



**Universitat  
Autònoma  
de Barcelona**

**Geometric and Structural-based Symbol  
Spotting. Application to Focused Retrieval  
in Graphic Document Collections**

A dissertation submitted by **Marçal Rusiñol Sanabra** at Universitat Autònoma de Barcelona to fulfil the degree of **Doctor en Informàtica**.

Bellaterra, 2009

Director: **Dr. Josep Lladós Canet**  
Universitat Autònoma de Barcelona  
Dep. Ciències de la Computació & Centre de Visió per Computador



This document was typeset by the author using L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>.

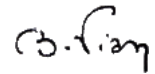
The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona.

This work is licensed under Creative Commons Attribution-Share Alike 3.0 Unported License © 2009 by Marçal Rusiñol Sanabra. You are free to copy, distribute and transmit the work as long as you attribute its author. If you alter, transform or build upon this work, you may distribute the resulting work only under the same, similar or compatible license.

ISBN: 978-84-936529-7-5

Printed by Ediciones Gráficas Rey, S.L.

“Les quelques pages de démonstration qui suivent  
tirent toute leur force du fait  
que l’histoire est entièrement vraie,  
puisque je l’ai imaginée d’un bout à l’autre.”



– *Boris Vian.*

“A theory that still doesn’t have any good  
counter-evidence is one worth pursuing.”

村上 春樹

– *Haruki Murakami.*

“Sand is overrated,  
it’s just tiny little rocks.”

–from *Eternal Sunshine  
of the Spotless Mind.*

## **Puedo empezar**

“Tengo ya preparadas las respuestas  
para las entrevistas periodísticas  
que me harán en la prensa, radio y tele.

Querrán saber qué opino y cómo soy.  
Me mostraré ingenioso y espontáneo.

Tengo ya preparadas unas listas  
de personalidades importantes  
e incluso redactados ya los textos,  
muy agudos, de las dedicatorias.

Tengo ya preparadas las metáforas  
que servirán como brillante ejemplo  
o síntesis que aclare lo que exponga.  
Saldrán como galaxias de las páginas.

Y tengo preparada mi postura  
al sentarme o de pie, tono de voz,  
expresión de los ojos y la boca.

Todo está preparado. Todo a punto.  
Puedo empezar, pues, a escribir mi libro.”

– *José María Fonollosa, Albert Pla, Eugenio.*

# Agraïments

Primer de tot vull agrair a en Juanjo Villanueva l'haver-me brindat l'oportunitat de poder entrar a formar part del CVC. Fa ja uns quants anys, ens va contractar a n'Eduard Vázquez i a mi com a becaris per picar codi. No va trigar massa en dipositar cada cop més confiança i en creure en nosaltres, per finalment animar-nos a emprendre l'aventura de fer un doctorat. És evident que sense ell, avui no estaria on soc ara, i per això li estaré eternament agraït. Voldria també donar les gràcies a en Felipe Lumbreras, que, per aquella mateixa època, em va dirigir el projecte de final de carrera i em va començar a introduir realment en el món de la visió.

Evidentment, tota aquesta feina no hagués estat possible sense el suport d'un director de tesi com en Josep Lladós. En ell també li he d'agrair la seva confiança i el seu recolzament així com l'haver dedicat part del seu (cada cop més escàs i preuat) temps en discutir, donar consells, orientar i dur a bon port la meva recerca. Moltes gràcies Josep per haver-me fet de director i haver-me aguantat durant aquests cinc anys.

Je tiens aussi à remercier à Karl Tombre et à Jean-Marc Ogier pour m'avoir accueilli dans ses groupes à Nancy et à La Rochelle pour pouvoir faire mes séjours, et aussi pour avoir accepté d'être membres du jury. Je voudrais aussi remercier l'équipe QGAR de Nancy: Bart Lamiroy, Gérald Masini, Antoine Tabbone, Laurent Wendling, Jan Rendek, Daniel Zuwala, Oanh Nguyen et Sabine Barrat pour avoir fait de ce stage une expérience merveilleuse. Merci spécialement à Philippe Dosch pour avoir orienté mon travail lors du stage. Merci à Rémy Mullot, Muriel Visani, Mickaël Coustaty, Stéphanie Guillas, Thomas Martin et Romain Raveaux pour votre support lors du stage à La Rochelle et en particulier à Karell Bertet pour m'avoir introduit au monde des tréillis.

