected keypoints in real time.

(Key words: computer vision, image processing, augmented reality, visual descriptors, machine learning, random forest, genetic algorithm)

Resumen

La demanda de aplicaciones de Realidad Aumentada y de otras aplicaciones de auto-localización por medio de visión por computadora ha aumentado significativamente en años recientes, esto implica al mismo tiempo una expansión en la variedad de lugares donde pueden emplearse, lo cual demanda mayores estándares de descriptores visuales que puedan usarse en ambientes exteriores. Las características de estos escenarios requieren mayor robustez e invariancia adicionalmente a la repetibilidad de las características visuales, los descriptores necesitan ser invariantes a las condiciones de iluminación y a cambios de transformación para dar soporte a una clasificación eficiente. En esta investigación se demuestra el uso de descriptores visuales específicamente diseñados son un medio efectivo para optimizar e incrementar la robustez de las aplicaciones de realidad aumentada en exteriores. Para lograrlo, en este trabajo se propone un nuevo método de auto-ajuste basado en el uso de algoritmos genéticos y un módulo de máquina de aprendizaje para crear y optimizar un descriptor modular extensible para un ambiente exterior específico. El algoritmo genera descriptores, mejora su eficiencia y los entrena para tareas de clasificación, el algoritmo controla la preparación de la imagen y los parámetros de la máquina de aprendizaje y al mismo tiempo optimiza el tamaño del descriptor a través de los miembros de los módulos activos y sus valores . Para demostrar el desempeño del descriptor, se compara con los descriptores estándares más conocidos en términos de: velocidad, exactitud, invariancia a las condiciones de iluminación, cambios de resolución de la imagen, transformaciones afines, escala y rotación, y el resultado muestra que el método propuesto obtiene los mejores resultados en la clasificación.

En la parte final del presente trabajo se integra en nuevo descriptor desarrollado con un modulo de bosques aleatorios en un sistema de seguimiento y mapeo paralelo (PTAM), de esta manera se es capaz de clasificar en grupos semánticos y remover los puntos clave detectados en tiempo real.

(Palbras claves: vision artificial, procesamiento de imágenes, realidad aumentada, descriptores, aprendizaje de máquinas, bosque aleatorio, algoritmo genético)

To Amanda for her patience and tireless support.

Acknowledgement

First and foremost I would like to thank my supervisor, Dr. Edgar Rivas Ariza, for his tireless support and endless proof-reading. Thank you to the Synode members for their help and observations.

My research was funded by the Consejo Nacional de Ciencia y Tecnología (grant 340519), and I am grateful to Universidad Autónoma de Quértaro for its facilities and support.

Throughout this work I have enjoyed the unfailing support of my family, and in particular from Amanda who has provided continual patience, support and encouragement.

Table of Contents

	Page
Abstract.....	i
Resumen	ii
Acknowledgement.....	iv
Table of Contents	iv
List of Tables	ix
List of Figures	xi
List of Abbreviations	xviii
I. INTRODUCTION	1
I.1 Introduction	1
I.1.1 Augmented Reality	1
I.1.2 Simultaneous Localization And Mapping	3
I.1.3 Descriptors	4
I.1.4 Machine Learning	5
I.1.5 Genetic Algorithm	6
I.2 Problem Description	7
I.3 Justification	7
I.4 Hypothesis	8
I.5 Objectives	8

I.5.1	General Objective	8
I.5.2	Particular Objectives	8
I.6	Layout of this Thesis	8
II.	LITERATURE REVIEW	10
II.1	Introduction	10
II.2	Research Groups	10
II.3	Augmented Reality	11
II.3.1	History	14
II.3.2	Display	16
II.3.3	AR Systems	19
II.3.4	Tracking	20
II.4	Simultaneous Localization And Mapping	24
II.5	Descriptors	28
II.5.1	Size Reduction	30
II.5.2	Invariance	31
II.6	Random Forest	33
II.7	Genetic Algorithm	34
III.	THEORETICAL FRAMEWORK	36
III.1	Introduction	36
III.2	Augmented Reality	36
III.2.1	Tracking	40
III.2.2	Registration	46

III.3	Simultaneous Localization And Mapping	46
III.3.1	Parallel Tracking And Mapping	46
III.4	Mathematics behind the application	50
III.4.1	Image Processing	51
III.4.2	Invariant features	60
III.4.3	Light condition modelling	64
III.4.4	Distance Transform	65
III.4.5	Distance metrics	65
III.4.6	Descriptor comparison techniques	66
III.4.7	Descriptor performance evaluation	67
III.4.8	Camera base tracking parameters	69
III.5	FAST: Features from Accelerated Segment Test	75
III.6	Descriptor extraction	77
III.6.1	Scale Invariant Feature Transform (SIFT)	77
III.6.2	Speeded-Up Robust Features (SURF)	80
III.6.3	Opponent SIFT	80
III.6.4	Opponent SURF	80
III.7	Random Forest	81
III.8	Genetic Algorithm	82
III.8.1	Initial Population	82
III.8.2	Evaluation	83
III.8.3	Genetic Operators	83

III.8.4	Fitness function	86
IV.	METHODOLOGY	87
IV.1	Introduction	87
IV.2	System's characteristics	87
IV.2.1	Datasets	87
IV.2.2	Libraries and equipment	89
IV.3	Phase 1: Environment Dedicated Descriptor (EDD)	89
IV.3.1	EDD setup	89
IV.3.2	Evaluation process	90
IV.4	Phase 2: Genetic Optimization framework	91
IV.4.1	Preprocessing stage	92
IV.4.2	Image Processing	94
IV.4.3	Module bank and activation	94
IV.4.4	Objective function	97
IV.5	Evaluation and Analysis	98
IV.6	Fitting of the algorithm into PTAM	99
V.	RESULTS & DISCUSSION.	101
V.1	Results	101
V.1.1	EDD results	101
V.1.2	Modular Descriptor	103
V.1.3	PTAM implementation	114
V.2	Discussion	115

Bibliography 117
APPENDIX 134

List of Tables

Tables	Page
2.1 Characteristics of visual AR displays	18
2.2 Summary of local float type feature descriptors.	29
2.3 Summary of local bit type feature descriptors.	30
2.4 Variations to which each color-invariant feature is invariant	31
2.5 Definition of labels for the color-invariant features (CIF) used in Table 2.4	32
3.1 Comparison of common tracking technologies. Range: size of the region that can be tracked within. Setup: amount of time for instru- mentation and calibration. Precision: granularity of a single output position. Time: duration for which useful tracking data is returned (be- fore it drifts too much). Environment: where the tracker can be used, indoors or outdoors	40
5.1 EDD evaluation results - Average true positives at the Brighton scene.(%)	102
5.2 EDD evaluation results - Average true positives with 100 images.(%) .	102
5.3 Optimized module genomes.	104
5.4 Optimized module genomes.	105
5.5 Optimized module values.	105
5.6 Modular descriptor - Average true positives at the Brighton scene.(%)	107
5.7 Modular descriptor "Else" class - Classification outcomes and Meas- urements.	110

5.8 Modular descriptor "Doors & Windows" class - Classification outcomes and Measurements.	111
5.9 Modular descriptor "Roof" class - Classification outcomes and Measurements.	112
5.10 Modular descriptor "Wall" class - Classification outcomes and Measurements.	113

List of Figures

Figures	Page
1.1 Simplified representation of Virtual Reality continuum	1
2.1 Human Computer Interaction styles	13
2.2 Source of system delay and possible solution	14
2.3 Sutherland's sword of Damocles	15
2.4 Touring Machine	16
2.5 ARToolKit	16
2.6 Display Taxonomy	17
3.1 Schemes based on the real world and camera relation: (a) World fixed AR, (b) Camera fixed AR	37
3.2 Display locations: (a) Head attached display, (b) Body attached display, (c) Spatial display	38
3.3 HMD schemes: (a) The Video see-through, (b) Optical see-through	38
3.4 The Head Mounted Projective Display (HMPD) scheme	39
3.5 Vision only tracking system overview	43
3.6 Distributed Tracking System overview	45
3.7 Tracked point overlaid on image	48
3.8 Initialization	49
3.9 Map points	49

3.10 Adaptive Height-Modified Histogram Equalization	56
3.11 Light changes: (a) original image, (b) intensity change ($a=3$), (c) light color change (3800K), (d) color shift ($\sigma_1=100$)	65
3.12 Results of the Canny edge detection and the distance transfer	66
3.13 Confusion Matrix	67
3.14 Confusion Matrix 2.	68
3.15 Pinhole camera model	69
3.16 Radial distortion cases	71
3.17 FAST Feature detection in an image patch	75
3.18 Gaussian Pyramid	78
3.19 Detection of maxima and minima	78
3.20 Keypoint descriptor creation from gradient magnitude and orientation .	79
3.21 Left images: Second order Gaussian partial derivative($y - (L_{yy})$ and xy -direction (L_{xy})); Right images: Approximation for the second order ($y - (D_{yy})$ and xy -direction (D_{xy}))	80
3.22 Random Forest in function	81
3.23 Stochastic Universal Sampling	84
3.24 Single Point Crossover	85
3.25 Two Point Crossover	85
3.26 Uniform Crossover	85
4.1 Image Datasets: (a)-(c) Oxford, (d) TILDE, (e) Brighton	88
4.2 Descriptor extraction	89
4.3 Modular Descriptor Extraction Process	92

4.4	Modular Descriptor Extraction Process	95
4.5	Average Gradient angles on the patch and sub-patches	96
4.6	Affine transformation cases in invariance evaluation: (a)-(j) Case 1-10	98
4.7	The parallel mapping and tracking thread	100
5.1	EDD results: (a) keypoint classification result, (b) segmentation result	102
5.2	EDD - Photometric Analysis results: (a) Intensity shift (with offset o_1), (b) Intensity scaling (with scalar a), (c) Illumination Color temperature (K)	103
5.3	Descriptor performance in time: (a) average object segmentation time, (b) average descriptor extraction time	104
5.4	EDD - Transformation Invariance results: (a) Gaussian blur (kernel size), (b) Image rotation (angle), (c) Image resize (size), (d) Affine transformation (cases)	106
5.5	Modular descriptor - Transformation Invariance results: (a) Gaussian blur (kernel size), (b) Image rotation (angle), (c) Image resize (size), (d) Affine transformation (cases)	107
5.6	Modular descriptor - Photometric Analysis results: (a) JPEG com- pression (cases), (b) Light change (cases), (c) Light condition change (Notre Dame), (d) Light condition change 2 (Mexico)	108
5.7	Classification Results for each Descriptor: (a) Base image, (b) Ground truth, (c) SIFT, (d) Opponent SIFT, (e) SURF, (f) Opponent SURF, (g) EDD, (h) Modular	109
5.8	"Else" class classification results and Measurements	110
5.9	"Doors & Windows" class classification results and Measurements	111
5.10	"Roof" class classification results and Measurements	112
5.11	"Wall" class classification results and Measurements	113

5.12 Original PTAM environment 114

5.13 PTAM with the classified interest points and parallel map 114

List of Abbreviations

π -SIFT	Photometric quasi-Invariant SIFT
A-GPS	Assisted Global Positioning System
ACC	Accuracy
APs	Access Points
AR	Augmented Reality
ASIFT	Affine-SIFT
AV	Augmented Virtuality
BF	Brute Force
BoW	Bag-Of-Words
BRIEF	Binary Robust Independent Elementary Features
CIE	Commission Internationale de l'Éclairage
CIF	Color-Invariant Features
CIF	Color-Invariant Features
CSIFT	Colored SIFT
CV	Computer Vision
DIRD	Dird is an Illumination Robust Descriptor
DOF	Degrees Of Freedom
EDD	Environment Dedicated Descriptor
FAST	Features from Accelerated Segment Test
FLANN	Fast Library for Approximate Nearest Neighbors
FN	False Negative

FOV	Field Of View
FP	False Positive
GA	Genetic Algorithm
GLOH	Gradient Location and Orientation Histogram
GPS	Global Positioning System
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HCI	Human-Computer Interaction
HMD	Head Mounted Display
iOS	iPhone OS
LIOP	Local Intensity Order Pattern
LSD-SLAM	Large-Scale Direct Monocular-SLAM
MAR	Mobile Augmented Reality
ML	Machine Learning
MR	Mixed Reality
NPV	Negative Prediction Value
oob	out-of-bag
ORB-SLAM	Oriented FAST and Rotated BRIEF-SLAM
OSID	Ordinal Spatial Intensity Distribution
PC	Personal Computer
PCA-SIFT	Principal Components Analysis SIFT
PPV	Precision or Positive Predictive Value
PTAM	Parallel Tracking And Mapping
PTAMM	Parallel Tracking and Multiple Mapping

SDK	Software Development Kit
SeqSLAM	Sequences of Places SLAM
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization And Mapping
SMD	Locally Stable Monotonic Change Invariant Feature Descriptor
SPC	Specificity or True Negative Rate
SURF	Speeded-Up Robust Features
TN	True Negative
TP	True Positive
TPR	Recall or True Positive Rate
USD	United States Dollar
USURF	Upright Speeded Up Robust Features
VR	Virtual Reality

I. INTRODUCTION

I.1. Introduction

I.1.1. Augmented Reality

Augmented Reality (AR) is part of a bigger conceptual group called Mixed Reality (MR) (Figure 1.1). MR synthesizes real scenery with synthetic computer generated data. The basic goal of the MR is to complement the user's understanding of and interaction with the physical world adding 3D virtual objects to the real environment (Azuma *et al.*, 2001). We can find two concepts on the opposite edges of the MR continuum: Real world and Virtual Reality (VR). The real world is the physical domain, which exists without any virtual item. The VR is a scenario where all the stimuli are computer generated and mediated via a different type of user interfaces like Head Mounted Displays (HMD), haptic devices and 3D pointing devices among others (Milgram *et al.*, 1994). This field between the antipodes covers various concepts categorized by the level of digital data presented. Augmented Virtuality (AV) is a primarily digital realm mixed with imagery from the real world, and AR is mainly real world scenery overlaid by virtual representation.

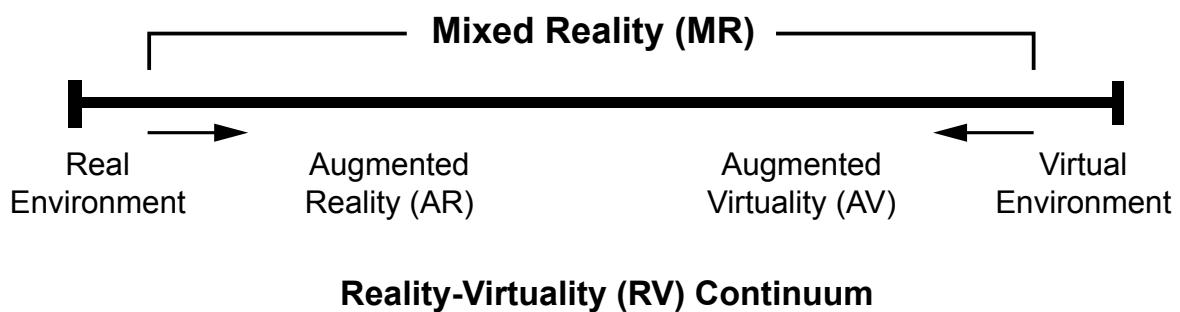


Figure 1.1: Simplified representation of Virtual Reality continuum
Source: Milgram *et al.*, 1994

The possibilities of the use of AR is continuously growing due to the latest advancements in the hardware development and computer vision research. AR have been successfully used in medicine for data visualization(Macedo *et al.*, 2014), in

the industry for aircraft maintenance (De Crescenzo *et al.*, 2011), in culture for archaeological sites (Vlahakis *et al.*, 2002) and AR museum guides (Miyashita *et al.*, 2008), in education as augmented chemistry workbench (Fjeld and Voegtli, 2002), in entertainment as haunted book (Scherrer *et al.*, 2008) and in defence for pilot to provide optimal flight information (Frantis, 2012). According to Weng *et al.* (2012) the "AR technology can be more widely used if its display device and registration method are carefully chosen, and specifically tailored".

AR typically means computer generated graphics overlaid the user's field-of-view to provide extra information about the surroundings to give guidance to complete a task. A basic AR application overlays key digital highlights, texts or pointers over the real scenery to the user, for example, navigation systems in unknown environments. More advanced AR systems can render and align complex 3D objects and animations in such a way that it appears as part of the environment, for example, Magic Leap™ or HoloLens™. It is possible to create x-ray vision, revealing objects or information which hidden from the user's view (e.g., surgery, industrial maintenance). This information can be previously designed (e.g., construction plans to show hidden cables) and created or acquired live (e.g., ultrasound in medicine). The virtual objects can represent "real" items (e.g., virtual chair), or can be "imaginary" (e.g., video games).

The AR systems have the following characteristics in common (Azuma *et al.*, 2001):

1. Combine the real and digital objects in a real environment
2. Run interactively and in real time
3. Register or align the real and virtual objects with one another in 3D

To achieve flawless registration, the system has to track accurately the surrounding environment. During the tracking, the AR systems specify their position in 3D environments from magnetic sensory information (gyroscope, Global Positioning System (GPS), accelerometer or compass) or analysing visual information from the video stream. The registration is crucial for the accurate alignment of real and virtual objects. "Without accurate registration, the illusion that the virtual objects exist in the real environment is severely compromised" (*ibid.*). The system fails to align the 3D objects without flaws, if the tracking framework is not robust enough.

As previously mentioned multiple tracking systems exist: infra-red, mechanical, inertial, ultrasound, vision-based and hybrid systems. The used Tracking technology is adopted to the different operation conditions with the final goal to provide high accuracy, low latency, low jitter and robustness (Hincapie *et al.*, 2011). For indoor environments, mainly vision-based systems are used combined with visual markers in the interest area. The tracker calculates its position and rotation angles through detecting the edges, corners and distinctive features of the marker's image. The visual tracking in outdoor scenery—where this investigation is heading—becomes a challenging task for various reasons. Filling the landscape with markers is impractical and unfeasible, the rapidly changing light conditions make hardly recognizable an already known environment, the lack of reference points (edges and corners) in sparse areas (e.g., plain walls) or the highly repetitive information (e.g., tree leaves or windows) confuses the device. Different techniques were developed to tackle these issues. During the first years of the last decade, mainly magnetic sensors (accelerometer, GPS, gyroscope or compass) defined the devices position (Azuma *et al.*, 2001). The same decade brought the development in the central processing units (CPU). That made possible in mobile devices to do tracking using the device's camera feed. The camera see-through AR applications started to exploit the visual information of the video stream using the advancements of robotics.

1.1.2. Simultaneous Localization And Mapping

The Simultaneous Localization and Mapping (SLAM) technique was developed in robotic to resolve the errors caused by the fallible sensors and actuators. The SLAM algorithms localize the devices' position based on the map formed by observing the surrounding environment. "The SLAM problem is hard due to its "chicken-egg" nature: robot should use constructed map to localize itself simultaneously with pose usage to update the map" (Huletski *et al.*, 2015). Over the many SLAM application has been developed primarily for AR. The most successful among these is the Parallel Tracking And Mapping (PTAM) by Klein and Murray (2007), which inspired many AR developments like PTAMM (Castle and Murray, 2009) PTAM for mobile devices, with Random Forest (Guan and Wang, 2009) and collaborative PTAM (Verbelen *et al.*, 2012), although these applications still use greyscale images for tracking. The recent SLAM developments like the LSD-SLAM (Engel *et al.*, 2014) and the ORB-SLAM (Mur-Artal *et al.*, 2015) use monocular cameras capturing color images. This advancement gives the potential for AR applications to exploit color

information from the environments with their cameras. As the calculation cost is high for outdoor AR, the applications—which localize their position from panoramic imagery—using cloud-based calculation (Wagner *et al.*, 2010; Ventura and Hollerer, 2011). To reduce the calculation cost is to create and optimize the feature descriptor algorithms.

I.1.3. Descriptors

During image detection and classification, we have to describe its properties in a unique manner. The descriptors must have invariant, reproducible and unique properties and ought to be calculated in an efficient way. An object can be classified into a category only if its descriptor is similar to the ones in the class. For that reason the descriptor has to have the following properties: repeatability, the resulted descriptors over a keypoint of the same object from two different images has to be identical; Invariance to scale, rotation, affine transformation, and light condition changes; Compact in size for optimum storing and calculation.

Many surveys (Mikolajczyk and Schmid, 2005; Li and Allinson, 2008; Gauglitz *et al.*, 2011; Miksik and Mikolajczyk, 2012; Huang *et al.*, 2013; Garcia-Fidalgo and Ortiz, 2015) were published during the past years to categorize and compare the techniques of low level image description. All these works concluded that there had not been developed yet a descriptor that works flawlessly in all environments and invariant to all type of image transformations. In the research literature, SIFT (Lowe, 1999) is the first robust and—according to Google Scholar, with more than 30,000 citations—the most referenced descriptor which is seen as a milestone in the development history of local invariant feature descriptors. SIFT is still the most reliable with high repeatability rate, despite the fact that many versions have been developed to address one of its weakness: ASIFT (Yu and Morel, 2009) for affine invariance, CSIFT (Abdel-Hakim and Farag, 2006) and Opponent SIFT (Van de Sande *et al.*, 2010) for color image description.

Recently the focus went beyond the repeatability rate addressing the calculation time, the dimensionality of the feature descriptors and the light change invariance for outdoor use. Different approaches were used to achieve dimensionality reduction and optimization using existing descriptors or creating new generation methods. For example using SIFT as a base calculation method PCA-SIFT (Ke and Sukthankar, 2004) or GLOH (Mikolajczyk and Schmid, 2005) using Principal Component Analysis

and SURF (Bay *et al.*, 2008) by simplifying its calculation process. For color invariance CSIFT (Abdel-Hakim and Farag, 2006) uses color invariant features or Opponent SIFT (Van de Sande *et al.*, 2010) with opponent RGB color channels. π -SIFT (Park, Park *et al.*, 2008) uses photometric invariant features. The other category is to create a descriptor from the root using different calculation methods. For illumination invariance, SMD (Gupta and Mittal, 2008) obtains invariance to a monotonic change in the intensities, OSID (Tang *et al.*, 2009) apply ordinal and spatial intensity histogram, LIOP (Wang, Fan *et al.*, 2011) considers the intensity order among all the sample points.

All these descriptors derive information from the actual image with the aim of performing in all type of environments. The resulted descriptors operate well in the dedicated environments but achieves poorly when other factors change. As a response DIRD (Lategahn *et al.*, 2013) creates and optimizes an illumination invariant descriptor for a particular environment using genetic algorithms. Their descriptor outperforms the existing hand-crafted like USURF (Bay *et al.*, 2008) or BRIEF (Calonder *et al.*, 2010), but the resulting dimension of their descriptor is still 216 which implies a long calculation time.

Machine Learning algorithms were proposed to enhance the descriptor's recognition and segmentation rate. The descriptors discriminative abilities are based on a probabilistic model that is learned from the incoming low-level image information.

I.1.4. Machine Learning

Machine Learning (ML) is one of computer science's subdomains that evolved from the combination of pattern recognition and artificial intelligence. Such algorithms make predictions, intelligent decisions based on the incoming data learn from the created prediction models. Applications use machine learning algorithms in multiple areas: data mining, optical character recognition, stock market analysis, medical diagnosis and image segmentation among others.

To respond the increased demand of mass 3D reconstruction and modeling in city planning—geo-applications like Google Earth or Microsoft Virtual Earth—and in 3D GPS navigation systems automatic facade recognition techniques were developed to reduce the rebuilding time and the data storage size (Van Gool *et al.*,

2007). Various recognition and segmentation techniques were developed over the years using Machine Learning Techniques.

The "bag of keyword" method (Csurka, Dance *et al.*, 2004)—a global image segmentation approach—assigns the pixel-level information to high-level semantic groups called "vocabularies" to train a multi-class classifier. This technique performed better using Random Forest learning method and color descriptors (Berg *et al.*, 2007; Fröhlich *et al.*, 2010; Delmerico *et al.*, 2011).

The Random Forest (Breiman, 2001) is a high-performance discriminative classifier, handling large set of features without having problem on high dimensionality (Fröhlich *et al.*, 2010). This supervised learning method—that can learn more than one class at a time—constructs an ensemble of recursively created random binary decision trees during the training period.

There is a clear opportunity improve a new type descriptor with machine learning algorithms, but also optimize all of the descriptor training and creation parameters for genetic algorithms.

I.1.5. Genetic Algorithm

The Genetic Algorithm (GA) is an effective stochastic algorithm taking as an example of the natural selection and genetics. It has been applied in Machine Learning and optimization problems successfully (Guo *et al.*, 2010). The algorithm creates and keep a population of fixed number of individuals (genomes) and modifies their composition based on probability, genetic operation—Selection, Crossover and Mutation—and evaluation function. The algorithm runs until reaches the termination criteria, creates the last set of generation or the perfect score reaches a particular threshold.

There has been a few application of GA used in object recognition (Sarfraz, Mehmood-ul-Hassan *et al.*, 2009; Behnam and Pourghassem, 2013), descriptor optimization (Trujillo, Legrand *et al.*, 2010) or creation (Lategahn *et al.*, 2013). The resulted algorithms used on binary greyscale descriptors and machine learning algorithms have not been applied in the objective function.

Creating descriptors with genetic algorithm and optimizing the parameters of each step of the process gives a new prospect in image processing.

I.2. Problem Description

The subject of the Project is to create a new descriptor for the tracking engine of a marker-less Augmented Reality application. The application has to fit certain characteristics, it has to be robust enough for continuous outdoor use, and the tracking has to be stable in congruent or incongruent settings. The AR applications have to be highly optimized for mobile devices, whose computational unit cannot support the laborious tracking and mapping operations.

I.3. Justification

The requirement for a stable open-air augmented reality application is the ability to process and understand the surrounding environment. In this case, the urban areas' primary structures are the buildings. Visual façade classification is the job of evaluating the position and size of different structural (e.g. window, door) and non-structural elements (e.g. sky, road, building) in a given image of a building or street scene (Fröhlich *et al.*, 2010). This recognition task has gained interest in the last years (Van Gool *et al.*, 2007), which is essentially due to the growing need to store the shape of buildings in extensive 3D city models. However, according to Guan and Wang (2009) the vision based registration systems for augmented reality are controlled by an ever changing environment (wide-area work scene, changes in illumination, sudden motion, and occlusion), and because of that it is under constant development.

As previously mentioned an outdoor augmented reality application needs multiple sensory inputs to be able to function correctly like in the case of Arth, Klopschitz *et al.* (2011), Gauglitz *et al.* (2011) and Ventura and Hollerer (2012). However, these solutions require a constant connection to a cloud server for source information and computing aid, which reduces the speed and accuracy of the system. There is a need for a robust system that can simplify the geometry of the surrounding environment, and via this process stabilizes the registration, reduces or avoid the communication time between the cloud server and the device.

I.4. Hypothesis

The Random Forest training method with a new type of descriptor, which takes into account the different color spaces and channels and also the geometric relation between the point and the camera position and rotation, is robust enough and capable of doing the façade segmentation in real time.

I.5. Objectives

I.5.1. General Objective

To perform robust and real-time facade segmentation using a newly generated and optimized feature descriptor. The descriptor is generated by genetic algorithm, trained with incorporating Random Forest and optimized without losing their descriptive strength to make suitable for AR applications.

I.5.2. Particular Objectives

1. Design a new type of low-level image descriptor that invariant to the changes that can occur in outdoor environments.
2. Evaluation and analysis of the structure and performance of the designed descriptor to optimize it in size, robustness and invariance to illumination changes.
3. Programme a self-adjusting algorithm for the optimization and evaluation using Genetic Algorithm.
4. Make Parallel Tracking And Mapping more robust in outdoor environments by integrating the generated optimized image descriptor to auto segment the dominant features of the facade.

I.6. Layout of this Thesis

The above introduction has outlined the main contributions described in the body of this thesis. Chapter II will present the latest advancement of the used technologies. The most recent developments in AR, in markerless SLAM, in the domain

of visual descriptors, in ML and GA that used in the field of image processing and segmentation. Chapter III explains the used technologies and the project development. We will see in details the base functions of the AR applications, the SLAM framework, the most referenced descriptor structures, the used image processing methods and the ML and GA applications. Chapter IV will describe the proposed algorithm and the used calculations and how we fit the resulted descriptor in the SLAM application. Chapter V presents the results and the completed objectives of the project and discuss the possible directions for continuing the investigations of the resulted framework.

II. LITERATURE REVIEW

II.1. Introduction

This chapter describes the taxonomies and state-of-the-art technologies relevant to AR, Computer Vision(CV), self-localization, local feature description and the use of machine learning in descriptor creation. The introduced articles and project served not only to learn about the current advancements but as inspiration in certain segments of this doctoral project. We will present in Section II.2 the research groups—around the world—working presently on AR, ML and CV. AR is first described in Section II.3. Mixing real and virtual is a broad term. Therefore, it will be presented through different taxonomies for better understanding. The latest developments in display technology, AR systems, and tracking are discussed in this chapter as well. The SLAM techniques—where the system can detect its outdoor position and orientation without external markers—are reviewed in Section II.4. Most of the discussed methods involve feature descriptors as anchors during their function. The overview of the latest extraction techniques of these local features is given in Section II.5. The use of Machine Learning algorithm to boost the descriptors capabilities is presented in Section II.6. Finally, a review of the incorporation of Genetic Algorithm in the descriptor creation is provided in Section II.7.

II.2. Research Groups

There are many outstanding research groups and laboratories around the world investigating a different aspect of Computer Vision and Artificial Intelligence. The Computer Vision Laboratory (CVLAB) at the École Polytechnique Fédérale de Lausanne in Switzerland—directed by Dr. Pascal Fua—covers various research fields. Image Description—e.g. the BRIEF (Calonder *et al.*, 2010) descriptor and BinBoost(Trzcinski, Christoudias, Fua *et al.*, 2013) was developed here—, Keypoint Detection, 3D object tracking or Tracking and Modeling people are of their interest. The Human Interface Technology Laboratory New Zealand (HITLabNZ) at University of Canterbury, leadered by Prof. Mark Billinghurst, is currently investigates the area of Augmented Reality, Human-Robot Interaction, and Visualization among oth-

ers. The Robotics Research Group at the University of Oxford (United Kingdom) has had an extensive list of high standard projects since 1985. It is divided into various laboratories. The most relevant in our case is the Active Vision Lab, leaded by Prof. David Murray, and among other things here was developed PTAM (Klein and Murray, 2007). Also worth mentioning the Machine Learning, Mobile Robotics, and Optimization for Vision and Learning laboratories too. Under the leadership of Prof. Vincent Lepetit, the Institute for Computer Graphics and Vision—funded in 1992—at the University of Graz (Austria), focuses among other things on studies Object Recognition, Object Reconstruction, Robotics, Virtual Reality and Augmented Reality. Also have to be mentioned without details, MiT Computer Vision Research Group (United States), Microsoft Research (e.g. Kinect cameras, Holo Lens), Magic Leap (e.g. Augmented Reality), Facebook Research (e.g. Oculus Rift, Virtual Reality), Google Research (e.g. Machine Intelligence).

In Mexico, the researchers in the Centro de Investigación y Desarrollo Tecnológico en Electroquímica (CIDETEQ) under the leadership of Dr. Ivan Ramon Terol Villalobos work with image processing and segmentation. In the Centro de Ingeniería y Desarrollo Industrial (CIDESI) the team led by Dr. Hugo Jiménez Hernández conduct investigations in CV, Tracking, and 3D reconstruction. Dr. Francisco Javier Cuevas de la Rosa with his team in the Centro de Investigaciones en Optica run investigations in the field of Computer Vision, Computational Intelligence, Evolutionary Algorithms and Digital Image Processing. At the National Autonomous University of Mexico (UNAM) in the Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS), Dr. Ernesto Bribiesca with his team researches robotic vision, image processing, and recognition.

II.3. Augmented Reality

The investigation of AR is a broad field of study—with over 28 thousand scholarly articles only in IEEE Xplore, Elsevier, and Springer—, and a booming industry with a predicted consumer market of 90 billion USD in 2020 (Digi-Capital, 2016). It is well researched and maturing technology with still many imperfections to resolve. To understand where the problem resides and how the researchers try to solve, we have to understand how a basic AR system works and what its building blocks.

Two simple taxonomy explains well the characteristics of AR. Milgram *et al.* (1994) categorizes the AR by the amount of virtual data being involved in the sys-

tem. The fact that there is a small amount of digital information is mixed with the real environment—as we can observe on Figure 1.1—does not make it easier the task. On the contrary, to maintain the illusion in an unpredictable environment that the virtual objects is part of the real environment is a demanding job. The taxonomy of Rekimoto and Nagao (1995) categorizes computer interfaces based on Human-Computer Interaction (HCI) and how well they become invisible to the humans meantime enhancing the interaction with the real world. The base concept is the Graphical User Interface (GUI) based desktop computer (Figure 2.1 (a)) which holds an apparent gap between the virtual and real domain. One of the approach to overcome this separation is the Ubiquitous computers, where the sensing and computing units are seamlessly built into the real world (Figure 2.1 (c)). We can observe a clear difference in contrast to the remaining techniques where the computer partially or entirely overwrites the interaction with the real world. In the case of the Virtual Reality (Figure 2.1 (b)) the user has fully separated from the real environment, and computer-generated graphics entirely replaces its information. In contrast the AR approach (Figure 2.1 (d)) is built to enhance interactions in the real environments. Also, this figure combined with the first categorization shows the difficulties that faces the AR systems. According to Azuma *et al.* (2001) stated "Without accurate registration, the illusion that the virtual objects exist in the real environment is severely compromised." In other words the virtual information has to be aligned correctly continuously with the real world because the user can sense each small errors or jitters as it has the real environment as a parallel reference. For that reason AR systems have to be able to find accurately their current position and compute their relation to the environment.

One of the most commonly used definition for AR coined by Azuma *et al.* (*ibid.*) depicts the previously mentioned theoretical and technical requirements. The AR systems have the following characteristics in common:

1. It combines real and virtual content,
2. It is interactive and real time,
3. It registers or aligns the real and virtual objects with one and other in 3D.

These three characteristics point towards the technological elements that build these systems (Billinghurst *et al.*, 2015):

1. A display that can fuse the real and virtual imagery,

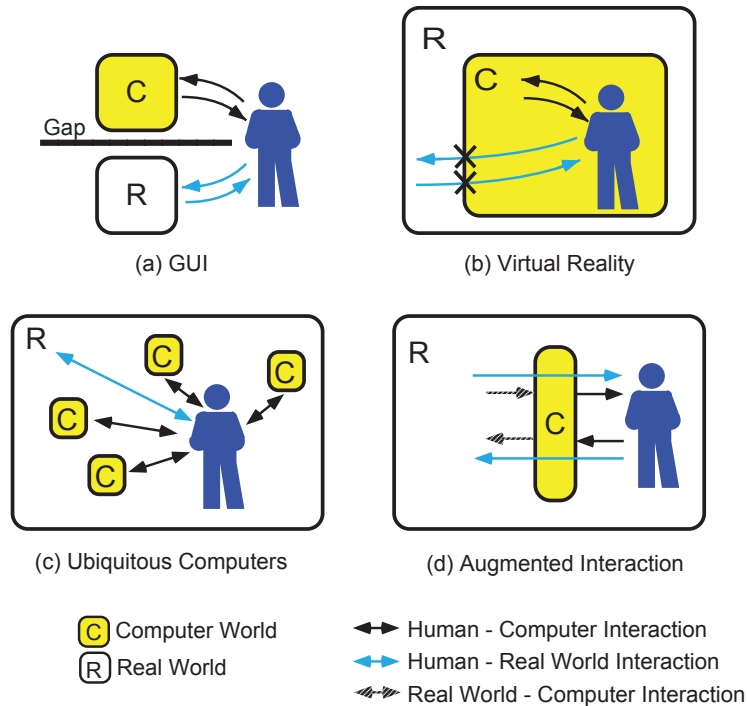


Figure 2.1: Human Computer Interaction styles
 Source: Rekimoto and Nagao, 1995

2. A computer system that can produce in real time the interactive response to the user's input,
3. A tracking method that can find the users viewpoint position, rotation and direction to register the virtual image in order to appear fixed in the real world.

These elements are not only the responsible for a faultless AR experience but the source of errors in the process (Figure 2.2). As previously mentioned, for an accurate Registration is inevitable for an AR illusion. The Registration deficiencies Chinthammit *et al.* (2003) categorized into two types: static and dynamic errors. The static errors occur when the user is stationary, and originated from calibration, tracking failures, and system misalignment. The dynamic errors appear when the user is in motion, and first and foremost is related to the system latency. The largest delay among the dynamic errors caused by tracking (Lincoln *et al.*, 2016). As can be seen, the tracking is an important source of errors with incorrect localization and latency. As a result, there is a need for fast and robust tracking techniques in AR applications.

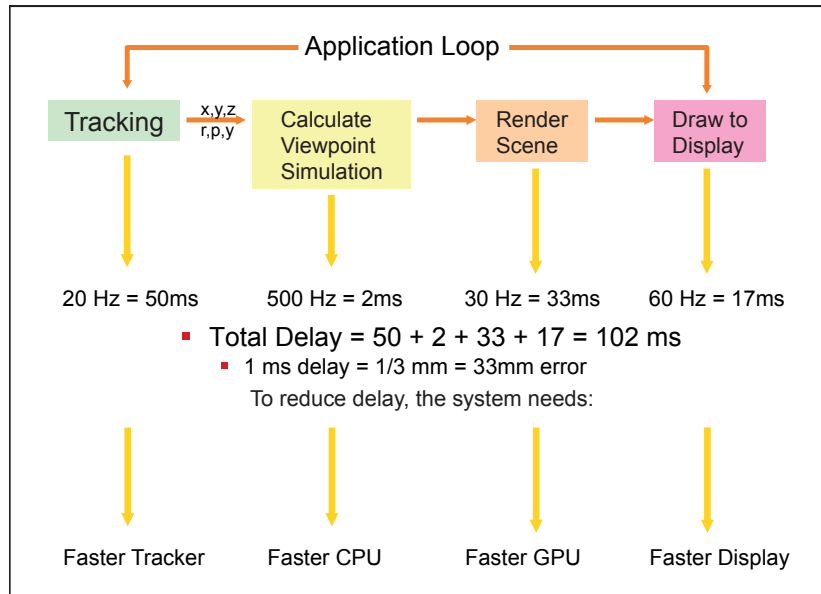


Figure 2.2: Source of system delay and possible solution
Source: Billinghurst, 2013

In this review, we will investigate the latest developments in three main areas of AR stated by Azuma *et al.* (2001). First, we will see the brief history of the AR evolution, where it started and what is the current state of AR support technologies.

II.3.1. History

Placing virtual images onto real environments using mirrors, lenses and light sources have long history goes beyond the beginning last century (Billinghurst *et al.*, 2015). For example, in the 17th century the "Pepper's Ghost"—an illusion applying plates of glass to combine the reflection with the real world—was already used in theaters and museums (Brooker, 2007).

The digital AR has only fifty years of history, but the technology and its potentials are growing exponentially. Sutherland (1968) created the first head mounted mixed VR - AR system. The display was suspended from the ceiling, and the system contained two different 6 Degree Of Freedom (DOF) tracker to follow the users movements, but the limited computer processing power only allowed simple wire-frame drawings (Figure 2.3).

The investigations from the 80s until the mid-90s laid down the fundamentals of the AR tracking and displays. The separation from VR came in the early 90s

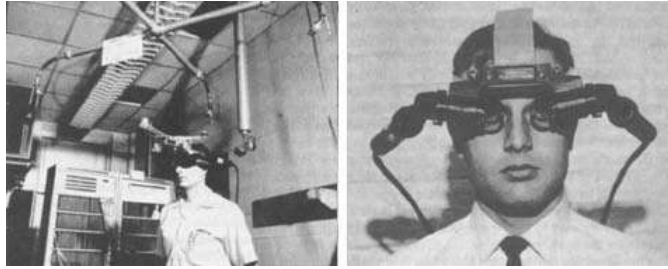


Figure 2.3: Sutherland's sword of Damocles
Source: Krevelen and Poelman, 2010

when Caudell and Mizell (1992) first coined the term Augmented Reality applying to the concept when a digitally created material is overlaid on the top of the real world. They stated that the advantages over the VR were that needed less pixel to render but to align the real and virtual objects correctly, an increased registration and positioning was required. During the 90s, the foundation technologies for the AR were developing simultaneously. There were multiple approaches to resolving accurate localization.

One was using sensors like accelerometer and gyroscope for orientation, and GPS for positioning. One of the most valuable aid for AR tracking the GPS originally started in the 70s as a military project and later became accessible for civilian use with less accuracy (El-Rabbany, 2002). The accuracy was set to 100 meters with the technique called "Selective Availability" (SA) degrading the signal for nonmilitary users, but during the years investigations and new technologies reduced this 15 meters. The GPS was fully functional in 1995 for both military and civil use. The GPS first appears in mobile AR reality system, the Touring Machine (Feiner *et al.*, 1997) see Figure 2.4. The system is still a computer in a backpack connected to a GPS receiver and a HMD amongst others. The GPS has flaws as the satellites not always visible to the devices and is not accurate enough for most of the AR application where the registration requires pixel-level accuracy.

The second approach was to process on pixel level the incoming imagery to localize the device position in the environment. The 2D markers proposed by Rekimoto (1996) was the first marker system which gave a six degree of freedom tracking for the cameras. This technique allowed a much more stable tracking and also became a crucial element for the AR applications. This method was further developed to ARToolKit (Kato and Billinghurst, 1999) tracking library 2.5. But the ARToolKit was still affected by pattern complexity, the marker orientation relative to



Figure 2.4: Touring Machine
Source: Feiner *et al.*, 1997

the camera, and the lighting conditions (Human Interface Technology Laboratory - University of Washington, 2003)

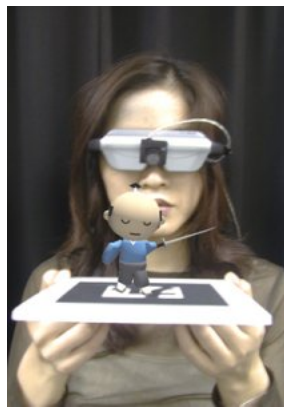


Figure 2.5: ARToolKit
Source: <http://www.hitl.washington.edu>

The third approach to mixing the sensory data with image processing. These applications were used GPS, mobile phones with cameras, but these days still using laptops or personal computers (PC) and HMD Mobile Augmented Reality System (MARS) (Feiner *et al.*, 1997) to Battlefield Augmented Reality System (BARS)(Julier *et al.*, 2000).