Vull també agrair profundament els companys de docència que he tingut al llarg d'aquests anys. N'Anna Salvatella, na Carme Julià, en Javier Vázquez, en Pau Baiget, n'Iván Huerta i als responsables de les assignatures, en Xavier Binefa, en Fernando Vilariño i na Petia Radeva. Voldria donar especialment les gràcies a en Ricardo Toledo i a en David Aldavert per aquests quatre anys entranyables d'estructura de dades.

Arriba l'hora d'agrair a la gent del CVC, però sou masses i segur que em deixo molta gent. Vull donar les gràcies en general per l'ambient que s'hi respira i en particular començaré per agrair a na Montse Culleré i a n'Ana María García per donar-me sempre un cop de mà a enquadrar, enviar faxos i ajudar-me a realitzar d'altres tasques altament complicades. També agrair als diversos companys de cervesetes i

soparets com en Xavier Baró, en Poal, en Ricard Borràs, na Raquel Gómez, en Coen Antens, na Mireia Montpart i un llarg etcètera. Agrair també als companys que presenten tesi aviat com n'Aura Hernández per les penes que hem compartit aquesta última temporada. Evidentment, donar moltes gràcies als companys de despatx, que al cap i a la fi son els que m'han aguantat més hores: n'Ignasi Rius, en Dani Rowe, n'Eric Sommerlade, en Carles Fernández, en Ferran Diego, en Mohammad Rouhani, i algun més que em dec estar deixant. Especialment, donar les gràcies a n'Alicia Fornés i a en Joan Mas que no només han estat companys de despatx sinó que també son companys de grup. De ben segur que recordaré durant temps les tertúlies que hem tingut. Agrair també finalment la resta del grup d'anàlisi de documents, en particular a en Dimosthenis Karatzas, n'Ernest Valveny, n'Agnés Borràs i en Miquel Ferrer.

No poden faltar els companys de la carrera que continuo veient i amb els quals hem gaudit de tants dinars els divendres a la penya bètica. Gràcies a en Juan Ignacio Toledo, n'Eduard Vázquez, n'Enric Sánchez, na Silvana Vacchina i n'Adolfo Zarrias. I tampoc poden faltar els companys de les escoles franceses que aquests si que fa temps que em suporten, en Dibi Morad, en Miguel García i especialment n'Enric Farrerons qui m'ha donat un munt de plànols sense els quals no haguès pogut ni començar a treballar.

Finalment, agrair-te a tu Silvia la paciència que has tingut aquests últims mesos d'estrès. Vull sobretot agrair-te els ànims que m'has donat sempre, com m'has alegrat cada dia i que em facis feliç dia rera dia. Per acabar, donar les gràcies a la Maria i l'Andreu, que sempre m'han animat a fer el que m'agradava, i sempre m'han donat el seu suport. Aquest cop us heu pogut lliurar de corretxir les faltas d'ortografia.

# Resum

Habitualment, els sistemes de reconeixement de patrons consisteixen en dos grans apartats. D'una banda, l'adquisició de les dades i, de l'altre, la classificació d'aquestes dades dins d'una certa categoria. Per tal de reconèixer a quina categoria pertany un cert element, un conjunt de patrons models han d'haver estat proporcionats per anticipat. Per tal d'entrenar el classificador i que pugui oferir una classificació de patrons robusta, es necessita una etapa d'aprenentatge que es pot dur a terme fora de línia. Dins del camp de reconeixement de patrons, estem interessats en el reconeixement d'elements gràfics i, en particular, en l'anàlisi de documents rics en informació gràfica. En el cas particular del reconeixement de símbols, certs descriptors s'extreuen del símbol a reconèixer i s'emparellen posteriorment amb el conjunt de símbols model. En aquest context, una de les principals inquietuds és assegurar-se que els sistemes que es proposen romanen escalables respecte al volum de dades i que poden fer front a quantitats creixents de models. Per tal d'evitar el fet de treballar amb una base de dades de símbols de referència, s'han proposat en els últims anys els sistemes de reconeixement de símbols al vol, o els sistemes de localització aproximada de símbols gràfics coneguts com a mètodes de *symbol spotting*.

En termes generals, es pot definir el problema de *symbol spotting* com la identificació d'un conjunt de regions d'interès d'un document que puguin contenir una instància d'un determinat símbol sense haver d'aplicar explícitament tot el procés de reconeixement de patrons. El nostre marc d'aplicació consisteix en la indexació d'una col·lecció de documents gràfics. Aquesta col·lecció es consulta donant un exemple del que es vol trobar, és a dir, amb una única instància del símbol a cercar i, gràcies als mètodes de *spotting* es retornen les regions d'interès dels documents on és probable trobar el símbol en qüestió. Aquest tipus d'aplicacions es coneixen com a recuperació d'informació dirigida.