II.3.2. Display

Following the definition of Azuma *et al.* (2001), there has to be a kind of display technology that merges the real and virtual for both to be seen at the same time. We will follow the taxonomy of Bimber and Raskar (2003) illustrated in Figure 2.6

the possible spatial places where the image can be reproduced, the display can be located on the user and real world object. They categorize the display technology into three different groups:

- **Head attached**

The display is directly connected with or attached to the head, as head mounted optical or video see-through displays or projectors which will be discussed later.

- **Body attached**

The display is still attached to the body but not so close to the head. In this class we talk about the handheld displays (e.g., mobile phone, tablet) or handheld projectors

- **Spatial**

The display is not joined up with the body and can have fair distance between them. Here are the spatial see-through displays(e.g., see-through displays in military cockpits) and the spatially aligned projectors.

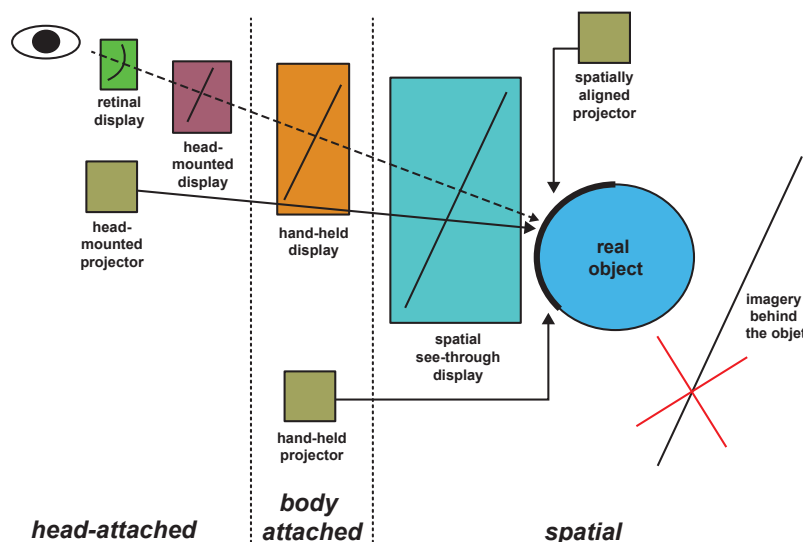


Figure 2.6: Display Taxonomy
Source: Bimber and Raskar, 2003

The most representative results in the AR/VR technologies are the developments in the display technology as this is the visible and vision part of the AR systems. (Krevelen and Poelman, 2010) organized the AR displays by their characteristics, drawbacks and opportunities (Table 2.1). To present an AR application currently,

the most popular and widespread platform is the hand-held video see-through solution, in other words, the mobile devices. Although their displays are limited in brightness, contrast resolution, and the camera field of view, these types of equipment offer the cheapest but also the most challenging way to create an AR system. These devices differ widely between the models, therefore, we limit our introduction to the characteristics presented in Table 2.1. We will focus on the most recent developments of the head-worn optical and video see-through displays. Also, worth mention the development of the smart contact lenses as has great potential. On Table 2.1 the retinal display refers not to the contact lenses but to the technology that project imagery directly to the human retina.

Table 2.1: Characteristics of visual AR displays

Positioning Technology	Head-worn				Hand-held	Spatial		
	Retinal	Optical	Video	Projective	All	Video	Optical	Projective
Mobile	+	+	+	+	+	-	-	-
Outdoor use	+	±	±	+	±	-	-	-
Interaction	+	+	+	+	+	Remote	-	-
Multi-user	+	+	+	+	+	+	Limited	Limited
Brightness	+	-	+	+	Limited	+	Limited	Limited
Contrast	+	-	+	+	Limited	+	Limited	Limited
Resolution	Growing	Growing	Growing	Growing	Limited	Limited	+	+
Field-of-view	Growing	Limited	Limited	Growing	Limited	Limited	+	+
Full-color	+	+	+	+	+	+	+	+
Stereoscopic	+	+	+	+	-	-	+	+
Dynamic refocus (eye strain)	+	-	-	+	-	-	+	+
Occlusion	+	+	+	Limited	±	+	Limited	Limited
Power economy	+	-	-	-	-	-	-	-
Opportunities	Future dominance	Current dominance			Realistic, mass-market	Cheap, off-the-shelf	Tuning, ergonomics	
Drawbacks		Tuning, tracking	Delays	Retro-reflective material	Processor, Memory limits	No see-through metaphor	Clipping	Clipping, shadows

Source: Krevelen and Poelman, 2010

There is a relatively small publication on AR or Bionic contact lenses. Researchers at the University of Washington created the first prototype of a "flexible, biologically safe contact lens with an imprinted electronic circuit and lights" (Hickey, 2008). They further developed the design into a wirelessly powered and controlled display that emits only one pixel of information (Lingley *et al.*, 2011). Samsung patented the design of a smart contact lens in 2016. According to (Adhikari, 2016), this lens holds a miniature screen, a camera, an antenna, and multiple sensors for the movement and eye blink detection. In the opinion of the experts, the current technology is still facing power and resolution problems among other, but the uses for smart contact lenses "are almost limitless" (*ibid.*).

The direction in the development of AR HMD goes towards an integrated vision system with cameras, microphones, and visual display systems. We have to mention the Oculus Rift™ goggles that put Virtual Reality back onto the map and according to Krewell (2016) it "will eventually include augmented reality (AR)". The goggles have a high-resolution screen at an accessible price.

HoloLens™ is a visual see through HMD—developed by Microsoft™—which is still available only for developers. It has a fully integrated visual system, with 4 environments understanding camera, 1 depth camera, 1 HD video camera, 4 microphones and 1 ambient light sensor. These sensors facilitate to use spatial sound, gaze, gesture and voice tracking. It is passively cooled eliminating this ambient noise. The hardware provides huge amount built in information for stable and robust AR tracking and experience. It comes with an SDK for the integration and development. Microsoft.com (2016) claims that the "see-through holographic lenses use an advanced optical projection system to generate multi-dimensional full-color holograms with very low latency".

Magic Leap™ is a recently founded company with a funding of 1.4 billion USD. The company is very secretive about the developed hardware and software, but according to Hempel (2016), Magic Leap™ "creates digital light field signals that mimic the way sight works", trying to clone the eyes "and make a digital version of it". Hempel (*ibid.*) says that the existing prototype is in a very early stage and it has to be reset after 15 minutes so it wouldn't crash. But she claims that the demos looked real without pixelation.

II.3.3. AR Systems

Vuforia™ is a semi-free Software Development Kit (SDK) for mixed Reality application development. This multiplatform package was launched in 2010 and currently is the version 5.5. With support for iOS, Android, and Unity 3D, this platform proves a tool for developers to create AR applications for a wide range of mobile devices. The SDK using planar and cylindrical object tracking also includes text, object and cloud recognition. The system allows the developers to upload unique marker to the cloud for their application and able the users download and use it without place restriction. Their current version supports the newly released HoloLens™ which brings a new stage for AR experience (Vuforia.com, 2016).

Junaio™—created by Metaio™—was an SDK and Augmented Reality browser opened for developers. According to Junaio.com (2015) the browser overcomes the GPS inaccuracy indoors by using LLA Markers (latitude, longitude, altitude marker). "Junaio only supports location-based AR, but also image based tracking" (*ibid.*). The users could choose between two different type: location or image based AR. During the location based AR the system uses the GPS, Compass, Accelerometer and Gyroscope to specify the points of interest and render the virtual objects in the surroundings. During the image based AR, the method uses images—created and uploaded by the users or developers— as markers gluing the virtual objects on the top of them. Metaio together with the AR technology was reportedly bought by Apple in May 2015 (Wakabayashi, 2015), which projects a new AR service on iOS devices.

II.3.4. Tracking

During the tracking, the AR systems specify their position and orientation in 3D environments, computing continuously 6 degrees of freedom (DOF) pose information: the position (x,y,z), and the rotation (roll, pitch, yaw).

According to Roberto *et al.* (2016), there has been a trend in the published papers on tracking technology between 2009 and 2014. The report shows that along with the increase of publications about tracking on mobile devices there was a growth among the sensory and the vision based tracking papers. According to the authors, the studies that use only sensory information to calculate the device's position and orientation information are the most common solution for the mobile applications. These applications do not require more precise information about the surrounding for the augmentation (e.g. the digital imagery is "floating" in the air and is not aligned with the environment). The exploitation of the mobile devices' built-in cameras and processing capacity shows the increased number of vision based investigation. These techniques demanded more processing power from the devices but combined with the sensory information provide a more robust tracking performance. Interestingly these vast majority of these calculations are run locally on the devices, but due to the increased need of information, the development of data transfer technology and, the improvement of the network infrastructure, the researchers are investigating the use of distributed tracking platforms. These techniques applying remote computers—connected to the mobile devices through the cloud—with more resources for the mobile devices, such as computing power, storage space

and memory.

This work will present briefly—with one interesting article—the sensory based and the marker based tracking literature as they are not related to the subject.

Sensory only tracking

In the papers on "sensory only" based tracking, the methods trying to improve the precision of the GPS position. Zandbergen (2009) measured the Assisted GPS (A-GPS) The proposed approach of Chon *et al.* (2012) exploits the users behavior via learning techniques to provide a better location service. The users construct a LifeMap with their personal point of interest locations during their activities. Thus, their method does not require previous data training. Using the information of the user's daily routine the system calculates and predicts the correct position reducing energy consumption and sensory errors. However, inherently their process failing where the inbuilt sensors do not produce reliable information—e.g. the GPS positioning deteriorates in underground environment—, also, the energy consumption increases after great distances as the system leaves the sensors ON throughout the trajectory.

Vision only tracking

Among the marker based technique, the ARToolKit (Kato and Billinghurst, 1999) was the first vision based tracking package for AR application. Since then the works trying to reduce the storage space of the marker templates and to resolve how to make more robust the tracking, as the markers are not trackable when the borders or edges are occluded. Li, Chen *et al.* (2012) proposed a new marker system which radically reduces the storage space and still provides enough information for registration even when the marker is occluded by 62.5%. The method—named CoP-Tag(Connected Points Tag)—uses 16 dots which are connected to a square region, at each side with five dots and a straight line through the center of the circles. The identification of the marker, based on the connectivity and a unique distance between the dots. According to the authors, the marker is robust to occlusion, noise, blur, and multi-marker registration, but over a certain distance, the marker is not detectable.

We will discuss in a Section II.4 the advances of the natural feature based

tracking techniques, namely the Simultaneous Localization And Mapping (SLAM) as it is closely related to the subject and brings more insight the latest development on the field.

Sensory and Vision based tracking

Kurz and Benhimane (2012) proposed a new type of combination of vision and sensory information to increase the realism of the augmented 3D objects. The authors used the inertial sensor data to improve the precision of the descriptor features. They created two gravity based descriptor using the gravitational force measured by inertial sensors (e.g. accelerometer). The Gravity-Aligned Feature Descriptors (GAFDs) created for static and vertical surfaces, and the Gravity-Rectified Feature Descriptors (GREFDs) are designed for horizontal surfaces. During their tests, the researchers reported an increase in the recognition rates in an AR museum guide using mobile phones using GAFD. Applying the gravitational vectors for image rectification they recorded improved precision–recall characteristics in the application. They also state that the "Gravity-rectified camera images also allow for real-time 6 Degree of Freedom (DoF) pose estimation using an edge-based object detection algorithm handling only 4 DoF similarity transforms." In other hands, according to the authors, the sensor hardware and processing power of the current mobile devices are not accurate and powerful enough for their approach.

Park, Lee *et al.* (2012) presented a method to improve the position and orientation detection for outdoor AR application. Their approach was to compare the preliminary information from the GPS, digital compass, and the vertical edges from visual detection with the visible corners of a one-dimensional cylindrical depth map. They determined the user's position optimizing non-linearly the location candidates. The maps included information about road and building areas, building types and GPS locations of each building corner. The base of this map was obtained from the National Spatial Information Cleaninghouse (NSIC) in Korea. They report an improvement localization and orientation, but the system failed when the GPS position error was too large.

Distributed tracking

The standard approach to the system that uses distributed tracking is that only a small fraction calculation occurs on the device. Most of the visual or sensory

information is sent to a server where all the localization and registration is computed and the results transmitted back for display.

Arth, Klopschitz *et al.* (2011) presented a wide area high-quality outdoor tracking and scene reconstruction system for mobile devices. Their system consists an incremental orientation tracking with 3 DoF and a model-based localization with 6 DoF. The authors recorded an increment of tracking and pose estimation accuracy with growing aperture angle. Therefore, they implemented geo-referenced panoramic images for the localization, as the mobile phones have little Field of View (FOV) cameras. They used panorama creation method—with image stitching—on the server to achieve the compelling cylindrical images. Then the resulted image is separated into tiles and compared to the online stored 3D model for localization. They demonstrated that the system is capable of real-time AR, but they assume that the users are static while record the panoramic images, which makes the approach less dynamic and realistic.

Wu, Choubassi *et al.* (2011) proposed a Mobile Augmented Reality (MAR) system which uses processes the localization and tracking on the mobile device. The client application computed SURF features from the frames compared with visual features of pre-processed database images. Unlike other systems where the data is calculated on the server, they used the server for storing the dataset. They downloaded and matched the incoming imagery against only a constrained dataset. The server selects the query pictures by location and orientation. The 3D urban dataset is stored as 3D point clouds which are linked to geotagged RGB images of the same urban scene. They report a good accuracy during their test, but the system suffers from latency in real time implementation. Only the detection algorithm, runs on average at one frame per second, plus we have to add the latency caused by the continuous communication with the server and the time method needs to download the corresponding dataset.

Ventura and Hollerer (2012) proposed a real-time AR system for wide area outdoor environments. They placed the localization module to a cloud server, to ease the real time tracking on the client device. The point cloud dataset, which was used for the localization, was constructed from omnidirectional videos and stored on the server. Their tracking method was similar to PTAM's (Klein and Murray, 2007) with the difference that they used a "full perspective patch warp rather than an affine approximation". This method was lightweight enough to use on mobile applications.

For the localization, the system used the SIFT detector and descriptor algorithm because its robustness. During their evaluation, they tested the accuracy and latency of the scheme. They achieved suitable results during the localization matching up to 95% of the query images. In the latency estimation, the tracking module on the device gave relatively good results with 15-20 frames per second, but as they state the feature-based localization is not fast enough for real-time applications. Also, they report various failures during the localization caused by sparse environments (lack of texture), repetitive structures (doors and windows), and occluding high-textured objects (e.g. trees and bushes).

We can observe that the tracking is a computationally intensive process, acts as constriction in the AR applications. The real-time systems need to achieve a minimum of 30 fps performance and highly accurate positioning for realism. There are various approaches presented in the literature to achieve this capacity. Combining the devices inertial sensors with the visual tracking modules for higher accuracy, or splitting the process and calculating in separate locations the tracking and localization functions. Typically the localization occurs on a remote server as it involves a more computation power and large dataset for image matching, and the tracking runs on the mobile device as the client application. The roots of the problem twofold, in one hand, is the insufficient computing unit, and unreliable sensors on the mobile devices, in the other the computationally heavy localization modules. The current applications for specifying the devices location uses calculation expensive or condition variant descriptors which imply more adjustments for a flawless experience. For that reason, the use of remote computing unit is a favorable response, but the coverage of wireless connections are still limited in outdoor areas. The most plausible answer to the question is to use an optimized and invariant descriptor to ease this bottleneck in tracking.

II.4. Simultaneous Localization And Mapping

Klein and Murray (2007) proposed the groundbreaking Parallel Tracking and Mapping (PTAM) method, which was the first work that parallelized the camera tracking and the mapping in separate processes. They use FAST (Rosten and Drummond, 2005) keypoint detector on a four-level grayscale pyramid—made from the incoming image—at each keyframe. The system loads detected points into the parallel 3D point map and compares them with the actual points for localization. They

proved that their approach was fast and robust enough for real-time augmented reality applications in small working areas. The method opened new ways in markerless tracking and influenced many later works. Among others Klein and Murray (2009) created its lightweight variant for mobile phones, Guan and Wang (2009) combined it with machine learning technique for extensive outdoor areas. The approach also presents limitations which come from the point description that corresponds to the FAST corner features which make the points usable only during tracking but not for place recognition. Also, it cannot detect large loops, due to its exponentially growing map and low invariance to viewpoint changes.

Arth, Wagner *et al.* (2009) presented in their paper a wide-area pose estimation method using recorded 3D feature models which have been created image collections and wide-angle lens. Their system acquires the visual data and register the data points offline (on the device) and runs the localization and matching online (on the server). They state that the matching performance increased—except with very low-resolution images— compared to the approaches that use standard camera lens. According to their finding, the matching performance does not elevate using images with higher resolution than 640×480 . They also illustrate that most of the processing time their application consumed in the feature extraction stage, as during the process the scale-space search for extremal points and the subsequent generation of descriptors are the computationally most intensive tasks. They state that the overall memory use is small enough to run it on a mobile phone, but the system was only tested indoor in a short period.

To overcome the PTAM's wide area registration problem, Guan and Wang (2009) proposed a method using multiple sub-maps and scene recognition techniques. They also presented a random forest-based incremental scene learning method which is processed online. The authors applied a hybrid, natural feature tracking approach using Graphics Processing Unit (GPU) accelerated SIFT descriptor for the initialization. They tested the system indoor and outdoor scenes. Their findings show that their approach overcomes the PTAM's tracking failure in low texture areas, and the sub-maps solves the overgrowing map and scaling issues that fail PTAM in broad areas. The cost in another hand is that the method is still computationally heavy using the GPU as aid which is not present in all devices.

The proposed wearable AR system in the paper of Castle and Murray (2011) is based on the PTAM (Klein and Murray, 2007) monocular visual SLAM application.

It tracks the camera position from frame-to-frame, and detects and recognizes known objects calculating SIFT features at each keyframe and attempts to reconstruct the 3D scene. The application runs the localization and object recognition on two separate processes as PTAM. Their method was tested in laboratory environment also in real indoor environments and street scenes. They report improvement in the freedom of the camera movement and the size of the explored area compared to their previous work (Castle, Klein *et al.*, 2010), but the SIFT feature calculation stays inadequate for the task as it creates a bottleneck method on a dual core machine. They recommend a cheaper feature descriptor as an alternative for the problem.

To resolve the failures of visual route-based navigation system caused by extreme perceptual changes, Milford and Wyeth (2012) propose a visual recognition algorithm that searches for "local best matches" in short sequences of images rather than calculating the single location called SeqSLAM. They do not use local features during their method, as—they state—SIFT and SURF are not suitable for such task during extreme light condition change (e.g. moving daytime to nighttime). Their method based on prerecorded footages, where each frame was combined with GPS position. During the preprocessing period, the system down samples divides into smaller regions the images, then normalizes each patch. They used Sum of Absolute Differences to find a match between the processed images and the dataset. They assumed a constant speed for the camera during the travel, but still their approach fails in the case of too extreme changes and suffers from viewpoint changes.

Another strategy to light changing problems was presented by Carlevaris-Bianco and Eustice (2014). In their work, the local feature descriptors are learned from recorded webcam footages using machine learning techniques. To specify interest areas they use SURF keypoint detector. For each keypoint, the method extracts a gradient patch which is loaded into a convolutional neural network (CNN). The learned large dataset (approximately 3.1 million feature) later used during the matching stage. The learning technique matches the candidate descriptors by closeness using Euclidean distance. They report a 10% increase in place recognition performance on a challenging robotic dataset than SIFT, SURF or other handcrafted feature descriptors, but the feature extraction time equal or worse than the mentioned local features.

To make the SLAM process feasible for mobile phones, Ventura, Arth *et al.* (2014) lifted the computationally most expensive from the cell phones and placed

into the cloud. The client application (phone) only responsible for the rotational and translational pose tracking and mapping, meantime the server calculates the localization and global registration information. Their method computes SIFT features at each keyframe which then is searched for the nearest neighbor on a previously built kd-forest. To reduce the query time and memory load in a large—and possibly growing—dataset, the authors applied visibility partitioning. These hexagonal grids partition the surrounding environment and the prebuilt model only allowing the system to load the corresponding part to the localization function. Despite the robustness, the average request time between the server and client takes almost 1 second in the outdoor environment.

Lowry *et al.* (2014) proposed a place recognition method with images treated with PCA to obtain a condition independent dataset. The method detects and removes the aspects and elements that were widespread on the scene. They keep the later PCA dimensions with the most distinctive elements removing the common elements (e.g. white snow in winter or darker ground in spring). As a result, the place recognition stability increased. They tested the approach on two different environments with single-image and sequence-based place recognition. They reported better performance in both cases than using the original images. They state that the required information can be acquired online from environmental data, and the training set can be built online. Although the technique is an excellent tool to obtain condition-invariant features from images, they do not exploit the potentials of merging the information gathered from data that was recorded in different circumstances to get a more robust tool.

Mur-Artal *et al.* (2015) created a new localization and tracking model, called ORB-SLAM based on an expanded PTAM (Klein and Murray, 2007) achieving a more robust application. Their method while keeps the core ideas of PTAM extends it with a loop closing function, the use of covisibility information, and the calculation of ORB local features for place recognition. They tested their approach on indoor and outdoor datasets, reporting a real-time operation in large environments. Their system—compared to PTAM—does not overload the memory as the after detecting an already known scene, it stops adding new points to the map, and because of the ORB features it has high invariance to viewpoint changes. Despite the advance robustness of their system, it has still weak points. They overcome the viewpoint invariance using ORB features, but these descriptors have only moderate brightness and contrast invariance (Krig, 2014), which means the system faces serious errors

in the case light condition changes, or if they want to extend their system with an image matching function. Also, the method needs a powerful computer for a flawless functioning.

Neubert *et al.* (2015) presented a new technique to treat images to solve the detection between images that have radical condition change (e.g. seasonal variations) predicting the appearance for the season using visual translation dictionaries. To calculate the transition they use superpixels—perceptual grouping of pixels—as image parts and cluster them to vocabularies using a SURF descriptor on these superpixels. As the images are aligned in pixel level, the seasonal vocabularies can be compared to create the transition dictionary. They use this dictionary to predict the look of the query superpixel to other seasons. During the testing period, the authors recorded a significant improvement in the SeqSLAM Milford and Wyeth (2012) algorithm. The main limitation among others that the images have to be pixel aligned in the training session which limits the training datasets.

In summary, the localization and tracking process of the SLAM approach is still under investigation for their massive processing power and memory space consumption. There have been various approaches to resolving the bottleneck in the operation caused by the local feature calculation for image matching, and the data overload by the preprocessed datasets and tracked point information. Strategies use remote servers to lift the massive computing from the mobile device—making dependent the system to internet connection stability—, computationally less expensive descriptors—reducing the invariance to conditional changes—or transformation descriptions between image conditions—with high sensitivity to viewpoint changes—for better tracking. Our attention turned towards the local feature descriptors as we deduced that for the robustness of these SLAM approaches the local descriptor has to be invariant and lightweight.

II.5. Descriptors

For detection or classification, we have to describe the properties of the image in a unique manner. The descriptors ought to have unique, reproducible and invariant properties and must be calculated in an efficient way. "To localize features in images, a local neighborhood of pixels needs to be analyzed, giving all local features some implicit spatial extent" (Tuytelaars and Mikolajczyk, 2008). "The higher the repeatability rate between two images, the more points can potentially be matched,

and the better the matching and recognition results are" (Mikolajczyk and Schmid, 2005). The classification error depends on the texture type and the dimensionality of the descriptors.

Garcia-Fidalgo and Ortiz (2015) reviewed the main approaches published about descriptor calculation methods in the last fifteen years. They categorized the descriptor extraction methods into four categories: global descriptors, local features, Bag-Of-Words (BoW) schemes and combined methods. The global descriptors derive general information from the whole image, which makes these features very fast to compute and used in scene classification without more detailed information about the environment. The local features hold more detailed information, analyzing the images on the pixel level around interest points. This method is slower to compute but excellent tools for image matching and further scene analysis. The BoW technique clusters the local features by their characteristic features, creating visual vocabularies. This approach is used for semantical images segmentation or object recognition.

In this section, we will investigate the latest techniques to resolve the crucial debilities of the local features: invariance and size among others. Table 2.2 and 2.3 groups the primary—most used—feature descriptors according to their type and presenting information about their dimensions and invariance.

Table 2.2: Summary of local float type feature descriptors.

Name	References	Component type	Number of components	Invariant to		
				Rotation	Scale	Affine
SIFT	(Lowe, 1999)	Float	128	✓	✓	
M-SIFT	(Andreasson and Duckett, 2004)	Float	128	✓	✓	
PCA-SIFT	(Ke and Sukthankar, 2004)	Float	36	✓	✓	
GLOH	(Mikolajczyk and Schmid, 2005)	Float	64,128	✓	✓	
SURF	(Bay <i>et al.</i> , 2008)	Float	32,64,128		✓	
U-SURF	(Bay <i>et al.</i> , 2008)	Float	32,64,128	✓	✓	
LESH	(Sarfraz and Hellwich, 2009)	Float	128	✓	✓	
ASIFT	(Yu and Morel, 2009)	Float	128	✓	✓	✓
DAISY	(Tola <i>et al.</i> , 2010)	Float	200	✓	✓	
KAZE	(Alcantarilla, Bartoli <i>et al.</i> , 2012)	Float	64	✓	✓	

Source: Garcia-Fidalgo and Ortiz, 2015

Table 2.3: Summary of local bit type feature descriptors.

Name	References	Component type	Number of components	Invariant to		
				Rotation	Scale	Affine
BRIEF	(Calonder <i>et al.</i> , 2010)	Bit	128,256,512			
BRISK	(Leutenegger <i>et al.</i> , 2011)	Bit	512	✓	✓	
ORB	(Rublee <i>et al.</i> , 2011)	Bit	256	✓	✓	
FREAK	(Alahi <i>et al.</i> , 2012)	Bit	512	✓	✓	
LDASHash	(Strecha <i>et al.</i> , 2012)	Bit	128	✓	✓	
D-BRIEF	(Trzcinski and Lepetit, 2012)	Bit	32	✓	✓	
LDB	(Yang and Cheng, 2012)	Bit	256, 512	✓	✓	
AKAZE	(Alcantarilla, Nuevo <i>et al.</i> , 2013)	Bit	488	✓	✓	
BinBoost	(Trzcinski, Christoudias and Lepetit, 2015)	Bit	64	✓	✓	

Source: Garcia-Fidalgo and Ortiz, 2015

II.5.1. Size Reduction

Fujiwara *et al.* (2013) proposed a method to reduce the local features based on similarity. The goal is to reduce the mismatch in image recognition due to repeated pattern feature. The approach measured the degree of similarity between the extracted feature descriptors and excluded the repeated pattern features in a certain threshold range. They specified a reference point and they measured its similarity with other keypoints using the cosine similarity. The keypoint was labeled and its feature was removed if the similarity was below the threshold. During the experiment, they were able to remove unnecessary features, but the process was still a time-consuming task, as they had to evaluate all features and then specify the optimal threshold value.

Su *et al.* (2013) proposed a more compact local feature—called CGCI-SIFT—, and claimed that their proposed algorithm was more efficient than SIFT and its two size reduced variants (PCA-SIFT and SURF). They assume that the pixel values around the center point are similar, therefore they divided the patch into two sub-regions: an inner region and a peripheral region. The method instead of storing the gradient information of all pixels, from the inner region obtains the gradient histogram information, and from the peripheral region the contrast intensity information. Then it applies a log-polar coordinate system as it is more sensitive to the closer pixels. Their experiments show that the CGCI-SIFT operational time is shorter than the SIFT, PCA-SIFT and SURF with better matching and recall performances.

II.5.2. Invariance

Muselet and Funt (2013) surveyed and presented most of the invariant color features used in color-based object recognition (Table 2.4 and 2.5). They state that "it is important to choose the optimal color invariant for each particular situation because there is a trade-off between invariance and discriminative power." Among other things, they collected various investigations that use these properties to lift the invariance level of existing descriptors. Like the work of Quelhas and Odobez (2006) where the authors concatenated to the PCA-SIFT descriptor the mean and standard deviation of each of the $L^*u^*v^*$ color components.

Table 2.4: Variations to which each color-invariant feature is invariant

Feature	Light int.	Light color	Light dir.	Viewpoint	Amb. light	Shadow/shading	Highlights
CIF_1	✓	✓		✓			
CIF_2	✓	✓	✓	✓			
CIF_3	✓		✓	✓			✓
CIF_4	✓		✓	✓			
CIF_5	✓	✓				✓	
CIF_6	✓	✓					
CIF_7	✓	✓					
CIF_8	✓	✓					
CIF_9	✓	✓			✓		
CIF_{10}	✓	✓					
CIF_{11}	✓	✓			✓		
CIF_{12}	✓	✓			✓		
CIF_{13}	✓		✓	✓			✓
CIF_{14}	✓		✓	✓			
CIF_{15}	✓						
CIF_{16}	✓	✓	✓	✓			
CIF_{17}						✓	
CIF_{18}	✓		✓	✓		✓	
CIF_{19}							✓
CIF_{20}						✓	✓
CIF_{21}	✓		✓	✓		✓	✓

Source: Muselet and Funt, 2013

Doretto and Yao (2010) presents descriptors crafted from region moment. They affirm that the descriptors can reach scale and rotation invariance if the moments and the image features are carefully designed. The authors tested various moments based descriptors: Central Moment (CM) descriptor, based on the central moments of the image features (μ_i); Central Moment Invariant (CMI) descriptor from the Hu moments (Hu, 1962) (ϕ_i); Radial Moment (RM) descriptor from the radial moments; and the Region Covariance (RC) descriptor based on the work of Tuzel *et al.* (2006). The results of their comparative evaluation show, that the CMI, and the RM

Table 2.5: Definition of labels for the color-invariant features (CIF) used in Table 2.4

Label	Description of the color invariant
CIF_1	Component levels ratio (Funt and Finlayson, 1995)
CIF_2	Color invariants m_1, m_2, m_3 (Gevers and Smeulders, 1999)
CIF_3	Color invariants l_1, l_2, l_3 (Gevers and Smeulders, 1999)
CIF_4	Color invariants c_1, c_2, c_3 (Gevers and Smeulders, 1999)
CIF_5	Entropy minimization (Finlayson and Hordley, 2001)
CIF_6	Histogram equalization (Finlayson, Hordley <i>et al.</i> , 2005)
CIF_7	Moment normalization (Lenz <i>et al.</i> , 1999)
CIF_8	Eigenvalue normalization (Healey and Slater, 1994)
CIF_9	Mean and standard deviation normalization
CIF_{10}	Color moments invariant to diagonal transform (Mindru <i>et al.</i> , 2004)
CIF_{11}	Color moments invariant to diagonal transform and translation (Mindru <i>et al.</i> , 2004)
CIF_{12}	Color moments invariant to affine transform (Mindru <i>et al.</i> , 2004)
CIF_{13}	Color-invariant feature H (Geusebroek <i>et al.</i> , 2001)
CIF_{14}	Color-invariant feature C (Geusebroek <i>et al.</i> , 2001)
CIF_{15}	Color-invariant feature W (Geusebroek <i>et al.</i> , 2001)
CIF_{16}	Color-invariant feature N (Geusebroek <i>et al.</i> , 2001)
CIF_{17}	Color feature "quasi-invariant" to shadow/shading (Weijer <i>et al.</i> , 2005)
CIF_{18}	Color feature "full invariant" to shadow/shading (Weijer <i>et al.</i> , 2005)
CIF_{19}	Color feature "quasi-invariant" to highlights (Weijer <i>et al.</i> , 2005)
CIF_{20}	Color feature "quasi-invariant" to shadow/shading and highlights (Weijer <i>et al.</i> , 2005)
CIF_{21}	Color feature "full invariant" to shadow/shading and highlights (Weijer <i>et al.</i> , 2005)

Source: Muselet and Funt, 2013

descriptors with isotropic—having identical property values in all directions—features are the best performers.

Takacs, Chandrasekhar *et al.* (2013) proposed a rotation invariant descriptor called Radial Gradient Transform (RGT) and Approximate RGT (ARGT). Their base concept is to convert the pixel gradient into a local, polar reference frame using two orthogonal basis vectors. They also developed a pipeline to fasten the feature extraction using look-up tables. Their experiments show that the performance with fewer than 8 directions in the ARGT decrease notably, and with more than 8 directions improves slightly, therefore they choose this number during the feature computation. They also showed that although they approach outperformed the state-of-the-art descriptors for retrieval— $16\times$ faster than SURF—, the matching performance stays below the mentioned descriptors. Also, the descriptor is evaluated against rotation variance, but there is no evidence about other invariances.

These works show that to create an all invariant, compact and accurate descriptor is not an easy task as we have seen in Table 2.2 and 2.3. In our work, we took many

elements to build our descriptor. The color invariant features of Muselet and Funt (2013), the gradient information of Lowe (1999), the image moments of Doretto and Yao (2010), or to calculate the local gradients relative to the dominant patch gradient (Takacs, Chandrasekhar *et al.*, 2013).

II.6. Random Forest

There is not a single receipt to choose the correct machine learning technique for image processing algorithm as among others the algorithm accuracy, training and prediction time, and setting complexity has to match the project requirements. In our research, the training time has to be short—as it is running on each genome in each generation—, the classification has to be accurate—because we test segmentation precision—and the algorithm has to cope with large datasets—as we work with a large amount of high dimension descriptor vector. In many types of research that compare supervised learning techniques like Caruana and Niculescu-Mizil (2006), Fernández-Delgado *et al.* (2014), Wahyono and Jo (2014) and Vink and Haan (2015) agree that the leading two algorithms—in terms of accuracy—are the Support Vector Machines (SVM) by Cortes and Vapnik (1995) and the Random Forests (RF) by Breiman (2001). Which learning technique comes out first largely depends on the project or the algorithm setup and the evaluation criteria. We choose the RF because Salhi *et al.* (2012) and Wahyono and Jo (2014) recommends it not only because it has good accuracy, but also has the fastest training and classification time.

Fröhlich *et al.* (2010) presented a pixelwise facade labeling approach using RF. The authors follow the work of Csurka and Perronnin (2008) extending with RF and using Opponent SIFT (Van de Sande *et al.*, 2010) descriptor. They extracted on different image scale the local features, which they classified independently, then the probability of each feature was computed, creating a probability map. The pixel-wise probability was a result of a gaussian smoothing on the probability map. They applied unsupervised segmentation techniques to integrate the results on different scales. They run the method on two different datasets and reported a significant improvement in time efficiency. They also tested the algorithm with color moments descriptor, but it reduced the recognition rates.

Teboul *et al.* (2010)—to create a multi-class facade segmentation algorithm—combined machinel learning techniques with procedural modeling as shape priors. The created shape grammars are constrained to building elements only and then

reduced in complexity by factorization. Then they used RF to link the grammar semantics with the image features. The evaluation of the technique show "promising results", but the technique is highly variant to light condition changes (e.g. shades).

Gondane and Devi (2015) proposed a Probabilistic Random Forest (PRF), using the concept of F-score and roulette wheel selection strategy to construct an ensemble learning model. Their approach—instead selecting randomly—is using the roulette wheel to choose the feature subsets for every tree. The method chooses the most discriminative features with a higher probability in the feature subset. During their experiments, the probabilistic random forest took more time to calculate than the classical random forest, but with higher accuracy.

We learned from the literature that the RF is a high-performance discriminative classifier, handling an extensive set of features without having problems due to the curse of dimensionality (Fröhlich *et al.*, 2010) trained and tested in a short timeframe, which is convenient in our framework.

II.7. Genetic Algorithm

Trujillo, Legrand *et al.* (2010) presented a new type of descriptor optimization technique. They used Genetic Algorithm (GA) on the Hölder descriptor (Trujillo, Olague *et al.*, 2007) with the goal to find the best performing reduced size variant of the canonical Hölder descriptor. They give the task to GA to choose the operating building points of the result descriptor. To evaluate the fitness of each, they used images with rotation transformation and with illumination changes. The authors state that it is possible to reduce the size of the descriptor without losing its invariant features. The resulted in reduced size Hölder descriptor—called H-GA—is near 70% smaller than it's canonical father and gives the same performance.

Behnam and Pourghassem (2013) used the GA to optimize the content-based medical image retrieval (CBMIR) system. The CBMIR system extracts shape and texture-based features independently. They evaluated precision and recall parameters of the retrieval system with different weights on the feature descriptors. Their results show that GA not only improved the accuracy of the scheme but also improved its recall for a predefined number of images.

Lategahn *et al.* (2013) proposed a new approach—called DIRD—to create

multiple descriptor with different characteristics. Their system is based on building blocks that are combined and optimized by the GA based on the fitness score of the actual genome. Each building block serves a function basket (e.g. filter banks holds Sobel filter, Gaussian blurring or Haar features among others) where from the GA chooses the most optimal element for the descriptor. The fitness function evaluated each descriptor on its matching performance under varying illumination conditions using images which were recorded at three different times of the day. The resulted descriptor was tested and compared to the state-of-the-art image descriptors like U-SURF and BRIEF. Their evaluations show that the GA optimized descriptor significantly outperforms the state-of-the-art local features. Though during their experiments they use greyscale images, which are easier to manage, loses valuable color information.

In essence, these papers show that the genetic algorithm can reduce in size, optimize or even create descriptors. The GA parameters can be adjusted uniquely depending on the project necessities. We used the core idea of Lategahn *et al.* (2013), using building block during the descriptor creation, but in a more controllable way. Meantime Lategahn *et al.* (*ibid.*) pools the functions and picks one at each generations randomly, we wanted to control their parameters and presence through performance.

III. THEORETICAL FRAMEWORK

III.1. Introduction

This chapter presents exhaustively the principal theories, applications, and functions for this investigation. Section III.2 outlines the essential components and working principles of the AR systems. Also the display, tracking and registration methods used in AR applications. Section III.3 describes the most investigated SLAM approach in details. The subsequent sections introduce the image processing algorithms, and local visual features the two most important building blocks of the AR mechanisms. Section III.4 outlines the mathematics behind the image processing functions, camera parameters, descriptor matching and the registration. Section III.5 explains the steps behind the most effective corner detector used in the SLAM applications. Section III.6 presents how the most popular feature descriptors—which are used for comparison in this work—organize the extracted low-level data to obtain robust local features. In Section III.7 we explain the method used in the Random Forest classifier followed by the final Section III.8 offering a detailed overview of the elements of the Simple Genetic Algorithm.

III.2. Augmented Reality

The definition of Azuma *et al.* (2001) synthesized AR in three common characteristics:

1. It combines the real and digital objects in a real environment
2. It runs interactively and real time
3. It registers or aligns the real and virtual objects with one and other in 3D

This general definition grasp the concept of the AR very well, but there are more categories and definitions to describe and explain better the large compound of AR applications.



Figure 3.1: Schemes based on the real world and camera relation: (a) World fixed AR, (b) Camera fixed AR

Source: Lovato, 2015; Velez, 2011

The AR systems can be organized in various categories. For the registration of virtual elements to the real objects we have to know the relation and position of the camera and real world. Based on this relationship we can talk about:

- **World Fixed**

Applications where the camera moving freely around the environment (eg. a mobile application) and the real environment is “fixed” (Figure 3.1 (a)).

- **Camera Fixed**

Set-up where the camera is fixed (eg. webcam on a computer for browser based AR) and the real world object (eg. a marker) is moving freely around the environment (Figure 3.1 (b)).

The other important factor is the display technology used in AR. It affects how and what we can show in the application. Billinghamurst *et al.* (2015) categorize the AR displays in two levels. The first is following the taxonomy of Bimber and Raskar (2003) illustrated in Figure 2.6 the possible spatial places where the image can be reproduced, the display can be located on the user and real world object. They categorize the display technology into three different groups:

- **Head attached**

The display is directly connected with or attached to the head, as head mounted optical or video see-through displays or projectors which will be discussed later.

- **Body attached**

The display is still attached to the body but not so close to the head. In this

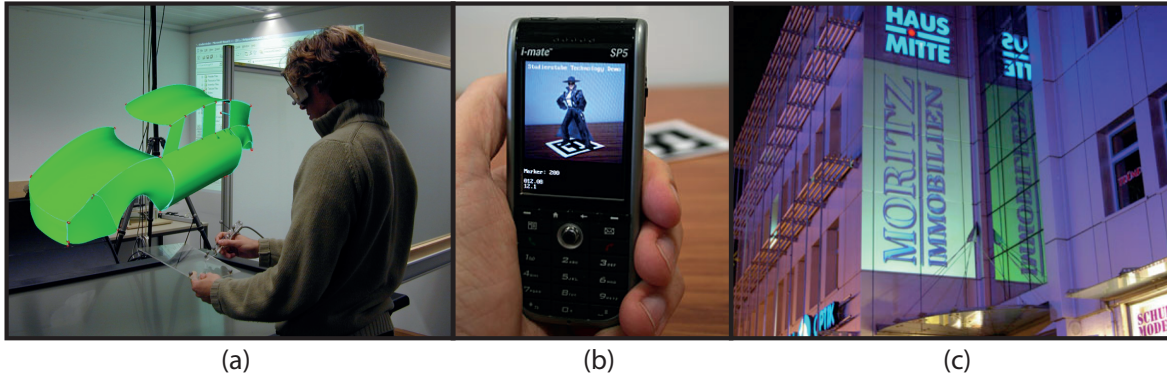


Figure 3.2: Display locations: (a) Head attached display, (b) Body attached display, (c) Spatial display

Source: Billinghurst *et al.*, 2015

class, we talk about the hand-held displays (e.g., mobile phone, tablet) or portable projectors

- **Spatial**

The display is not joined up with the body and can have fair distance between them. Here are the spatial see-through displays(e.g., see-through displays in military cockpits) and the spatially aligned projectors.

The displays also can be categorized via the techniques how the virtual and the real images are mixed. Billinghurst *et al.* (2015) separate the displays also into three classes:

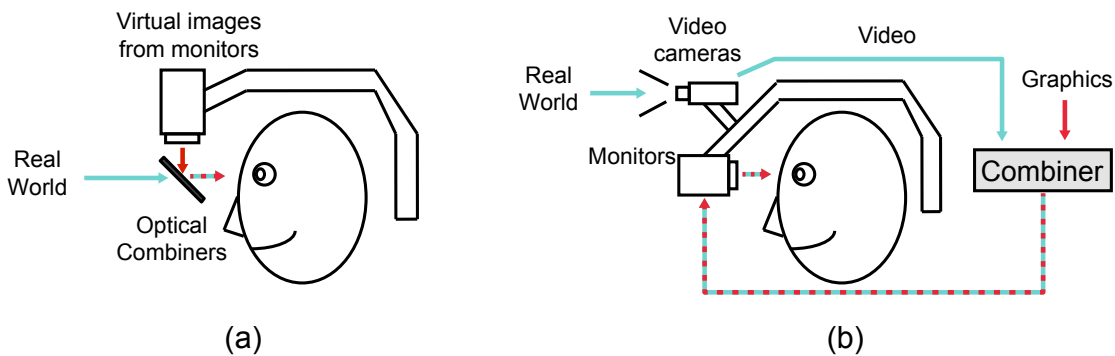


Figure 3.3: HMD schemes: (a) The Video see-through, (b) Optical see-trough
Source: Billinghurst *et al.*, 2015

- **Video see-through displays**

The scene is observed by a camera, and the graphics are overlaid through a

combiner unit on the video stream and sent to the monitors (Figure 3.3 (a)). The strength of this display is that it creates a total occlusion blocking the real world image. To the monitor, the real world arrives as digitalized stream which makes more flexible the composition and offers more registration and calibration strategies. But this approach has its limitations because the camera's narrow field of view and low contrast allow seeing only a fraction of the real world and if the power goes, nothing can be seen. This mixing technique is used also in the hand-held displays and monitor based AR.

- **Optical see-through display**

The user wears a see-through display which contains an optical combiner where the real world image and the digital graphics merged (Figure 3.3 (b)). The advantage of this process is that we have a wide field of view with high contrast and color resolution, and when the power goes the real world is still visible. The significant disadvantages are that the projected virtual objects always appear as translucent, low contrast ghosts (Maimone *et al.*, 2013) floating in front of the actual scene because there is no technology to entirely blocking out the real scene with a beam splitter.

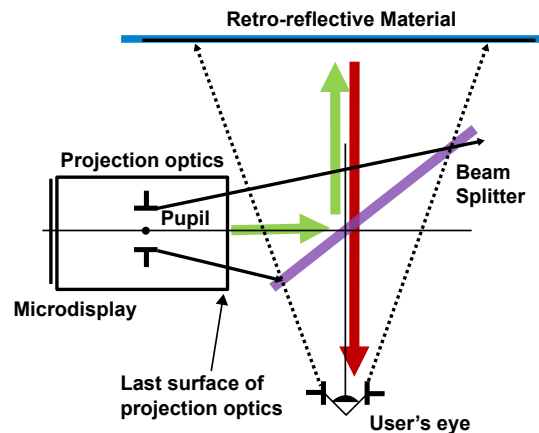


Figure 3.4: The Head Mounted Projective Display (HMPD) scheme
Source: Billinghamurst *et al.*, 2015

- **Projective displays**

The user wears a projector on the head that casts an image onto the scene which reflects back a visible imagery using a reflective material in the scenery (figure 3.4). The advantage of this method is that the combination of real and

virtual appears in the physical real world object in the correct depth, and the reflective material can be used to occlude disruptive elements of the environment. But the retro-reflective tape cannot be put everywhere in the real environment, and another negative point is that only the closer objects can be seen sharp as the intensity falls with the square of the distance.

III.2.1. Tracking

According to Chinthammit *et al.* (2003) in both static and dynamic errors of the Rendering is accounted for the major flaws in the AR systems. In the following we will see how the different tracking technologies try to define the position (x, y, z) and orientation (r, p, y) of the device indoor or outdoor based on the available incoming data.

Table 3.1: Comparison of common tracking technologies. **Range:** size of the region that can be tracked within. **Setup:** amount of time for instrumentation and calibration. **Precision:** granularity of a single output position. **Time:** duration for which useful tracking data is returned (before it drifts too much). **Environment:** where the tracker can be used, indoors or outdoors

Technology	Range (m)	Setup time(h)	Precision (mm)	Time (s)	Environment
<i>Optical: marker-based</i>	10	0	10	∞	in/out
<i>Optical: markerless</i>	50	0-1	10	∞	in/out
<i>Optical: outside-in</i>	10	10	10	∞	in
<i>Optical: inside-out</i>	50	0-1	10	∞	in/out
<i>GPS</i>	∞	0	8,000	∞	out
<i>WiFi</i>	100	10	74,000 (outside) 1,000 (inside)	∞	in/out
<i>Accelerometer</i>	1,000	0	100	1,000	in/out
<i>Magnetic</i>	1	1	1	∞	in/out
<i>Ultrasound</i>	10	1	10	∞	in
<i>Inertial</i>	1	0	1	10	in/out
<i>Hybrid</i>	30	10	1	∞	in/out

Source: Carmigniani and Furht, 2011

Sensory only tracking

These applications use only sensors to estimate the device's pose about the real world. The approach can use a single sensor or a fusion of various sensors to perform the tracking. The most common sensory information that is used in AR applications are the following (Roberto *et al.*, 2016):

Active sensors

We are talking about *active sensors* when it receives external data for tracking and localization. The most common active sensor used in AR:

- (1) The *GPS* receiver is available in most of the latest model of mobile phones. It can provide location, speed, and time information to the client under any weather condition. Most of these mobile phones use Assisted GPS (A-GPS) technology (Zandbergen, 2009). As the cell phones are equipped with a very basic GPS device, the A-GPS processes the "satellite orbit and clock information, the initial position and time estimate, the satellite selection, range and range date, and position computation" (*ibid.*) on a remote server. The GPS and A-GPS regularly fail indoors and is inaccurate in urban areas due to reflections from buildings.
- (2) *WiFi* positioning exploits the WiFi access points (APs) to define position. The WiFi signals normally have an omnidirectional reach of several hundred meters. The approach calculates the position from the APs in range taking advantage that in urban areas the APs has high density, and the signals usually overlap. The method is functioning well indoor environments and densely populated outdoor areas. Also, advantage that the device does not require connection to the WiFi network for the positioning.
- (3) The tracking based on *Cell location* uses the cellular towers as reference point to calculate the position of the device by triangulation. The technique was created to track the device while it is traveling through the network, as it is connecting to the tower with the strongest field strength. The accuracy of this method depends on the tower density, topography, and obstacles among others.

According to Zandbergen (2009) the biggest problem with these sensors is the insufficient position accuracy therefore only usable for applications where precision is not necessary. The measured average median error was 8 meters in the case of *GPS* units, 74 meters through *WiFi* and 600 meters when using *cell towers* for positioning.

Inertial

We call *inertial sensors* the sensors that are measuring the relative change in the device's state of motion.

- (1) The accelerometer detects tri-axis motion changes of the mobile devices. According to Williamson and Murray-Smith (2010) the accelerometers has a relatively fast update rate, but the errors can amplify quickly as each measurement depends on the preceding data. Frequently needed error correction or a high-pass filter to eliminate drifts and inaccuracies.
- (2) Gyroscope returns information about the device's orientation measuring angular velocity. It does not provide information about location only reports values about the yaw, pitch, and roll. This denomination of angles depicted from the aircraft's rotation around the principal axes. The yaw, pitch and roll refer accordingly to the rotation around the z, y, and x-axes.
- (3) The compass give old fashion information about the orientation on the cardinal directions.

Vision only tracking

This tracking method only uses camera image for position and orientation calculation. As we can see on Figure 3.5, the visual tracking pipeline is the following:

- (1) The image frame is being captured by the systems camera. Usually, the image is converted to a one channel greyscale image because it is faster to calculate and cheaper to store.

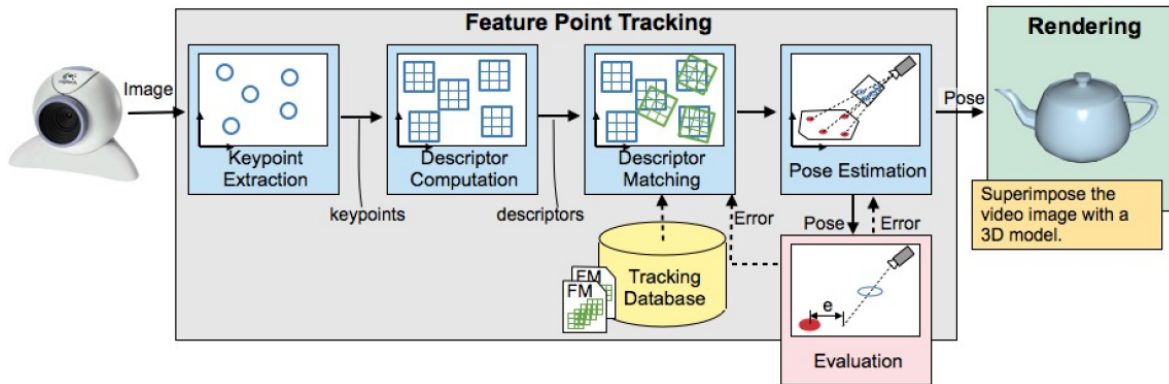


Figure 3.5: Vision only tracking system overview
Source: Radkowski and Oliver, 2013

- (2) A series of image processing function—e.g. histogram equalization or blurring among others—are applied to images to optimize the keypoint detection and descriptor extraction.
- (3) The keypoints are specified. There are many techniques developed during the years, some defines the interest points around the corner features, like FAST (in Section III.5), others combine the keypoint detection with the descriptor extraction, using the prior as distinctive feature or as orientation information, like SIFT and SURF (in Section III.6). Also, some approaches specify the keypoint all over the frame to extract the most information possible.
- (4) At the keypoint coordinates an area is specified from where the algorithm collects a vector of selective and distinctive information (in Section III.6). The information has to be invariant to light condition and geometrical changes as in the sequential frames the descriptor is matched with vectors that are taken from images captured under different light conditions and camera orientation. The composition of the descriptor has to be balanced as its calculation cannot create latency, and its features cannot cause misalignment. In other words it has to be small in size and robust in description (see the size and characteristics of latest and most used descriptor in Table 2.2 and 2.3).
- (5) The system matches the calculated descriptors with descriptors extracted from a visual marker or with descriptors saved in previous frames into a parallel map. Either way they are compared calculating the vector distance between the descriptors using Brute Force (Section III.4.6.1) or FLANN (Section III.4.6.2) technique. The distance measure depends on the descriptor value type. If the descriptors are floating point types—like the descriptors from Table 2.2—the

algorithm uses Euclidean distance , in the case of binary descriptors—like the examples from Table 2.3—the Hamming Distance is used.

- (6) After the algorithm matched from different frames the same keypoints, the camera pose for localization and the Homography matrix for transformation is being estimated (Section III.4.8).
- (7) In the last step, the Homography matrix is used for the conversion of the virtual object to align with the marker or natural feature.

These steps are taken in a loop along the functioning of the AR application.

Sensory and vision based tracking

These mixed tracking methods use the same pipeline for tracking, but with the aid of one or multiple sensory. The sensory information adjusts and completes the visual tracking information to achieve a more robust registration. For example, Kurz and Benhimane (2011) proposed a mixed tracking technique was calculating gravity vector at the feature point. First, the approach calculates the gravity vector of the internal sensor and the ground truth

$$g_{sensor} = \begin{bmatrix} \cos \alpha_s \cos \beta_s \\ \sin \alpha_s \cos \beta_s \\ \sin \beta_s \end{bmatrix} \quad \text{and} \quad g_{truth} = \begin{bmatrix} \cos \alpha_g \cos \beta_g \\ \sin \alpha_g \cos \beta_g \\ \sin \beta_g \end{bmatrix} \quad (III.1)$$

and then the gravity vector is synthesized as

$$g_{synth}(t) = \begin{bmatrix} \cos(\alpha_g(t + \delta^*) + X) \cos(\beta_g(t + \delta^*) + Y) \\ \sin(\alpha_g(t + \delta^*) + X) \cos(\beta_g(t + \delta^*) + Y) \\ \sin(\beta_g(t + \delta^*) + Y) \end{bmatrix} \quad (III.2)$$

Another example is the sensor fusion with PTAM by Porzi *et al.* (2012). Here the base approach is to improve PTAM's positioning estimation using the mobile device's accelerometer and gyroscope. The problem of PTAM that the pose estimation gets increasingly difficult with the time as the recorded data exponentially growing and the pose estimation function has to compare more and more information.

Distributed tracking

The distributed tracking approaches present the most sophisticated tracking structure among all. As we can observe on Figure 3.6, the technique pushes one step ahead the tracking, exploiting all the resources a mobile device can offer. The separated client and cloud modules serve different purposes. The client side used only acquisition and display while the server takes care of the heavy calculation. The information from the camera and the inertial sensors are sent to the server for pose estimation and localization to the remote server. The system here runs the descriptor acquisition, matching, pose estimation, and localization before sending back to the client the necessary information for display. The framework weakness also comes from the distributed workflow, as the system's speed mainly depends on the wireless connection.

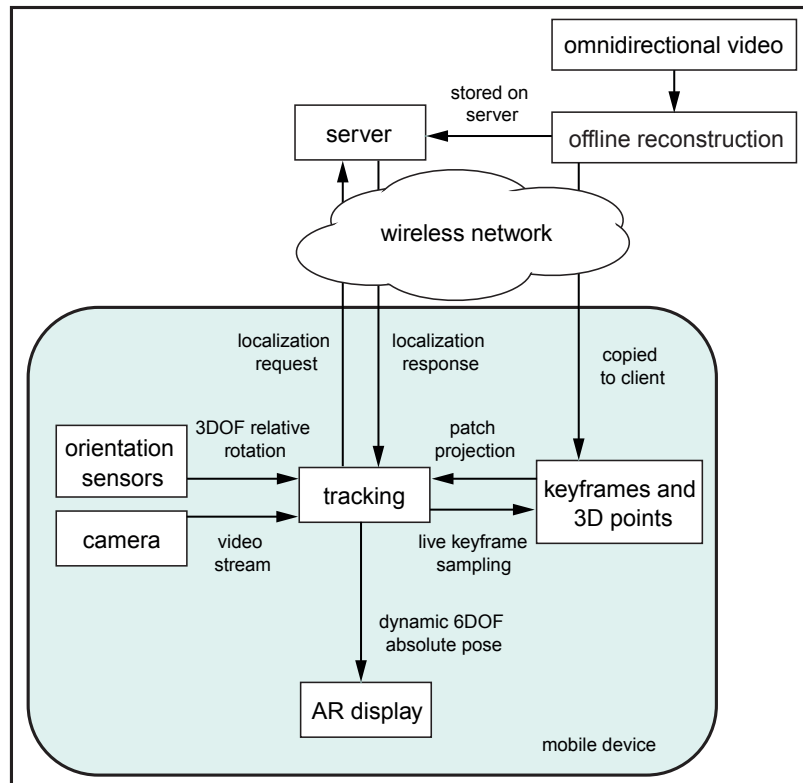


Figure 3.6: Distributed Tracking System overview
Source: Ventura and Hollerer, 2012

III.2.2. Registration

To create plausible and convincing AR application, it is inevitable that the registration being accurate and robust. The registration is the precise alignment of real and virtual objects. The aim of the registration is to anchor the virtual 2D planes or 3D objects frame by frame to the real world marker or natural feature. It is necessary that the application solves all the obstacles that cause the registration's static and dynamic errors. To resolve the static errors the camera has to be calibrated precisely, the extrinsic parameters, the detected points 3D world coordinates and consequently the homography matrix (Section III.4.8) needs to be calculated accurately. The latency that causes the dynamic errors in the application can be resolved by choosing the pipeline elements well. In other words robust and fast keypoint detector and descriptor calculator, suitable for the devices parameters and for the environment.

III.3. Simultaneous Localization And Mapping

III.3.1. Parallel Tracking And Mapping

The reason Klein and Murray (2007) ground breaking work on Parallel Tracking And Mapping has been chosen for the project not only for the ability of tracking and translating feature points of an unknown the environment into 3D map point, but for its architecture which offers the opportunity to run data processing without breaking the feature tracking, and for the large amount of data presented by the system. The essence of the method is encapsulated in the name of the system. The feature tracking and the map making are separated into two parallel threads.

Tracker

The tracking is processing every frame received from a hand-held camera and estimates the camera pose relative to the map. The initialization sets up the base of the map which used by the tracker throughout the course. The following steps are taken during the tracking:

1. The incoming images are converted into a four-level greyscale pyramid. On each level they run the FAST-10 corner detector by Rosten and Drummond

(2006), and estimate a prior for the camera pose E_{CW} in world coordinate frame \mathcal{W} . This 4×4 matrix contains the rotation and translation vector of the camera.

2. The map points are projected firstly by transforming the map points from the world coordinates frame to the camera-centred coordinate frame using the previously estimated camera pose E_{CW} . After that using a calibrated camera projection model to project the points to the image. The changes in the camera pose is represented by a 4×4 M camera motion matrix. $E'_{CW} = ME_{CW}$
3. The current frame is searched for 50 map points over a large radius. A fixed range search is done around the predicted locations on each pyramid level, to be able to find the map point on the image.
4. After the successful search, the new camera pose is computed and updated from the found patch positions.
5. After the new camera pose is computed, the system re-projects a far larger number of potentially visible image patches and performs a tight patch search on the same image.
6. The system evaluates the tracking quality. If it falls under a certain threshold, the tracker runs as normal but is not allowed to send a new keyframe to the mapmaker. If the poor tracking continues for a few frames and the system is not able to connect with previous keyframes, the tracking is considered lost and a recovery initiated. If the tracking is right, the final pose estimate is computed for the frame.

As on Figure 3.7 we can observe as the system projects back the tracked feature points to the actual frame. This feature can be switched off when a virtual object is augmented. The colours of the points represent the level of pyramid they were tracked. The red colour represents the features tracked on the top level of the pyramid, meanwhile the other colours are the reference to the feature patches tracked in lower level. They are only references because the actual pixel information of patch is not stored in the system, just receiving a reference point on the top level of the pyramid.

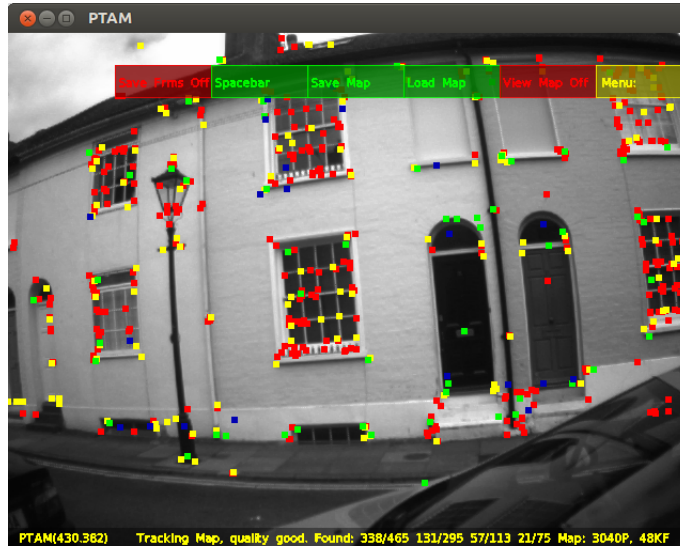


Figure 3.7: Tracked point overlaid on image
Source: Takács, 2013

The Mapmaker

The simultaneously running Mapmaker is being initialised from a stereo pair image and extended with every newly added keyframe. The initialisation is done by the user panning the camera horizontally minimum 10 cm to get a good pair of image (Figure 3.8.). 1000 2D FAST corners are tracked between the pair from which the five-point algorithm and RANSAC estimate and triangulate the base map. After the initialization, the system inserts a keyframe passing a minimum distance in space and time after the last keyframe and with epipolar search the correspondence is established with the new keyframe. Because the full bundle adjustment for all the keyframes would be an increasingly expensive problem in a steadily growing map, local bundle adjustment is used after the first two frames. The pose of a subset of keyframes, the most recent and its closest neighbours, are adjusted. The Map consists a collection of data:

1. M point features, each j th point:

- represents a locally planar 8×8 square texture patch
- has coordinates $p_{j\mathcal{W}} = (x_{j\mathcal{W}} \ y_{j\mathcal{W}} \ z_{j\mathcal{W}} \ 1)^T$ in coordinate frame \mathcal{W}
- has unit patch normal n_j
- if the point is not from the top level of the image pyramid, has a reference to the Source patch on the Source level of the pyramid



Figure 3.8: Initialization
Source: Takács, 2013

2. N keyframes each contains:

- snapshot taken by the camera
- a four levels pyramid of greyscale images, the sub-samples of the snapshot, constructed in the tracking
- the transformation between a unique camera-centred coordinate frame and the world

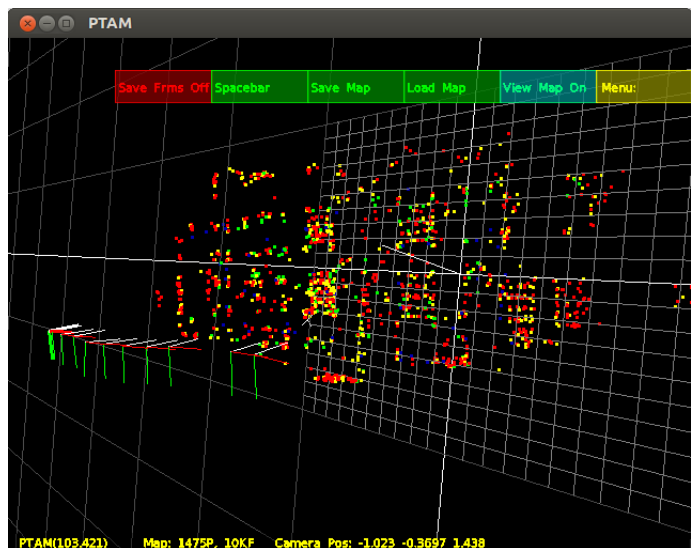


Figure 3.9: Map points
Source: Takács, 2013

Despite the robustness of the application, Klein and Murray (2007) describe various ways when the system can fail:

- The tracking
 - Rapid camera movements decimates the tracked corner features in the image.
 - Repeated structure produce large number of outliers
- The mapping
 - When the inserted information into the map is incorrect
 - When the "real-world scene" changes substantially and permanently.

And most importantly PTAM does not contain information about object's type and nature where the points are detected from. That structural information could help for the system to recover from a tracking and registration failure such as Guan and Wang (2009) proposed in their work.

To be able to overcome this flaw in the system, and make "visible" for the system these features, we aim to create an auto-segmentation function extending PTAM's arms towards the wall.

As PTAM runs parallel the tracking and the map making, and the tracking has to run and exceed at least 20 frames before adding a new keyframe to the map, the map making thread has free time to do computation without stopping the tracking from running. Therefore, the project is focusing on the adjustment of the map maker. To enhance PTAM's senses first, we have to structure the existing point cloud into clusters make and then train PTAM to be able to segment this already segmented data into more meaningful classes.

III.4. Mathematics behind the application

In this section we will present the mathematics behind the research algorithm. First we will introduce the image processing functions used in the preparation and descriptor calculation period. The comparison method that used for matching evaluation Also we will explain the geometry behind the camera, that is used for alignment, projection and detection.

III.4.1. Image Processing

These functions are important elements during the preparation for the local feature acquisition. All of these function has its own function to extract the most distinctive feature from the image.

Color Spaces

Most the classical AR systems and CV applications (e.g. PTAM (Klein and Murray, 2007) or SIFT (Lowe, 2004)) use greyscale images for image processing and information acquisition as it is convenient in size and computing. On the other hand, the color data is relevant in CV because it offers extra information about the same environment that can be exploited during the process. Along the years has been developed different theories to descodes the same tristimulus light wave. In the following list, we will present the color channels that we considered during the investigation.

RGB

The most common additive colour model with Red, Green, and Blue channels. The color camera obtains the R, G, B values through

$$R = \int_{\lambda} E(\lambda)S(\lambda)f_R(\lambda)d\lambda, \quad G = \int_{\lambda} E(\lambda)S(\lambda)f_G(\lambda)d\lambda, \quad B = \int_{\lambda} E(\lambda)S(\lambda)f_B(\lambda)d\lambda \quad (\text{III.3})$$

where $E(\lambda)$ is the spectral power distribution, $S(\lambda)$ is the light reflected by objects and three separate color matching function $f_C(\lambda)$ for $C \in \{R, G, B\}$ of the camera or observer. According to Shih and Liu (2005) the RGB channels are sensitive to luminance, surface orientation, and other photographic conditions. Therefore the RGB values are normalized to the rgb space to reduce the sensitivity.

$$r = \frac{R}{(R + G + B)}, \quad g = \frac{G}{(R + G + B)}, \quad b = \frac{B}{(R + G + B)} \quad (\text{III.4})$$

CIE XYZ

As the RGB values cannot represent all visible colors, using only positive values, therefore Commission Internationale de l'Éclairage (CIE) in 1931 defined the XYZ . The tristimulus values for a color with a spectral power distribution $E(\lambda)$, are given in terms of the standard observer by:

$$X = \int_{\lambda} E(\lambda)S(\lambda)\bar{x}(\lambda)d\lambda, \quad Y = \int_{\lambda} E(\lambda)S(\lambda)\bar{y}(\lambda)d\lambda, \quad Z = \int_{\lambda} E(\lambda)S(\lambda)\bar{z}(\lambda)d\lambda \quad (\text{III.5})$$

where \bar{x}, \bar{y} and \bar{z} are the CIE color matching functions of the 2° CIO standard observer (Gevers, 2001). The other way is to compute based on the RGB values:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.607 & 0.174 & 0.200 \\ 0.299 & 0.587 & 0.114 \\ 0.000 & 0.066 & 1.116 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (\text{III.6})$$

These color features are hardly visible for human eye, as the they are too saturated. The characteristics of the XYZ system (Stanescu *et al.*, 2009): the luminance is represented only by the Y value, device independent, not perceptual uniform, not intuitive, linear transformation, dependent on viewing direction, object geometry, direction, intensity or color of the illumination and highlights.

CIE $L^*u^*v^*$

This color model is a simple-to-compute transformation of the CIE XYZ color space trying to achieve perceptual uniformity. L for lightness—determined only by Y value—and (u, v) for chromaticity.

$$L^* = \begin{cases} (\frac{29}{3})^3 Y/Y_n, & Y/Y_n \leq (\frac{6}{29})^3 \\ 116(Y/Y_n)^{\frac{1}{3}} - 16, & Y/Y_n > (\frac{6}{29})^3 \end{cases}$$

$$u^* = 13L^* \cdot (u' - u'_n)$$

$$v^* = 13L^* \cdot (v' - v'_n)$$

where

$$\begin{aligned} u' &= \frac{4X}{X + 15Y + 3Z} \\ v' &= \frac{9Y}{X + 15Y + 3Z} \end{aligned} \quad (\text{III.7})$$

device independent, perceptual uniform, not intuitive, nonlinear transformation and dependent on viewing direction, object geometry, direction, intensity or color of the illumination and highlights.

CIE $L^*a^*b^*$

Has similar characteristics as the $L^*u^*v^*$ model, L for lightness, a and b for the color-opponent dimensions, based on non-linearly compressed CIE XYZ color space coordinates

$$\begin{aligned} L^* &= 116f(Y/Y_n) - 16 \\ a^* &= 500 [f(X/X_n) - f(Y/Y_n)] \\ b^* &= 200 [f(Y/Y_n) - f(Z/Z_n)] \end{aligned}$$

where

$$f(t) = \begin{cases} t^{\frac{1}{3}} & \text{if } t > (\frac{6}{29})^3 \\ \frac{1}{3}(\frac{29}{6})^2 t + \frac{4}{29} & \text{otherwise} \end{cases} \quad (\text{III.8})$$

The characteristics of both CIE $L^*u^*v^*$ and CIE $L^*a^*b^*$ according to Stanescu *et al.* (2009): device independent, perceptual uniform, not intuitive, nonlinear transformation and dependent on viewing direction, object geometry, direction, intensity or color of the illumination and highlights.

$YC_B C_R$

In this model the Y is the luma component and C_B and C_R are the blue-difference and red-difference chroma components calculated from the RGB color

space. The $YC_B C_R$ is the digital version of the YUV color model. Its characteristics (Chiang *et al.*, 2012): is a linear transformation and it is easier for design and operation in the hardware implementation, the current display devices for LCD-TV, digital TV, and HDTV all use the $YCbCr$ color space model.

$$\begin{aligned}
 Y &= 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \\
 C_B &= (B - Y) \cdot 0.564 + \text{delta} \\
 C_R &= (R - Y) \cdot 0.713 + \text{delta}
 \end{aligned}$$

where

$$\text{delta} = \begin{cases} 128 & \text{for 8-bit images} \\ 3278 & \text{for 16-bit images} \\ 0.5 & \text{for floating-point images} \end{cases} \quad (\text{III.9})$$

HSV

Cylindrical representation colour space with channel H for Hue—the color—, S Saturation—the intensity of the color—, and V for Value—the brightness of the color—. This model describes the colors in more visually understandable terms, close to the artists' terms.

$$\begin{aligned}
 V &= \max(R, G, B) \\
 S &= \begin{cases} \frac{V - \min(R, G, B)}{V} & \text{if } V \neq 0 \\ 0 & \text{otherwise} \end{cases} \\
 H &= \begin{cases} 60(G - B)/(V - \min(R, G, B)) & \text{if } V = R \\ 120 + 60(B - R)/(V - \min(R, G, B)) & \text{if } V = G \\ 240 + 60(R - G)/(V - \min(R, G, B)) & \text{if } V = B \end{cases} \quad (\text{III.10})
 \end{aligned}$$

The characteristics of this space (Stanescu *et al.*, 2009):device dependent, not perceptual uniform, intuitive, nonlinear transformation.

$$\text{If } H < 0 \text{ then } H = H + 360$$

HSL

Cylindrical representation colour space with channel H for Hue, S Saturation, and L for Lightness. It is really similar to the HSV color model. Apart of the name "lightness" replaces the "brightness" the difference between the two color mode is that the lightness of a pure color is equal to the lightness of a medium gray, meantime the brightness is equal to the brightness of white.

$$\begin{aligned}V_{max} &= \max(R, G, B) \\V_{min} &= \min(R, G, B) \\L &= \frac{V_{max} + V_{min}}{2} \\S &= \begin{cases} \frac{V_{max}-V_{min}}{V_{max}+V_{min}} & \text{if } L < 0.5 \\ \frac{V_{max}-V_{min}}{2-(V_{max}+V_{min})} & \text{if } L \geq 0.5 \end{cases} \\H &= \begin{cases} 60(G - B)/S & \text{if } V_{max} = R \\ 120 + 60(B - R)/S & \text{if } V_{max} = G \\ 240 + 60(R - G)/S & \text{if } V_{max} = B \end{cases} \end{aligned} \quad (III.11)$$

If $H < 0$ then $H = H + 360$

Opponent RGB

The Opponent RGB color channels, where the color space calculated from the basic R, G, B values and contains one intensity (O_3) and two chromaticity or color channels (O_1 and O_2).

$$\begin{pmatrix} O_1 \\ O_2 \\ O_3 \end{pmatrix} = \begin{pmatrix} \frac{R-G}{\sqrt{2}} \\ \frac{R+G-2B}{\sqrt{6}} \\ \frac{R+G+B}{\sqrt{3}} \end{pmatrix} \quad (III.12)$$

According to Van de Sande *et al.* (2010) the model is invariant to light intensity changes and shifts.

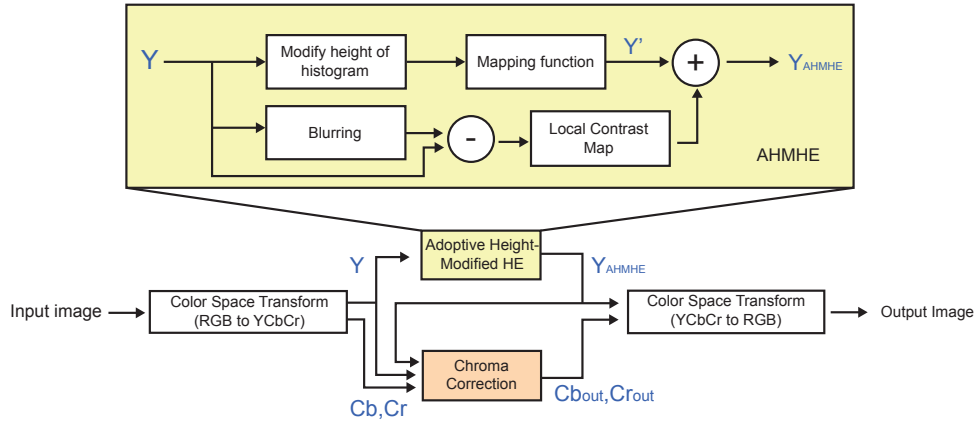


Figure 3.10: Adaptive Height-Modified Histogram Equalization
Source: Kang *et al.*, 2011

Histogram equalization

To create more illumination balanced images, the function calculates the probability density ($P(k)$) of the image from which remaps the pixel values.

$$P(k) = \frac{h(k)}{N} \quad \text{for } k = 0, 1, \dots, L - 1 \quad (\text{III.13})$$

where $h(k)$ is the number of pixels with gray-level k and N is the total number of pixels. The cumulative density function can be obtained:

$$C(k) = \sum_{m=0}^k P(m) \quad \text{for } k = 0, 1, \dots, L - 1 \quad (\text{III.14})$$

Finally the mapping function is calculated from $C(k)$

$$\tilde{k} = (L - 1) \times C(k) \quad (\text{III.15})$$

where \tilde{k} is the remapped luminance level for k input value. In the case of color images Kang *et al.* (2011) uses YCbCr color space as it preserves detailed information of luminance component better than any other color spaces. We make the histogram Equalization on the Y intensity plane and also chroma correction (Eq. III.16) to enhance color purity before transform back to the RGB color space (Figure 3.10).

$$\begin{aligned}
Y_{out} &= Y_{AHMHE} \\
Cb_{out} &= s \times (Y_{AHMHE}/Y_{in}) \times (Cb_{in} - 128) + 128 \\
Cr_{out} &= s \times (Y_{AHMHE}/Y_{in}) \times (Cr_{in} - 128) + 128
\end{aligned} \tag{III.16}$$

Where 128 is subtracted from the chroma components, then multiplied with the Y_{AHMHE} and Y_{in} ration. Next step, the values are multiplied by parameter s where $s > 1$, and finally add 128 to get the corrected chroma components (Cb, Cr).

Gaussian blur

The Gaussian filter is used to smooth the image and clean the noise from the image. In general form the function follows the Gaussian distribution in concentric circles. Parameter σ define how flattened is the Gaussian curve around point x, y , in other words the larger the sigma, the larger is the area modified by the function.

$$G_{2D}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{III.17}$$

The method adds to the center pixel from its value multiplied by the kernel weight. In image processing the calculation is done with filters where the kernel size is $(2k + 1) \times (2k + 1)$. As it follows the probability function the kernel values always have to add up to 1. For example (with $\sigma = 1.4$):

$$B = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * A \tag{III.18}$$

Where A is the incoming image and B is the blurred image.

Pixel Gradient calculation

The gradient of an image is used for obtain information from images. The gradient measures the intensity changes on pixel level on x and y direction. The

general form is the following

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}, \quad \theta = \tan^{-1} \left[\frac{g_y}{g_x} \right] \quad (\text{III.19})$$

where the $\frac{\partial f}{\partial x}$ returns the gradient in x direction and $\frac{\partial f}{\partial y}$ in y direction. Then the gradient direction θ is calculated from:

$$\theta = \tan^{-1} \left[\frac{g_y}{g_x} \right] \quad (\text{III.20})$$

Normally used convolutional filter in both direction to extract such information. The most used filters are the Sobel operator (Eq. III.21) and the Prewitt operator (Eq. III.22). Prewitt is much simpler to calculate but is more sensitive to noise.

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * A \quad \text{and} \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * A \quad (\text{III.21})$$

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} * A \quad \text{and} \quad G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} * A \quad (\text{III.22})$$

Canny edge detection

The method was proposed by Canny (1986), detects structural information about objects on the image. The Canny detectors follow the following steps:

1. The algorithm blurs the image with Gaussian blur to remove noise and unnecessary details.
2. Calculates intensity gradients on the image running Sobel or Prewitt operators
3. Applies non-maximum suppression to remove the points that are not part of the local maxima.

4. Applies double threshold to determine potential edges
5. Tracking edges by hysteresis, following the candidates in orthogonal direction to the gradient, until the gradient magnitude drops below a threshold.

Image transformation

During the project we used two type image transformation matrices, the Rotation and Affine Transformation matrices that are calculated by OpenCV (Bradski, 2000).

Rotation and Resize

The matrix is calculated using the following parameters: the *center* of rotation in the image, the rotation *angles*, and the isotropic *scale* factor to change the size of the image.

$$M = \begin{bmatrix} \alpha & \beta & (1 - \alpha) \cdot \text{center}.x - \beta \cdot \text{center}.y \\ -\beta & \alpha & \beta \cdot \text{center}.x + (1 - \alpha) \cdot \text{center}.y \end{bmatrix}$$

where

$$\begin{aligned} \alpha &= \text{scale} \cdot \cos \text{angle} \\ \beta &= \text{scale} \cdot \sin \text{angle} \end{aligned} \tag{III.23}$$

Affine Transformation

The Affine Transformation Matrix builds up from rotation matrix A and the translation matrix B . The result 2×3 matrix is used for the affine transformation.

$$\begin{aligned} A &= \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}_{2 \times 2} & B &= \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix}_{2 \times 1} \\ T &= A \cdot \begin{bmatrix} x \\ y \end{bmatrix} + B = \begin{bmatrix} a_{00}x + a_{01}y + b_{00} \\ a_{10}x + a_{11}y + b_{10} \end{bmatrix} \end{aligned} \tag{III.24}$$

III.4.2. Invariant features

Invariant Color features

Gevers' invariant features

Gevers and Smeulders (1999) presented a color model l_1, l_2, l_3 is proposed uniquely determining the direction of the triangular color plane in RGB-space. According to Stanescu *et al.* (2009) the feature characteristics are: nonlinear transformation, object geometry, direction and intensity of the illumination and highlights.

$$\begin{aligned}l_1 &= \frac{(R - G)^2}{(R - G)^2 - (R - B)^2 - (G - B)^2}, \\l_2 &= \frac{(R - B)^2}{(R - G)^2 - (R - B)^2 - (G - B)^2}, \\l_3 &= \frac{(G - B)^2}{(R - G)^2 - (R - B)^2 - (G - B)^2}\end{aligned}\tag{III.25}$$

Gevers and Smeulders (1999) proposed another group of invariant features, denoted c_1, c_2, c_3 . These are photometric color invariant features for Matte, Dull surfaces, with the assumption that for a "matte surface under ideal white illumination, the ratio of color component pairs is invariant to the light's intensity and direction, and to the view direction"(Muselet and Funt, 2013).

$$\begin{aligned}c_1 &= \arctan\left(\frac{R}{\max\{G, B\}}\right), \\c_2 &= \arctan\left(\frac{G}{\max\{R, B\}}\right), \\c_3 &= \arctan\left(\frac{B}{\max\{R, G\}}\right)\end{aligned}\tag{III.26}$$

Geusebroek's invariant features

Geusebroek *et al.* (2001) based the color invariant features on the Gaussian color model, where the \hat{E} , \hat{E}_λ , and $\hat{E}_{\lambda\lambda}$ represents the three sensors whose spectral

sensitivities are the 0, 1st, and 2nd-order derivatives of the Gaussian function. The Gaussian color model calculated from the RGB terms:

$$\begin{bmatrix} \hat{E} \\ \hat{E}_\lambda \\ \hat{E}_{\lambda\lambda} \end{bmatrix} = \begin{pmatrix} 0.06 & 0.63 & 0.27 \\ 0.3 & 0.04 & -0.35 \\ 0.34 & -0.6 & 0.17 \end{pmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (\text{III.27})$$

The ratio at pixel P of the Gaussian color components $\hat{E}_\lambda(P)$ and $\hat{E}_{\lambda\lambda}(P)$ is independent of the intensity and incident angle of the light as well as of the view direction and the presence of specular highlights for a surface with neutral interface reflection under ideal white illumination (Muselet and Funt, 2013).

$$H = \frac{\hat{E}_\lambda(P)}{\hat{E}_{\lambda\lambda}(P)} \quad (\text{III.28})$$

The ratio between the components $\hat{E}_\lambda(P)$ and $\hat{E}(P)$ is independent of the light intensity or direction of the incident light and view direction.

$$C = \frac{\hat{E}_\lambda(P)}{\hat{E}(P)} \quad (\text{III.29})$$

Mean and Standard deviation

The mean (μ) calculates the average value of an image or patch, summing up the pixel values ($I(x, y)$) and divided by the number pixels (N). The standard deviation (σ) defines the dispersion around the mean.

$$\mu = \frac{1}{N} \sum_x \sum_y I(x, y), \quad \sigma = \sqrt{\frac{1}{N} \sum_x \sum_y I(x, y) - \mu} \quad (\text{III.30})$$

Central Moments

To get shape information we calculate the normalized central moments for each channel over the image patch. At first we calculate the zero-order M_{00} —represents the total mass of the image or patch—and the two first-order M_{10}, M_{01}

raw image moments.

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y) \quad (\text{III.31})$$

then the two components of the mass center (\bar{x}, \bar{y}) :

$$\bar{x} = \frac{M_{10}}{M_{00}}, \quad \bar{y} = \frac{M_{01}}{M_{00}} \quad (\text{III.32})$$

The Central moment—where the sum of the $p, q \in \mathbb{N}$ subscript of μ_{pq} specify the order of the moment—defined as:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I(x, y) \quad (\text{III.33})$$

Finally the moment is normalized by a proper power of μ_{00} :

$$\tilde{\mu}_{pq} = \frac{\mu_{pq}}{\mu_{00}^w}, \quad \text{where } w = \frac{p+q}{2} + 1 \quad (\text{III.34})$$

The following characteristics have the different order central moments:

- 2nd order central moments describe the variance and measure 2D elliptical shape.
- 3rd order central moments give symmetry information about the 2D shape or skewness.
- 4th order central moments define 2D distribution as tall, short, thin or fat .
- 5th order central moments calculates further shape parameters.