Per tal que els sistemes de recuperació d'informació puguin gestionar grans col·leccions de documents necessitem proporcionar un accés eficient als grans volums d'informació que es poden emmagatzemar. Farem ús d'estratègies d'indexació que siguin capaces de retornar localitzacions on apareguin parts similars del símbol consultat. En aquest escenari, els patrons gràfics s'hauran d'emprar com a índexs afavorint l'accés i la navegació de la col·lecció de documents. Aquests mecanismes d'indexació permeten a l'usuari buscar elements semblants utilitzant informació gràfica en lloc de formular consultes textuais.

Al llarg d'aquesta tesi es presenten una arquitectura i diferents mètodes de localització aproximada de símbols, per tal de construir una aplicació de recuperació dirigida fent front a una col·lecció de documents gràfics.

S'han proposat diferents descriptors de símbols que codifiquen informació geomètrica i estructural. L'objectiu d'aquests descriptors és descriure les parts dels símbols de manera molt compacta i eficient. Signatures vectorials, cadenes amb atributs i descriptors de forma genèrics s'han fet servir per agrupar símbols per semblança.

S'han utilitzat diverses estratègies de cerca d'informació gràfica per semblança. Per tal de recuperar les localitzacions dins de la col·lecció de documents on apareixen les parts dels símbols, hem utilitzat *lookup tables* i *grid files* indexades a partir de patrons gràfics. S'ha introduït una fase final de verificació per validar les hipotètiques localitzacions on és probable que es trobi un cert símbol. Aquesta etapa de validació està formulada en termes d'informació espacial i relacional.

A més a més, es proposa un protocol per avaluar el rendiment dels mètodes de localització aproximada de símbols en termes de taxes de reconeixement, precisió en la localització i escalabilitat. Es mostra que les mesures que es proposen permeten determinar els punts forts i febles dels mètodes analitzats. Totes les contribucions proposades s'han posat a prova experimentalment amb una col·lecció de planells arquitectònics, amb la corresponent base de dades de referència.

**Paraules clau:** *Reconeixement de Patrons, Reconeixement de Gràfics, Descripció de Símbols, Localització Aproximada de Símbols, Recuperació d'Informació Dirigida, Indexació de Patrons Gràfics, Avaluació del Rendiment.*



# Abstract

Usually pattern recognition systems consist in two main parts. On the one hand, the data acquisition and, on the other hand, the classification of this data on a certain category. In order to recognize which category a certain query element belongs to, a set of pattern models must be provided beforehand. An off-line learning stage is needed to train the classifier and offer a robust classification of the patterns. Within the pattern recognition field, we are interested in the recognition of graphics and, in particular, on the analysis of documents rich in graphical information. In the particular case of graphical symbol recognition, descriptors are extracted from the symbol to recognize and are subsequently matched with the set symbol models. In this context, one of the main concerns is to see if the proposed systems remain scalable with respect to the data volume so as it can handle growing amounts of symbol models. In order to avoid to work with a database of reference symbols, symbol spotting and on-the-fly symbol recognition methods have been introduced in the past years.

Generally speaking, the symbol spotting problem can be defined as the identification of a set of regions of interest from a document image which are likely to contain an instance of a certain queried symbol without explicitly applying the whole pattern recognition scheme. Our application framework consists on indexing a collection of graphic-rich document images. This collection is queried by example with a single instance of the symbol to look for and, by means of symbol spotting methods we retrieve the regions of interest where the symbol is likely to appear within the documents. This kind of applications are known as focused retrieval methods.

In order that the focused retrieval application can handle large collections of documents there is a need to provide an efficient access to the large volume of information that might be stored. We use indexing strategies in order to efficiently retrieve by similarity the locations where a certain part of the symbol appears. In that scenario, graphical patterns should be used as indices for accessing and navigating the collection of documents. These indexing mechanism allow the user to search for similar elements using graphical information rather than textual queries.

Along this thesis we present a spotting architecture and different methods aiming to build a complete focused retrieval application dealing with a graphic-rich document collections.