Hu moments

Based on normalized central moments ($\tilde{\mu}_{pq}$), Hu (1962) introduced seven moments, that are invariant to translation, scale and an in-plane rotation around the origin.

$$\phi_1 = \tilde{\mu}_{20} + \tilde{\mu}_{02}, \quad (\text{III.35a})$$

$$\phi_2 = (\tilde{\mu}_{20} - \tilde{\mu}_{02})^2 + \tilde{\mu}_{11}^2, \quad (\text{III.35b})$$

$$\phi_3 = (\tilde{\mu}_{30} - \tilde{\mu}_{12})^2 + (\tilde{\mu}_{21} - \tilde{\mu}_{03})^2, \quad (\text{III.35c})$$

$$\phi_4 = (\tilde{\mu}_{30} + \tilde{\mu}_{12})^2 + (\tilde{\mu}_{21} + \tilde{\mu}_{03})^2, \quad (\text{III.35d})$$

$$\begin{aligned} \phi_5 = & (\tilde{\mu}_{30} - 3\tilde{\mu}_{12})(\tilde{\mu}_{30} + \tilde{\mu}_{12}) \left[(\tilde{\mu}_{30} + \tilde{\mu}_{12})^2 - 3(\tilde{\mu}_{21} + \tilde{\mu}_{03})^2 \right] \\ & + (3\tilde{\mu}_{21} - \tilde{\mu}_{03})(\tilde{\mu}_{21} + \tilde{\mu}_{03}) \left[3(\tilde{\mu}_{30} + \tilde{\mu}_{12})^2 - (\tilde{\mu}_{21} + \tilde{\mu}_{03})^2 \right], \end{aligned} \quad (\text{III.35e})$$

$$\begin{aligned} \phi_6 = & (\tilde{\mu}_{20} - \tilde{\mu}_{02}) \left[(\tilde{\mu}_{30} + \tilde{\mu}_{12})^2 - (\tilde{\mu}_{21} + \tilde{\mu}_{03})^2 \right] \\ & + 4\tilde{\mu}_{11}(\tilde{\mu}_{30} + \tilde{\mu}_{12})(\tilde{\mu}_{21} + \tilde{\mu}_{03}), \end{aligned} \quad (\text{III.35f})$$

$$\begin{aligned} \phi_7 = & (3\tilde{\mu}_{21} - \tilde{\mu}_{03})(\tilde{\mu}_{30} + \tilde{\mu}_{12}) \left[(\tilde{\mu}_{30} + \tilde{\mu}_{12})^2 - 3(\tilde{\mu}_{21} + \tilde{\mu}_{03})^2 \right] \\ & - (\tilde{\mu}_{30} - 3\tilde{\mu}_{12})(\tilde{\mu}_{21} + \tilde{\mu}_{03}) \left[3(\tilde{\mu}_{30} + \tilde{\mu}_{12})^2 - (\tilde{\mu}_{21} + \tilde{\mu}_{03})^2 \right], \end{aligned} \quad (\text{III.35g})$$

Affine Moment Invariant

Suk and Flusser (2004) presented 10 affine moment invariant feature up to the 5th order in explicit forms based on central moments(μ_{pq}). They recommended for object recognition application as they provide discrimination power.

$$I_1 = (\mu_{20}\mu_{02} - \mu_{11}^2) / \mu_{00}^4, \quad (\text{III.36a})$$

$$I_2 = (-\mu_{30}^2\mu_{03}^2 + 6\mu_{30}\mu_{21}\mu_{12}\mu_{03} - 4\mu_{30}\mu_{12}^3 - 4\mu_{21}^3\mu_{03} + 3\mu_{21}^2\mu_{12}^2) / \mu_{00}^{10}, \quad (\text{III.36b})$$

$$\begin{aligned} I_3 = & (\mu_{20}\mu_{21}\mu_{03} - \mu_{20}\mu_{12}^2 - \mu_{11}\mu_{30}\mu_{03} + \mu_{11}\mu_{21}\mu_{12} \\ & + \mu_{02}\mu_{30}\mu_{12} - \mu_{02}\mu_{21}^2) / \mu_{00}^7, \end{aligned} \quad (\text{III.36c})$$

$$\begin{aligned} I_4 = & (-\mu_{20}^3\mu_{03}^2 + 6\mu_{20}^2\mu_{11}\mu_{12}\mu_{03} - 3\mu_{20}^2\mu_{02}\mu_{12}^2 - 6\mu_{20}\mu_{11}^2\mu_{21}\mu_{03} \\ & - 6\mu_{20}\mu_{11}^2\mu_{12}^2 + 12\mu_{20}\mu_{11}\mu_{02}\mu_{21}\mu_{12} - 3\mu_{20}\mu_{02}^2\mu_{21}^2 \\ & + 2\mu_{11}^3\mu_{30}\mu_{03} + 6\mu_{11}^3\mu_{21}\mu_{12} - 6\mu_{11}^2\mu_{02}\mu_{30}\mu_{12} \\ & - 6\mu_{11}^2\mu_{02}\mu_{21}^2 + 6\mu_{11}\mu_{02}^2\mu_{30}\mu_{21} - \mu_{02}^3\mu_{30}^2) / \mu_{00}^{11}, \end{aligned} \quad (\text{III.36d})$$

$$I_5 = (\mu_{40}\mu_{04} - 4\mu_{31}\mu_{13} + 3\mu_{22}^2) / \mu_{00}^6, \quad (\text{III.36e})$$

$$I_6 = (\mu_{40}\mu_{22}\mu_{04} - \mu_{40}\mu_{13}^2 - \mu_{31}^2\mu_{04} + 2\mu_{31}\mu_{22}\mu_{13} - \mu_{22}^3) / \mu_{00}^9, \quad (\text{III.36f})$$

$$\begin{aligned} I_7 = & (\mu_{20}^2\mu_{04} - 4\mu_{20}\mu_{11}\mu_{13} + 2\mu_{20}\mu_{02}\mu_{22} + 4\mu_{11}^2\mu_{22} \\ & - 4\mu_{11}\mu_{02}\mu_{31} + \mu_{02}^2\mu_{40}) / \mu_{00}^7, \end{aligned} \quad (\text{III.36g})$$

$$\begin{aligned}
I_8 = & (\mu_{20}^2\mu_{22}\mu_{04} - \mu_{20}^2\mu_{13}^2 - 2\mu_{20}\mu_{11}\mu_{31}\mu_{04} + 2\mu_{20}\mu_{11}\mu_{22}\mu_{13} \\
& + \mu_{20}\mu_{02}\mu_{40}\mu_{04} - 2\mu_{20}\mu_{02}\mu_{31}\mu_{13} + \mu_{20}\mu_{02}\mu_{22}^2 \\
& + 4\mu_{11}^2\mu_{31}\mu_{13} - 4\mu_{11}^2\mu_{22}^2 - 2\mu_{11}\mu_{02}\mu_{40}\mu_{13} \\
& + 2\mu_{11}\mu_{02}\mu_{31}\mu_{22} + \mu_{02}^2\mu_{40}\mu_{22} - \mu_{02}^2\mu_{31}^2)/\mu_{00}^{10},
\end{aligned} \tag{III.36h}$$

$$\begin{aligned}
I_9 = & (\mu_{30}^2\mu_{12}^2\mu_{04} - 2\mu_{30}^2\mu_{12}\mu_{03}\mu_{13} + \mu_{30}^2\mu_{03}^2\mu_{22} - 2\mu_{30}\mu_{21}^2\mu_{12}\mu_{04} \\
& + 2\mu_{30}\mu_{21}^2\mu_{03}\mu_{13} + 2\mu_{30}\mu_{21}\mu_{12}^2\mu_{13} - 2\mu_{30}\mu_{21}\mu_{03}^2\mu_{31} \\
& - 2\mu_{30}\mu_{12}^3\mu_{22} + 2\mu_{30}\mu_{12}^2\mu_{03}\mu_{31} + 4\mu_{21}^4\mu_{04} - 2\mu_{21}^3\mu_{12}\mu_{13} \\
& - 2\mu_{21}^3\mu_{03}\mu_{22} + 3\mu_{21}^2\mu_{12}^2\mu_{22} + 2\mu_{21}^2\mu_{12}\mu_{03}\mu_{31} + \mu_{21}^2\mu_{03}^2\mu_{40} \\
& - 2\mu_{21}\mu_{12}^3\mu_{31} - 2\mu_{21}\mu_{12}^2\mu_{03}\mu_{40} + \mu_{12}^4\mu_{40})/\mu_{00}^{13},
\end{aligned} \tag{III.36i}$$

$$\begin{aligned}
I_{10} = & (-\mu_{50}^2\mu_{05}^2 + 10\mu_{50}\mu_{41}\mu_{14}\mu_{05} - 4\mu_{50}\mu_{32}\mu_{23}\mu_{05} - 16\mu_{50}\mu_{32}\mu_{14}^2 \\
& + 12\mu_{50}\mu_{23}^2\mu_{14} - 16\mu_{41}^2\mu_{23}\mu_{05} - 9\mu_{41}^2\mu_{14}^2 + 12\mu_{41}\mu_{32}^2\mu_{05} \\
& + 76\mu_{41}\mu_{32}\mu_{23}\mu_{14} - 48\mu_{41}\mu_{23}^3 - 48\mu_{32}^3\mu_{14} + 32\mu_{32}^2\mu_{23}^2)/\mu_{00}^{14}.
\end{aligned} \tag{III.36j}$$

III.4.3. Light condition modelling

We modelled the extreme lighting condition variations—which occurs in real-world scenarios—by a diagonal mapping. We applied the photometric analysis and a diagonal model of Van de Sande *et al.* 2010 to rate the robustness of the descriptor against light changes. The matrix maps the colors captured under an unknown light source u to their equivalent under the canonical illuminant c . *Light intensity change* alters the pixel values multiplying with a constant factor (Equation III.37).

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} \tag{III.37}$$

Light intensity shift amends all color channels with the same constant across the image (Equation III.38).

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix} \tag{III.38}$$

Light color change modifies the color channels independently (Equation III.39). This type of modification can model changes in the illuminant color and light scattering

Van de Sande *et al.* 2010. To model these changes we used the RGB values corresponding to light color temperatures.

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} \quad (\text{III.39})$$



Figure 3.11: Light changes: (a) original image, (b) intensity change ($a=3$), (c) light color change (3800K), (d) color shift ($o_1=100$)
Source: Takacs, Toledano-Ayala *et al.*, 2016

III.4.4. Distance Transform

Various segmentation algorithm (Fabbri *et al.*, 2008; Wang and Tan, 2011) uses this feature estimating the distance between the pixel and the nearest detected edge point. Its result values are higher when the points are farther away from the edge, gradually changing color with the distance. Distance Transform helps the forest to decide between patches from flat area and from information dense areas as flat areas return the maximum values adding a real distinctive component to the descriptor.

III.4.5. Distance metrics

For descriptor evaluation the algorithms use normally vector distance calculations to compare the similarity of a high dimensional descriptor. Different techniques were used for floating point based or binary vectors, because of their characteristics.

Hamming Distance

This technique measures the binary difference or agreement between two binary vector with equal length. It can be implemented with a *XOR* operation followed by

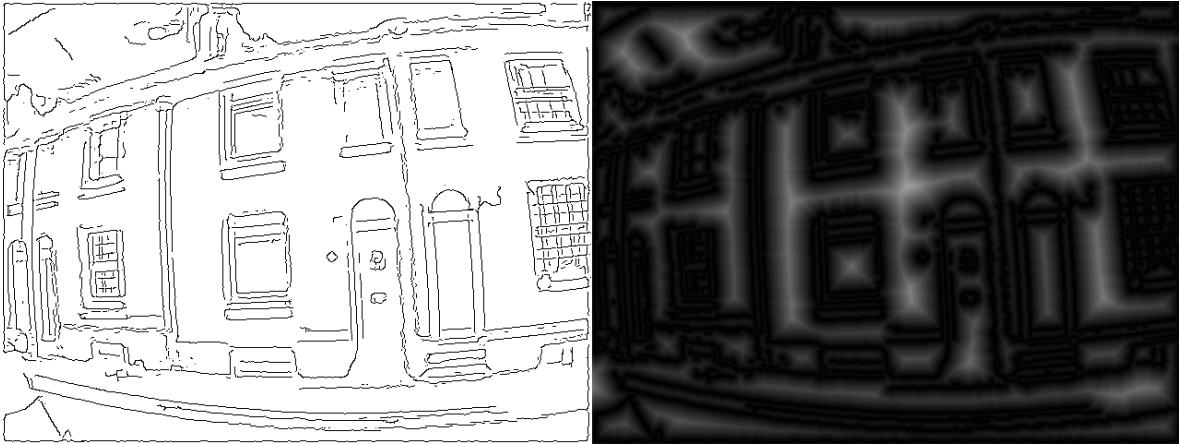


Figure 3.12: Results of the Canny edge detection and the distance transfer
Source: Takacs, Toledano-Ayala *et al.*, 2016

a bit count operation. Descriptors like ORB, FREAK, BRISK, or BRIEF are measured with this technique

$$\text{Binary distance: } 3 = 10100010 = (01001110) \text{ XOR } (11001100) \quad (\text{III.40})$$

Euclidean Distance

The classical distance measure in the Cartesian coordinate space calculates the distance between point p and q .

$$\begin{aligned} d(p, q) = d(q, p) &= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2} \\ &= \sqrt{\sum_{i=1}^n (q_i - p_i)^2} \end{aligned} \quad (\text{III.41})$$

III.4.6. Descriptor comparison techniques

Brute Force matcher

The Brute Force (BF) comes from the fact that the technique compares the descriptor with all the descriptors from the counterpart of the operation to find the best match. It is a slow operation but brings the closest pair for the descriptor. For

the operation it uses Hamming or Euclidean distances depending on the descriptor type.

Fast Library for Approximate Nearest Neighbors

Fast Library for Approximate Nearest Neighbors (FLANN) is an optimized library for matching in high dimensional spaces using fast approximate nearest neighbor search. It is a much faster method, but it brings back an approximate, that is not necessarily the best match.

III.4.7. Descriptor performance evaluation

Confusion Matrix

A confusion matrix (or error matrix) in machine learning used for measuring the performance of supervised or unsupervised learning algorithm (Ting, 2010). It is a 2-dimensional matrix (Figure 3.13) representing the predicted instances in the columns and the actual instances in the rows. The True Positive (TP) cell represents the all the positive examples that are correctly classified by a classification model. The True Negative (TN) instances are elements that the correctly categorized as non-member. There are two types of errors during the classification: False Positive (FP) and False Negative (FN). The FP instances are examples of negative class that has been incorrectly classified as positive. The elements in the FP class are examples of positive class that have been incorrectly classified as negative.

		Predicted Condition	
		Positive	Negative
True Condition	Positive	TP	FN
	Negative	FP	TN

Figure 3.13: Confusion Matrix
Source: Ting, 2010

When we are making multiple class segmentation evaluation, we have to calculate the positive and negative instances separately for each class. The cells in the column under "Class 2" in the matrix (shown in Figure 3.14) present all the positive predicted instances (the TP and the FP examples). The rest of the columns are the negatively predicted instances: the FN, where the actual conditions are true, and the TN cases, where the elements are correctly .

		Predicted Condition			
		Class 1	Class 2	Class 3	Class 4
True Condition	Class 1	TN	FP	TN	
	Class 2	FN	TP	FN	
	Class 3	TN		FP	TN
	Class 4	TN		FP	TN

Figure 3.14: Confusion Matrix 2.

Performance evaluation metrics

Accuracy is a description of systematic errors. It shows how close are the correct predictions to the actual conditions.

$$ACC = \frac{\sum TruePositive + \sum TrueNegative}{TotalPopulation} \quad (III.42)$$

Precision or Positive Predictive Value is the percent of the selected instances that are correct.

$$PPV = \frac{\sum TruePositive}{\sum TruePositive + \sum FalsePositive} \quad (III.43)$$

Recall or True Positive Rate is the measure of the retrieval performance. It shows the ratio between the total retrieved relevant example and the total relevant instances.

$$TPR = \frac{\sum TruePositive}{\sum TruePositive + \sum FalseNegative} \quad (III.44)$$

Specificity or True Negative Rate measures the ratio of the correctly classified negative instances.

$$TNR = \frac{\sum TrueNegative}{\sum TrueNegative + \sum FalsePositive} \quad (III.45)$$

Negative Prediction Value is the percent of the negative instances that are correctly selected from the actual model.

$$NPV = \frac{\sum TrueNegative}{\sum TrueNegative + \sum FalseNegative} \quad (III.46)$$

F-Measure is used to evaluate the performance of the predictions.

$$F_1 = 2 \cdot \frac{\textit{Precision} \cdot \textit{Recall}}{\textit{Precision} + \textit{Recall}} \quad (\text{III.47})$$

III.4.8. Camera base tracking parameters

The geometric model behind the camera image capture, largely influences the tracking and registration. The goal of the pinhole camera model is to represent a $w = [u, v, w]^T$ point of the 3D world coordinate system as $x = [x, y]^T$ on the 2D image plane (Figure 3.15), learning the *intrinsic*, *extrinsic*, and *camera distortion* parameters. The intrinsic parameters are related to the internal geometry of the camera like focal length, offset or skew. The extrinsic parameters describe the external parameters like position or orientation. Lastly the camera distortion parameters compensate the lens deformations. In our description we will follow the model description of Prince (2012).

Intrinsic and extrinsic parameters

The w or optical axis—which crosses the image plane at the Principal point—originates from the center point of the coordinate system is the Optical center or Pinhole. The focal length is the distance measured between the Optical center and the Principal point.

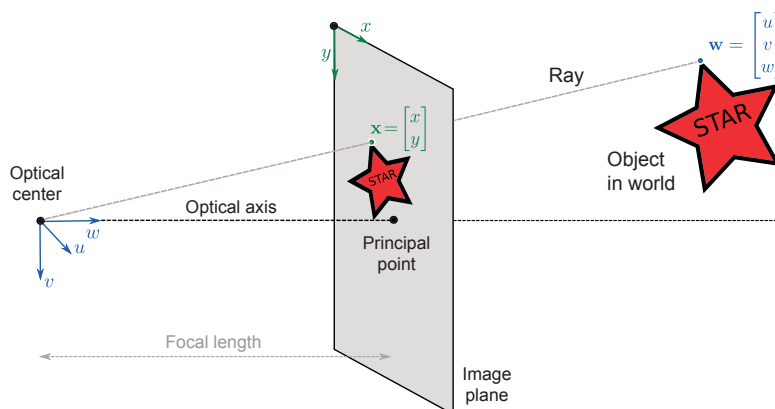


Figure 3.15: Pinhole camera model
Source: Prince, 2012

The simplest camera model express x and y simply normalizing the u and

v parameters by w , assuming that the focal length is 1. But the spacing of the photoreceptors can differ in x and y directions, therefore the focal length parameters (or scaling factors ϕ_x and ϕ_y) need to be applied:

$$\begin{aligned}x &= \frac{\phi_x u}{w}, \\y &= \frac{\phi_y v}{w}\end{aligned}\tag{III.48}$$

As the principal point in digital images is at the top left corner, we have to add δ_x and δ_y offset parameters to replace it to the center of the image plane. To complete the camera model, the γ skew term—that moderates x as a function of v —need to be introduced.

$$\begin{aligned}x &= \frac{\phi_x u + \gamma v}{w} + \delta_x, \\y &= \frac{\phi_y v}{w} + \delta_y\end{aligned}\tag{III.49}$$

The intrinsic parameters then can be represented in the intrinsic matrix:

$$\Lambda = \begin{bmatrix} \phi_x & \gamma & \delta_x \\ 0 & \phi_y & \delta_y \\ 0 & 0 & 1 \end{bmatrix}\tag{III.50}$$

To express the w point in an arbitrary world coordinates we have to transform it with the extrinsic parameters, where Ω is the 3×3 rotation matrix, and τ is the 3×1 translation matrix:

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{21} & \omega_{22} & \omega_{23} \\ \omega_{31} & \omega_{32} & \omega_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix},\tag{III.51}$$

or

$$\mathbf{w}' = \Omega \mathbf{w} + \tau\tag{III.52}$$

Radial distortion

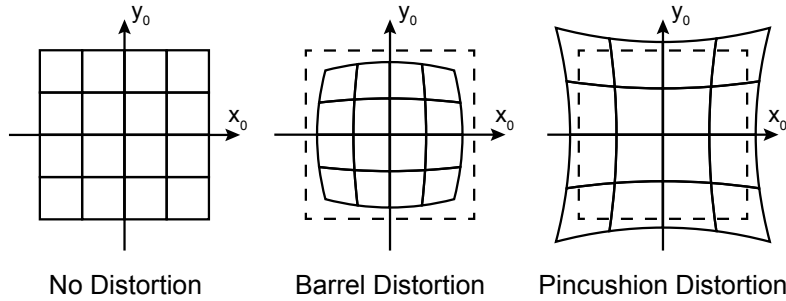


Figure 3.16: Radial distortion cases
Source: Majewski, 2015

The fact that the system counts with lenses to capture visible information from larger are leads to deviation from the classical pinhole model. The curvature of the optics cause different type of nonlinear warping (Figure 3.16), which magnitude linked to the distance (r) from the image center. The corrected x' and y' are calculated by

$$\begin{aligned} x' &= x (1 + \beta_1 r^2 + \beta_2 r^4), \\ y' &= y (1 + \beta_1 r^2 + \beta_2 r^4) \end{aligned} \quad (\text{III.53})$$

where β_1 and β_2 are the control parameters for degree of distortion. According to Prince (2012), these distortion is applied "after perspective projection (division by w) but before the effect of intrinsic parameters (focal length, offset, etc.)".

Resolve parameters

There are three basic parameter group to find for a flawless AR system. All three are prediction tasks for the different parameters of the model **pinhole** $[w, \Lambda, \Omega, \tau]$. These parameters define the cameras position, orientation or projection parameters.

Camera calibration

To be able to use the CV system, first needs define the camera's intrinsic parameters Λ . The process estimates Λ as maximum likelihood learning problem based on I different 3D points $\{\mathbf{w}_i\}_{i=1}^I$ of a known 3D object and their projections to the image plane $\{\mathbf{x}_i\}_{i=1}^I$:

$$\hat{\Lambda} = \operatorname{argmax}_{\Lambda} \left[\max_{\Omega, \tau} \left[\sum_{i=1}^I \log [Pr(\mathbf{x}_i | \mathbf{w}_i, \Lambda, \Omega, \tau)] \right] \right] \quad (\text{III.54})$$

The system eventually finds predictions for the extrinsic parameters Ω, τ , but in this step only the calibration parameters Λ are important as they are saved in the system and applied while the corresponding camera is in use. Meantime Ω, τ are temporary parameters as the orientation and position—in the case of world fixed AR—are constantly changing.

Exterior orientation

Once we know the camera's intrinsic parameters Λ the next step is to infer the orientation and position of the camera in the real world. The task is to predict with maximum likelihood the extrinsic parameters Ω for rotation, and τ for translation. The prediction is based on I 3D points of a known object $\{\mathbf{w}_i\}_{i=1}^I$ and their projections to the image plane $\{\mathbf{x}_i\}_{i=1}^I$:

$$\hat{\Omega}, \hat{\tau} = \operatorname{argmax}_{\Omega, \tau} \left[\sum_{i=1}^I \log [Pr(\mathbf{x}_i | \mathbf{w}_i, \Lambda, \Omega, \tau)] \right] \quad (\text{III.55})$$

Deduce world points

The third problem is to estimate the 3D world position of point w based on its projections $\{\mathbf{x}_j\}_{j=1}^J$ (where $J \geq 2$) to the image plain. When $J = 2$, the process called calibrated stereo reconstruction—same as the initialization step of PTAM—, and multiview reconstruction in the case of $J > 2$, which used to create in the SLAM

applications the parallel 3D point map.

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmax}} \left[\sum_{j=1}^J \log [Pr(\mathbf{x}_j | \mathbf{w}, \Lambda_j, \Omega_j, \tau_j)] \right] \quad (\text{III.56})$$

In other words, the given J calibrated cameras—knowing their $\Lambda_j, \Omega_j, \tau_j$ parameters—try to establish the position of \mathbf{w} via triangulation, having their own projected points $\{\mathbf{x}_j\}_{j=1}^J$ of these 3D world points.

Homography Matrix

In order to calculate the transformation between consequent image plains, the system uses projective transformation or homography matrix. This matrix maps point $\mathbf{w} = [u, v, 0]^T$ —where w -coordinate is always zero as it is perpendicular to the image plane—to $\mathbf{x} = [x, y]^T$:

$$\begin{aligned} \lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} &= \begin{bmatrix} \phi_x & \gamma & \delta_x \\ 0 & \phi_y & \delta_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \tau_x \\ \omega_{21} & \omega_{22} & \omega_{23} & \tau_y \\ \omega_{31} & \omega_{32} & \omega_{33} & \tau_z \end{bmatrix} \begin{bmatrix} u \\ v \\ 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \phi_x & \gamma & \delta_x \\ 0 & \phi_y & \delta_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \omega_{11} & \omega_{12} & \tau_x \\ \omega_{21} & \omega_{22} & \tau_y \\ \omega_{31} & \omega_{32} & \tau_z \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \end{aligned} \quad (\text{III.57})$$

We get the generalized homography matrix, by multiplying the 3×3 matrices:

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} \\ \phi_{21} & \phi_{22} & \phi_{23} \\ \phi_{31} & \phi_{32} & \phi_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad (\text{III.58})$$

or in Cartesian coordinates:

$$\begin{aligned}
x &= \frac{\phi_{11}u + \phi_{12}v + \phi_{13}}{\phi_{31}u + \phi_{32}v + \phi_{33}} \\
y &= \frac{\phi_{21}u + \phi_{22}v + \phi_{23}}{\phi_{31}u + \phi_{32}v + \phi_{33}}
\end{aligned} \tag{III.59}$$

In short form $x = \text{hom}[w, \Phi]$, where the matrix Φ has nine members, but only 8 DoF as they are redundant to scale. This matrix is used for the transformation of the rendered objects.

We have to rewrite in matrix form the Equation III.58, where both side represents the same direction, therefore their cross product is 0.

$$\tilde{x} \times \Phi \tilde{w} = 0 \tag{III.60}$$

Writing out in full

$$\begin{bmatrix}
y(\phi_{31}u + \phi_{32}v + \phi_{33}) - (\phi_{21}u + \phi_{22}v + \phi_{23}) \\
(\phi_{11}u + \phi_{12}v + \phi_{13}) - x(\phi_{31}u + \phi_{32}v + \phi_{33}) \\
x(\phi_{21}u + \phi_{22}v + \phi_{23}) - y(\phi_{11}u + \phi_{12}v + \phi_{13})
\end{bmatrix} = 0 \tag{III.61}$$

We use the only 2 independent equations to form the system for all the detected n pair of points ($n > 4$).

$$\begin{bmatrix}
0 & 0 & 0 & -u_1 & -v_1 & -1 & y_1u_1 & y_1v_1 & y_1 \\
u_1 & v_1 & 1 & 0 & 0 & 0 & -x_1u_1 & -x_1v_1 & -x_1 \\
& & & & \vdots & & & & \\
0 & 0 & 0 & -u_n & -v_n & -1 & y_nu_n & y_nv_n & y_n \\
u_n & v_n & 1 & 0 & 0 & 0 & -x_nu_n & -x_nv_n & -x_n
\end{bmatrix}
\begin{bmatrix}
\phi_{11} \\
\phi_{12} \\
\phi_{13} \\
\phi_{21} \\
\phi_{22} \\
\phi_{23} \\
\phi_{31} \\
\phi_{32} \\
\phi_{33}
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix} \tag{III.62}$$

As these equations have the form $A\phi = 0$ we have to solve with the constraint $\phi^T\phi = 1$ to prevent trivial solution $\phi = 0$. For the solution we compute the SVD $A = ULV^T$ and choose ϕ to be the last column of V . This reshapes into a 3×3 matrix Φ .

III.5. FAST: Features from Accelerated Segment Test

The FAST corner detector was proposed by Rosten and Drummond (2005) for fast real time feature detection. The authors trade off quality for speed by dividing the detection steps in their approach. They partitioned a Bresenham circle with a 3 pixel radius, into 16 pixels (Figure 3.17) around the pixel I_p . The method investigates the intensity values of these surrounding pixels. If at least n contiguous pixels are brighter than $I_p + t$ or darker than $I_p - t$ (where t is some threshold value), the feature is detected. They chosen $n = 12$ as it allows the high-speed test. To reject faster the candidate pixels, first they examine pixels no. 1, 9, 5, and 13, as a feature can only exist if at least 3 of these points are above or below the intensity of I_p by t .

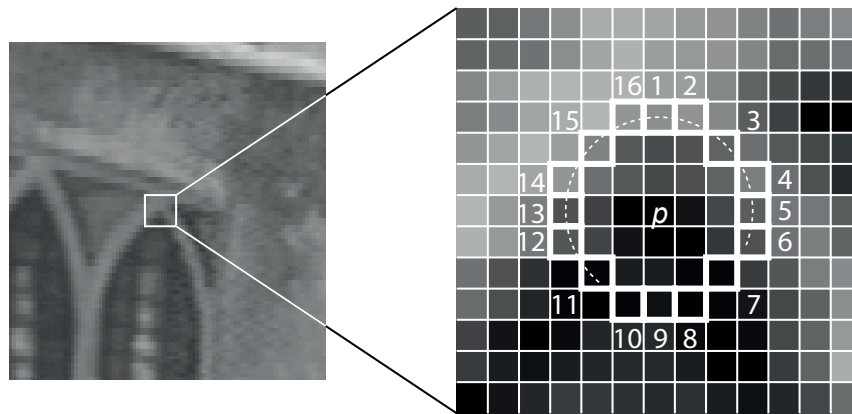


Figure 3.17: FAST Feature detection in an image patch
Source: Rosten and Drummond, 2005

Although the detector is proved to be a high performance method, according to the authors it exhibits some weaknesses:

- (1) In the case of $n < 12$, the high speed test fails to reject the correct number of candidates.
- (2) The choice and ordering of the fast test pixels contains implicit assumptions about the distribution of feature appearance.

- (3) The outcomes of the high-speed tests are discarded.
- (4) Multiple features are detected adjacent to one another.

They proposed to use machine learning to resolve the first three issues and non-maximal suppression for the last point. The first process operates in two stages. To train with a set of images the detector for a given n . They FAST detector on the images, but storing all the 16 pixels in a feature vector. Each pixel in the vector $x \in \{1..16\}$, position relative to p (denoted $p \rightarrow x$), can have one of the following states:

$$S_{p \rightarrow x} = \begin{cases} d, & I_{p \rightarrow x} \leq I_p - t & (\text{darker}) \\ s, & I_p - t < I_{p \rightarrow x} < I_p + t & (\text{similar}) \\ b, & I_p + t \leq I_{p \rightarrow x} & (\text{brighter}) \end{cases} \quad (\text{III.63})$$

They Choose an x and compute $S_{p \rightarrow x}$ for all $p \in P$ (the set of all pixels in all training images) partitions P into three subsets, P_d, P_s, P_b , where each p is assigned to $P_{S_{p \rightarrow x}}$. Then they specify K_p boolean variable which becomes true if p is a corner. After they used a decision tree classifier (ID3 algorithm) to select x that provide the most information—based on the entropy of K_p —during the corner evaluation. This is applied recursively to the subsets until their entropy reaches zero.

The second process addresses the problem of the adjacent locations using Non-maximum Suppression. The method calculates a score function V for each detected corner. V is equal with the sum of the absolute difference between I_p and the pixels in the contiguous arc around (Eq. III.64). The corner with lower V gets discarded.

$$V = \max \left(\sum_{x \in S_{\text{bright}}} |I_{p \rightarrow x} - I_p| - t, \sum_{x \in S_{\text{dark}}} |I_p - I_{p \rightarrow x}| - t \right) \quad (\text{III.64})$$

According to their findings, the approach is significantly faster than other corner detectors but with high sensitivity to noise. For its speed it is widely used in the SLAM applications, like PTAM (Klein and Murray, 2007) or ORB-SLAM (Mur-Artal *et al.*, 2015).

III.6. Descriptor extraction

Local descriptors extracted from the interest are has wide variety of use in object recognition (specific object or category recognition) (Lowe, 1999; Ferrari *et al.*, 2006), scene classification (Shimazaki and Nagao, 2013; Raja *et al.*, 2013),

Local features are used because it can represent an image, object or scene and it helps in recognition without pre-segmentation. The goal is to analyse statistically the features, so there is no importance find them accurately or match them individually (Tuytelaars and Mikolajczyk, 2008).

To localize features in images, a local neighborhood of pixels needs to be analyzed, giving all local features some implicit spatial extent. (*ibid.*)

The higher the repeatability rate between two images, the more points can potentially be matched and the better the matching and recognition results are. The classification error depends on the texture type and the dimensionality of the descriptors.

III.6.1. Scale Invariant Feature Transform (SIFT)

SIFT has been the most used and referenced descriptor (according to Wu, Cui *et al.* (2013) it has more than 12,000 references) in the past 10 years. The SIFT algorithm calculates the Difference of Gaussian (DoG) blurring the image with two different σ (in this case σ and $k\sigma$).

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \quad (\text{III.65})$$

The DoG is more convenient to compute as the Laplacian of Gaussian is computationally expensive and the DoG is a close approximation to the scale-normalized Laplacian of Gaussian (Lowe, 1999).

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k - 1)\sigma^2 \nabla^2 G \quad (\text{III.66})$$

This is repeated on each octave (scale) of the image in the Gaussian Pyramid (Figure 3.18). After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated (*ibid.*). After obtaining the DoG, keypoints are identified as local extrema over scale and space by comparing a pixel to its 26 neighbors in 3×3

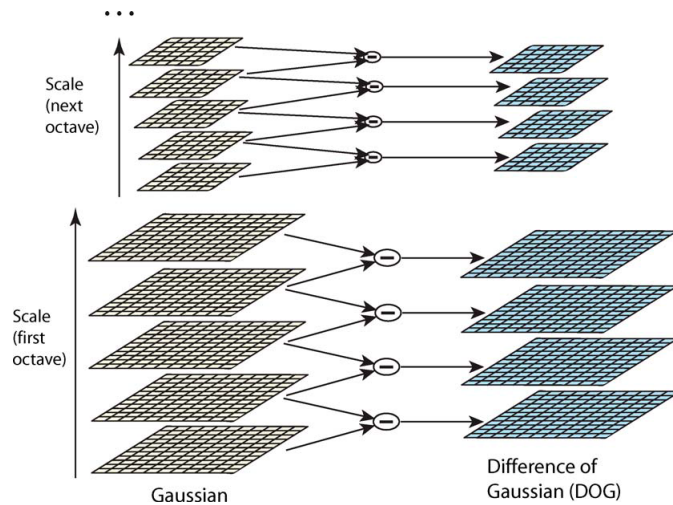


Figure 3.18: Gaussian Pyramid
Source: Lowe, 1999

regions at the current and adjacent scales(Figure 3.18). After localizing the keypoint

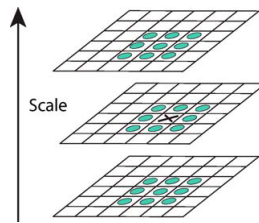


Figure 3.19: Detection of maxima and minima
Source: Lowe, 1999

candidates, they use Taylor expansion to search a better location of the extrema, and discarding the low intensity points. To remove the possible edges—as the DoG is more sensitive to edges—they compute a simpler Harris corner detector, calculating only the eigen value ratios of the curvatures Hessian matrix discarding the keypoints which are greater then the threshold. To make invariant to rotation each keypoint is assigned with an orientation. The gradient magnitude and direction is calculated from neighbourhood around the keypoint location. These gradient orientations are used to create an orientation histogram is created. They select the highest peaks and those within 80% of the highest peak. That means that various keypoints are created with the same location with similar magnitude but different directions. Once these features are calculated a 16×16 region around the keypoint is specified and divided into 4×4 sub-blocks(Figure 3.20). 8 bin orientation histogram is created at each sub-block which add up total of 128 bin values ($4 \times 4 \times 8 = 128$). At the last step the created feature vector is normalized to unit length to enhance the descriptor's

invariance to illumination change.

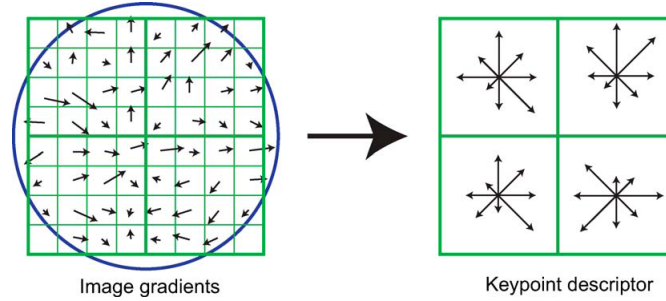


Figure 3.20: Keypoint descriptor creation from gradient magnitude and orientation
Source: Lowe, 1999

Since the publication of SIFT, many variations were created but in reference to Wu, Cui *et al.* (2013) the most used and investigated are the following. ASIFT (Ke and Sukthankar, 2004) simulates the camera rotation axis and adds rotation and tilt transformation to the image in order to add affine transformation invariance to SIFT. An extension, the Gradient location-orientation histogram (GLOH) by Mikolajczyk and Schmid (2005) was created to increase the robustness and distinctiveness of the SIFT. Corresponding to the authors GLOH gives the best results on those fields. PCA-SIFT (Ke and Sukthankar, 2004) calculates horizontal and vertical image gradient vectors on a 39×39 region ($39 \times 39 \times 2 = 3042$), then uses Principle Component Analysis (PCA) to reduce the descriptor vector's dimensionality to 36. GSIFT (Mortensen *et al.*, 2005) adding global texture information to the descriptor. The two parts of the 188 dimensional descriptor vector (local(S) and global(G)) are regulated with a weight factor (Eq III.67).

$$F = \begin{bmatrix} \omega S \\ (1 - \omega)G \end{bmatrix} \quad (\text{III.67})$$

CSIFT (Abdel-Hakim and Farag, 2006) adds color invariance to SIFT by defining a color image invariant H matrix using Kubelka-Munk theory. The calculated H then replaces I for the SIFT calculation method. SURF by Bay *et al.* (2008) (we will discuss in the next section) was created to speed up the extraction process. According to Wu, Cui *et al.* (2013) their test shows that each variation has its own advantage depending on the design, but the PCA-SIFT comes second in all tests although PCA slows down feature computation and the worst in performance and during the invariance tests but far the fastest.

III.6.2. Speeded-Up Robust Features (SURF)

This 64-dimension vector descriptor is proposed by Bay *et al.* (2008) to speed up the SIFT calculation. To avoid the calculation of the expensive LoG, they approximate it by calculating the Hessian matrix determinant with box filters (Figure 3.21). A Hessian $\mathcal{H}(x, \sigma)$ at a given point $x = (x, y)$ at scale σ defined as

$$\mathcal{H}(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{yx}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix} \quad (\text{III.68})$$

$$\det(\mathcal{H}_{approx}) = D_{xx}D_{yy} - (wD_{xy})^2 \quad (\text{III.69})$$

The relative weight w of the filter responses is used to balance the expression for

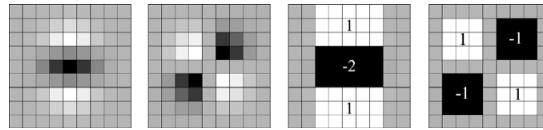


Figure 3.21: Left images: Second order Gaussian partial derivative($y - (L_{yy})$ and xy -direction (L_{xy})); Right images: Approximation for the second order ($y - (D_{yy})$ and xy -direction (D_{xy}))

Source: Bay *et al.*, 2008

the Hessian's determinant.

III.6.3. Opponent SIFT

This is the best performing color image SIFT descriptor Van de Sande *et al.* 2010. Its estimation results a 384-dimension vector applying the classical SIFT descriptor calculation for all the highly decorrelated opponent color channel (Equation III.12), where the color space contains one intensity and two chromaticity channels.

III.6.4. Opponent SURF

This descriptor was first applied for foreground/background discrimination Chu and Smeulders 2010 during tracking. The extraction method calculates the original SURF descriptor on the 3 opponent color space (Equation III.12) resulting a 192-dimensional vector. The authors state that Opponent SURF increased invariance and discriminative power of the tracker.

III.7. Random Forest

The Random Forest (Breiman, 2001) is a high-performance discriminative classifier, handling large feature set without having difficulties in high dimensionality (Fröhlich *et al.*, 2010). This supervised learning method—which can learn more than one class at a time—constructs an ensemble of recursively created random binary decision trees during the training period (Figure 3.22). To avoid overfitting, the method applies out-of-bag error estimates (oob) for each tree. The oob estimate means, that one-third of the input vectors are left out of the tree creation, and used for the error rate evaluation at the end of the construction. The classification process returns the class (κ) probability (p) of a given feature vector (v_i) via averaging the final votes (p_τ) of each tree with the total number of trees (T) (Equation III.70).

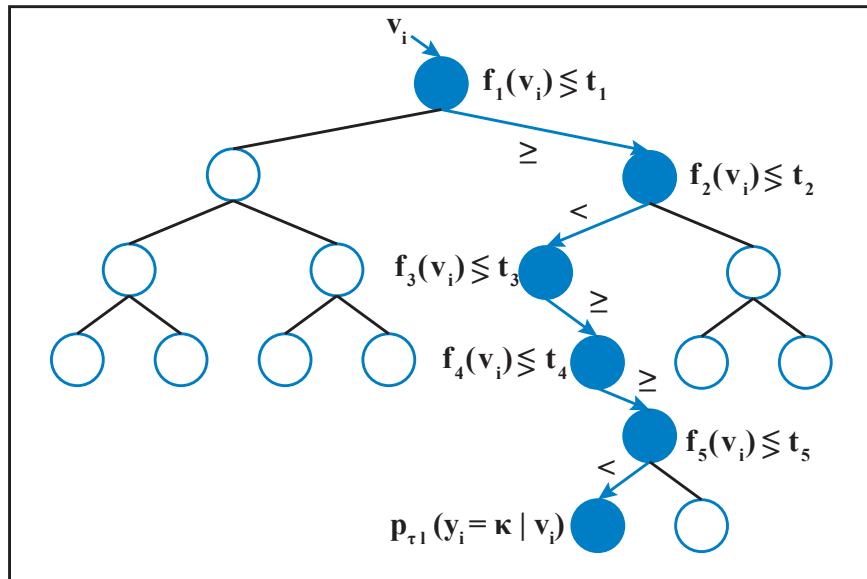


Figure 3.22: Random Forest in function

Source: Fanelli *et al.*, 2013

$$p(y_i = \kappa | v_i) = \frac{1}{T} \sum_{\tau=1}^T p_\tau(y_i = \kappa | v_i) \tag{III.70}$$

The process aggregates randomness at two stages during the building of the forest in the training session. With this step, Random Forest solves the overfitting problem which in other algorithms like the random decision trees causes major issues. First, the Bootstrap Aggregation creates random subsets of data from which the trees are learned, and second the split functions use only a random fraction of

all features during the creation of the decision trees. Breiman (2001) states that using "significant test variation (1 to 50) of random inputs selected at each node, the strength of the forest remained constant after passing the value 4", in other words adding more data did not improve the performance. Though our experience, that the optimal split number is the square root of the feature dimensionality.