Different symbol descriptors encoding geometric and structural information are proposed in this thesis. These descriptors aim to describe parts of the symbols in a very compact and efficient way. Vectorial signatures, attributed strings and off-the-shelf shape descriptors are used to cluster parts of the symbols by similarity.

Several strategies aiming to search for graphical information by similarity are used in this thesis. In order to retrieve locations from the document collection where parts of the symbols appear we use lookup tables and grid files indexed by graphical patterns. A final validation phase is introduced to validate the hypothetic locations where a symbol is likely to be found. This validation stage is formulated in terms of spatial and relational information.

In addition, a protocol to evaluate the performance of symbol spotting systems in terms of recognition abilities, location accuracy and scalability is proposed. We show that the evaluation measures allow to determine the weaknesses and strengths of the methods under analysis. All the proposed contributions have been tested with an experimental scenario consisting of a collection of architectural drawings with its corresponding ground-truth.

**Keywords:** *Pattern Recognition, Graphics Recognition, Symbol Description, Symbol Spotting, Focused Retrieval, Graphical Pattern Indexation, Performance Evaluation.*

# Contents

<b>Agraiments</b>	<b>i</b>
<b>Resum</b>	<b>iii</b>
<b>Abstract</b>	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Document Image Analysis and Recognition Context . . . . .	1
1.1.1 Accessibility to Large Document Collections . . . . .	2
1.1.2 Information Spotting . . . . .	3
1.2 Symbol Spotting . . . . .	4
1.3 Objectives and Contributions of this Thesis . . . . .	6
1.4 Organization . . . . .	9
<b>2 State of the Art in Symbol Spotting</b>	<b>13</b>
2.1 Introduction . . . . .	13
2.2 Spotting Graphical Elements . . . . .	14
2.2.1 Word Spotting . . . . .	14
2.2.2 Symbol Spotting . . . . .	16
2.3 Symbol Description . . . . .	18
2.3.1 Photometric Description . . . . .	20
2.3.2 Geometric Description . . . . .	21
2.3.3 Syntactic and Structural Description . . . . .	29
2.4 Descriptors Organization and Access . . . . .	31
2.4.1 Sequential Access . . . . .	32
2.4.2 Hierarchical Organization . . . . .	33
2.4.3 Prototype-based Search . . . . .	35
2.4.4 Hashing Approaches . . . . .	35
2.4.5 Spatial Access Methods . . . . .	37
2.4.6 Curse of Dimensionality . . . . .	38
2.5 Hypotheses Validation . . . . .	38
2.5.1 Statistical Validation . . . . .	39
2.5.2 Voting Strategies and Alignment . . . . .	39
2.6 Conclusions and Discussion . . . . .	40

<b>Part I – On the Use of Photometric Descriptors for Symbol Spotting</b>	<b>43</b>
<b>3 Symbol Spotting for Document Categorization</b>	<b>45</b>
3.1 Introduction and Related Work . . . . .	45
3.2 Outline of the Approach . . . . .	47
3.3 Document Categorization by Logo Detection . . . . .	47
3.3.1 Feature Extraction and Description . . . . .	48
3.3.2 Logo Representation and Matching . . . . .	50
3.3.3 Bag-of-visual-words . . . . .	51
3.4 Introducing Spatial Density for Logo Spotting . . . . .	52
3.5 Experiments . . . . .	54
3.5.1 Evaluation Methodology . . . . .	54
3.5.2 Performance Comparison . . . . .	55
3.6 Conclusions and Discussion . . . . .	55
<b>Part II – On the Use of Geometric and Structural Constraints for Symbol Spotting</b>	<b>59</b>
<b>4 Vectorial Signatures for Symbol Recognition and Spotting</b>	<b>61</b>
4.1 Introduction and Related Work . . . . .	61
4.2 Pre-processing Step: Raster-to-vector Conversion . . . . .	63
4.2.1 Document Binarization . . . . .	64
4.2.2 Skeletonization . . . . .	65
4.2.3 Polygonal Approximation . . . . .	66
4.3 A Vectorial Signature for Symbol Description . . . . .	67
4.3.1 Representing Symbols by Attributed Graphs . . . . .	68
4.3.2 Building the Vectorial Signature . . . . .	68
4.4 Sequential Access to Signatures: Defining Regions of Interest . . . . .	71
4.5 Experimental Results . . . . .	73
4.6 Conclusions and Discussion . . . . .	78
4.6.1 Limitations of the Vectorial Signatures . . . . .	78
<b>5 Symbol Spotting Through Prototype-based Search</b>	<b>81</b>
5.1 Introduction and Related Work . . . . .	81
5.2 String Matching Theory and Algorithms . . . . .	84
5.2.1 Definitions . . . . .	84
5.2.2 Linear String Matching . . . . .	85
5.2.3 Cyclic String Matching . . . . .	87
5.2.4 A String Matching Cost Function for Polygon Recognition . . . . .	87
5.3 Spotting Method . . . . .	88
5.3.1 Symbol Representation in Terms of String Primitives . . . . .	89
5.3.2 Off-line Lookup Table Construction . . . . .	89
5.3.3 On-line Querying of Symbols: Activating Table Entries . . . . .	92
5.3.4 Hough-like Voting Scheme to Validate Location Hypotheses . . . . .	92
5.4 Experimental Results . . . . .	93

5.4.1	Silhouette Shape Matching . . . . .	94
5.4.2	Evaluation of the Contours as Primitives . . . . .	96
5.4.3	Symbol Spotting in a Document Database . . . . .	99
5.5	Conclusions and Discussion . . . . .	100
<b>6</b>	<b>A Relational Indexing Method for Symbol Spotting</b>	<b>103</b>
6.1	Introduction and Related Work . . . . .	103
6.2	Description of Graphical Symbols in Terms of Vectorial Primitives . . . . .	104
6.2.1	Vectorial Primitives . . . . .	105
6.3	Off-the-shelf Shape Descriptors Applied to Vectorial Data . . . . .	106
6.3.1	Moment Invariants . . . . .	107
6.3.2	Simple Shape Description Ratios . . . . .	109
6.3.3	Fourier Descriptors . . . . .	109
6.4	Multidimensional Hashing to Index Primitives . . . . .	110
6.5	Relational Indexing and Hypotheses Validation . . . . .	111
6.5.1	Relational Indexing . . . . .	112
6.5.2	Voting Scheme . . . . .	114
6.6	Experimental Results . . . . .	115
6.7	Conclusions and Discussion . . . . .	119
	<b>Part III – A Performance Evaluation Protocol for Symbol Spotting Systems</b>	<b>121</b>
<b>7</b>	<b>Performance Evaluation of Symbol Spotting Systems</b>	<b>123</b>
7.1	Introduction . . . . .	123
7.2	Related Work . . . . .	125
7.3	An Overview on Measures to Evaluate Retrieval Effectiveness . . . . .	126
7.3.1	Precision and Recall . . . . .	127
7.3.2	$P@n$ and $P(r)$ . . . . .	128
7.3.3	Precision and Recall Plots . . . . .	128
7.3.4	Measures of Quality . . . . .	128
7.3.5	Fall-Out and Generality . . . . .	130
7.3.6	Central Tendency of Precision and Recall . . . . .	131
7.4	Precision and Recall for Spotting Systems . . . . .	131
7.4.1	Precision and Recall of Regions of Interest . . . . .	132
7.4.2	Measures of Quality, Fall-out and Generality . . . . .	133
7.4.3	Measures at Symbol Level . . . . .	134
7.4.4	Scalability Test . . . . .	135
7.5	Evaluating a Symbol Spotting System . . . . .	136
7.5.1	Ground-truthing . . . . .	137
7.5.2	Spotting Methods Under Test . . . . .	138
7.5.3	The FPLAN-POLY Dataset . . . . .	139
7.5.4	Evaluation . . . . .	140
7.6	Conclusions and Discussion . . . . .	143
<b>8</b>	<b>Conclusions</b>	<b>149</b>

8.1	Summary of the Contributions . . . . .	149
8.2	Discussion . . . . .	151
8.3	Open Challenges . . . . .	153
<b>A</b>	<b>Databases</b>	<b>155</b>
A.1	GREC 2005 Database . . . . .	155
	A.1.1 Variation GREC-SEG . . . . .	156
	A.1.2 Variation GREC-POLY . . . . .	160
A.2	MPEG Database . . . . .	160
	A.2.1 Variation MPEG-POLY . . . . .	160
A.3	FPLAN-POLY Database . . . . .	164
	<b>Bibliography</b>	<b>169</b>