III.8. Genetic Algorithm

The Genetic Algorithm is an effective stochastic algorithm taking as an example of the natural selection and genetics. It has been applied in Machine Learning and optimization problems successfully (Guo *et al.*, 2010). The base of our design is Goldberg (1989) Simple Genetic Algorithm (SGA) (Algorithm 1). The algorithm creates and keep a population of fixed number of individuals (genomes) and modifies their composition based on probability, genetic operation—Reproduction, Crossover and Mutation—and evaluation function. The algorithm runs until reaches the termination criteria—creates the last set generation or the optimum score reaches a specific threshold—.

Algorithm 1 Simple Genetic Algorithm pseudocode

```
1: procedure GENETIC ALGORITHM
2:   Initialize Population with randomly generated genomes
3:   repeat
4:     Evaluation with the Objective Function to Accuracy and Invariance evaluation
5:     Select individuals for mating
6:     Mate individuals to produce offspring
7:     Mutate offspring
8:     Insert offspring into population
9:   until reached the termination criteria
10: end procedure
```

III.8.1. Initial Population

The genetic algorithm creates an initial population for the first step. The population members (individuals) are fixed-length binary strings. The length of the genome depends on the precision and complexity of the project. The binary strings can be created in various ways. The most common that the algorithm generates them

randomly, other option to import a preset population or specify a single individual which would be mixed up by the genetic algorithm.

III.8.2. Evaluation

This project specific function evaluates each individual in the population in each generation returning an objective score. The score would be increasing or decreasing based on that we want to minimize or maximize the optimum. The raw objective scores are linearly scaled to obtain the fitness (Wall, 1996). This fitness score is used during the Reproduction operation for the creation of new offsprings.

III.8.3. Genetic Operators

There are three key genetic operators: Reproduction, Crossover and Mutation, which controlled by their probability parameter(P_R, P_C, P_M).

Reproduction

During this operation the best individuals are selected from the current population to carry over to the next generation. There are many different strategies to choose an individual and their use depends on the goal of the optimization:

1. During the **Random Selection**, the parents are randomly selected from the population.
2. The **Roulette Wheel Selection** uses a "proportionate reproductive operator where a string is selected from the mating pool with a probability proportional to the fitness" (Sivanandam and Deepa, 2008). The individual probability (p_i) is the ratio of the fitness (f_i) of the individual i chromosome and the total fitness of the population (Eq. III.71). In this way individuals with greater fitness has major probability to be selected. The roulette wheel spins N times to choose N individuals for the next generation.

$$p_i = \frac{f_i}{\sum_{j=1}^N f_j} \quad (\text{III.71})$$

3. The **Rank Selection** overcomes the problem of the Roulette Wheel selection, when there are big differences between the fitness values. The selection probability (p_i) of the individuals is based on their ranking. The individuals are ranked based on their individual fitness (f_i).
4. The **Tournament Selection** uses randomly selected subgroup of individuals from the population. This method uses a selection tournament to choose the best individuals from subgroups. The tournaments are running until the mating pool is filled.
5. The **Stochastic Uniform Sampling** is advanced version of the Roulette Wheel technique mapping the individuals to a continuous line with a size according to their fitness (Figure 3.23). This method does not choose the candidates by repeated random sampling, but uses a single random value to sample the pointers with evenly spaced intervals. This approach gives to opportunity for the weaker member of the population to be chosen.

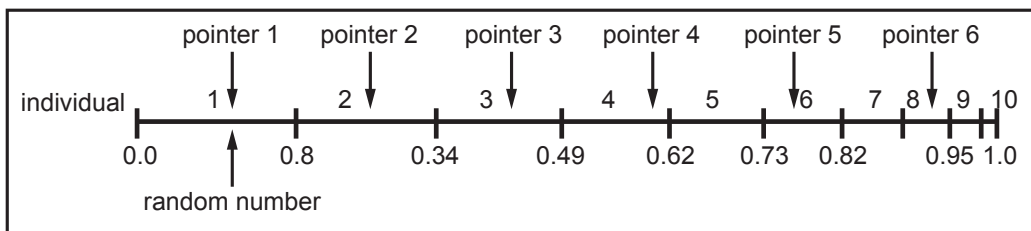


Figure 3.23: Stochastic Universal Sampling
Source: Sivanandam and Deepa, 2008

Crossover

This operation defines the form to create new offsprings from the selected individuals. The crossover swaps sections between the parental genomes. The Crossover probability P_C parameter defines how often the crossover is performed. All the children is created by crossover if it is 100% and the parents exact copy goes to the next generation if it is 0%. There are various technique that can be applied:

1. The **Single Point Crossover** used in the traditional genetic algorithms. The randomly selected crossover point—a point between 1 and N-1, where N is the number of elements in the genome—specifies the section starting point. For

example g_1 and g_2 parental genomes (N=15) with a crossover point 4, creating the following g'_1 and g'_2 siblings (Figure 3.24).

$$\begin{array}{l} g_1 = \langle 1001|10011110101 \rangle \\ g_2 = \langle 0101|00100001010 \rangle \end{array} \times \begin{array}{l} g'_1 = \langle 1001|00100001010 \rangle \\ g'_2 = \langle 0101|10011110101 \rangle \end{array}$$

Figure 3.24: Single Point Crossover

2. The **Two Point Crossover** chooses two random crossover points, and the sections between these points exchanged between the parents g_1 and g_2 (Figure 3.25). According to Sivanandam and Deepa (2008) this technique has the advantage that the good genetic information of both the head and the tail of the chromosome can be passed to the next generation g'_1 and g'_2 .

$$\begin{array}{l} g_1 = \langle 1001|100111|10101 \rangle \\ g_2 = \langle 0101|001000|01010 \rangle \end{array} \times \begin{array}{l} g'_1 = \langle 1001|001000|10101 \rangle \\ g'_2 = \langle 0101|100111|01010 \rangle \end{array}$$

Figure 3.25: Two Point Crossover

3. During the **Uniform Crossover** each gene of the child is copied from one or the other parent's chromosome using a random generated binary crossover. If at the gene's position the mask is 1, the gene from the first parent (g_1) is copied, if it is 0, the second parent's (g_2) gene is used as in Figure 3.26.

g_1	100110011110101
g_2	010100100001010
mask	110110000111101
g'_1	100110100110111
g'_2	010100011001000

Figure 3.26: Uniform Crossover

Mutation

Random changes produced with this operator in the genomes structure. This can be beneficial as it causes that result escapes from the local optimum. The form of mutation depends on the data type controlled by the mutation probability (P_M). The P_M indicates the section section of the chromosome that will be changed. With 0%

nothing changes, if it is 100% the whole chromosome changes. There are different mutation methods:

1. The **Bit Flip** mutator changes the randomly chosen binary values from 1 to 0 or 0 to 1.
2. When the **Interchanging** mutator is active, two randomly chosen bits interchanging position.
3. During the **Reversing** method a randomly chosen bit the one next to it changes place, so they reverse order in the chromosome.

III.8.4. Fitness function

During the genome evaluation step, this function returns an objective score from each genome whom later converts to fitness score in each cycle. The results of each evaluation operation can be weighted to strengthen a particular feature or invariance. For example, if we would like the descriptor to be more accurate in classification we would multiply with higher values the segmentation scores. Consequently, the genetic algorithm only accepts the genomes with the lower scores, therefore, better segmentation results.

IV. METHODOLOGY

IV.1. Introduction

The base concept of this investigation is to create lightweight descriptors with simple, computationally cheap elements that are specialized to the particular environment (buildings, green areas, sparse areas) after training and empowering their segmentation accuracy with machine-learning techniques. This chapter will present the development of this descriptor. First in Section IV.2 we will show the used datasets, libraries, and equipment. Section IV.3 will describe the design of a preliminary descriptor, what kind of tests were used during the evaluation and performance analysis. In Section IV.4 we will present a self-adjusting Algorithm for descriptor creation, optimization and evaluation using Genetic Algorithm. Section IV.6 will introduce the fitting algorithm to PTAM.

IV.2. System's characteristics

IV.2.1. Datasets

We used three sets of images with different characteristics during the evaluation. The datasets (Figure 4.1) served to evaluate the descriptors adaptabilities in a particular and a general environment.

The Brighton images (Figure 4.1 (e)) were specially chosen key frames from video recordings made on two different occasions with different light conditions on Queens Gardens street in Brighton, United Kingdom. The database contains 52 pixel-wise labeled images with four classes. The classes are meaningful regions of house pictures (doors and windows, wall, roof, else). 90% of the images were set for training and the rest were destined for the testing period. This dataset was used for the Segmentation evaluation and during the Rotation, Affine Transformation, Resize and Blur Invariance test.

The LabelMeFacade(Fröhlich *et al.*, 2010) dataset was assembled from the

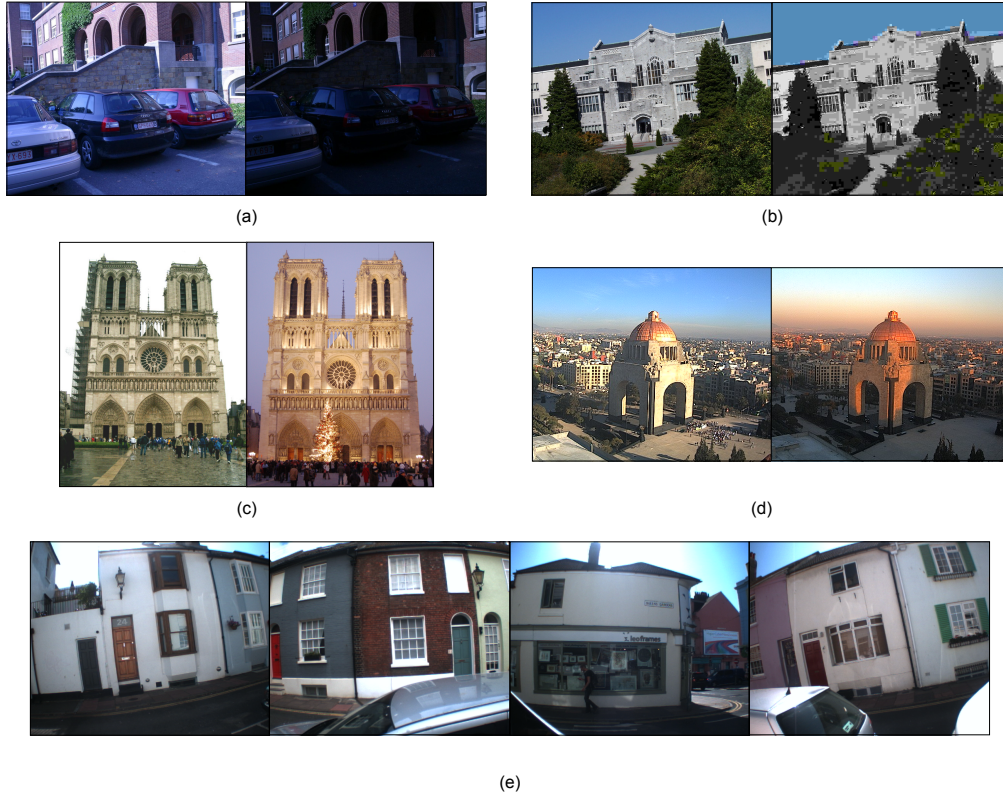


Figure 4.1: Image Datasets: **(a)-(c)** Oxford, **(d)** TILDE, **(e)** Brighton

LabelMe Russell *et al.* 2008 database. It contains 945 labeled images with nine classes (building, car, door, pavement, road, sky, vegetation, window, else) including the most important semantical regions of an urban scene. We divided the dataset into 100 mixed scene images for training and 845 for testing following (Brust *et al.*, 2015) description.

The Oxford dataset (Mikolajczyk and Schmid, 2005) contains real images with different geometric and photometric transformations and different scene types (Figure 4.1 (a)-(c)). Each image group contains six images with the Homography matrices for the transformation between the pictures for the comparison. The images were used for the JPEG compression, Illumination, and Light condition change evaluation.

The TILDE dataset (Verdie *et al.*, 2014) was designed to test the robustness of keypoint detectors to temporal changes (Figure 4.1 (d)). The database consists images taken by webcams in a different period of the day and the year. The cameras located on the top of a building in a different part of the world—Mexico, Frankfurt and

Chamonix among others—. The images were used for the Light condition change evaluation

IV.2.2. Libraries and equipment

To compute the framework, we used the OpenCV (Bradski, 2000) for the image processing and Machine Learning calculations and GAlib (Wall, 1996) for the Genetic Algorithm on a PC with 15.6 GB RAM and Intel®Core™i7-4790 CPU with 3.60GHz × 8.

IV.3. Phase 1: Environment Dedicated Descriptor (EDD)

IV.3.1. EDD setup

The proposed descriptor—published by Takacs, Toledano-Ayala *et al.* 2016—is an 113-dimension vector computed from a 9×9 patch selected around each key-point (Figure 4.2). The size was chosen to be big enough to pick up edges and low-level changes on the image, but also was sized to reduce the forest-training time and size. The following elements make up the descriptor:

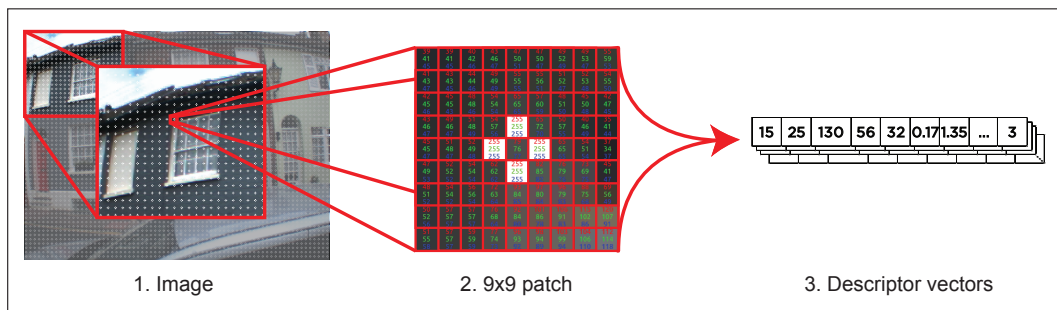


Figure 4.2: Descriptor extraction
Source: Takacs, Toledano-Ayala *et al.*, 2016

(1) **Position** - 2 values - 2D image coordinates of the patch centers to separate points which are on the top (sky), on the bottom (street), and in the middle (wall).

(2) **Patch mean** - 6 values - The mean of the Red, Green, Blue (from the RGB channels), and Saturation values (from the HSV channels) over the patch, to exploit the color changes on the images. Sine and Cosine of the mean of the discontinuous

Hue values were calculated over the patch. In consequence of the channels, angular design, the color values differ significantly at the opposite ends (0° and 359°), while these colors have neighboring RGB values. With the Sine and Cosine pair, we equalise them (ex. $\sin(0^\circ) = \sin(360^\circ) = 0$ and similarly $\cos(0^\circ) = \cos(360^\circ) = 1$).

(3) **The third order central moments** - 24 values - Generated to get distinctive shape description of the patch using the third order central moments— μ_{03} , μ_{30} , μ_{21} , μ_{12} —of the RGB and HSV channels over the patch measure the skew and the symmetry of the point spread around the mean of the patch.

(4) **Distance Transform** - 81 values

Algorithm 2 EDD Creation pseudocode

```

1: procedure DESCRIPTOR CREATION AND TRAINING
2:   Keypoint Creation with a constant (9px) distance
3:   Descriptor Extraction from the  $9 \times 9$  patch around each keypoint
4:   repeat
5:     Position
6:     Patch mean
7:     Central Moments
8:     Distance Transform
9:   until descriptor vector is calculated for all keypoint
10:  Random Forest Training
11: end procedure

```

IV.3.2. Evaluation process

To compare the descriptor with the most used descriptors, the evaluation was divided into two operations: accuracy and speed analysis, with different training database sizes and characteristics during the test session; and invariance evaluation to size, rotation, blur, light intensity changes, light intensity shifts, and light color changes in a separate function (Algorithm 3).

Algorithm 3 Descriptor Evaluation pseudocode

```
procedure ACCURACY EVALUATION
  repeat
    Descriptor Extraction for all keypoints
    Average descriptor extraction time measure
    Random Forest classification
    Feature Segmentation
    Overall and Average Recognition Rate calculation
    Average Segmentation Time computation
  until reached the last test image
procedure INVARIANCE EVALUATION
  repeat
    Evaluation setup for specific invariance
    repeat
      Image transformation
      Descriptor Extraction for all keypoints for the image pair
      Brute Force descriptor vector matching
      Result storing
    until reached the last test image
  until all invariance is being tested
  Result plotting by invariance case
end procedure
```

IV.4. Phase 2: Genetic Optimization framework

Phase 1 showed us the strength of the preliminary design and that the direction is right, but the parameters and elements lacked optimization. We turned to the Genetic Algorithms knowing its high optimization power in complex systems. In this section, we present the scheme that creates multiple, scene dedicated optimized descriptors. Each element of the algorithm has responsibility for the size reduction and descriptor invariance: The module bank for storing of the building blocks that hold the image processing functions for the descriptor; the activation function for the dimensionality reduction which also delivers information about the relation between the structure of the descriptor and the scene characteristics; the objective function the training period which shapes the descriptor depending on the features of the database.

IV.4.1. Preprocessing stage

The binary chromosome known from genetic algorithm typically represent a solution for an optimization problem. The binary string represents a real number, and the length of the chain depends on the precision requirements. In our case, this series has two purposes: the first part serves as a control panel being in charge of the image processing and evaluation function parameters, each bit of the string is responsible for activating or deactivating a value in a specified module. During this stage, the algorithm—based on the current genome—sets the image processing and evaluation parameters and assembles the modular descriptor. The 211 element long genome splits in two—a 26 and an 185 element long—binary substrings (Figure 4.3).

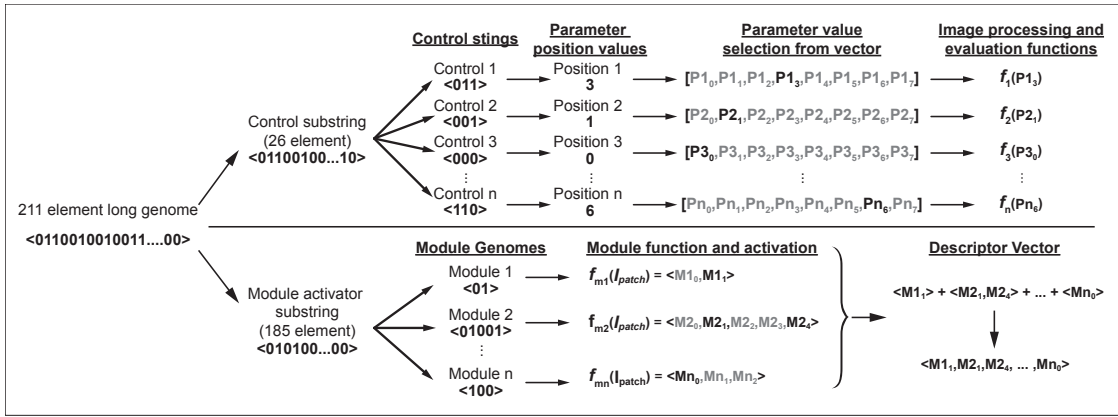


Figure 4.3: Modular Descriptor Extraction Process

The first 26 bits of the genome specifies ten different vector position. Each vector holds a list of parameters corresponding to an image processing or evaluation stage. The binary strings $\langle b_0 b_1 \dots b_n \rangle$ converted to a decimal number (Equation IV.1) to select the corresponding value from the parameter vector. The selected parameters cover the most important stages of the optimization with the possibility to enhance precision, speed, and complexity of the descriptor among others.

$$x = \sum_{i=0}^n b_i 2^i \quad (IV.1)$$

The parameters categories, binary string size and parameter values are the following:

- (1) **Color space** - 3 bits - Eight different color space: RGB, Lab, $L^*u^*v^*$, XYZ, HSV,

HLS, YCrCb, Opponent RGB (Section III.4.1.1).

- (2) **Patch Radius** - *3 bits* - Defines the patch size by $\text{Patch}_{width}, \text{Patch}_{height} = 2 \times \text{Radius} + 1$. The radius values: 3, 4, 5, 6, 7, 8, 9, 10.
- (3) **Filter Kernel** - *3 bits* - The width and height of the 2D Gaussian blur kernel. Values: 1, 3, 5, 7, 9, 11, 13, 15 (Section III.4.1.3).
- (4) **Sigma** - *3 bits* - Gaussian kernel standard deviation. Values: 0.8, 1.1, 1.5, 3, 6, 12, 24, 30 (Section III.4.1.3).

The next sequences define three Random Forest parameters for better learning capacities (Section III.7).

- (5) **Depth** - *3 bits* - The depth of each tree in the forest. Values: 5, 10, 15, 20, 25, 30, 35, 40.
- (6) **Size** - *3 bits* - The number of trees in the forest. Values: 40, 50, 60, 70, 80, 100, 120, 140.
- (7) **Best Split** - *3 bits* - The number of randomly selected variables at each node for the best split. Values: 4, 8, 12, 16, 20, 25, 30, 35.

The last sequences select the parameters for the edge detection and intensity gradient calculation (Section III.4.1.5).

- (8) **Canny kernel size** - *1 bits* - The Canny Edge Detector's kernel width and height. Values: 3,5 .
- (9) **Canny Threshold** - *3 bits* - The values specify the lower threshold of the edge evaluation function. The upper and lower threshold ratio is 3 : 1. The pixel accepted as edge if the pixel gradient is higher than the upper threshold, or it is between the two thresholds, but connected to a pixel above the upper threshold. The Values: 15, 25, 35, 45, 55, 65, 75, 85.
- (10) **Pixel Gradient Calculator** - *1 bits* - Select the method for image gradient calculation for the later gradient angle calculation Values: Sobel, Sharr (Section III.4.1.4).

IV.4.2. Image Processing

Before calculating the descriptor, we process the incoming image through various image processing steps to obtain the necessary module information. The functions process the whole image and stores it for to import later the patch is for the module calculation. The primary processing functions are the following:

- (1) **Histogram Equation** - Making the Histogram Equalization on the Y intensity plane and transform back to the RGB color space (Section III.4.1.2).
- (2) **Image Blurring** - Using Gaussian blur with the preset parameters—in the Pre-processing stage—the image blurred and stored (Section III.4.1.3).
- (3) **Color Space transform** - The blurred image is transformed into one of the genome specified color space (Section III.4.1.1).
- (4) **Distance Transform** - Calculates the Canny edges and the Euclidean distance of each pixels from the these edges (Section III.4.4).
- (5) **Calculate image gradients** - Calculates the g_x and g_y gradients of the image using the Image gradient calculator. The angles (θ) computed from the resulted gradients (Section III.4.1.4).
- (6) **Invariant Color features** - The blurred RGB image is used to calculate Gevers' and Geusebroek's invariant color features (Section III.4.2.1).

IV.4.3. Module bank and activation

The second part of the genome controls the descriptor assembly (Figure 4.4). The 185 element long binary substring is subdivided into modules. These sub-strings are the binary representation of the modules. They have an equal length as the number of values are produced by the corresponding module, and the binary value specifies that the corresponding value is active (=1) or inactive (=0). For example if a module m_n —which returns in total four values—and is represented by $b_n = \langle 1001 \rangle$ string in the genome, $m_n(1)$ and $m_n(4)$ are inserted into the descriptor as active values and the inactive $m_n(2), m_n(3)$ are spared. When a module has only inactive elements, the algorithm flushes from the descriptor module list. This organization gives the flexibility to extend the module list infinitely. The following features, filters, and calculations are contained in our Module Bank:

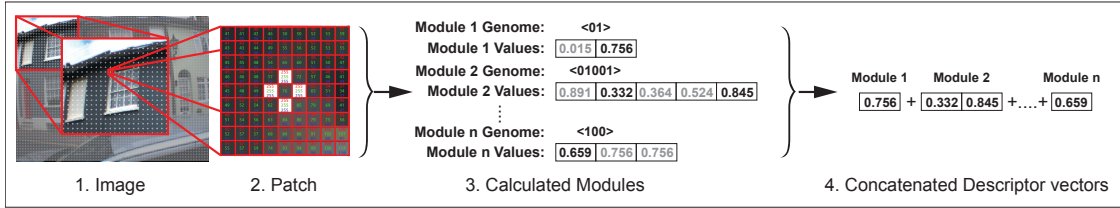


Figure 4.4: Modular Descriptor Extraction Process

- (1) **Normalized Position** - 2 values - 2D normalized image coordinates of the patch centers ($np(1) = I_x/I_{width}$, $np(2) = I_y/I_{height}$) to aid the classification. The image points at the top of the image would be classified as roof or sky, at the bottom more likely as street and at the middle as wall, door or window.
- (2) **Mean(μ) and Standard Deviation(σ) of each color channel** - 6 values - To obtain descriptive color information and not to overload the descriptor we calculate only the μ and σ of each channel of the actual color space.
- (3) **2nd Order Central Moments** ($\mu'_{20}, \mu'_{11}, \mu'_{02}$) - 9 values - Information about the principal axes of inertia of the patch (Section III.4.2.3).
- (4) **3rd Order Central Moments** ($\mu'_{30}, \mu'_{21}, \mu'_{12}, \mu'_{03}$) - 12 values - Symmetry information about the 2D shape, or skewness of the patch (Section III.4.2.3).
- (5) **4th Order Central Moments** ($\mu'_{40}, \mu'_{31}, \mu'_{22}, \mu'_{13}, \mu'_{04}$) - 15 values - Measurements of the patch the kurtosis, describes the peakedness of the distribution (Section III.4.2.3).
- (6) **5th Order Central Moments** ($\mu'_{50}, \mu'_{41}, \mu'_{32}, \mu'_{23}, \mu'_{14}, \mu'_{05}$) - 18 values - Estimation of further shape parameters (Section III.4.2.3).
- (7) **Hu Moments** ($\phi_1, \phi_2, \phi_3, \phi_4, \phi_5, \phi_6, \phi_7, \phi_8$) - 21 values - The rotation invariant Hu moments (Section III.4.2.4) calculated for each image channels
- (8) **Gevers' color invariant l features** (l_1, l_2, l_3) - 6 values - Mean(μ) and Standard Deviation(σ) of the photometric color invariant features for both Matte and Shiny surfaces (Section III.4.2.1) for each RGB channels.
- (9) **Gevers' color invariant c features** (c_1, c_2, c_3) - 6 values - Mean(μ) and Standard Deviation(σ) of the photometric color invariant features for Matte and Dull surfaces (Section III.4.2.1) for each RGB channels.

- (10) **Geusebroek color invariant C and H features** - 4 values - Mean(μ) and Standard Deviation(σ) of Geusebroek's color invariants (Section III.4.2.1) for each channels
- (11) **Affine Moment Invariant features** ($I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8, I_9, I_{10}$) - 30 values - The invariant features (Section III.4.2.5) for each channels
- (12) **Distance Transform** - 2 values - Mean(μ) and Standard Deviation(σ) of the Distance Transform (Section III.4.4) over the patch.
- (13) **Eigen values** ($\lambda_1, c\lambda_2, c\lambda_3$)- 9 values - The three highest Eigen values of each color channel
- (14) **Gradient Angles** - 45 values - The average gradient angles (Section III.4.1.4). The average of all the patch and the averages of 4 subregions in the patch (Figure 4.5), and the differences of these gradient averages ($\theta_{00-11}, \theta_{00-12}, \theta_{00-21}, \theta_{00-22}, \theta_{11-12}, \theta_{11-21}, \theta_{11-22}, \theta_{12-21}, \theta_{12-22}, \theta_{21-22}$). The subregions size was set to radius to be aligned with the parameter changes during the optimization.

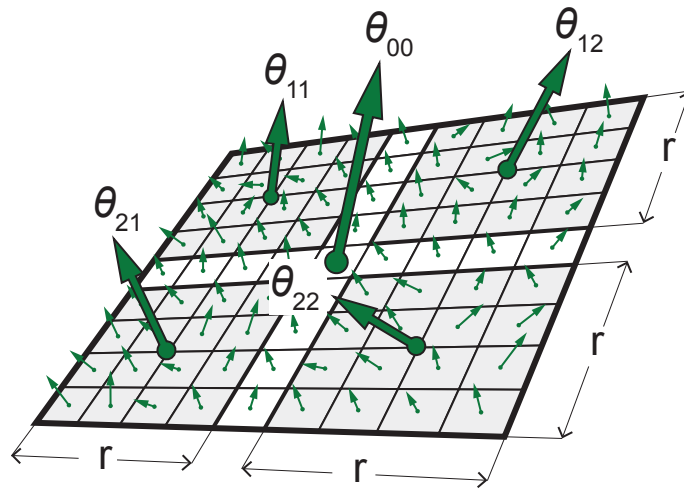


Figure 4.5: Average Gradient angles on the patch and sub-patches

- (15) **Affine Moment Invariant for colour images** ($I_{C0}^{(a,b)}, I_{C1}^{(a,b)}, I_{C12}^{(a,b)}, I_{C2}^{(a,b)}, I_{C23}^{(a,b)}, I_{C4}^{(a,b,c)}$)- 7 values by Suk and Flusser (2009)

IV.4.4. Objective function

To evaluate the genome we split the objective function into two function: the segmentation (Algorithm 4) and the invariance optimization(Algorithm 5). Both functions return an objective score which later is converted in each cycle to fitness score. The results of each evaluation operation can be weighted to strengthen a particular feature or invariance. For example, if we would like the descriptor active in segmentation we would multiply with higher values the segmentation scores, with this, the genetic algorithm only accepts the genomes with the lower scores, therefore, better segmentation results.

Algorithm 4 Objective Function pseudocode 1

```
1: procedure ACCURACY EVALUATION
2:   repeat
3:     Descriptor Extraction for all keypoints
4:     Average descriptor extraction time measure
5:     Random Forest classification
6:     Feature Segmentation
7:     Overall and Average Recognition Rate calculation
8:     Average Segmentation Time computation
9:     Objective Score calculation
10:  until reached the last test image
11: end procedure
```

Algorithm 5 Objective Function pseudocode 2

```
1: procedure INVARIANCE EVALUATION
2:   repeat
3:     Evaluation setup for specific invariance
4:     repeat
5:       Image transformation
6:       Descriptor Extraction for all keypoints for the image pair
7:       Brute Force descriptor vector matching
8:       Result storing
9:     until reached the last test image
10:  until all invariance is being tested
11:  Result plotting by invariance case
12: end procedure
```

IV.5. Evaluation and Analysis

We applied the objective function's evaluation process to the most referenced descriptors in order to compare their performance. After the evaluation, we generated confusion matrices from the Random Forests' test results to compare the performance of each descriptor. As there is multiple class in the scene, we created one confusion matrix for each class results. We used this data to calculate the performance metrics (Section III.4.7.2.) of the class. We calculated the Accuracy, Precision, Recall, Specificity, Negative Predictive Value and the F-Measure of each descriptor over the class.

The speed probe measured two different properties. The *average descriptor extraction time* shows the time of the descriptor retrieval from the whole image with an equal amount of key-points. The *average segmentation time* demonstrates the duration of the forest evaluation and image segmentation.

A set of two identical images served for the invariance test. To compare the impact of the deformation, we kept one of the pairs as ground truth and transformed the other. During the rotation, resize, and affine transformation assessments (Figure 4.6), the same transformation matrix converts the keypoints and image pixels to place the reference points at the correct spot. Then we compare the extracted descriptor vectors with the Brute Force matching function (Bradski, 2000).



Figure 4.6: Affine transformation cases in invariance evaluation: **(a)-(j)** Case 1-10
Source: Takacs, Toledano-Ayala *et al.*, 2016

IV.6. Fitting of the algorithm into PTAM

The proposal of this project is to fit the new modular descriptor into the Parallel Tracking and Mapping application for classification. There were three points where we inserted modifications in the algorithm (Figure 4.7):

1. During the start, the system loads the trained "Random Forest", for the classification process
2. After the stereo initialization, the algorithm calculates the feature descriptors at the position of the first set of points on the virtual map.
3. Between the tracking and mapping module. Based on the newly tracked corner points the descriptors are calculated and set their classes.

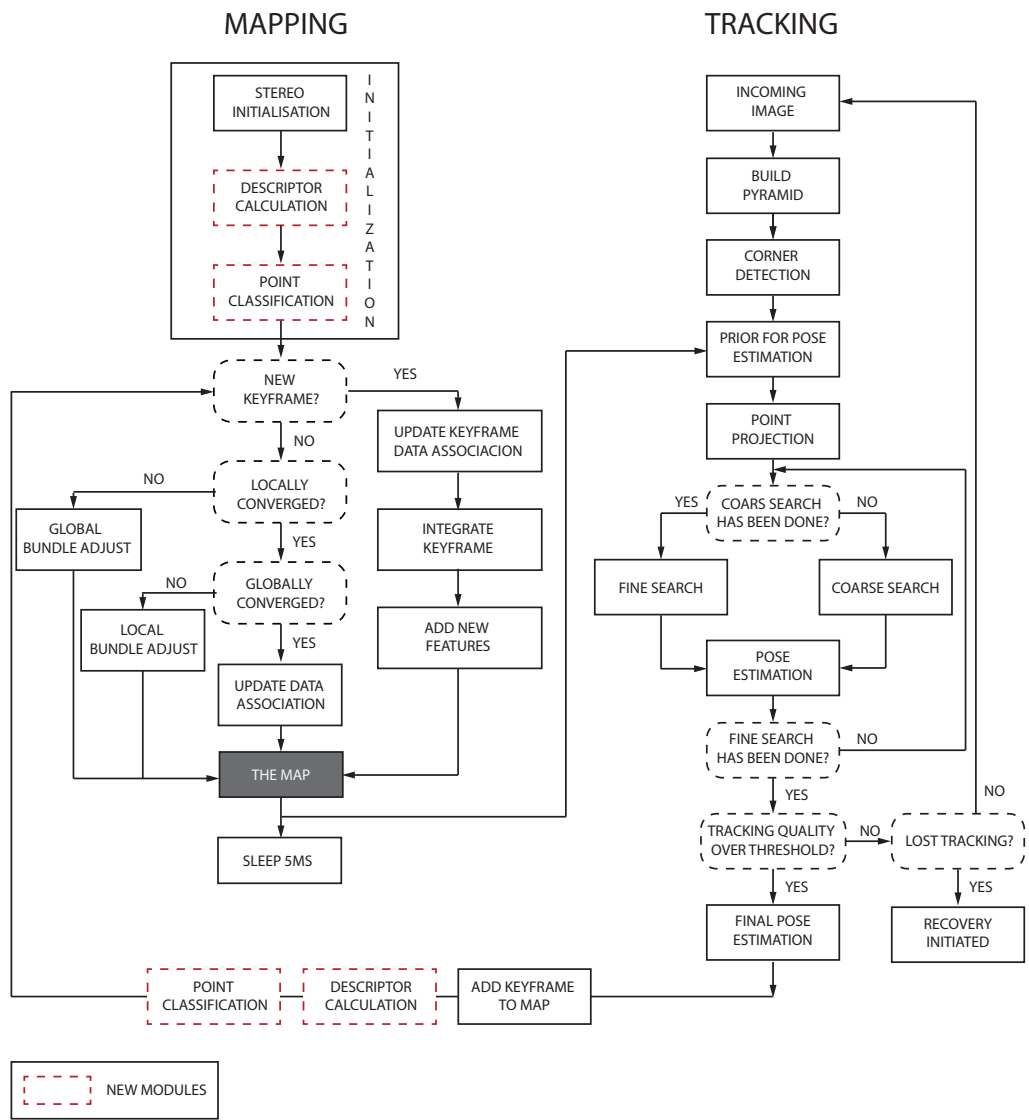


Figure 4.7: The parallel mapping and tracking thread
 Source: Klein and Murray, 2007

V. RESULTS & DISCUSSION

V.1. Results

In the following sections we will present the outcomes of the particular objectives. In Section V.1.1 we will show the performance of the proposed low level image descriptor. Despite its variance to transformation, the descriptor is effective in segmentation and is calculated very fast. Section V.1.2 will present the outcomes of the self-adjusting optimization algorithm and the evaluation of the new modular descriptor. The performance evaluation shows that the framework produce a descriptor with good variance to geometric and illumination changes and with exceptional segmentation performance. In Section V.1.3 we present the performance of the PTAM () algorithm with the integrated new descriptor.

V.1.1. EDD results

We contrasted the EDD with the state-of-the-art descriptors' performance. Table 5.1 and 5.2 show the accuracy evaluation results. Each table reports the *overall recognition rate* in the first column and the *average recognition rate* of each class in the rest of the table. These analyses show a similar pattern with all the tested descriptors. After the training to a specific scene, the descriptors gave better results in classification. Throughout the testing, the Opponent SIFT, and SIFT descriptors gave the most reliable performance, above 70% with the specific and 58% with the general environments. The Opponent SIFT descriptor was designed for a color environment, but in the evaluations, it had a poorer performance than the greyscale SIFT descriptor. The EDD came out third overall, after the two SIFT descriptors. This shows that a trained environment-specific descriptor with carefully chosen modules can substitute the state-of-the-art descriptors in classification tasks efficiently.

Throughout the Brighton scene (Table 5.1), the EDD was operating with high precision in most of the classes except in the "Roof" class. Sparse (little image gradient) areas like the "Wall" were still producing good results but gave its best performance classifying the "else" group. The classification results in Figure 5.1a show the outcome, where each color represents a class: the yellow circles the wall

type, the red circles the window or door class, the blue circles the roof class, and the green circles the else class. We can observe that despite the outliers and the misclassified object (Figure 5.1b), the descriptor’s overall performance is good.

Table 5.1: EDD evaluation results - Average true positives at the Brighton scene.(%)

Descriptor	Average true positive by class				
	All	Else	Doors & Windows	Roof	Wall
SIFT	74.67	82.58	61.03	70.15	79.14
Opponent SIFT	72.92	83.97	56.27	77.91	79.40
SURF	53.45	63.26	46.68	17.55	59.48
Opponent SURF	55.43	74.74	43.89	34.24	62.55
EDD	67.58	92.36	45.68	55.91	72.39

Table 5.2: EDD evaluation results - Average true positives with 100 images.(%)

Descriptor	All	Average true positive by class								
		Else	Building	Door	Window	Car	Pavement	Road	Sky	Vegetation
SIFT	58.27	14.51	70.21	0	0	26.58	17.31	49.12	68.28	20.15
Opp. SIFT	58.01	0	69.65	0	0	0	0	48.00	66.13	22.11
SURF	23.05	5.36	62.81	3.24	11.91	9.25	7.00	28.78	53.27	13.18
Opp. SURF	28.94	5.66	64.49	0	10.65	9.51	7.81	40.23	55.54	14.37
EDD	46.63	9.88	72.48	0	9.47	17.32	14.50	53.31	64.22	14.98



Figure 5.1: EDD results: **(a)** keypoint classification result, **(b)** segmentation result
Source: Takacs, Toledano-Ayala *et al.*, 2016

The photometric analysis and the invariance test (Figure 5.2) gave two different kinds of results for the EDD. During the *Light intensity change*, *light intensity shift*, *light color change*, and *blur* test, it gave the most reliable performance. This is due to the local normalization on the patch level and the blurring during the image-preparation period. On the other hand, the descriptor still had a poor performance during the image transformation assessment.

The measured *average segmentation time* (Figure 5.3a) and *average descriptor extraction time* (Figure 5.3b) indicate that the EDD is by far the fastest in both cases.

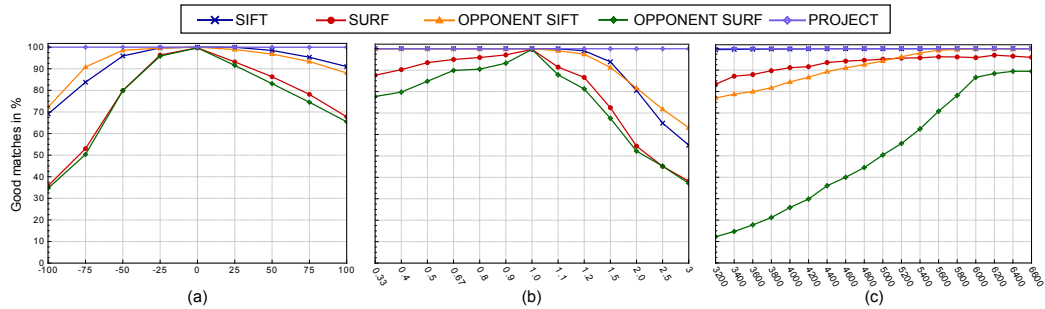


Figure 5.2: EDD - Photometric Analysis results: **(a)** Intensity shift (with offset o_1), **(b)** Intensity scaling (with scalar a), **(c)** Illumination Color temperature (K)
 Source: Takacs, Toledano-Ayala *et al.*, 2016

In the extraction time, we included the image-processing and the feature-calculation times. We can observe in Figure 5.3a how much time it takes for the computer to extract all the descriptor vectors from the whole image (average 3600 vector/image). The EDD is on average 26 times faster than the Opponent SIFT and needs 4.5 times less time than the SURF. However, the algorithm has been run on different computers, the ratios between the descriptors' calculation time maintained invariant. The results are due to the calculation efficient descriptor features. The average time the computer takes (Figure 5.3a) to read and calculate the necessary data for the descriptor is based on the outcome of the classification, which segmented the window and door areas. The result data shows, while the most effective SIFT descriptor needed 44 seconds on average, the OpponentSIFT descriptor for the same work needed two times as much effort in time. The OpponentSURF descriptor occupied 56 seconds for the work, which is two times slower than the EDD, and its performance in detection was inferior to this descriptor. These results correlated with the finding of Valgren and Lilienthal 2010, where the SIFT descriptor was more accurate in feature matching but took a considerably longer time frame.

Figure 5.1 shows the outcome of the segmentation algorithm, where we see the strength of window detection and the weakness of the descriptor vector regarding distortion and rotation.

V.1.2. Modular Descriptor

The genetic algorithm after 150 generations resulted an optimized descriptor with good segmentation power and similar invariance to rotation, blur, size and affine changes as the most used state-of-the-art descriptors. The final optimized paramet-

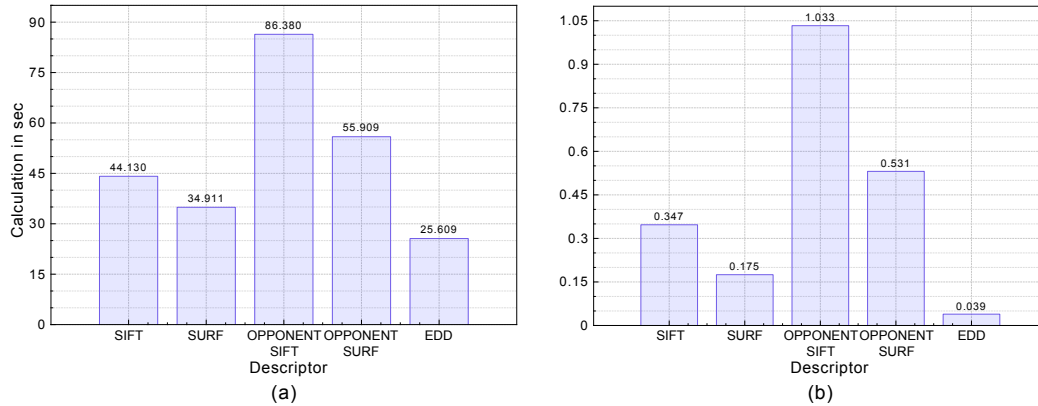


Figure 5.3: Descriptor performance in time: **(a)** average object segmentation time, **(b)** average descriptor extraction time
 Source: Takacs, Toledano-Ayala *et al.*, 2016

ers listed in Table 5.3 and final descriptor setup listed in Table 5.4 and 5.5.