# List of Tables

2.1	State-of-the-art symbol spotting approaches. . . . .	18
2.2	State-of-the-art photometric symbol descriptors. . . . .	23
2.3	State-of-the-art geometric symbol descriptors. . . . .	28
2.4	State-of-the-art syntactic and structural symbol descriptors. . . . .	32
3.1	Evaluation measures for the document categorization experiment. . . . .	55
4.1	Examples of sub-shapes composing the vectorial signature. . . . .	70
4.2	Results of the recognition of GREC-SEG database (1). . . . .	74
4.3	Results of the recognition of GREC-SEG database (2). . . . .	75
5.1	FPR and threshold values for several TPR. . . . .	96
5.2	False positives for a certain number of retrieved zones. . . . .	97
5.3	Retrieval performance measures for symbol spotting. . . . .	100
6.1	Recognition results of the relational indexing method. . . . .	118
7.1	Retrieval Matrix. . . . .	127
7.2	Measures of quality. . . . .	141
7.3	Scalability test details. . . . .	143
A.1	GREC 2005 Database (1). . . . .	157
A.2	GREC 2005 Database (2). . . . .	158
A.3	GREC 2005 Database (3). . . . .	159
A.4	Some characteristics of GREC-SEG dataset. . . . .	160
A.5	Some characteristics of GREC-POLY dataset. . . . .	161
A.6	MPEG Database (1). . . . .	162
A.7	MPEG Database (2). . . . .	163
A.8	Some characteristics of MPEG-POLY dataset. . . . .	164
A.9	Some characteristics of FPLAN-POLY dataset. . . . .	165





# List of Figures

1.1	Symbol spotting applied to a focused retrieval application overview. . .	6
1.2	General architecture of a symbol spotting system. . . . .	7
2.1	Word image signature for word spotting. . . . .	14
2.2	Dendrogram representation for symbol spotting. . . . .	17
2.3	Classification of symbol description techniques. . . . .	19
2.4	Generic Fourier descriptor example. . . . .	20
2.5	Example of the Zernike and Legendre moments. . . . .	22
2.6	Computation of arc-length-based signatures. . . . .	24
2.7	Shape context descriptor for shape matching. . . . .	25
2.8	Examples of graphical tokens-based description. . . . .	26
2.9	Example of vectorial signatures for symbol description. . . . .	29
2.10	String representation of closed contours. . . . .	30
2.11	Attribute relational graph for symbol description. . . . .	30
2.12	Symbol spotting by using sliding windows. . . . .	33
2.13	Hierarchical organization of descriptors. . . . .	34
2.14	Using hash tables for descriptor access. . . . .	36
2.15	Quadtree representation of a shape. . . . .	37
3.1	Overview of the proposed document categorization method. . . . .	48
3.2	SIFT and shape context descriptors. . . . .	50
3.3	Matching logos in documents with the SIFT features. . . . .	52
3.4	Introducing spatial density information to spot logos. . . . .	53
3.5	Confusion matrices for the document categorization experiment. . . .	56
3.6	Matching symbols in line-drawings with the SIFT features. . . . .	58
4.1	Binarization of an old handwritten document. . . . .	65
4.2	(3,4)-Distance-based skeletonization. . . . .	66
4.3	Three levels of the straight line approximation. . . . .	67
4.4	Attributed graph representation of graphical symbols. . . . .	69
4.5	Sub-shapes extracted from a symbol at different levels. . . . .	70
4.6	Computing a region of interest from a reference segment. . . . .	71
4.7	Obtaining ROIs from a low-res representation of line-drawings. . . . .	73
4.8	Example of synthetical distortion from the GREC-SEG dataset. . . . .	76
4.9	Average true positive rates for three different symbol categories. . . .	77