Table 5.3: Optimized module genomes.

Parameter	Value
Color Space	XYZ
Patch Radius	10
Gaussian Kernel	7
Gaussian Blur sigma	1.5
Random Tree size	100
Random Tree depth	35
Number of values for split	16
Canny Kernel size	5
Canny low threshold	65
Gradient calculator	Sobel

The invariance results plotted on separate graphs with the average match rate on the y axes and the transformation cases on the x axes (Figure 5.5 and 5.6). The modular descriptor also reached a significant improvement in the transformation invariance compared to EDD, and shown a midrange invariance towards affine, blur, and size changes, but it kept highly variant towards rotation (Figure 5.5). Also performed poorly during the photometric analysis (Figure 5.5) which suggests that for that kind of environments need a different type of approach and new modules invariant for this kind of changes.

We contrasted the Modular Descriptor with the most referenced descriptors' performance with the same Random Forest Training parameters. Figure 5.7 shows

Table 5.4: Optimized module genomes.

Module	Size		Genome
	Original	Optimized	
Normalized Position	2	2	11
Mean and Standard Deviation of each image channel	6	4	101,110
2 nd Order Normalized Central Moments of each channel	9	4	011,000,110
3 rd Order Normalized Central Moments of each channel	12	5	1001,0100,0101
4 th Order Normalized Central Moments of each channel	15	6	00101,00001,01110
5 th Order Normalized Central Moments of each channel	18	10	100111,100101,111000
Hu moments	21	12	0100101,1011101,0001111
Mean and Standard Deviation of l_1, l_2, l_3	6	4	101,101
Mean and Standard Deviation of c_1, c_2, c_3	6	4	100,111
Mean and Standard Deviation of the H and C features	4	2	10,01
Affine moments	30	11	0101000110,1010010100,1000000110
Mean and Standard Deviation of Distance Transform	2	2	1,1
Eigen Values	9	4	100,110,100
Gradient angels	45	23	01110,01010,01110,0010111101,1010110100,0001011010
Total	188	93	

Table 5.5: Optimized module values.

	Module	Active Values		Module	Active Values
1	Position	normalized x and y	8	Mean and StD. of l_1, l_2, l_3	$\mu(l_1, l_3)$ $\sigma(l_1, l_3)$
2	Mean and StD. of image channels	$\mu(\text{CH1, CH3})$ $\sigma(\text{CH1, CH2})$	9	Mean and StD. of c_1, c_2, c_3	$\mu(c_1)$ $\sigma(c_1, c_2, c_3)$
3	2 nd Cent. Mom.	μ'_{11}, μ'_{02} (CH1) μ'_{20}, μ'_{11} (CH3)	10	Mean and StD. of the H and C	$\mu(\text{H})$ $\sigma(\text{C})$
4	3 rd Cent. Mom.	μ'_{30}, μ'_{03} (CH1) μ'_{21} (CH2) μ'_{21}, μ'_{03} (CH3)	11	Affine moments	I_2, I_4, I_8, I_9 (CH1) I_1, I_3, I_6, I_8 (CH2) I_1, I_8, I_9 (CH3)
5	4 th Cent. Mom.	μ'_{22}, μ'_{04} (CH1) μ'_{04} (CH2) $\mu'_{31}, \mu'_{22}, \mu'_{13}$ (CH3)	12	Mean and StD. of Dist. Trans.	μ and σ Distance Transform
6	5 th Cent. Mom.	$\mu'_{50}, \mu'_{23}, \mu'_{14}, \mu'_{05}$ (CH1) $\mu'_{50}, \mu'_{23}, \mu'_{05}$ (CH2) $\mu'_{50}, \mu'_{41}, \mu'_{32}$ (CH3)	13	Eigen values	λ_1 (CH1) λ_1, λ_2 (CH2) λ_1 (CH3)
7	Hu moments	ϕ_2, ϕ_5, ϕ_7 (CH1) $\phi_1, \phi_3, \phi_4, \phi_5, \phi_7$ (CH2) $\phi_4, \phi_5, \phi_6, \phi_7$ (CH3)	14	Gradient angles averages and differences	$\theta_{11}, \theta_{12}, \theta_{21}$ (CH1) θ_{11}, θ_{21} (CH2) $\theta_{11}, \theta_{12}, \theta_{21},$ (CH3) $\theta_{00-21}, \theta_{11-12}, \theta_{11-21},$ $\theta_{11-22}, \theta_{12-21}, \theta_{21-22}$ (CH1) $\theta_{00-11}, \theta_{00-21}, \theta_{11-12},$ $\theta_{11-21}, \theta_{12-21}$ (CH2) $\theta_{00-22}, \theta_{11-21}, \theta_{11-22},$ θ_{12-22} (CH3)

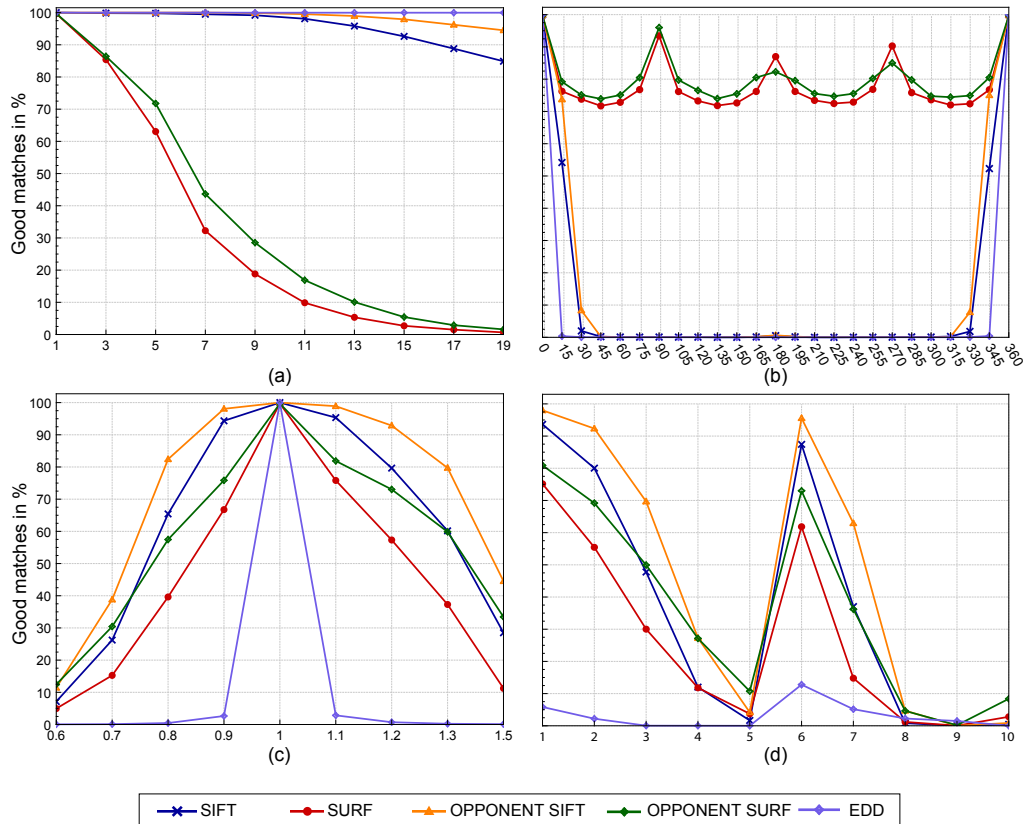


Figure 5.4: EDD - Transformation Invariance results: **(a)** Gaussian blur (kernel size), **(b)** Image rotation (angle), **(c)** Image resize (size), **(d)** Affine transformation (cases)

Source: Takacs, Toledano-Ayala *et al.*, 2016

how each descriptor performed classifying the image points. Each color represent a class: red - Windows & Doors, green - else, yellow - wall, blue - roof. We present the accuracy evaluation results in Table 5.6. Each table reports the *overall recognition rate* in the first column and the *average recognition rate* of each class in the rest of the table. Tables 5.7, 5.8, 5.10 and 5.9 show the confusion matrices and performance evaluation of each descriptor at with the four classes. The same performance metrics are plotted in Figures 5.8, 5.9, 5.11 and 5.10.

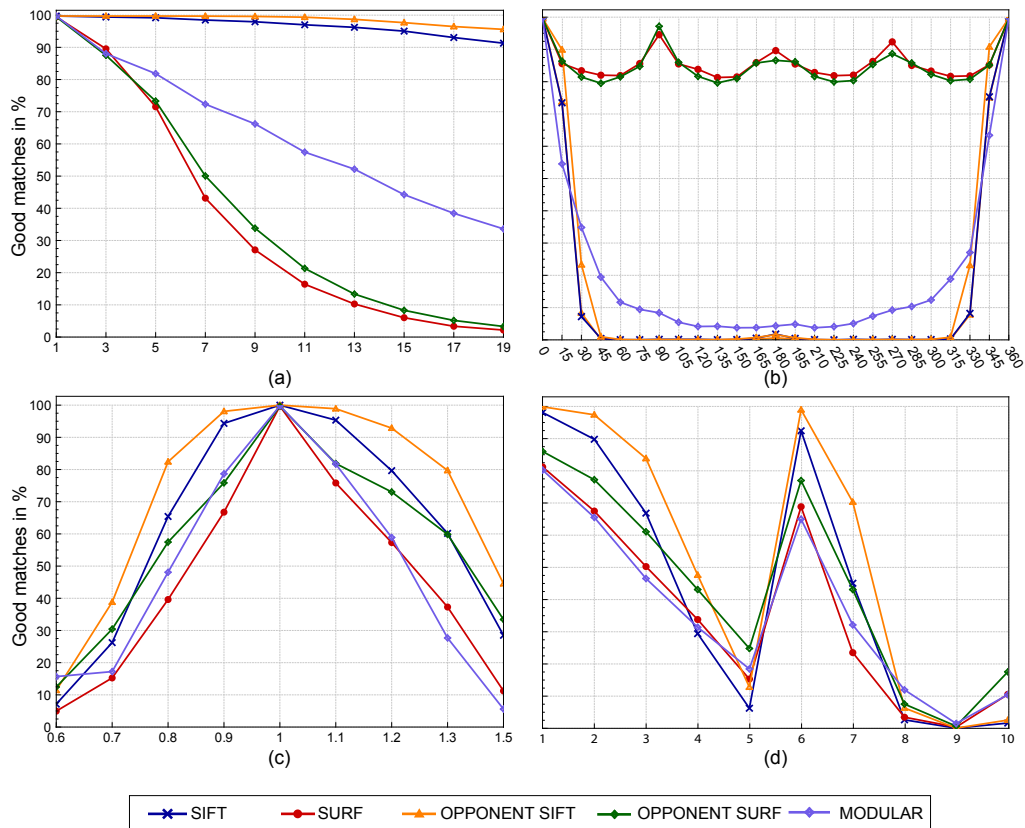


Figure 5.5: Modular descriptor - Transformation Invariance results: **(a)** Gaussian blur (kernel size), **(b)** Image rotation (angle), **(c)** Image resize (size), **(d)** Affine transformation (cases)

Table 5.6: Modular descriptor - Average true positives at the Brighton scene.(%)

Descriptor	All	Average true positive by class			
		Else	Doors, Windows	Roof	Wall
SIFT	73.45	78.53	67.23	69.07	71.80
Opponent SIFT	72.45	77.66	66.76	79.15	69.54
SURF	56.18	57.58	55.94	13.96	58.34
Opponent SURF	57.63	65.16	52.30	34.72	57.84
EDD	67.42	63.89	58.17	59.77	61.92
Modular Descriptor	73.70	91.86	55.03	72.13	73.84

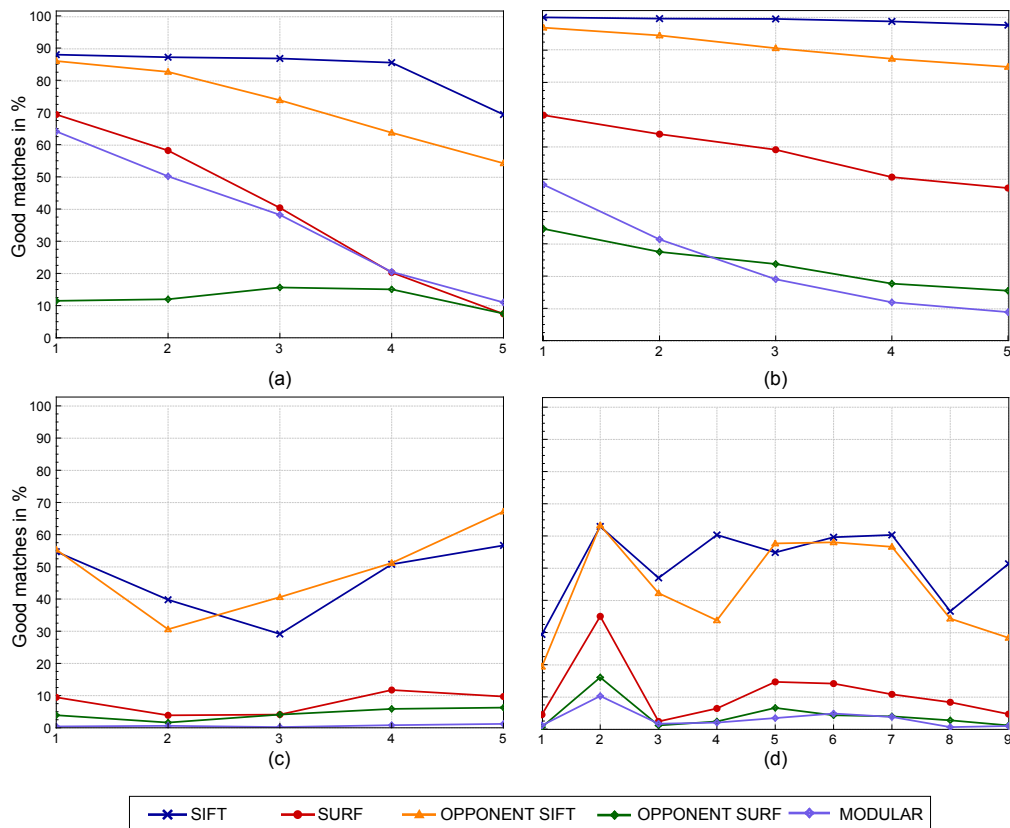


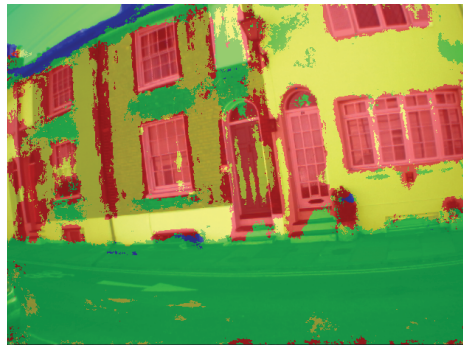
Figure 5.6: Modular descriptor - Photometric Analysis results: **(a)** JPEG compression (cases), **(b)** Light change (cases), **(c)** Light condition change (Notre Dame), **(d)** Light condition change 2 (Mexico)



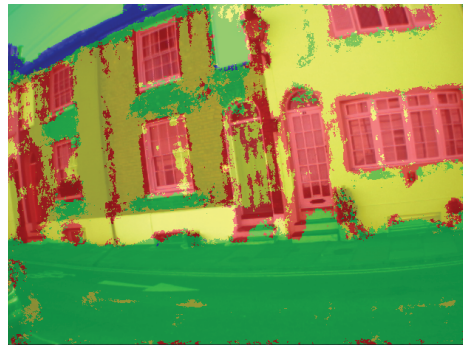
(a)



(b)



(c)



(d)



(e)



(f)



(g)



(h)

Figure 5.7: Classification Results for each Descriptor: **(a)** Base image, **(b)** Ground truth, **(c)** SIFT, **(d)** Opponent SIFT, **(e)** SURF, **(f)** Opponent SURF, **(g)** EDD, **(h)** Modular

Table 5.7: Modular descriptor "Else" class - Classification outcomes and Measurements.

	SIFT	SURF	Opponent SIFT	Opponent SURF	EDD	Modular
True Positive	95095.40	52986.40	92506.60	56373.80	92347.00	89470.40
False Positive	23847.80	37013.40	23372.00	25829.40	14408.40	7177.60
False Negative	19112.40	61221.40	21701.20	57834.00	21860.80	24737.40
True Negative	147144.41	133978.80	147620.20	145162.80	156583.80	163814.61
Accuracy	84.94%	65.56%	84.20%	70.67%	87.28%	88.81%
Precision	80.06%	58.48%	80.16%	68.42%	86.70%	92.71%
Recall	83.93%	46.53%	81.57%	50.11%	81.28%	79.26%
Specificity	85.81%	78.02%	85.96%	84.65%	91.35%	95.58%
Neg.Pred.Val.	88.30%	68.51%	86.89%	71.54%	87.50%	86.73%
F-Measure	81.83%	51.75%	80.76%	57.67%	83.89%	85.36%

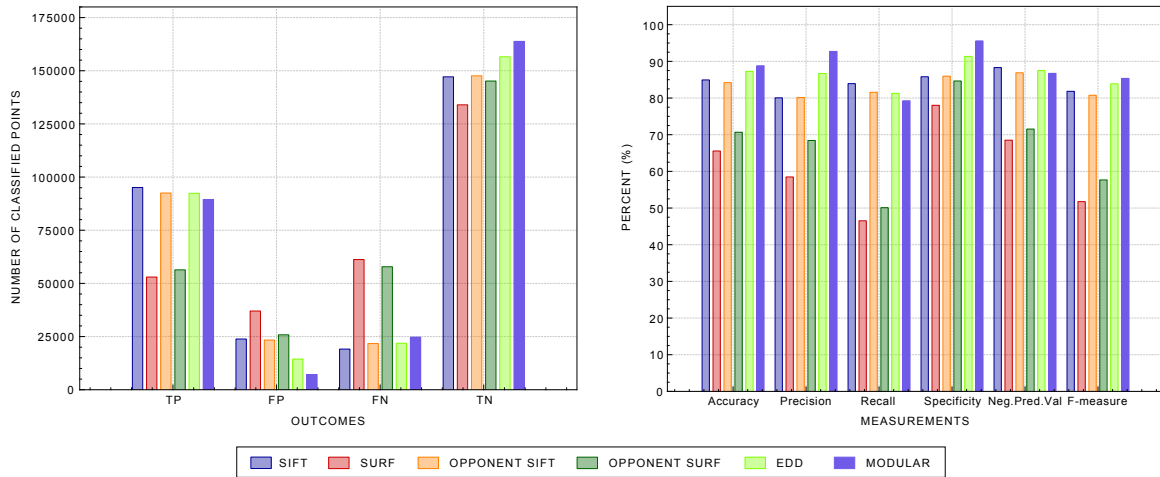


Figure 5.8: "Else" class classification results and Measurements

Table 5.8: Modular descriptor "Doors & Windows" class - Classification outcomes and Measurements.

	SIFT	SURF	Opponent SIFT	Opponent SURF	EDD	Modular
True Positive	948997.00	42877.40	47187.60	46515.80	43579.80	50882.00
False Positive	26151.20	41503.60	27632.60	48476.20	40014.40	47167.40
False Negative	22131.60	28251.20	23941.00	24612.80	27548.80	20246.60
True Negative	187920.20	172567.80	186438.80	165595.20	174057.00	166904.00
Accuracy	83.07%	75.54%	81.92%	74.37%	76.31%	76.36%
Precision	65.00%	51.52%	63.63%	50.21%	51.54%	52.27%
Recall	68.41%	59.90%	65.98%	65.13%	60.59%	71.58%
Specificity	87.84%	80.78%	87.19%	77.57%	81.31%	78.04%
Neg.Pred.Val.	89.49%	85.91%	88.63%	87.01%	86.37%	89.27%
F-Measure	66.52%	55.07%	64.45%	56.20%	55.69%	60.04%

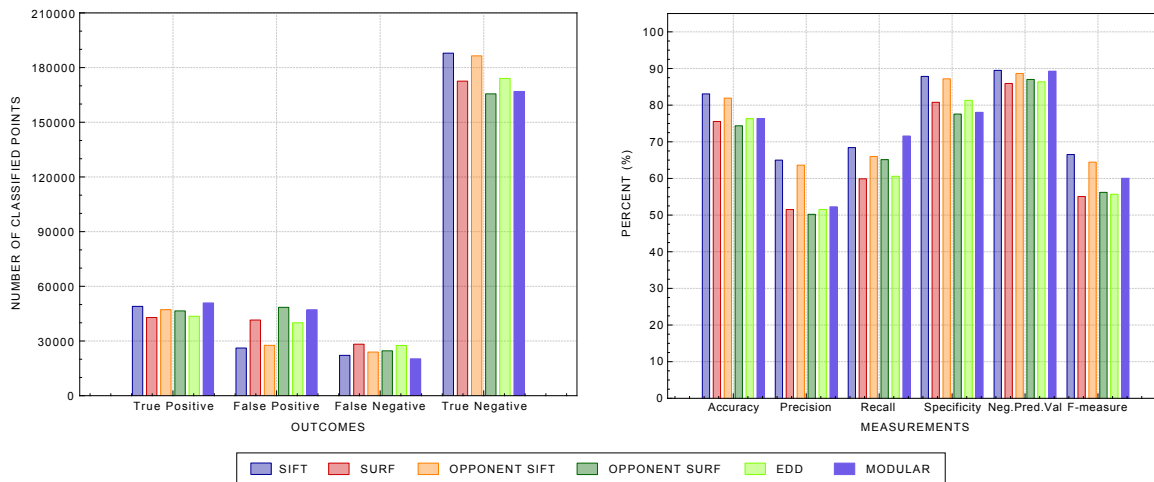


Figure 5.9: "Doors & Windows" class classification results and Measurements

Table 5.9: Modular descriptor "Roof" class - Classification outcomes and Measurements.

	SIFT	SURF	Opponent SIFT	Opponent SURF	EDD	Modular
True Positive	2527.20	793.80	3566.00	1514.80	2465.80	4035.40
False Positive	1242.20	4733.40	682.20	2565.60	1811.00	2817.00
False Negative	3275.20	5008.60	2236.40	4287.60	3336.60	1767.00
True Negative	278155.41	274664.22	278715.41	276832.00	277586.59	276580.59
Accuracy	98.42%	96.58%	98.98%	97.60%	98.20%	98.39%
Precision	70.19%	15.94%	81.20%	39.67%	60.13%	61.21%
Recall	44.70%	14.48%	62.18%	27.26%	43.62%	69.74%
Specificity	99.55%	98.31%	99.76%	99.08%	99.35%	98.99%
Neg.Pred.Val.	98.84%	98.21%	99.21%	98.47%	98.81%	99.37%
F-Measure	53.60%	14.67%	69.11%	31.42%	48.92%	63.86%

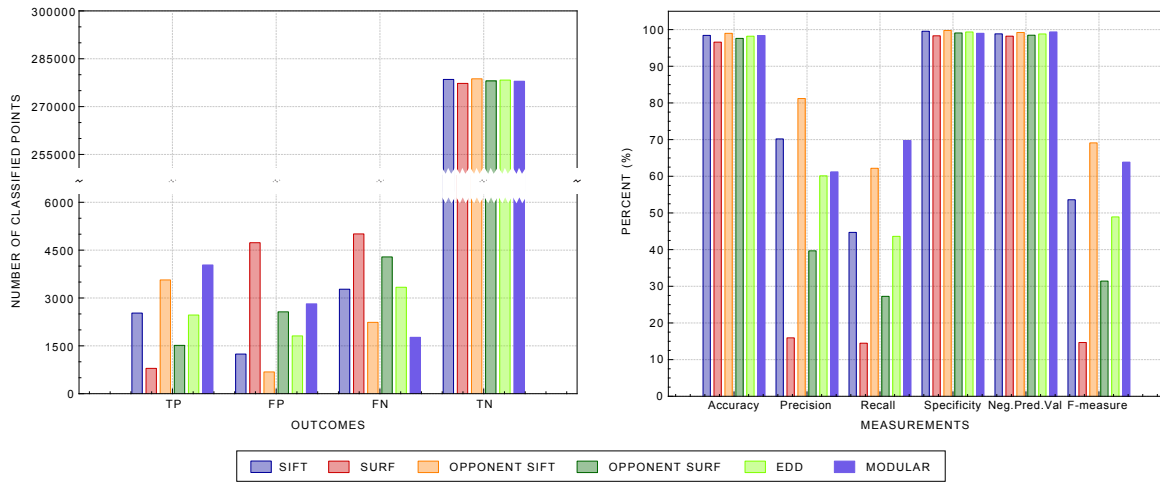


Figure 5.10: "Roof" class classification results and Measurements

Table 5.10: Modular descriptor "Wall" class - Classification outcomes and Measurements.

	SIFT	SURF	Opponent SIFT	Opponent SURF	EDD	Modular
True Positive	63567.40	61284.80	64182.80	63534.00	59182.40	62722.60
False Positive	23771.80	44007.20	26070.20	40390.40	31391.20	20927.60
False Negative	30493.80	32776.40	29878.40	30527.20	34878.80	31338.60
True Negative	167367.00	147131.61	165068.61	150748.41	159747.61	170211.20
Accuracy	80.97%	73.08%	80.38%	75.13%	76.76%	81.67%
Precision	72.58%	58.71%	70.52%	61.12%	64.98%	75.20%
Recall	67.16%	64.72%	67.55%	67.16%	62.43%	66.49%
Specificity	87.66%	76.93%	86.38%	78.85%	83.70%	89.10%
Neg.Pred.Val.	84.62%	81.69%	84.75%	83.14%	82.13%	84.49%
F-Measure	69.65%	61.31%	68.93%	63.90%	63.61%	70.33%

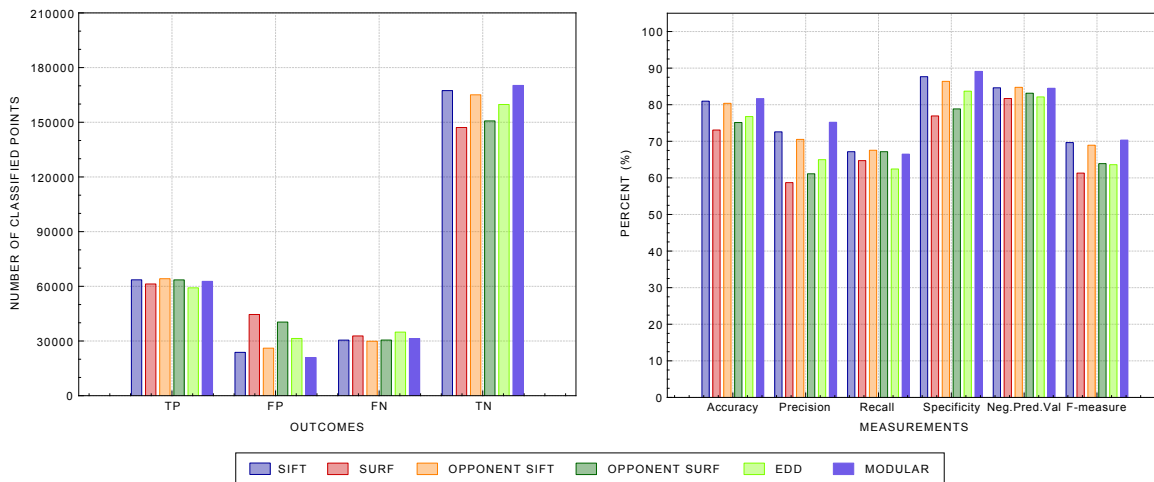


Figure 5.11: "Wall" class classification results and Measurements

V.1.3. PTAM implementation

The results of the implementation of the new optimized modular descriptor can be compared with the following two image pairs in Figure 5.12 and in Figure 5.13. The left image shows the virtual map built by PTAM, and the right image visualizes the interest points by augmenting on the real video feed.

The color code in the images in Figure 5.12 (PTAM's original output) represents the source level of the interest points in the four level image pyramid. The red points are from level 0, the yellow from level 1 green from level 2 and blue level 3. In contrast, in Figure 5.13 the colors represent the classification results. The red points show the else class, the yellow color represents the window class, the green points are from the roof class, and the blue color presents the points from the wall class.

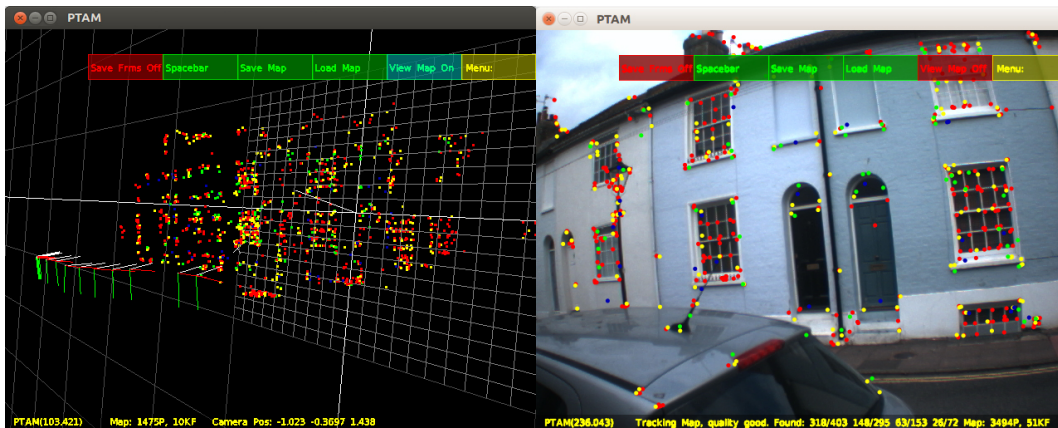


Figure 5.12: Original PTAM environment
Source: Takács, 2013

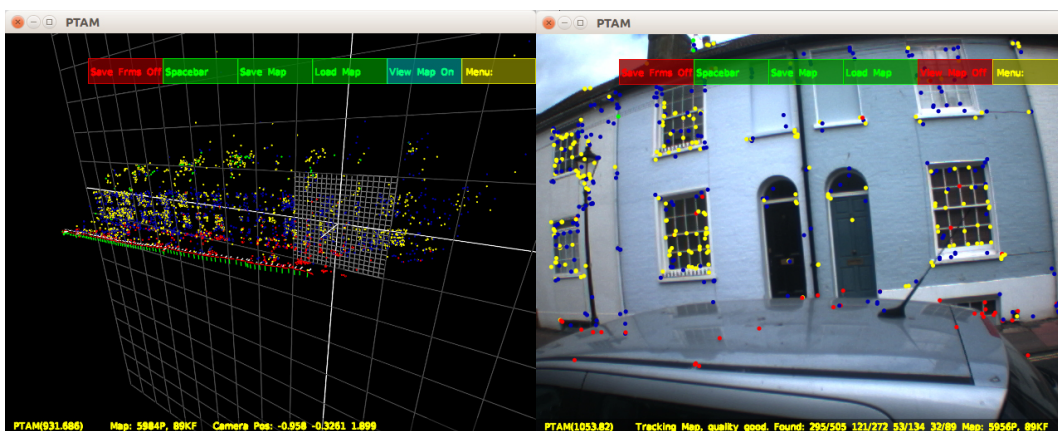


Figure 5.13: PTAM with the classified interest points and parallel map

V.2. Discussion

This research showed that newly designed feature descriptor—generated and optimized by the proposed proposed framework—integrated into PTAM performs robust and real-time facade segmentation. The genetic algorithm created, trained and optimized the modular descriptor with the aid of the incorporated Random Forest module. The descriptor was optimized without losing their descriptive strength to make suitable for AR applications.

Chapter IV showed the first contribution of this work, the design and evaluation of the Environment Dedicated Descriptor. The low level image descriptor that invariant to the changes that can occur in outdoor environments. The shows that a compound of a few existing variables—that is cheap to calculate and chosen for the environment—is capable of the same local feature description as a common descriptor. The distance transform brings valuable information about sparse and dense areas, color information holds extra information and the central moments can be calculated easily while being independent. The results support the concept that although these calculations are weak alone to describe a local feature but in concatenated form strengthen each other. The proposed descriptor is fast to calculate, robust in segmentation and invariant to light condition changes.

During the evaluation period the feature extraction with the new environment-dedicated descriptor was studied with respect to speed, accuracy, and invariance. EDD examined the scene and calculated quickly the local features with distinctive information. The complete descriptor was used for semantic-feature extraction, with the aid of a trained Random Forest classifier. The results show that although the most popular descriptors have reliable performance in feature detection, a descriptor which is dedicated to a specific environment—that is, a street environment with buildings—can have similar accuracy but in a shorter time period. The EDD also responds well to the light-condition variations. This projects a new path to investigate a trained dynamic descriptor that can adjust characteristics of the retrieved information according to the environment. Based on the results, the EDD should be stabilized for transformation invariance, and another version should be created for different environmental characteristics.

Chapter IV presented the detailed description of a new, environment dedicated modular descriptor generator and optimizer algorithm. The framework gener-

ates descriptors through genetic algorithm cycles training and improving their efficiency for classification. The algorithm controls the image processing and random forest parameters and optimizes the descriptors' size through managing the active modules and values. The results show that although the mainstream descriptors have reliable performance in feature detection and invariance, a modular descriptor which is dedicated to a particular environment and optimized for size and image preparation can have better classification accuracy and similar invariance to the environmental changes.

As the results show the framework has good potential to optimize a list of candidates of a module bank, and image processing parameters into a descriptor with high recognition rate and invariance features. The algorithm also has the potential not only optimize the size of an existing descriptor (Lategahn *et al.*, 2013), but refine the image processing parameters and steps. Interesting to see that after three different genetic optimization with 200 generation each, the XYZ was the most effective color space for our purposes with 100 trees in the Random Forest training period which is identical to the original training parameter of Breiman (2001). Also there is a substantial change comparing the patch with to EDD(Takacs, Rivas-Araiza *et al.*, 2015). Meantime they use 9×9 pixel size patch, in our case for the optimum size is 21×21 . This implies more information but also more computation time.

After the genetic optimization to a specific scene, the modular descriptor developed a higher class recognition power, and became more invariant to transformation changes. The modular descriptor's *overall recognition rate* is the best among the contrasted descriptors and gave the best performance—except in the "Doors and Windows" class—in the *average recognition rate*.

Based on the results a new, two cycle recognition function is proposed where the first class best-recognized class results would be removed from the second classification period. Also new modules would be designed to enhance the invariance to light condition changes.

BIBLIOGRAPHY

- Abdel-Hakim, A. and Farag, A. (2006). "CSIFT: A SIFT Descriptor with Color Invariant Characteristics". In: *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. Vol. 2, pp. 1978–1983.
- Adhikari, R. (2016). *Samsung's AR Vision Includes Smart Contact Lenses*. URL: <http://www.technewsworld.com/story/83354.html> (visited on 12/04/2016).
- Alahi, A., Ortiz, R. and Vandergheynst, P. (2012). "FREAK: Fast Retina Keypoint". In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 510–517.
- Alcantarilla, P. F., Nuevo, J. and Bartoli, A. (2013). "Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces". In: *British Machine Vision Conf. (BMVC)*. Bristol, UK.
- Alcantarilla, P. F., Bartoli, A. and Davison, A. J. (2012). "KAZE Features". In: *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI. ECCV'12*. Florence, Italy: Springer-Verlag, pp. 214–227.
- Andreasson, H. and Duckett, T. (2004). "Topological localization for mobile robots using omni-directional vision and local features". In: *In Proceedings of the 5th IFAC Symposium on Intelligent Autonomous Vehicles*.
- Arth, C., Klopschitz, M., Reitmayr, G. and Schmalstieg, D. (2011). "Real-time self-localization from panoramic images on mobile devices". In: *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pp. 37–46.
- Arth, C., Wagner, D., Klopschitz, M., Irschara, A. and Schmalstieg, D. (2009). "Wide area localization on mobile phones". In: *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pp. 73–82.
- Azuma, R., Bailiot, Y., Behringer, R., Feiner, S., Julier, S. and MacIntyre, B. (2001). "Recent Advances in Augmented Reality". In: *IEEE Comput. Graph. Appl.* 21.6, pp. 34–47.
- Bay, H., Ess, A., Tuytelaars, T. and Van Gool, L. (2008). "Speeded-Up Robust Features (SURF)". In: *Comput. Vis. Image Underst.* 110.3, pp. 346–359.
- Behnam, M. and Pourghassem, H. (2013). "Feature descriptor optimization in medical image retrieval based on Genetic Algorithm". In: *Biomedical Engineering (ICBME), 2013 20th Iranian Conference on*, pp. 280–285.

- Berg, A., Grabler, F. and Malik, J. (2007). "Parsing Images of Architectural Scenes". In: *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8.
- Billinghamurst, M. (2013). *Lecture notes on Augmented Reality*. URL: <http://www.slideshare.net/marknb00/2013-lecture3-ar-tracking>.
- Billinghamurst, M., Clark, A. and Lee, G. (2015). "A Survey of Augmented Reality". In: *Foundations and Trends in Human–Computer Interaction* 8.2-3, pp. 73–272.
- Bimber, O. and Raskar, R. (2003). "Alternative augmented reality approaches: Concepts, techniques, and applications". In: *Eurographics 2003 (Tutorial Notes)*.
- Bradski, G. (2000). "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools*.
- Breiman, L. (2001). "Random Forests". In: *Mach. Learn.* 45.1, pp. 5–32.
- Brooker, J. (2007). "THE POLYTECHNIC GHOST - Pepper's Ghost, Metempsychosis and the Magic Lantern at the Royal Polytechnic InstitutionB". In: *Early Popular Visual Culture* 5.2, pp. 189–206. eprint: <http://dx.doi.org/10.1080/17460650701433517>.
- Brust, C., Sickert, S., Simon, M., Rodner, E. and Denzler, J. (2015). "Convolutional Patch Networks with Spatial Prior for Road Detection and Urban Scene Understanding". In: *CoRR* abs/1502.06344.
- Calonder, M., Lepetit, V., Strecha, C. and Fua, P. (2010). "BRIEF: Binary Robust Independent Elementary Features". In: *Computer Vision – ECCV 2010*. Ed. by K. Daniilidis, P. Maragos and N. Paragios. Vol. 6314. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 778–792.
- Canny, J. (1986). "A Computational Approach to Edge Detection". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI*-8.6, pp. 679–698.
- Carlevaris-Bianco, N. and Eustice, R. M. (2014). "Learning visual feature descriptors for dynamic lighting conditions". In: *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 2769–2776.
- Carmigniani, J. and Furht, B. (2011). "Handbook of Augmented Reality". In: ed. by B. Furht. New York, NY: Springer New York. Chap. Augmented Reality: An Overview, pp. 3–46.
- Caruana, R. and Niculescu-Mizil, A. (2006). "An Empirical Comparison of Supervised Learning Algorithms". In: *Proceedings of the 23rd International Conference on Machine Learning. ICML '06*. Pittsburgh, Pennsylvania, USA: ACM, pp. 161–168.

- Castle, R. and Murray, D. (2009). "Object recognition and localization while tracking and mapping". In: *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pp. 179–180.
- Castle, R., Klein, G. and Murray, D. (2010). "Combining monoSLAM with object recognition for scene augmentation using a wearable camera". In: *Image and Vision Computing* 28.11, pp. 1548–1556.
- Castle, R. and Murray, D. (2011). "Keyframe-based recognition and localization during video-rate parallel tracking and mapping". In: *Image and Vision Computing* 29.8, pp. 524–532.
- Caudell, T. and Mizell, D. (1992). "Augmented reality: an application of heads-up display technology to manual manufacturing processes". In: *System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on*. Vol. ii, 659–669 vol.2.
- Chiang, J. S., Hsia, C. H., Peng, H. W., Lien, C. H. and Li, H. T. (2012). "Saturation adjustment method based on human vision with YCbCr color model characteristics and luminance changes". In: *Intelligent Signal Processing and Communications Systems (ISPACS), 2012 International Symposium on*, pp. 136–141.
- Chinthammit, W., Seibel, E. J. and Furness, T. A. (2003). "A Shared-Aperture Tracking Display for Augmented Reality". In: *Presence* 12.1, pp. 1–18.
- Chon, Y., Talipov, E. and Cha, H. (2012). "Autonomous Management of Everyday Places for a Personalized Location Provider". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.4, pp. 518–531.
- Chu, D. M. and Smeulders, A. W. M. (2010). "Color Invariant SURF in Discriminative Object Tracking". In: *ECCV Workshop on Color and Reflectance in Imaging and Computer Vision*.
- Cortes, C. and Vapnik, V. (1995). "Support-Vector Networks". In: *Machine Learning* 20.3, pp. 273–297.
- Csurka, G. and Perronnin, F. (2008). "A Simple High Performance Approach to Semantic Segmentation". In: *Proceedings of the British Machine Vision Conference*. BMVA Press, pp. 22.1–22.10.
- Csurka, G., Dance, C. R., Fan, L., Willamowski, J. and Bray, C. (2004). "Visual categorization with bags of keypoints". In: *In Workshop on Statistical Learning in Computer Vision, ECCV*, pp. 1–22.

- De Crescenzo, F., Fantini, M., Persiani, F., Di Stefano, L., Azzari, P. and Salti, S. (2011). "Augmented Reality for Aircraft Maintenance Training and Operations Support". In: *Computer Graphics and Applications, IEEE 31.1*, pp. 96–101.
- Delmerico, J., David, P. and Corso, J. (2011). "Building facade detection, segmentation, and parameter estimation for mobile robot localization and guidance". In: *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 1632–1639.
- Digi-Capital (2016). *Augmented/Virtual Reality revenue forecast revised to hit 120 billion USD by 2020*. URL: <http://www.digi-capital.com/news/2016/01/augmentedvirtual-reality-revenue-forecast-revised-to-hit-120-billion-by-2020/#.vw723xurkyj> (visited on 17/03/2016).
- Doretto, G. and Yao, Y. (2010). "Region moments: Fast invariant descriptors for detecting small image structures". In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 3019–3026.
- Engel, J., Schöps, T. and Cremers, D. (2014). "Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II". In: ed. by D. Fleet, T. Pajdla, B. Schiele and T. Tuytelaars. Cham: Springer International Publishing. Chap. LSD-SLAM: Large-Scale Direct Monocular SLAM, pp. 834–849.
- Fabbri, R., Costa, L. D. F., Torelli, J. C. and Bruno, O. M. (2008). "2D Euclidean Distance Transform Algorithms: A Comparative Survey". In: *ACM Comput. Surv.* 40.1, 2:1–2:44.
- Fanelli, G., Dantone, M. and Van Gool, L. (2013). "Real time 3D face alignment with Random Forests-based Active Appearance Models". In: pp. 1–8.
- Feiner, S., MacIntyre, B., Hollerer, T. and Webster, A. (1997). "A touring machine: prototyping 3D mobile augmented reality systems for exploring the urban environment". In: *Wearable Computers, 1997. Digest of Papers., First International Symposium on*, pp. 74–81.
- Fernández-Delgado, M., Cernadas, E., Barro, S. and Amorim, D. (2014). "Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?" In: *Journal of Machine Learning Research* 15, pp. 3133–3181. URL: <http://jmlr.org/papers/v15/delgado14a.html>.
- Ferrari, V., Tuytelaars, T. and Van Gool, L. (2006). "Simultaneous Object Recognition and Segmentation by Image Exploration". English. In: *Toward Category-Level Object Recognition*. Ed. by J. Ponce, M. Hebert, C. Schmid and A.

- Zisserman. Vol. 4170. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 145–169.
- Finlayson, G. D. and Hordley, S. D. (2001). “Color constancy at a pixel”. In: *J. Opt. Soc. Am. A* 18.2, pp. 253–264.
- Finlayson, G., Hordley, S., Schaefer, G. and Tian, G. Y. (2005). “Illuminant and device invariant colour using histogram equalisation”. In: *Pattern Recognition* 38.2, pp. 179–190.
- Fjeld, M. and Voegtli, B. (2002). “Augmented Chemistry: an interactive educational workbench”. In: *Mixed and Augmented Reality, 2002. ISMAR 2002. Proceedings. International Symposium on*, pp. 259–321.
- Frantis, P. (2012). “Pilot suit with integrated avionics”. In: *Digital Avionics Systems Conference (DASC), 2012 IEEE/AIAA 31st*, pages.
- Fröhlich, B., Rodner, E. and Denzler, J. (2010). “A Fast Approach for Pixelwise Labeling of Facade Images”. In: *Pattern Recognition (ICPR), 2010 20th International Conference on*, pp. 3029–3032.
- Fujiwara, Y., Okamoto, T. and Kondo, K. (2013). “SIFT feature reduction based on feature similarity of repeated patterns”. In: *Intelligent Signal Processing and Communications Systems (ISPACS), 2013 International Symposium on*, pp. 311–314.
- Funt, B. V. and Finlayson, G. D. (1995). “Color constant color indexing”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17.5, pp. 522–529.
- Garcia-Fidalgo, E. and Ortiz, A. (2015). “Vision-based topological mapping and localization methods: A survey”. In: *Robotics and Autonomous Systems* 64, pp. 1–20.
- Gauglitz, S., Höllerer, T. and Turk, M. (2011). “Evaluation of Interest Point Detectors and Feature Descriptors for Visual Tracking”. In: *International Journal of Computer Vision* 94.3, pp. 335–360.
- Geusebroek, J.-M., Van den Boomgaard, R., Smeulders, A. and Geerts, H. (2001). “Color invariance”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 23.12, pp. 1338–1350.
- Gevers, T. (2001). “Principles of Visual Information Retrieval”. In: ed. by M. S. Lew. London: Springer London. Chap. Color-Based Retrieval, pp. 11–49.
- Gevers, T. and Smeulders, A. W. (1999). “Color-based object recognition”. In: *Pattern Recognition* 32.3, pp. 453–464.

- Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. 1st. Artificial Intelligence. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.
- Gondane, R. and Devi, V. S. (2015). "Classification Using Probabilistic Random Forest". In: *Computational Intelligence, 2015 IEEE Symposium Series on*, pp. 174–179.
- Guan, T. and Wang, C. (2009). "Registration Based on Scene Recognition and Natural Features Tracking Techniques for Wide-Area Augmented Reality Systems". In: *Multimedia, IEEE Transactions on* 11.8, pp. 1393–1406.
- Guo, P., Wang, X. and Han, Y. (2010). "The enhanced genetic algorithms for the optimization design". In: *Biomedical Engineering and Informatics (BMEI), 2010 3rd International Conference on*. Vol. 7, pp. 2990–2994.
- Gupta, R. and Mittal, A. (2008). "SMD: A Locally Stable Monotonic Change Invariant Feature Descriptor". English. In: *Computer Vision – ECCV 2008*. Ed. by D. Forsyth, P. Torr and A. Zisserman. Vol. 5303. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 265–277.
- Healey, G. and Slater, D. (1994). "Global color constancy: recognition of objects by use of illumination-invariant properties of color distributions". In: *J. Opt. Soc. Am. A* 11.11, pp. 3003–3010.
- Hempel, J. (2016). *I Went Inside Magic Leap's Mysterious HQ. Here's What I Saw*. URL: <http://www.wired.com/2016/04/went-inside-magic-leaps-mysterious-hq-heres-saw/> (visited on 08/05/2016).
- Hickey, H. (2008). *Contact Lenses for Superhuman Vision*. URL: http://www.engr.washington.edu/facresearch/highlights/ee_contactlens.html (visited on 12/04/2016).
- Hincapie, M., Caponio, A., Rios, H. and Mendivil, E. (2011). "An introduction to Augmented Reality with applications in aeronautical maintenance". In: *Transparent Optical Networks (ICTON), 2011 13th International Conference on*, pp. 1–4.
- Hu, M.-K. (1962). "Visual pattern recognition by moment invariants". In: *Information Theory, IRE Transactions on* 8.2, pp. 179–187.
- Huang, W., Wei, Y., Xie, Y. and Jin, H. (2013). "Survey of local invariant feature description". In: *Chinese Automation Congress (CAC), 2013*, pp. 353–358.
- Huletski, A., Kartashov, D. and Krinkin, K. (2015). "Evaluation of the modern visual SLAM methods". In: *Artificial Intelligence and Natural Language and Inform-*

- ation Extraction, Social Media and Web Search FRUCT Conference (AINL-ISMW FRUCT), 2015*, pp. 19–25.
- Human Interface Technology Laboratory - University of Washington (2003). *AR-Toolkit Documentation*. URL: <http://www.hitl.washington.edu/artoolkit/documentation/userarwork.htm> (visited on 24/10/2015).
- Julier, S., Baillet, Y., Lanzagorta, M., Brown, D. and Rosenblum, L. (2000). “BARS: Battlefield Augmented Reality System”. In: *In NATO Symposium on Information Processing Techniques for Military Systems*, pp. 9–11.
- Junaio.com (2015). *Developer Documentation junaio*. URL: <https://my.metaio.com/dev/junaio/documentation/index.html> (visited on 08/05/2016).
- Kang, B., Jeon, C., Han, D. K. and Ko, H. (2011). “Adaptive height-modified histogram equalization and chroma correction in {YCbCr} color space for fast backlight image compensation”. In: *Image and Vision Computing 29.8*, pp. 557–568.
- Kato, H. and Billinghurst, M. (1999). “Marker tracking and HMD calibration for a video-based augmented reality conferencing system”. In: *Augmented Reality, 1999. (IWAR '99) Proceedings. 2nd IEEE and ACM International Workshop on*, pp. 85–94.
- Ke, Y. and Sukthankar, R. (2004). “PCA-SIFT: a more distinctive representation for local image descriptors”. In: *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. Vol. 2, pages.
- Klein, G. and Murray, D. (2007). “Parallel Tracking and Mapping for Small AR Workspaces”. In: *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pp. 225–234.
- Klein, G. and Murray, D. (2009). “Parallel Tracking and Mapping on a camera phone”. In: *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pp. 83–86.
- Krevelen, D. W. F. van and Poelman, R. (2010). “A Survey of Augmented Reality Technologies, Applications and Limitations”. In: *The International Journal of Virtual Reality 9.2*, pp. 1–20.
- Krewell, K. (2016). *Oculus Has Plans For Augmented Reality*. URL: <http://www.forbes.com/sites/tiriasresearch/2016/04/01/oculus-has-plans-for-augmented-reality/#7a15eee52a8a> (visited on 09/05/2016).
- Krig, S. (2014). “Interest Point Detector and Feature Descriptor Survey”. English. In: *Computer Vision Metrics*. Apress, pp. 217–282.

- Kurz, D. and Benhimane, S. (2011). "Inertial sensor-aligned visual feature descriptors". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 161–166.
- Kurz, D. and Benhimane, S. (2012). "Augmented Reality: Handheld Augmented Reality involving gravity measurements". In: *Computers & Graphics* 36.7, pp. 866–883.
- Lategahn, H., Beck, J., Kitt, B. and Stiller, C. (2013). "How to learn an illumination robust image feature for place recognition". In: *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pp. 285–291.
- Lenz, R., Tran, L. V. and Meer, P. (1999). "Moment based normalization of color images". In: *Multimedia Signal Processing, 1999 IEEE 3rd Workshop on*, pp. 103–108.
- Leutenegger, S., Chli, M. and Siegwart, R. (2011). "BRISK: Binary Robust invariant scalable keypoints". In: *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2548–2555.
- Li, J. and Allinson, N. M. (2008). "A comprehensive review of current local features for computer vision". In: *Neurocomputing* 71.10–12. *Neurocomputing for Vision Research Advances in Blind Signal Processing*, pp. 1771–1787.
- Li, Y., Chen, Y., Lu, R., Ma, D. and Li, Q. (2012). "A novel marker system in augmented reality". In: *Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference on*, pp. 1413–1417.
- Lincoln, P., Blate, A., Singh, M., Whitted, T., State, A., Lastra, A. and Fuchs, H. (2016). "From Motion to Photons in 80 Microseconds: Towards Minimal Latency for Virtual and Augmented Reality". In: *IEEE Transactions on Visualization and Computer Graphics* 22.4, pp. 1367–1376.
- Lingley, A. R., Ali, M., Liao, Y., Mirjalili, R., Klonner, M. *et al.* (2011). "A single-pixel wireless contact lens display". In: *Journal of Micromechanics and Microengineering* 21.12, p. 125014.
- Lovato, L. (2015). *Rideon, la maschera da sci con la realt*. URL: http://www.ilturista.info/blog/11139-rideon_la_maschera_da_sci_con_la_realta_aumentata._sciare_giocando_e_non_solo/#.vz0icfkrliu (visited on 18/05/2016).
- Lowe, D. G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". In: *Int. J. Comput. Vision* 60.2, pp. 91–110.

- Lowe, D. (1999). "Object recognition from local scale-invariant features". In: *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. Vol. 2, 1150–1157 vol.2.
- Lowry, S., Wyeth, G. and Milford, M. (2014). "Unsupervised online learning of condition-invariant images for place recognition". In: *Australasian Conference on Robotics and Automation (ACRA2014)*. University of Melbourne, Melbourne, Australia: Australian Robotics & Automation Association ARAA.
- Macedo, M., Lopes Apolinario Junior, A., Souza, A. and Giraldo, G. (2014). "A Semi-automatic Markerless Augmented Reality Approach for On-Patient Volumetric Medical Data Visualization". In: *Virtual and Augmented Reality (SVR), 2014 XVI Symposium on*, pp. 63–70.
- Maimone, A., Yang, X., Dierk, N., State, A., Dou, M. and Fuchs, H. (2013). "General-purpose telepresence with head-worn optical see-through displays and projector-based lighting". In: *Virtual Reality (VR), 2013 IEEE*, pp. 23–26.
- Majewski, S. R. (2015). *Lecture Notes on Telescope Optics II: Aberrations, Diffraction Effects and Image Quality*. URL: <http://www.faculty.virginia.edu/ASTR5110/lectures/optics2/optics2.html> (visited on 25/05/2016).
- Microsoft.com (2016). *Microsoft HoloLens*. URL: <https://www.microsoft.com/microsoft-hololens/en-us/development-edition> (visited on 09/05/2016).
- Mikolajczyk, K. and Schmid, C. (2005). "A performance evaluation of local descriptors". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 27.10, pp. 1615–1630.
- Miksik, O. and Mikolajczyk, K. (2012). "Evaluation of local detectors and descriptors for fast feature matching". In: *Pattern Recognition (ICPR), 2012 21st International Conference on*, pp. 2681–2684. URL: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6460718.
- Milford, M. J. and Wyeth, G. F. (2012). "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights". In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 1643–1649.
- Milgram, P., Takemura, H., Utsumi, A. and Kishino, F. (1994). "Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum". In: *Telemanipulator and Telepresence Technologies*. International Society for Optics and Photonics, pp. 282–292.
- Mindru, F., Tuytelaars, T., Gool, L. V. and Moons, T. (2004). "Moment invariants for recognition under changing viewpoint and illumination". In: *Computer Vision*

- and Image Understanding* 94.1–3. Special Issue: Colour for Image Indexing and Retrieval, pp. 3–27.
- Miyashita, T., Meier, P., Tachikawa, T., Orlic, S., Eble, T., Scholz, V., Gapel, A., Gerl, O., Arnaudov, S. and Lieberknecht, S. (2008). “An Augmented Reality museum guide”. In: *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*, pp. 103–106.
- Mortensen, E. N., Deng, H. and Shapiro, L. (2005). “A SIFT Descriptor with Global Context”. In: *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05) - Volume 1 - Volume 01*. CVPR ’05. Washington, DC, USA: IEEE Computer Society, pp. 184–190.
- Mur-Artal, R., Montiel, J. and Tardos, J. (2015). “ORB-SLAM: A Versatile and Accurate Monocular SLAM System”. In: *Robotics, IEEE Transactions on* 31.5, pp. 1147–1163.
- Muselet, D. and Funt, B. (2013). “Color Invariants for Object Recognition”. English. In: *Advanced Color Image Processing and Analysis*. Ed. by C. Fernandez-Maloigne. Springer New York, pp. 327–376.
- Neubert, P., Sünderhauf, N. and Protzel, P. (2015). “Superpixel-based appearance change prediction for long-term navigation across seasons”. In: *Robotics and Autonomous Systems* 69. Selected papers from 6th European Conference on Mobile Robots, pp. 15–27.
- Park, J., Park, K., Baeg, S. and Baeg, M. (2008). “pi-SIFT: A photometric and Scale Invariant Feature Transform”. In: *19th International Conference on Pattern Recognition (ICPR 2008), December 8-11, 2008, Tampa, Florida, USA*, pp. 1–4.
- Park, J., Lee, D. and Park, J. (2012). “Digital map based pose improvement for outdoor Augmented Reality”. In: *Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. ISMAR ’12. Washington, DC, USA: IEEE Computer Society, pp. 309–310.
- Porzi, L., Ricci, E., Ciarfuglia, T. and Zanin, M. (2012). “Visual-inertial tracking on Android for Augmented Reality applications”. In: *Environmental Energy and Structural Monitoring Systems (EESMS), 2012 IEEE Workshop on*, pp. 35–41.
- Prince, S. J. D. (2012). *Computer Vision: Models, Learning, and Inference*. 1st. New York, NY, USA: Cambridge University Press. URL: <http://www.computervisionmodels.com/>.

- Quelhas, P. and Odobez, J.-M. (2006). "Natural Scene Image Modeling Using Color and Texture Visterms". English. In: *Image and Video Retrieval*. Ed. by H. Sundaram, M. Naphade, J. Smith and Y. Rui. Vol. 4071. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 411–421.
- El-Rabbany, A. (2002). *Introduction to GPS: the Global Positioning System*. Artech House.
- Radkowski, R. and Oliver, J. (2013). "Natural Feature Tracking Augmented Reality for On-Site Assembly Assistance Systems". English. In: *Virtual, Augmented and Mixed Reality. Systems and Applications*. Ed. by R. Shumaker. Vol. 8022. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 281–290.
- Raja, R., Md Mansoor Roomi, S. and Dharmalakshmi, D. (2013). "Outdoor scene classification using invariant features". In: *Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), 2013 Fourth National Conference on*, pp. 1–4.
- Rekimoto, J. (1996). "Augmenetd Reality using the 2D matrix code". In: *Interactive Systems and Software IV. Kindaikagaku-sha*, pp. 199–208.
- Rekimoto, J. and Nagao, K. (1995). "The World Through the Computer: Computer Augmented Interaction with Real World Environments". In: *Proceedings of the 8th Annual ACM Symposium on User Interface and Software Technology*. UIST '95. Pittsburgh, Pennsylvania, USA: ACM, pp. 29–36.
- Roberto, R., Lima, J. P. and Teichrieb, V. (2016). "Tracking for mobile devices: A systematic mapping study". In: *Computers & Graphics* 56, pp. 20–30.
- Rosten, E. and Drummond, T. (2005). "Fusing points and lines for high performance tracking." In: *IEEE International Conference on Computer Vision*. Vol. 2, pp. 1508–1511.
- Rosten, E. and Drummond, T. (2006). "Machine learning for high-speed corner detection". In: *European Conference on Computer Vision*. Vol. 1, pp. 430–443.
- Rublee, E., Rabaud, V., Konolige, K. and Bradski, G. (2011). "ORB: An efficient alternative to SIFT or SURF". In: *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2564–2571.
- Russell, B. C., Torralba, A., Murphy, K. P. and Freeman, W. T. (2008). "LabelMe: A Database and Web-Based Tool for Image Annotation". In: *Int. J. Comput. Vision* 77.1-3, pp. 157–173.
- Salhi, A. I., Kardouchi, M. and Belacel, N. (2012). "Fast and efficient face recognition system using random forest and histograms of oriented gradients". In:

- Biometrics Special Interest Group (BIOSIG), 2012 BIOSIG - Proceedings of the International Conference of the*, pp. 1–11. URL: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6313557>.
- Sarfraz, M. S. and Hellwich, O. (2009). “Computer Vision and Computer Graphics. Theory and Applications: International Conference, VISIGRAPP 2008, Funchal-Madeira, Portugal, January 22-25, 2008. Revised Selected Papers”. In: ed. by A. Ranchordas, H. J. Araújo, J. M. Pereira and J. Braz. Berlin, Heidelberg: Springer Berlin Heidelberg. Chap. On Head Pose Estimation in Face Recognition, pp. 162–175.
- Sarfraz, M., Mehmood-ul-Hassan and Iqbal, M. (2009). “Object Recognition Using Fourier Descriptors and Genetic Algorithm”. In: *Soft Computing and Pattern Recognition, 2009. SOCPAR '09. International Conference of*, pp. 318–323.
- Scherrer, C., Pilet, J., Fua, P. and Lepetit, V. (2008). “The haunted book”. In: *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*, pp. 163–164.
- Shih, P. and Liu, C. (2005). “Comparative Assesment of Content-Based Face Image Retrieval in Different Color Spaces”. In: *International Journal of Pattern Recognition & Artificial Intelligence* 19.7, pp. 873–893. URL: <http://search.ebscohost.com.etechnicryt.idm.oclc.org/login.aspx?direct=true&db=a9h&AN=18867254&lang=es&site=eds-live>.
- Shimazaki, K. and Nagao, T. (2013). “Scene classification using color and structure-based features”. In: *Computational Intelligence Applications (IWCI/A), 2013 IEEE Sixth International Workshop on*, pp. 211–216.
- Sivanandam, S. and Deepa, S. (2008). “Terminologies and Operators of GA”. In: *Introduction to Genetic Algorithms*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 39–81. URL: http://dx.doi.org/10.1007/978-3-540-73190-0_3.
- Stanescu, L., Dan Burdescu, D. and Brezovan, M. (2009). “Biomedical Data and Applications”. In: ed. by A. S. Sidhu and T. S. Dillon. Berlin, Heidelberg: Springer Berlin Heidelberg. Chap. Multimedia Medical Databases, pp. 71–141.
- Strecha, C., Bronstein, A., Bronstein, M. and Fua, P. (2012). “LDAHash: Improved Matching with Smaller Descriptors”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 34.1, pp. 66–78.

- Su, D., Wu, J., Cui, Z., Sheng, V. S. and Gong, S. (2013). “CGCI-SIFT: A More Efficient and Compact Representation of Local Descriptor”. In: *Measurement Science Review* 13, pp. 132–141.
- Suk, T. and Flusser, J. (2004). “Graph method for generating affine moment invariants”. In: *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*. Vol. 2, 192–195 Vol.2.
- Suk, T. and Flusser, J. (2009). “Affine Moment Invariants of Color Images”. English. In: *Computer Analysis of Images and Patterns*. Ed. by X. Jiang and N. Petkov. Vol. 5702. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 334–341.
- Sutherland, I. E. (1968). “A head-mounted three dimensional display”. In: *Proceedings of the December 9-11, 1968, fall joint computer conference, part I. AFIPS '68 (Fall, part I)*. ACM, pp. 757–764.
- Takács, A. (2013). “Façade labelling for better AR registration”. MA thesis. University College London.
- Takacs, A., Rivas-Araiza, E. A. and Pedraza-Ortega, J. C. (2015). “Scene Dedicated Feature Descriptor with Random Forest Training for Better Augmented Reality Registration”. In: *Research in Computing Science* 102, pp. 51–61. URL: http://www.rcs.cic.ipn.mx/rcs/2015_102/Scene%20Dedicated%20Feature%20Descriptor%20with%20Random%20Forest%20Training%20for%20Better%20Augmented%20Reality.pdf.
- Takacs, A., Toledano-Ayala, M., Pedraza-Ortega, J. C. and Rivas-Araiza, E. A. (2016). “Dedicated feature descriptor for outdoor augmented reality detection”. In: *Pattern Analysis and Applications*, pp. 1–12. URL: <http://dx.doi.org/10.1007/s10044-016-0581-8>.
- Takacs, G., Chandrasekhar, V., Tsai, S., Chen, D., Grzeszczuk, R. and Girod, B. (2013). “Fast Computation of Rotation-Invariant Image Features by an Approximate Radial Gradient Transform”. In: *Image Processing, IEEE Transactions on* 22.8, pp. 2970–2982.
- Tang, F., Lim, S. H., Chang, N. and Tao, H. (2009). “A novel feature descriptor invariant to complex brightness changes”. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 2631–2638.
- Teboul, O., Simon, L., Koutsourakis, P. and Paragios, N. (2010). “Segmentation of building facades using procedural shape priors”. In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 3105–3112.

- Ting, K. M. (2010). "Confusion Matrix". In: *Encyclopedia of Machine Learning*. Ed. by C. Sammut and G. I. Webb. Boston, MA: Springer US, pp. 209–209. URL: http://dx.doi.org/10.1007/978-0-387-30164-8_157.
- Tola, E., Lepetit, V. and Fua, P. (2010). "DAISY: An Efficient Dense Descriptor Applied to Wide-Baseline Stereo". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 32.5, pp. 815–830.
- Trujillo, L., Legrand, P., Olague, G. and Pérez, C. (2010). "Optimization of the HöLder Image Descriptor Using a Genetic Algorithm". In: *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation. GECCO '10*. Portland, Oregon, USA: ACM, pp. 1147–1154.
- Trujillo, L., Olague, G., Legrand, P. and Lutton, E. (2007). "Regularity based descriptor computed from local image oscillations". In: *Opt. Express* 15.10, pp. 6140–6145.
- Trzcinski, T., Christoudias, M. and Lepetit, V. (2015). "Learning Image Descriptors with Boosting". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 37.3, pp. 597–610.
- Trzcinski, T., Christoudias, M., Fua, P. and Lepetit, V. (2013). "Boosting Binary Key-point Descriptors". In: *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition. CVPR '13*. Washington, DC, USA: IEEE Computer Society, pp. 2874–2881.
- Trzcinski, T. and Lepetit, V. (2012). "Efficient Discriminative Projections for Compact Binary Descriptors". English. In: *Computer Vision – ECCV 2012*. Ed. by A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato and C. Schmid. Vol. 7572. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 228–242.
- Tuytelaars, T. and Mikolajczyk, K. (2008). "Local Invariant Feature Detectors: A Survey". In: *Found. Trends. Comput. Graph. Vis.* 3.3, pp. 177–280.
- Tuzel, O., Porikli, F. and Meer, P. (2006). "Region Covariance: A Fast Descriptor for Detection and Classification". In: *Proceedings of the 9th European Conference on Computer Vision - Volume Part II. ECCV'06*. Graz, Austria: Springer-Verlag, pp. 589–600.
- Valgren, C. and Lilienthal, A. J. (2010). "SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments". In: *Robotics and Autonomous Systems* 58.2, pp. 149–156.
- Van de Sande, K. E. A., Gevers, T. and Snoek, C. G. M. (2010). "Evaluating Color Descriptors for Object and Scene Recognition". In: vol. 32. 9, pp. 1582–1596.

- Van Gool, L., Zeng, G., Van den Borre, F. and Müller, P. (2007). "Towards Mass-produced Building Models". In: *Photogrammetric Image Analysis*. Ed. by U. Stilla, H. Mayer, F. Rottensteiner, C. Heipke and S. Hinz. Institute of Photogrammetry and Cartography, Technische Universitaet Muenchen, pp. 209–220.
- Velez, D. (2011). *Business in Two Worlds*. URL: <http://www.bayforce.com/2011/02/business-in-two-worlds/> (visited on 18/05/2016).
- Ventura, J., Arth, C., Reitmayr, G. and Schmalstieg, D. (2014). "Global Localization from Monocular SLAM on a Mobile Phone". In: *IEEE Transactions on Visualization and Computer Graphics* 20.4, pp. 531–539.
- Ventura, J. and Hollerer, T. (2011). "Outdoor mobile localization from panoramic imagery". In: *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pp. 247–248.
- Ventura, J. and Hollerer, T. (2012). "Wide-area scene mapping for mobile visual tracking". In: *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pp. 3–12.
- Verbelen, T., Simoons, P., De Turck, F. and Dhoedt, B. (2012). "A component-based approach towards mobile distributed and collaborative PTAM". In: *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pp. 329–330.
- Verdie, Y., Moo Yi, K., Fua, P. and Lepetit, V. (2014). "TILDE: A Temporally Invariant Learned DETector". In: *ArXiv e-prints*. arXiv: 1411.4568 [cs.CV].
- Vink, J. P. and Haan, G. de (2015). "Comparison of machine learning techniques for target detection". In: *Artificial Intelligence Review* 43.1, pp. 125–139.
- Vlahakis, V., Ioannidis, N., Karigiannis, J., Tstros, M., Gounaris, M., Stricker, D., Gleue, T., Daehne, P. and Almeida, L. (2002). "Archeoguide: an augmented reality guide for archaeological sites". In: *Computer Graphics and Applications, IEEE* 22.5, pp. 52–60.
- Vuforia.com (2016). *Augmented Reality -(Vuforia)*. URL: <https://developer.vuforia.com/> (visited on 08/05/2016).
- Wagner, D., Mulloni, A., Langlotz, T. and Schmalstieg, D. (2010). "Real-time panoramic mapping and tracking on mobile phones". In: *Virtual Reality Conference (VR), 2010 IEEE*, pp. 211–218.
- Wahyono and Jo, K. H. (2014). "A comparative study of classification methods for traffic signs recognition". In: *Industrial Technology (ICIT), 2014 IEEE International Conference on*, pp. 614–619.

- Wakabayashi, D. (2015). *Apple Buys German Augmented-Reality Firm Metaio*. URL: <http://blogs.wsj.com/digits/2015/05/28/apple-buys-german-augmented-reality-firm-metaio/> (visited on 08/05/2016).
- Wall, M. (1996). *GAlib: A C++ Library of Genetic Algorithm Components*. Massachusetts Institute of Technology.
- Wang, J. and Tan, Y. (2011). "Efficient Euclidean distance transform using perpendicular bisector segmentation". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1625–1632.
- Wang, Z., Fan, B. and Wu, F. (2011). "Local Intensity Order Pattern for feature description". In: *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 603–610.
- Weijer, J. van de, Gevers, T. and Geusebroek, J. M. (2005). "Edge and corner detection by photometric quasi-invariants". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.4, pp. 625–630.
- Weng, D., Cheng, D., Wang, Y. and Liu, Y. (2012). "Display systems and registration methods for augmented reality applications". In: *Optik - International Journal for Light and Electron Optics* 123.9, pp. 769–774.
- Williamson, J. and Murray-Smith, R. (2010). "The Engineering of Mixed Reality Systems". In: ed. by E. Dubois, P. Gray and L. Nigay. London: Springer London. Chap. Multimodal Excitatory Interfaces with Automatic Content Classification, pp. 233–250. URL: http://dx.doi.org/10.1007/978-1-84882-733-2_12.
- Wu, J., Cui, Z., Sheng, V. S., Zhao, P., Su, D. and Gong, S. (2013). "A Comparative Study of SIFT and its Variants". In: *Measurement Science Review* 13(3), pp. 122–131.
- Wu, Y., Choubassi, M. E. and Kozintsev, I. (2011). "Augmenting 3D urban environment using mobile devices". In: *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pp. 241–242.
- Yang, X. and Cheng, K.-T. (2012). "LDB: An ultra-fast feature for scalable Augmented Reality on mobile devices". In: *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pp. 49–57.
- Yu, G. and Morel, J.-M. (2009). "A fully affine invariant image comparison method". In: *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pp. 1597–1600.
- Zandbergen, P. A. (2009). "Accuracy of iPhone Locations: A Comparison of Assisted GPS, WiFi and Cellular Positioning". In: *Transactions in GIS* 13, pp. 5–25.

APPENDIX

APPENDIX

CONACyT PRODUCT - Published Article

Dedicated feature descriptor for outdoor augmented reality detection

*Dedicated feature descriptor for outdoor
augmented reality detection*

**Andras Takacs, Manuel Toledano-Ayala,
Jesus Carlos Pedraza-Ortega & Edgar
A. Rivas-Araiza**

Pattern Analysis and Applications

ISSN 1433-7541

Pattern Anal Applic

DOI 10.1007/s10044-016-0581-8



Your article is protected by copyright and all rights are held exclusively by Springer-Verlag London. This e-offprint is for personal use only and shall not be self-archived in electronic repositories. If you wish to self-archive your article, please use the accepted manuscript version for posting on your own website. You may further deposit the accepted manuscript version in any repository, provided it is only made publicly available 12 months after official publication or later and provided acknowledgement is given to the original source of publication and a link is inserted to the published article on Springer's website. The link must be accompanied by the following text: "The final publication is available at link.springer.com".

Dedicated feature descriptor for outdoor augmented reality detection

Andras Takacs¹ · Manuel Toledano-Ayala¹ · Jesus Carlos Pedraza-Ortega¹ · Edgar A. Rivas-Araiza¹

Received: 26 December 2015 / Accepted: 8 September 2016
© Springer-Verlag London 2016

Abstract Stable augmented reality applications consist of an accurate registration supported by a robust tracking module. In outdoor locations, the changing environmental and light conditions compromise this tracking. Reliable descriptors under unsettled conditions are essential for this process. The most used descriptors have this distinctive capacity, but computers and mobile devices process them in a long time frame. This paper investigates a new lightweight environment dedicated descriptor (EDD) trained with a machine-learning algorithm. The descriptor analyzes the scene characteristics with elements that can be computed fast and that have distinctive information about the selected area. The complete descriptor is used for semantic feature extraction with the aid of a trained random forest classifier. The descriptor is compared with the most popular descriptors—with respect to speed, accuracy, and invariance to illumination changes, scale, affine transformation, and rotation—and the results show that it is faster and in most cases equally reliable .

Keywords Augmented reality · Image processing · Descriptor · Random forest learning algorithm · Machine learning · Computer vision

1 Introduction

The use of augmented reality (AR) applications has been increasing in the past years. Despite the continuous improvement, AR applications have lots of flaws in outdoor environments. The rapidly changing light and environmental circumstances and the mobile devices' limited storage and processing capacity destabilize this software. To address this issue, we created a lightweight and robust application that can handle these factors in outdoor environments. An AR system using an EDD and semantical segmentation could master this problem. An EDD enhances the computational efficiency, and the environment segmentation reduces storage data. These applications overcome internet dependencies for calculation and large stored datasets.

The outdoor AR applications face two main issues: changing light conditions and large computational data. Illumination change lowers the precision, and the continuous calculation overloads the device's memory. The used floating-point grayscale descriptors [12, 27] are computationally expensive and do not take into account color information. The segmentation techniques aim to decimate the stored data, but the used color descriptors with machine-learning techniques are creating a large amount of data.

There have been various approaches to reduce computational time and data and to make keypoint detection and description invariant to outdoor changes. Descriptors trained with machine-learning algorithms [6, 28, 31, 32]

✉ Andras Takacs
andras.takacs.m@gmail.com

Manuel Toledano-Ayala
toledano@uaq.mx

Jesus Carlos Pedraza-Ortega
caryoko@yahoo.com

Edgar A. Rivas-Araiza
erivas@uaq.mx

¹ Facultad de Ingeniería, Universidad Autónoma de Querétaro, Cerro de las Campanas s/n, Las Campanas, 76010 Santiago de Querétaro, Querétaro, Mexico

brought promising results in terms of discrimination power. These studies use existing descriptors or image filtering techniques to extract low-level discriminative information in all types of scenery. Other works use scene-specific trained filters [37], but they focus on the keypoint detection only.

The previously mentioned descriptors are robust in just only one aspect: They are fast to calculate or have good segmentation power or invariant to structural and light condition changes. Our aim is to show that an environment specialized feature descriptor can fully satisfy multiple evaluation areas at once, can produce similar results in terms of performance, and can be as efficient as the most popular feature descriptors. Also, we aim to show that a descriptor can be created for a specific scenery, where it operates with high efficiency. The base concept is to create lightweight descriptors with simple, computationally cheap elements that are specialized to the corresponding environment (buildings, dense, and sparse areas) after training and empowering their segmentation accuracy with machine-learning techniques. The results show that although the descriptor made of ordinary elements, if they are carefully chosen for the environment, the concatenated form can be as effective as a common descriptor. A further goal is to create a new optimized modular descriptor—robust in repeatability and invariant to light changes—where the system can automatically detect the scenery and decide the composition of the descriptor with the objective to reduce the computed and later stored data on mobile devices.

This paper presents the results of a comprehensive performance evaluation of a specialized feature descriptor in terms of computational efficiency, retrieval performance, and invariance. It will start with the presentation of the random forest classifier, which will be followed by the state-of-the-art descriptors that were used for the evaluation of EDD. The second part of the article begins with an overview of the experimental setup and finishes with the results and discussion.

1.1 Related work

Tracking is how AR systems specify their positions in 3D environments, a crucial support for the stable registration. For outdoor conditions (changing light settings, markerless, and sparse areas), different techniques were developed over the years. At the beginning of the last decade, mainly magnetic sensors (gyroscope, GPS, accelerometer, or compass) specified the devices position [1]. Along the same decade developments, in computer and mobile central processing units (CPU) made possible to do tracking using the device's camera feed. The camera see-through AR applications started to exploit the visual information through the video stream using image-processing

functions. Using local image features became the most successful approach to analyze and describe such information. Many surveys [20, 25, 29, 30] were published during the past decade to compare and categorize the local image descriptor techniques. All these works conclude a descriptor, one that works flawlessly in all environments and is invariant to all types of image transformations, has not been developed. In the research literature, SIFT [27] is the first robust and—according to Google Scholar, with more than 30,000 citations—the most referenced descriptor, which is seen as a milestone in the development of local invariant feature descriptors. SIFT is still the most reliable with high repeatability rate—the percentage of points simultaneously present in two images—despite the fact that many versions have been developed to address one of its weaknesses: LLA-SIFT [22] for size reduction and face recognition, Opponent SIFT [36] and yCQ-SIFT [40] for color image description.

Recently, the focus went beyond the repeatability rate focusing on the calculation time, the dimensionality of the feature descriptors, and the light-change invariance for outdoor use. Different approaches were used to achieve dimensionality reduction, GLOH [29] with principal component analysis, with similar pattern removal [16] and DIRD [24] with genetic algorithm optimization. For illumination invariance, OSID [34] applies ordinal and spatial intensity histogram, LIOP [39] considers the intensity order among all the sample points, and LOIND [14] uses depth information.

The parallel tracking and mapping (PTAM) [23] application—to substitute the magnetic sensors with only camera tracking—localizes itself with the simultaneous localization and mapping (SLAM) technique. The application loses the tracking easily in the outdoor environment, quickly overloading the memory with data. To reduce memory load, different approaches were used, cloud-based, machine-learning techniques [18] to lift the calculation from the mobile device or the reduced size I-BRIEF descriptor [26].

The automatic facade-recognition techniques were a response to the growing need of mass 3D reconstruction and modeling in city planning—geo-applications like Google Earth or Microsoft Virtual Earth—and in 3D GPS navigation systems to reduce the reconstruction time and the storage size of the data [17]. Researchers developed various techniques for this recognition and segmentation over the years. The “bag of keypoint” method [10]—a general image categorization technique—uses pixel-level image processing assigned to high-level image clusters called “vocabularies” for training a multiclass classifier. This technique reached better results combined with color descriptors and the random forest learning method [3, 15]. The latest investigations achieved real-time feature extraction, using discrete-time cellular neural networks and

an accelerated KAZE algorithm [21], or multiscale convolutional network [13].

2 Proposed method

2.1 Random forest

The random forest [5] is a high-performance discriminative classifier, handling a large set of features without having difficulties, due to the curse of dimensionality [15]. This supervised learning method—which is able to learn more than one class at a time—constructs an ensemble of recursively created random binary decision trees (Fig. 1) during the training period. The algorithm learns the optimal threshold (t) at each node to split the training data by using the same number of random features in each tree. The classification process returns the class (κ) probability (p) of a given feature vector (v_i) via averaging the final votes (p_τ) of each tree with the total number of trees (T) (Eq. 1).

$$p(y_i = \kappa | v_i) = \frac{1}{T} \sum_{\tau=1}^T p_\tau(y_i = \kappa | v_i) \quad (1)$$

The process aggregates randomness at two stages during the building of the forest in the training session. With this step, random forest solves the overfitting problem, which in other algorithms like the random decision trees causes major issues. First, the bootstrap aggregation creates random subsets of data, from which the trees are learned. Second, the split functions use only an m random value of all features during the creation of the decision trees [5]. The random forest calculates the out-of-bag (OOB) error rate for each tree to guarantee unbiased results. This means that one-third of the cases are left out and not used in the construction of each tree, making unnecessary an extra test set for cross-validation, where bias still exists with an unknown extent [5].

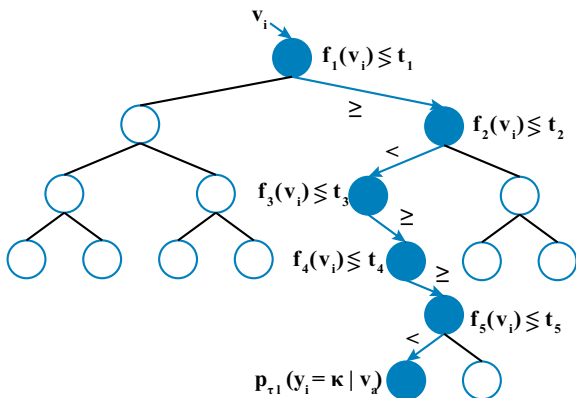


Fig. 1 Binary decision tree

2.2 The descriptors

For image detection or classification, the image properties need to be described in a unique manner. The local features group in different ways—by the used computation method (e.g., gradient- or histogram-based), by the used component data (e.g., binary or floating point-type), or by the working environment (e.g., inside or outside)—but they have common properties. The use of local descriptors follows the aforementioned principal steps: feature detection, descriptor calculation, and feature matching. The features ought to have unique, reproducible, and invariant properties and an efficient calculation technique. For this reason, the descriptor has to be repeatable, invariant to scale, rotation, affine transformation, and light condition changes and be compact in size to provide optimum storage and calculations to cite a few. In other words, the resulting descriptor over a keypoint of the same object from two different images has to be identical, even if the environment has changed.

2.2.1 State-of-the-art descriptors

Scale-invariant feature transform (SIFT) SIFT has been the most used and referenced descriptor in the past 10 years. It contains 128 elements calculated from a set of orientation histograms on 4×4 pixel over a 16×16 region around the keypoint. The magnitudes are weighted by a Gaussian function afterward [27].

Speeded up robust features (SURF) It is inspired by SIFT, but it has a faster calculation performance and smaller dimensionality [2]. The technique uses a 64-dimension vector calculated from a square region centered on the keypoint. The region is split into 4×4 subregion. They calculate a Gaussian-weighted, horizontal-and-vertical Haar wavelet—which is summed over the subregions—and the absolute values of the same responses.

Opponent SIFT This is the best performing SIFT descriptor on colored images [36]. It is calculated in the same way as the classical SIFT descriptor, in each of the highly decorrelated opponent color channels (Eq. 2), where the color space calculated from the basic *RGB* values contain one intensity (O_3) and two chromaticity channels (O_1 and O_2). This leads to a 384-dimension vector.

$$\begin{pmatrix} O_1 \\ O_2 \\ O_3 \end{pmatrix} = \begin{pmatrix} \frac{R - G}{\sqrt{2}} \\ \frac{R + G - 2B}{\sqrt{6}} \\ \frac{R + G + B}{\sqrt{3}} \end{pmatrix} \quad (2)$$

Opponent SURF This descriptor retrieves color information and was first applied for tracking using foreground/

background discrimination [9]. The extraction method uses the original SURF calculation steps in the three opponent color channels (Eq. 2), producing a 192-dimensional vector. According to the authors, the invariance and discriminative power of the tracker increased using this descriptor.

2.2.2 Environment dedicated descriptor (EDD)

The descriptor was shaped thinking on the characteristics of the environment. The used databases contain images from urban scenery, combining organized shapes, parallel lines, sparse and dense areas. Therefore, the descriptor was aimed to capture this information in short period of time. EDD is a 113-dimension vector computed from a 9×9 patch selected around each keypoint. The size was chosen to be big enough to pick up edges and low-level changes on the image, but also was sized to reduce the forest training time and size. The following elements make up the descriptor:

(1) **Position**—2 values—2D image coordinates of the patch centers to separate points which are on the top (sky), on the bottom (street), and in the middle (wall).