4.10	Qualitative results of spotting symbols by using vectorial signatures. . .	79
5.1	Polyline decomposition of a vectorized symbol. . . . .	82
5.2	Example of the string matching algorithm. . . . .	86
5.3	Attributed representation of a chain of adjacent segments. . . . .	88
5.4	Symbol representation from polygonally approximated contours. . . .	90
5.5	Anti-aliasing voting scheme. . . . .	94
5.6	ROC curve for the silhouette matching experiment. . . . .	95
5.7	Different symbol primitive representations. . . . .	97
5.8	Precision and recall plot. . . . .	98
5.9	ROC curve for the symbol matching experiment. . . . .	98
5.10	Precision and recall plot for symbol spotting in the document database. .	99
5.11	Qualitative results for spotting symbols. . . . .	101
6.1	Primitive symbol decomposition. . . . .	106
6.2	Grid file to index vectorial primitives. . . . .	111
6.3	Relational indexing example. . . . .	112
6.4	Relational indexing architecture. . . . .	113
6.5	Center mapping function for pose estimation. . . . .	114
6.6	Qualitative results of the relational indexing method (1). . . . .	116
6.7	Qualitative results of the relational indexing method (2). . . . .	117
6.8	Illustration of a result which is difficult to evaluate. . . . .	118
7.1	$F^1(r)$ -score plots for different synthetic precision and recall plots. . . .	130
7.2	Central tendency of precision and recall. . . . .	131
7.3	Overlapping between results and ground-truth. . . . .	132
7.4	Scalability test example. . . . .	136
7.5	Sketching annotation tool for ground-truth generation. . . . .	137
7.6	Symbol models and an example of a document in the database. . . . .	139
7.7	Precision recall and fall-out recall plots. . . . .	140
7.8	$F_A^1(r)$ -score plots. . . . .	142
7.9	Scalability tests (1). . . . .	144
7.10	Scalability tests (2). . . . .	145
7.11	Scalability tests at symbol level. . . . .	146
A.1	Example of variations for the GREC database. . . . .	156
A.2	Example of the distortions of the MPEG-POLY database. . . . .	161
A.3	Example of the distortions of the FPLAN-POLY database. . . . .	164

# Chapter 1

## Introduction

---

In this chapter we put in context the symbol spotting problem. By giving a general overview of the Document Image Analysis and Recognition field and, in particular, of the Graphics Recognition research topic, we present the motivations of this work. We summarize the objectives and contributions of this work as well as the contents of each chapter.

---

### 1.1 Document Image Analysis and Recognition Context

Document Image Analysis and Recognition (DIAR) is one of the most important subfields of Pattern Recognition. In its early years, the research efforts were mainly focused on the processing of textual documents. In particular, most research efforts were centered in the development of automatic reader systems which entailed the design of effective Page Layout Analysis (PLA) methods and Optical Character Recognition (OCR) techniques. However, nowadays, commercial OCR software performing good recognition results in type-written documents can be purchased, and we can say that OCR in type-written documents is a mature problem from the scientific point of view. Today, the interests of the Document Image Analysis and Recognition community cover a wide spectrum of open challenges. Let us enumerate a few of them. For instance, the processing of hand-written documents, for both off-line [BB08] and on-line [SMVG08] inputs, is still an important research topic. The huge variability of the character shapes among different writers make hand-written character recognition to be a much more complex and interesting problem than type-written OCR. Another research topic which has attracted the attention of researchers in the last years is the problem of processing documents acquired with low-resolution digital cameras, as in [LD08]. This problem has emerged with the presence of such cameras in ordinary devices like PDAs or cell-phones and the big amount of interesting applications that can be envisaged with the inclusion of recognition tasks in portable devices. As an

example, the cell-phone *Motorola A1200* has a built-in OCR able to process business cards by finding names, phone numbers, addresses and automatically import them to the phone-book. Another actual and interesting problem is the analysis of web documents, as presented in [EFM07, YN07]. Although the process of web documents seems quite easy since they are digital-born documents, they may be a great challenge since they can contain a great amount of artwork, a great variability of font types, different font sizes, non-standard layouts, a large variety of colors, etc. which difficult recognition tasks. Finally, another example of open problem that nowadays is receiving a lot of interest is the management of digital libraries of cultural heritage documents like in [CCL07, BvKS07]. Usually the main problem to tackle in such applications is the management of historic documents which may be very old and degraded. In addition, these document collections are quite large and the methods to analyze these documents should be conceived to provide an efficient access to such amounts of information.

### 1.1.1 Accessibility to Large Document Collections

Nowadays, there is still a huge amount of information stored in paper format. Libraries are the main example. For instance, the Spanish National Library<sup>1</sup> has about eight millions of paper documents (besides books) of different kinds such as musical scores, maps, plans, engravings, etc. Great efforts are made to digitize such amount of information mainly for space saving and preservation issues, but also to avoid physical boundaries and to facilitate the information retrieval. For example, *Gallica*<sup>2</sup> is the digital library for on-line users of the French National Library. It provides free access to 90.000 scanned and OCRed books and has made available more than 80.000 document images. The interest of providing access to books through the web has also gained importance with big initiatives such as *Google Books*<sup>3</sup>. However, the need of digitizing paper documents is not just a specific problem of libraries, and it is not just focused to old and rare documents which need preservation. Hundreds or even thousands of invoices, receipts, faxes, etc. can be managed per day by big companies. Obviously, the cost of storing and consulting this information in paper format becomes unaffordable and the use of a digital collection becomes a must.