Before the calculation of the color values and the central moments, we normalize the patch (P) pixel values with the local maxima.

$$P_{\text{average}} = \frac{\sum_0^x \sum_0^y (I^{\text{max}} - I(x, y))}{P_{\text{size}}} \quad (3)$$

$$I_{\text{norm}}(x, y) = \frac{I^{\text{max}} - I(x, y)}{P_{\text{average}}}$$

(2) **Patch mean**—6 values—The mean of the red, green, blue (from the RGB channels), and saturation values (from the HSV channels) over the patch, to exploit the color changes on the images. Sine and cosine of the mean of the discontinuous hue values were calculated over the patch. In consequence of the channels, angular design, the color values differ significantly at the opposite ends (0° and 359°), while these colors have neighboring RGB values. With the sine and cosine pair, we equalize them (ex. $\sin(0^\circ) = \sin(360^\circ) = 0$ and similarly $\cos(0^\circ) = \cos(360^\circ) = 1$).

(3) **The third-order central moments**—24 values—generated to obtain distinctive shape description of the patch. The central moments can be easily calculated and are independent [29]. The third-order central moments— $\mu_{03}, \mu_{30}, \mu_{21}, \mu_{12}$ —of the RGB and HSV channels over the patch measure the skew and the symmetry of the point spread around the mean of the patch. At first the zeroth- (M_{00}) and first-order (M_{10}, M_{01}) raw image moments are calculated by

$$M_{ij} = \sum_x \sum_y x^i y^j I_{\text{norm}}(x, y) \quad (4)$$

then the two components of the patch centroid:

$$\bar{x} = \frac{M_{10}}{M_{00}}, \quad \bar{y} = \frac{M_{01}}{M_{00}} \quad (5)$$

The third-order central moments—where the sum of the $p, q \in \mathbb{N}$ subscript of μ_{pq} specifies the order of the moment—defined as:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I_{\text{norm}}(x, y) \quad (6)$$

The higher-order moments describe more fine variations in the shape, but they are more sensitive to noise and left out for that reason.

(4) **Distance transform**—81 values—Distance transform measures the distance between the pixel and the nearest detected canny edge [8] point. This feature was used in a variety of segmentation algorithms [11, 38]. The values of the distance transfer are growing as the points are farther away from the edge. They reach their maximum on flat areas, which is a good distinctive component in the descriptor, helping the forest to separate flat area patches from patches on areas with lots of transition (Fig. 2).

2.3 Datasets

We used two sets of images with different characteristics during the evaluation. Both datasets contain street sceneries with buildings, serving to evaluate the descriptors adaptabilities in a specific and in a general environment.

The Brighton images were specifically chosen key frames from video recordings made on two different occasions, with different light conditions on Queens Gardens Street in Brighton, UK. The database contains 52 images labeled with four classes. The classes are meaningful regions of house images (doors and windows, wall, roof, else). Ninety percent of the images were set for training, and the rest were destined for the testing period.

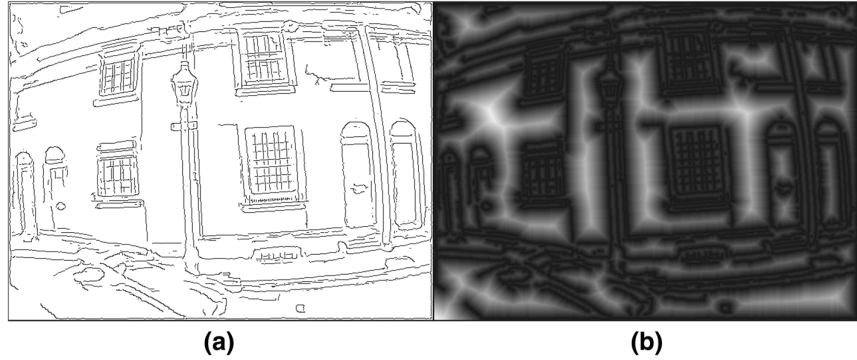
The LabelMeFacade [15] dataset was assembled from the LabelMe [33] database. It contains 945 labeled images with nine classes (building, car, door, pavement, road, sky, vegetation, window, else), including the most important semantical regions of an urban scene. We divided the dataset into 100-mixed-scene images for training and 845 for testing following [7] the description.

In both cases, the testing images were taken in different circumstances and in a different part of the environment as the training images, to avoid overfitting in the testing period.

2.4 Experiment

The experiment followed a uniform strategy for all investigated descriptors: The system extracted the descriptor vector using the same keypoint matrix for all the images;

Fig. 2 Edges and distances: **a** canny edges, **b** distance transform



the training method loaded the calculated descriptor vectors from all the training images to random forest; the classification method segmented the object features using the trained decision trees (Algorithm 1). During the experiment, we used the OpenCV [4] for the image processing, to compute the SIFT, Opponent SIFT, SURF, Opponent SURF descriptors, and for the random forest calculation on a PC with 15.6 GB RAM and Intel®Core™i7-4790 CPU with 3.60 GHz × 8.

Algorithm 1 EDD Creation pseudocode

```

1: procedure DESCRIPTOR CREATION AND TRAINING
2:   Keypoint Creation with a constant (9px) distance
3:   Descriptor Extraction from the 9×9 patch around each keypoint
4:   repeat
5:     Position
6:     Patch mean
7:     Central Moments
8:     Distance Transform
9:   until descriptor vector is calculated for all keypoint
10:  Random Forest Training
11: end procedure
    
```

The images were processed in order to remove the noise and undesired edges to gain the most characteristic information. For the purpose, a Gaussian blur filter with an 11×11 kernel and $\sigma = 30$ was used. From our experiment, that setup produced the least noisy and most prominent edges. The resulting images were saved for the descriptor extraction and to estimate the canny edges [8]. This served for computing the pixel distances from their closest edge pixel (Fig. 2). These blurred-color and pixel-distance

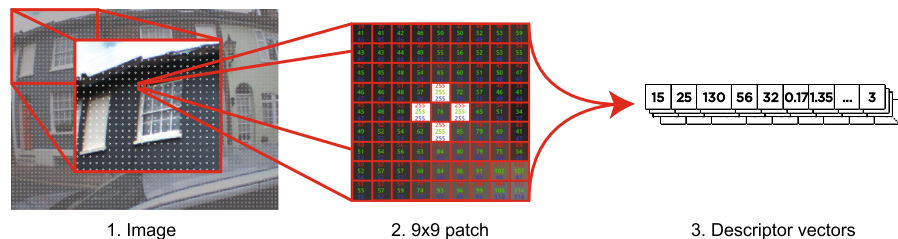
images were used for calculating the descriptor vector (Fig. 3). To obtain the most data keypoints were set across the image with a distance of nine pixels. This ensured that the patches cover all area of the image to retrieve all the necessary training information.

The random forest training operation followed the proposed framework by [5]. A group of 100 decision trees consisted of a forest with a depth maximized in 15. The incoming vectors reached the lowest point after at each of the 15 levels a decision node, with m number of randomly selected variables for a good split at each node. We first considered Brieman’s [5] setup values, as he stated that, “the strength of the forest remained constant after passing the value $m = 4$, in other words adding more inputs do not help.” Then we chose m equal to the square root of the feature dimensionality p ($m = \sqrt{p}$) [19]. We saved the forests for later use and measurement. To recognize the objects of interest, we used the same descriptor retrieval procedure on the test images and loaded the information to the trained decision trees. The resulting votes on the interest areas were applied for creating binary images. The bounding boxes were arranged and reprojected to the original images after removing outlier points (caused by the misclassification) with morphological operations.

2.5 Evaluation

The descriptor evaluation was divided into two operations: accuracy and speed analysis, with different training database sizes and characteristics during the test session; and

Fig. 3 Descriptor extraction



invariance evaluation to size, rotation, blur, light intensity changes, light intensity shifts, and light color changes in a separate function (Algorithm 2).

Algorithm 2 Descriptor Evaluation pseudocode

```

1: procedure ACCURACY EVALUATION
2:   repeat
3:     Descriptor Extraction for all keypoints
4:     Average descriptor extraction time measure
5:     Random Forest classification
6:     Feature Segmentation
7:     Overall and Average Recognition Rate calculation
8:     Average Segmentation Time computation
9:   until reached the last test image
10: procedure INVARIANCE EVALUATION
11:   repeat
12:     Evaluation setup for specific invariance
13:     repeat
14:       Image transformation
15:       Descriptor Extraction for all keypoints for the image pair
16:       Brute Force descriptor vector matching
17:       Result storing
18:     until reached the last test image
19:   until all invariance is being tested
20:   Result plotting by invariance case
21: end procedure

```

We generated confusion matrices from the random forests' test results to compare the precision of each descriptor. This formed the base to calculate the *overall recognition rate*—the average of all correctly classified patches on the test set—and the *average recognition rate*—the average per class recognition rate on the test images. The speed probe measured two different properties. The *average descriptor extraction time* shows the time of the descriptor retrieval from the whole image with an equal amount of keypoints. The *average segmentation time* demonstrates the duration of the forest evaluation and image segmentation.

A set of two identical images served for the invariance test. To compare the impact of the deformation, we kept

one of the pair as ground truth and transformed the other. During the rotation, resize (Eq. 7), and affine transformation assessments (Eq. 8), the same transformation matrix converts the keypoints and image pixels (Fig. 4) to place the reference points at the correct spot. Then we compare the extracted descriptor vectors with the brute-force matching function [4].

$$M = \begin{bmatrix} \alpha & \beta & (1 - \alpha) \cdot \text{center.x} - \beta \cdot \text{center.y} \\ -\beta & \alpha & \beta \cdot \text{center.x} + (1 - \alpha) \cdot \text{center.y} \end{bmatrix} \quad \text{where} \quad (7)$$

$$\alpha = \text{scale} \cdot \cos \text{angle}$$

$$\beta = \text{scale} \cdot \sin \text{angle}$$

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}_{2 \times 2} \quad B = \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix}_{2 \times 1} \quad (8)$$

$$T = A \cdot \begin{bmatrix} x \\ y \end{bmatrix} + B = \begin{bmatrix} a_{00}x + a_{01}y + b_{00} \\ a_{10}x + a_{11}y + b_{10} \end{bmatrix}$$

The large lighting condition variations—which often happen in real-world scenarios—can be modeled by a diagonal mapping. We used [36] photometric analysis and a diagonal model to rate the descriptors' robustness against light changes. The matrix maps the colors captured under an unknown light source u to their equivalent under the canonical illuminant c . *Light intensity change* alters the pixel values multiplying with a constant factor (Eq. 9).

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} \quad (9)$$

Light intensity shift amends all color channels with the same constant across the image (Eq. 10).

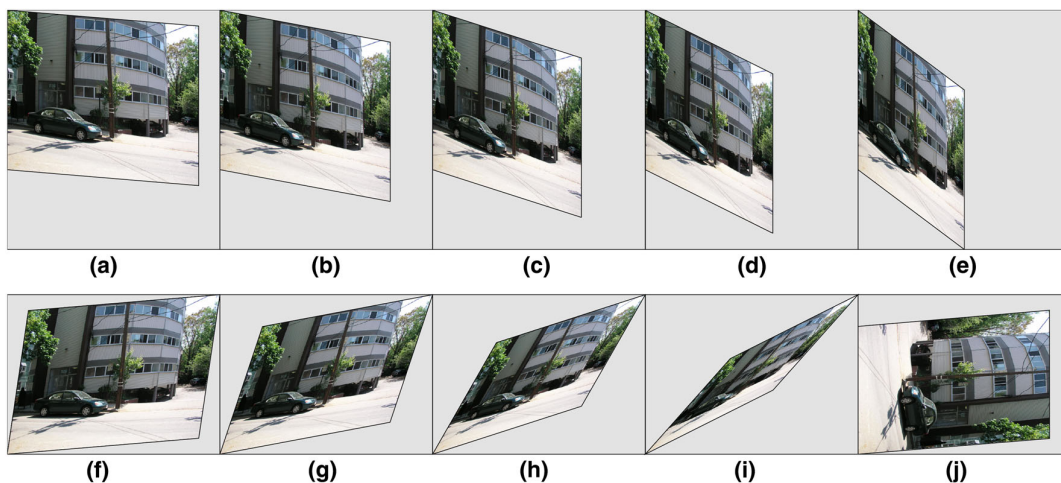


Fig. 4 Affine transformation cases in invariance evaluation: a–j Case 1–10

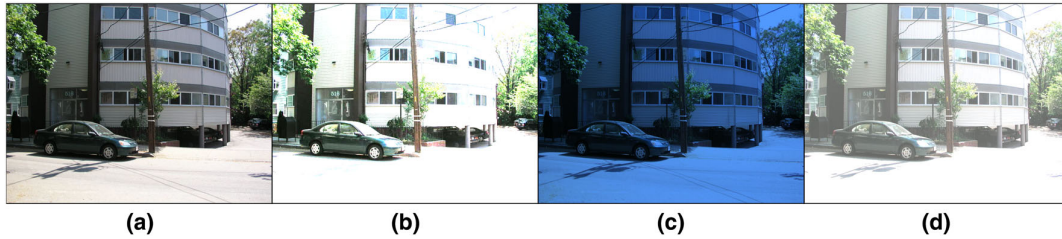


Fig. 5 Light changes: **a** original image, **b** intensity change ($\alpha = 3$), **c** light color change (3800 K), **d** color shift ($o_1 = 100$)

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix} \quad (10)$$

Light color change modifies the color channels independently (Eq. 11). This type of modification can model changes in the illuminant color and light scattering [36]. To model these changes, we used the RGB values corresponding to light color temperatures (Fig. 5).

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} \quad (11)$$

3 Results

We contrasted the EDD with the state-of-the-art descriptors' performance. Tables 1 and 2 show the accuracy evaluation results. Each table reports the outcome of classification—using the trained forest with parameter $m = \sqrt{p}$ —with the overall recognition rate in the first

column and the average recognition rate of each class in the rest of the table. These analyses show a similar pattern with all the tested descriptors. After the training to a specific scene, the descriptors gave better results in classification. Throughout the testing, the Opponent SIFT, and SIFT descriptors gave the most reliable performance, above 70 % with the specific and 58 % with the general environments. The Opponent SIFT descriptor was designed for a color environment, but in the evaluations it had a poorer performance than the grayscale SIFT descriptor. The EDD came out third overall, after the two SIFT descriptors. This shows that a trained environment-specific descriptor with carefully chosen modules can substitute effectively the state-of-the-art descriptors in classification tasks.

Throughout the Brighton scene (Table 1), the EDD was operating with high precision in most of the classes except in the “Roof” class. Sparse (low image gradient) areas like the “wall” were still producing good results, but gave its best performance classifying the “else” group. The classification results (Fig. 6a) show visually the outcome, where each color represents a class: the yellow circles the wall class, the red circles the window or door class, the

Table 1 Average true positives at the Brighton scene (%)

Descriptor	All	Average true positive by class			
		Else	Doors and windows	Roof	Wall
SIFT	74.67	82.58	61.03	70.16	79.15
Opponent SIFT	72.92	83.97	56.27	77.92	79.41
SURF	53.46	63.27	46.69	17.55	59.48
Opponent SURF	55.43	74.74	43.90	34.25	62.55
EDD	67.58	92.36	45.68	55.92	72.39

Table 2 Average true positives with 100 images (%)

Descriptor	All	Average true positive by class								
		Else	Building	Door	Window	Car	Pavement	Road	Sky	Vegetation
SIFT	67.27	0	65.68	0	0	29.66	21.02	88.74	73.24	33.09
Opp. SIFT	64.15	0	60.85	0	0	29.16	24.44	86.59	79.06	35.26
SURF	46.13	0	47.10	0	0	12.57	3.79	59.03	62.56	20.57
Opp. SURF	51.54	0	48.39	0	0	0	0	75.46	63.39	33.31
EDD	60.48	0	63.81	0	0	14.67	9.56	85.07	65.14	31.33

Fig. 6 EDD results: **a** keypoint classification result, **b** segmentation result



blue circles the roof class, and the green circles the else class. We can observe that despite the outliers and the misclassified object (Fig. 6b), the descriptor's overall performance is good.

The different scene characteristics of the LabalMeFacade dataset reduced the exactness of the descriptors (Table 2) in feature recognition (doors, window, car, pavement, vegetation, and else). However, the EDD

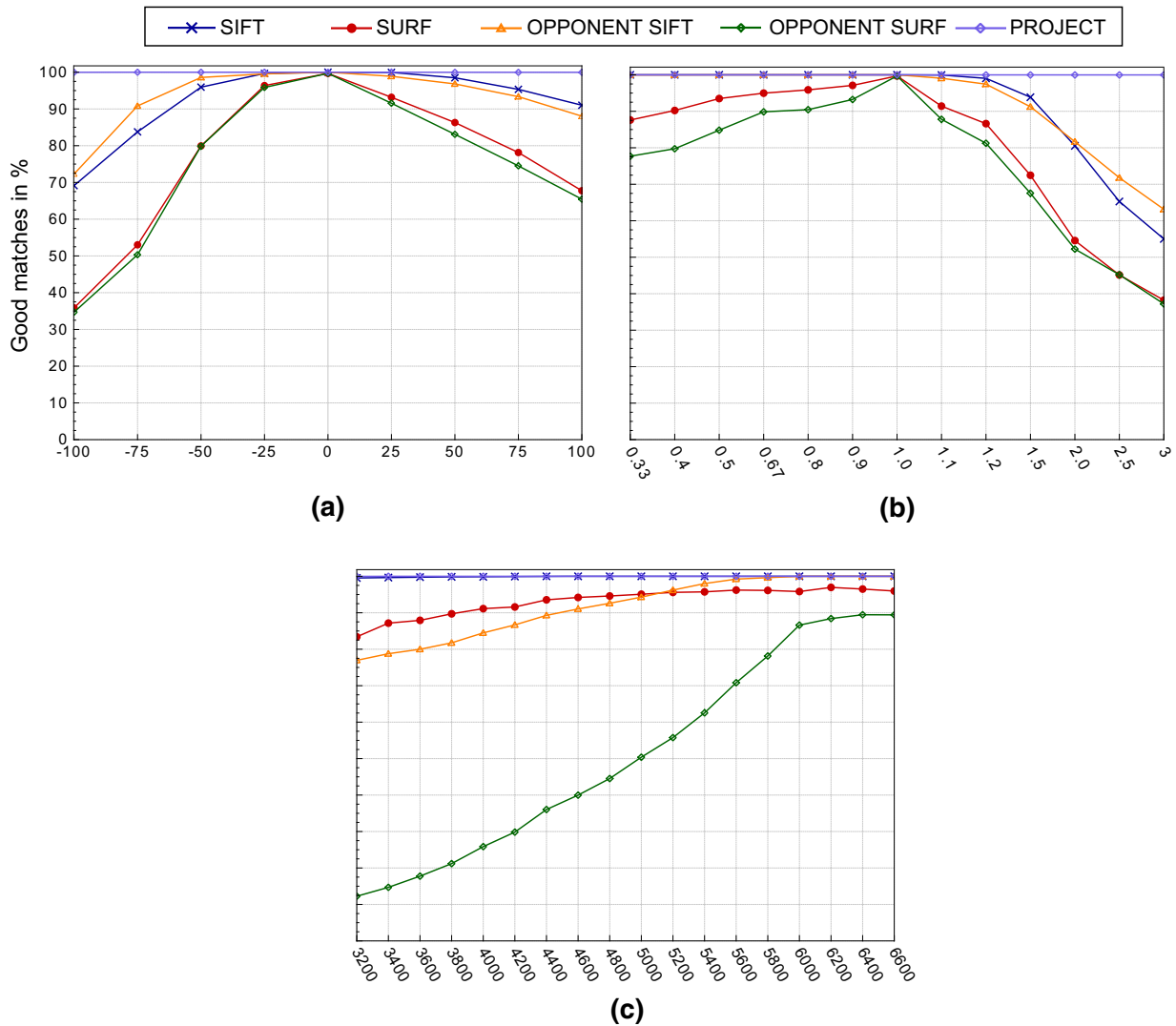


Fig. 7 Photometric analysis results: **a** intensity shift (with offset σ_1), **b** intensity scaling (with scalar a), **c** illumination color temperature (K)

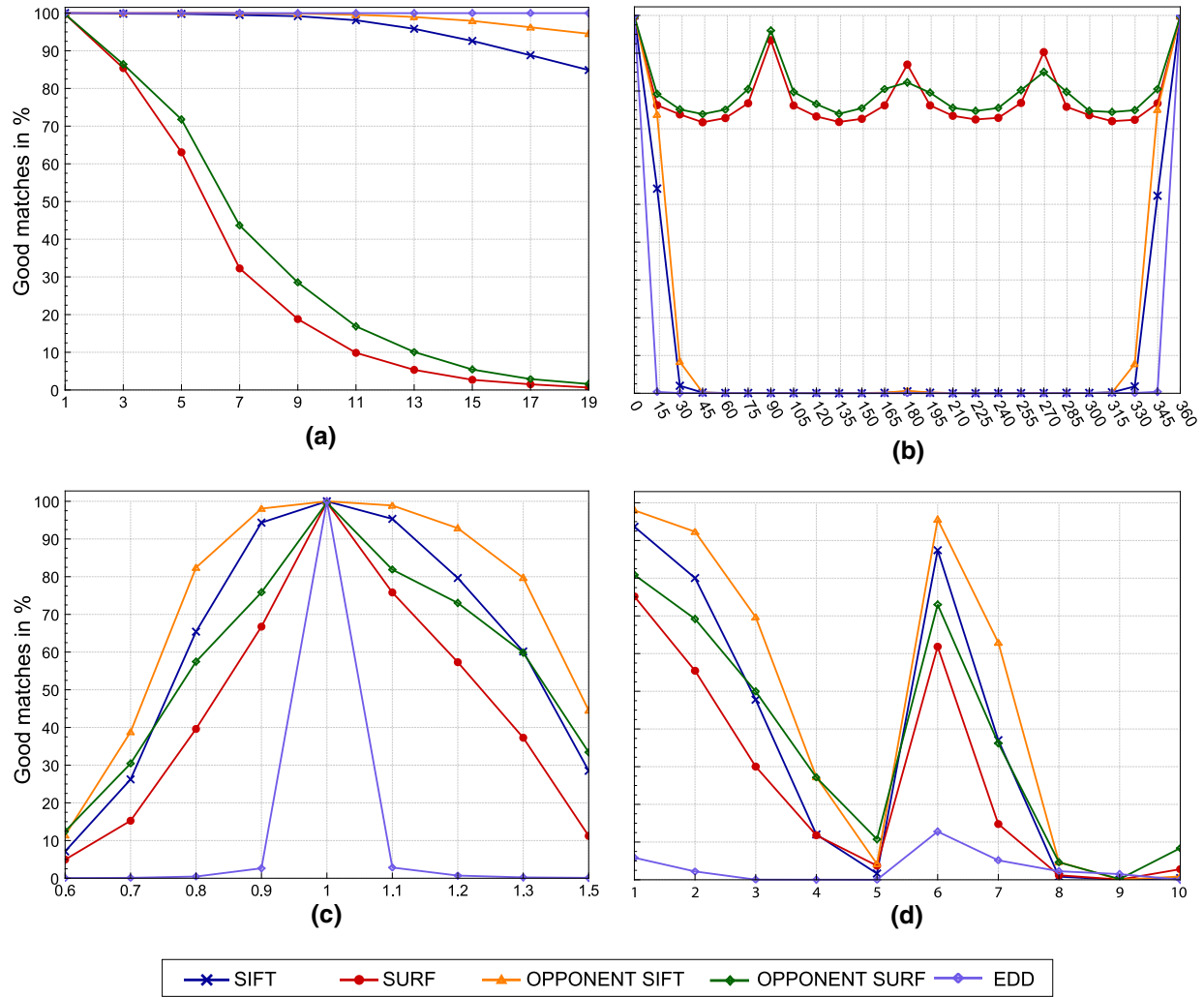


Fig. 8 Transformation invariance results: (a) Gaussian blur (kernel size), (b) image rotation (angle), (c) image resize (size), (d) affine transformation (cases)

produced high-performance results in detecting buildings, roads, and the sky, and we can clearly see that to capture the other classes correctly, other type of modules need to be incorporated to the descriptor. Using this dataset, we observed a significant improvement in the classification performance by changing the number of random values during the random forest training at the split points moving from $m = 4$ to $m = \sqrt{p}$. This could be because in this dataset we have nine segmentation categories, and the more feature chosen at the nodes resulted more accurate in categorizing the features (Figs. 7, 8).

The photometric analysis and the invariance test gave two different kinds of results for the EDD. During the *light intensity change*, *light intensity shift*, *light color change*, and *blur* test, it gave the most reliable performance. This is

due to the local normalization on the patch level and the blurring during the image preparation period. On the other hand, the descriptor still had a poor performance during the image transformation assessment.

The measured *average segmentation time* (Fig. 9a) and *average descriptor extraction time* (Fig. 9b) indicate that the EDD is by far the fastest in both cases. In the extraction time, we included the image-processing and the feature calculation times. We can observe in Fig. 9a how much time it takes for the computer to extract all the descriptor vectors from the whole image (average 3600 vector/image). The EDD is on average 26 times faster than the Opponent SIFT and needs 4.5 times less time than the SURF. However, the algorithm has been run on different computers, and the ratios between the descriptors' calculation time maintained invariant. The

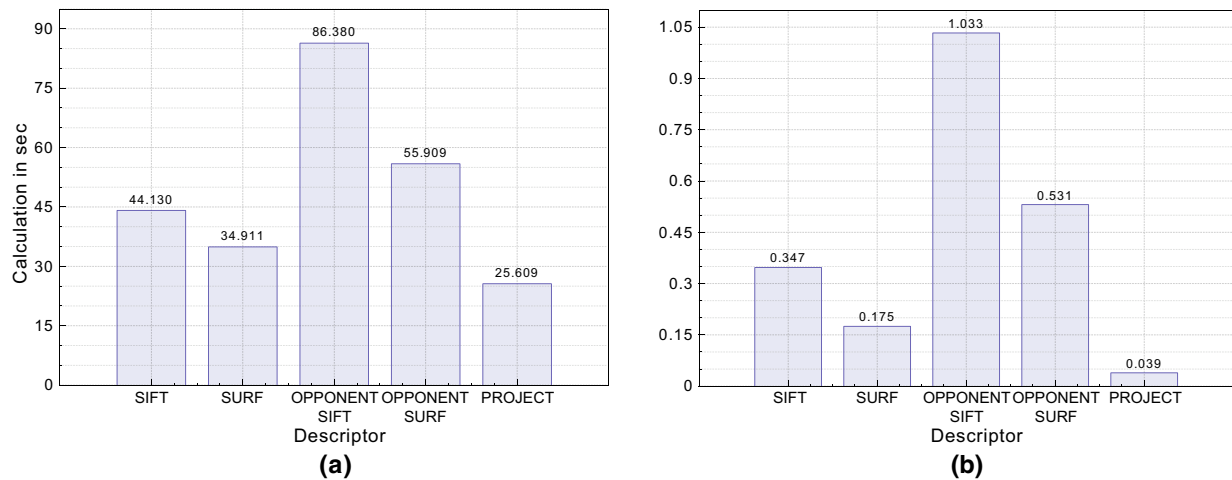


Fig. 9 Descriptor performance in time: **a** average object segmentation time, **b** average descriptor extraction time

results are due to the calculation efficient descriptor features. The average time the computer takes (Fig. 9a) to read and calculate the necessary data for the descriptor is based on the outcome of the classification, which segmented the window and door areas. The result data show while the most accurate SIFT descriptor needed 44 s on average, the Opponent SIFT descriptor for the same work needed two times as much effort in time. The Opponent SURF descriptor occupied 56 s for the work, which is two times slower than the EDD, and its performance in detection was inferior to this descriptor. These results correlate with the finding of [35], where the SIFT descriptor was more accurate in feature matching but took a considerably longer time frame. Figure 6 shows the final outcome of the segmentation algorithm, where we see the strength of window detection and the weakness of the descriptor vector in terms of distortion and rotation.

4 Conclusion

The main contribution of this work is a new approach to create robust descriptors satisfying multiple requirements at once. We proved that a compound of a few existing variables—that is cheap to calculate and chosen for the environment—is capable of the same local feature description as a common descriptor. The distance transform brings valuable information about sparse and dense areas, color information holds extra information, and the central moments can be calculated easily while being independent. The results support the concept that although these calculations are weak alone to describe a local feature but in concatenated form strengthen each other. The

proposed descriptor is fast to calculate, robust in segmentation, and invariant to light condition changes. In this paper, feature extraction with a new environment dedicated descriptor was studied with respect to speed, accuracy, and invariance. EDD examined the scene and calculated quickly the local features with distinctive information. The complete descriptor was used for semantic feature extraction, with the aid of a trained random forest classifier. The results show that although the most popular descriptors have reliable performance in feature detection, a descriptor which is dedicated to a specific environment—that is, a street environment with buildings—can have similar accuracy but in a shorter time period. The EDD also responds well to the light condition variations. This projects a new path to investigate a trained dynamic descriptor that can adjust characteristics of the retrieved information according to the environment. Based on the results, the EDD should be stabilized for transformation invariance and another version should be created for different environmental characteristics.

Acknowledgments I would like to thank the Consejo Nacional de Ciencia y Tecnología through the project number 340519 without whom this paper could not have been completed. Also, I would like to thank the Universidad Autónoma de Quértaro for its facilities and support.

References

1. Azuma R, Baillet Y, Behringer R, Feiner S, Julier S, MacIntyre B (2001) Recent advances in augmented reality. *IEEE Comput Graph Appl* 21(6):34–47. doi:[10.1109/38.963459](https://doi.org/10.1109/38.963459)
2. Bay H, Ess A, Tuytelaars T, Van Gool L (2008) Speeded-up robust features (surf). *Comput Vis Image Underst* 110(3):346–359. doi:[10.1016/j.cviu.2007.09.014](https://doi.org/10.1016/j.cviu.2007.09.014)

3. Berg A, Grabler F, Malik J (2007) Parsing images of architectural scenes. In: IEEE 11th International Conference on Computer vision. ICCV 2007. pp 1–8. doi:[10.1109/ICCV.2007.4409091](https://doi.org/10.1109/ICCV.2007.4409091)
4. Bradski G (2000) The opencv library. Dr. Dobb's J Softw Tools
5. Breiman L (2001) Random forests. *Mach Learn* 45(1):5–32. doi:[10.1023/A:1010933404324](https://doi.org/10.1023/A:1010933404324)
6. Brown M, Hua G, Winder S (2011) Discriminative learning of local image descriptors. *IEEE Trans Pattern Anal Mach Intell* 33(1):43–57. doi:[10.1109/TPAMI.2010.54](https://doi.org/10.1109/TPAMI.2010.54)
7. Brust C, Sickert S, Simon M, Rodner E, Denzler J (2015) Convolutional patch networks with spatial prior for road detection and urban scene understanding. *CoRR*. abs/1502.06344. <http://arxiv.org/abs/1502.06344>
8. Canny J (1986) A computational approach to edge detection. *IEEE Trans Pattern Anal Mach Intell PAMI*–8(6):679–698. doi:[10.1109/TPAMI.1986.4767851](https://doi.org/10.1109/TPAMI.1986.4767851)
9. Chu DM, Smeulders AWM (2010) Color invariant surf in discriminative object tracking. In: ECCV Workshop on Color and Reflectance in Imaging and Computer Vision. <http://www.science.uva.nl/research/publications/2010/ChuCRICV2010>
10. Csurka G, Dance CR, Fan L, Willamowski J, Bray C (2004) Visual categorization with bags of keypoints. In: In Workshop on Statistical Learning in Computer Vision, ECCV, pp 1–22. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.72.604>
11. Fabbri R, Costa LDF, Torelli JC, Bruno OM (2008) 2D euclidean distance transform algorithms: a comparative survey. *ACM Comput Surv* 40(1):2:1–2:44. doi:[10.1145/1322432.1322434](https://doi.org/10.1145/1322432.1322434)
12. Fan B, Wu F, Hu Z (2011) Aggregating gradient distributions into intensity orders: A novel local image descriptor. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2377–2384. doi:[10.1109/CVPR.2011.5995385](https://doi.org/10.1109/CVPR.2011.5995385)
13. Farabet C, Couprie C, Najman L, LeCun Y (2013) Learning hierarchical features for scene labeling. *IEEE Trans Pattern Anal Mach Intell* 35(8):1915–1929. doi:[10.1109/TPAMI.2012.231](https://doi.org/10.1109/TPAMI.2012.231)
14. Feng G, Liu Y, Liao Y (2015) Loind: An illumination and scale invariant rgb-d descriptor. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp 1893–1898. doi:[10.1109/ICRA.2015.7139445](https://doi.org/10.1109/ICRA.2015.7139445)
15. Fröhlich B, Rodner E, Denzler J (2010) A fast approach for pixelwise labeling of facade images. In: 2010 20th International Conference on Pattern Recognition (ICPR), pp 3029–3032. doi:[10.1109/ICPR.2010.742](https://doi.org/10.1109/ICPR.2010.742)
16. Fujiwara Y, Okamoto T, Kondo K (2013) Sift feature reduction based on feature similarity of repeated patterns. In: 2013 International Symposium on Intelligent Signal Processing and Communications Systems (ISPACS), pp 311–314. doi:[10.1109/ISPACS.2013.6704567](https://doi.org/10.1109/ISPACS.2013.6704567)
17. Gool LV, Zeng G, den Borre FV, Müller P (2007) Towards mass-produced building models. In: Stilla U, Mayer H, Rottensteiner F, Heipke C, Hinz S (eds) Photogrammetric image analysis. Institute of Photogrammetry and Cartography, Technische Universität München, Munich, pp 209–220
18. Guan T, Wang C (2009) Registration based on scene recognition and natural features tracking techniques for wide-area augmented reality systems. *IEEE Trans Multimedia* 11(8):1393–1406. doi:[10.1109/TMM.2009.2032684](https://doi.org/10.1109/TMM.2009.2032684)
19. Hastie T, Tibshirani R, Friedman J (2009) Random forests. In: *The elements of statistical learning: data mining, inference, and prediction*, vol 2019, pp 587–604. doi:[10.1007/978-0-387-84858-7](https://doi.org/10.1007/978-0-387-84858-7)
20. Huang W, Wei Y, Xie Y, Jin H (2013) Survey of local invariant feature description. *Chin Autom Congr (CAC)* 2013:353–358. doi:[10.1109/CAC.2013.6775758](https://doi.org/10.1109/CAC.2013.6775758)
21. Jiang G, Liu L, Zhu W, Yin S, Wei S (2015) A 181 gops akaze accelerator employing discrete-time cellular neural networks for real-time feature extraction. *Sensors* 15(9):22,509. doi:[10.3390/s150922509](https://doi.org/10.3390/s150922509). <http://www.mdpi.com/1424-8220/15/9/22509>
22. Jihua Y, Shuxia S, Yahui C (2015) A face recognition algorithm based on l1e-sift feature descriptors. In: 2015 10th International Conference on Computer Science Education (ICCSE), pp 729–734. doi:[10.1109/ICCSE.2015.7250341](https://doi.org/10.1109/ICCSE.2015.7250341)
23. Klein G, Murray D (2007) Parallel tracking and mapping for small ar workspaces. In: 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007. ISMAR 2007. pp 225–234. doi:[10.1109/ISMAR.2007.4538852](https://doi.org/10.1109/ISMAR.2007.4538852)
24. Lategahn H, Beck J, Kitt B, Stiller C (2013) How to learn an illumination robust image feature for place recognition. In: Intelligent Vehicles Symposium (IV), 2013 IEEE, pp 285–291. doi:[10.1109/IVS.2013.6629483](https://doi.org/10.1109/IVS.2013.6629483)
25. Li J, Allinson NM (2008) A comprehensive review of current local features for computer vision. *Neurocomputing* 71(10–12):1771–1787. doi:[10.1016/j.neucom.2007.11.032](https://doi.org/10.1016/j.neucom.2007.11.032). <http://www.sciencedirect.com/science/article/pii/S0925231208001124>. Neurocomputing for Vision Research Advances in Blind Signal Processing
26. Liu J, Liang X (2011) I-brief: A fast feature point descriptor with more robust features. In: 2011 Seventh International Conference on Signal-Image Technology and Internet-Based Systems (SITIS), pp 322–328. doi:[10.1109/SITIS.2011.11](https://doi.org/10.1109/SITIS.2011.11)
27. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vision* 60(2):91–110. doi:[10.1023/B:VISI.0000029664.99615.94](https://doi.org/10.1023/B:VISI.0000029664.99615.94)
28. McManus C, Upcroft B, Newmann P (2014) Scene signatures : localised and point-less features for localisation. In: Robotics: Science and Systems X. University of California, Berkeley, CA. <http://eprints.qut.edu.au/76158/>
29. Mikolajczyk K, Schmid C (2005) A performance evaluation of local descriptors. *IEEE Trans Pattern Anal Mach Intell* 27(10):1615–1630. doi:[10.1109/TPAMI.2005.188](https://doi.org/10.1109/TPAMI.2005.188)
30. Miksik O, Mikolajczyk K (2012) Evaluation of local detectors and descriptors for fast feature matching. In: 2012 21st International Conference on Pattern Recognition (ICPR), pp 2681–2684
31. Ozuysal M, Calonder M, Lepetit V, Fua P (2010) Fast keypoint recognition using random ferns. *IEEE Trans Pattern Anal Mach Intell* 32(3):448–461. doi:[10.1109/TPAMI.2009.23](https://doi.org/10.1109/TPAMI.2009.23)
32. Rigamonti R, Lepetit V, Gonzalez G, Treten E, Benmansour F, Brown M, Fua P (2014) On the relevance of sparsity for image classification. *Comput Vision Image Underst* 125:115–127. doi:[10.1016/j.cviu.2014.03.009](https://doi.org/10.1016/j.cviu.2014.03.009). <http://www.sciencedirect.com/science/article/pii/S1077314214000757>
33. Russell BC, Torralba A, Murphy KP, Freeman WT (2008) Labelme: a database and web-based tool for image annotation. *Int J Comput Vision* 77(1–3):157–173
34. Tang F, Lim SH, Chang N, Tao H (2009) A novel feature descriptor invariant to complex brightness changes. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pp. 2631–2638. doi:[10.1109/CVPR.2009.5206550](https://doi.org/10.1109/CVPR.2009.5206550)
35. Valgren C, Lilienthal AJ (2010) Sift, SURF and seasons: appearance-based long-term localization in outdoor environments. *Robot Auton Syst* 58(2):149–156. doi:[10.1016/j.robot.2009.09.010](https://doi.org/10.1016/j.robot.2009.09.010)
36. Van de Sande KEA, Gevers T, Snoek CGM (2010) Evaluating color descriptors for object and scene recognition. *IEEE Trans Pattern Anal Mach Intell*. doi:[10.1109/TPAMI.2009.154](https://doi.org/10.1109/TPAMI.2009.154)
37. Verdie Y, Moo Yi K, Fua P, Lepetit V (2014) Tilde: A temporally invariant learned detector. *ArXiv e-prints*. doi:[10.1109/CVPR.2015.7299165](https://doi.org/10.1109/CVPR.2015.7299165)
38. Wang J, Tan Y (2011) Efficient euclidean distance transform using perpendicular bisector segmentation. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1625–1632. doi:[10.1109/CVPR.2011.5995644](https://doi.org/10.1109/CVPR.2011.5995644)

39. Wang Z, Fan B, Wu F (2011) Local intensity order pattern for feature description. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp 603–610. doi:[10.1109/ICCV.2011.6126294](https://doi.org/10.1109/ICCV.2011.6126294)
40. Xiao P, Cai N, Tang B, Weng S, Wang H (2014) Efficient sift descriptor via color quantization. In: 2014 IEEE International Conference on Consumer Electronics - China, pp 1–3. doi:[10.1109/ICCE-China.2014.7029876](https://doi.org/10.1109/ICCE-China.2014.7029876)

CONACyT PRODUCT - Technological Transference

Ciudad de México, a 26 de Octubre de 2016

ESTIMADOS SEÑORES DEL SIN
CONACYT

P R E S E N T E

Por este conducto se hace constar que el C. Andras Takacs ha realizado la transferencia tecnológica de un “**Algoritmo organizador de descriptores con algoritmo genético**”, el cual se encuentra registrado ante el Instituto Nacional del Derecho de Autor (INDAUTOR) con número de registro **03-2016-091410443600-01**, el cual ha sido incorporado como un módulo de un sistema de visión artificial desarrollado dentro del marco de un proyecto denominado “Investigación y desarrollo de sistemas de captura por visión artificial con aplicaciones para sistemas hápticos, e-learning y videojuegos” realizado conjuntamente entre la empresa y la Universidad Autónoma de Querétaro.

Se extiende la presente para los fines que al interesado convengan a los veintiséis días del mes de octubre del año en curso, sin otro particular que agregar por el momento, quedo a sus órdenes y aprovecho la ocasión para enviarle un cordial saludo.

Atentamente

A handwritten signature in black ink, appearing to read "Alyed Tzompa Sosa".

Alyed Tzompa Sosa
Director General

CERTIFICADO

Registro Público del Derecho de Autor

Para los efectos de los artículos 13, 162, 163 fracción I, 164 fracción I, 168, 169, 209 fracción III y demás relativos de la Ley Federal del Derecho de Autor, se hace constar que la **OBRA** cuyas especificaciones aparecen a continuación, ha quedado inscrita en el Registro Público del Derecho de Autor, con los siguientes datos:

AUTOR: TAKACS ANDRAS
TITULO: OPTIMIZADOR DE DESCRIPTORES CON ALGORITMO GENETICO
RAMA: PROGRAMAS DE COMPUTACION
TITULAR: TAKACS ANDRAS

Con fundamento en lo establecido por el artículo 168 de la Ley Federal del Derecho de Autor, las inscripciones en el registro establecen la presunción de ser ciertos los hechos y actos que en ellas consten, salvo prueba en contrario. Toda inscripción deja a salvo los derechos de terceros. Si surge controversia, los efectos de la inscripción quedarán suspendidos en tanto se pronuncie resolución firme por autoridad competente.

Con fundamento en los artículos 2, 208, 209 fracción III y 211 de la Ley Federal del Derecho de Autor; artículos 64, 103 fracción IV y 104 del Reglamento de la Ley Federal del Derecho de Autor; artículos 1, 3 fracción I, 4, 8 fracción I y 9 del Reglamento Interior del Instituto Nacional del Derecho de Autor, se expide el presente certificado.

Número de Registro: 03-2016-091410443600-01

México D.F., a 14 de septiembre de 2016

EL DIRECTOR DEL REGISTRO PÚBLICO DEL DERECHO DE AUTOR

JESUS PARETS GOMEZ



SECRETARÍA DE CULTURA
INSTITUTO NACIONAL DEL
DERECHO DE AUTOR
DIRECCIÓN DEL REGISTRO
PÚBLICO DEL DERECHO DE AUTOR