However, these huge amounts of digitized information are usually stored in poor formats making difficult the accessibility to the contained information. On the one hand, type-written documents are scanned and then transcribed by an OCR software to provide access to the text. The fact of storing these collections by using the ASCII character encoding allow to retrieve desired contents from the collection by using textual queries. In this particular scenario, the main challenge nowadays is to add semantic information to these digital documents in order to permit a higher level information extraction process. On the other hand, there is a lot of documents which can not be processed by an OCR software since they are hand-written or contain non-textual information. In those cases, digital libraries use facsimile representations

---

<sup>1</sup><http://www.bne.es/>

<sup>2</sup><http://gallica.bnf.fr/>

<sup>3</sup><http://books.google.com/>

of these documents, i.e. the image arising from the scanning process, to store them. Even if the use of facsimile representation is useful for storing and preservation issues, it still presents a great drawback, which is the lack of accessibility. Nowadays, recognition methods for hand-written documents or non-textual elements do not reach such reliable recognition rates as OCR systems. In addition, the computational cost of recognizing type-written characters is very low in comparison with hand-written character recognition or graphic recognition schemes. These constraints provoke that usually, facsimile documents are just manually annotated with a set of previously harvested metadata. This means that the only information we have about these documents is a set of predefined keywords and these documents can not be queried in terms of their contents but they can be retrieved just by querying the predefined keywords. This problem is common to any search tool that has to face non-textual information. For example, *Google Image Search*<sup>4</sup> service bases its search engine on the images filenames and text adjacent to the images. In this context, there is a need of creating tools aiming to provide efficient categorization, indexation, browsing, information retrieval in terms of visual contents, etc. for non-textual documents, without any human inspection of each document. In particular, one of the main motivations of our work is the adaptation of the idea of text mining techniques to non-textual elements. In that scenario, graphical patterns should be used as indices for accessing and navigating large collections of documents.

### 1.1.2 Information Spotting

The use of graphic indices to access non-textual documents is not straightforward. One of the strategies proposed to enhance the accessibility of large data collections that may result suitable in the case of non-textual documents is the *Information Spotting* technique. We can define the term spotting as the task of locating and retrieving specific information from large datasets without explicitly recognizing it. That means that if we want to provide a retrieval tool for non-textual documents, with a spotting approach there is no need to fully recognize all the objects conforming a document in the database but to coarsely locate some regions of interest where the queried object is likely to be found. These spotting approaches were already proposed some years ago within the speech recognition field in order to spot spoken words from a sound recording. In the works of [GN93, RJN93, JFJ95], the use of Hidden Markov Models (HMM) allowed to process the speech signal and to focus the attention on a set of time intervals where a certain keyword is likely to be pronounced. This problem is known as phonetic word spotting.

Spotting techniques have also been applied to textual document images in the recent years by following the same ideas of the word spotters used in the speech recognition field. Even if images are two-dimensional structures, the text lines appearing on those documents are segmented and they are taken as one-dimensional signals. These signals are then processed by a HMM or a neural network as presented in [WPW95]. The locations where the output of the network has higher responses are the ones likely to contain the queried word. These techniques can be applied to

---

<sup>4</sup><http://images.google.com/>

both type-written documents, as the case of automatic fax routing applications like in [VRL04], or hand-written documents, as the historical word spotting method presented in [RM03]. The use of spotting methodologies to treat textual documents allow to process large amounts of documents without the need to apply an OCR software to get the ASCII characters. This methods have particular interest in applications dealing with documents where an OCR would not produce a reliable result.

These methods are however hardly useful when we want like to treat graphic-rich documents. All word spotting methods make the strong assumption that all the objects in the document can be segmented and transformed into a one-dimensional signal in order to be processed in a linear way. This assumption is no longer valid when we want to spot graphic elements instead of textual ones. In the last years, the problem of spotting symbols within graphical documents has been an emerging research topic.

## 1.2 Symbol Spotting

Among the Graphics Recognition community, a lot of efforts have been devoted over the years to the problem of recognize symbols. Several contests of Symbol Recognition have been held during the last editions of the *Graphics Recognition Workshop* (GREC). These contests are an excellent way to track the progress of the research on this specific problem and aim to determine which are the challenges and the future research directions. In the last edition<sup>5</sup>, an important challenge to be addressed has been identified. For many years researchers of the Symbol Recognition community centered their methods in recognizing isolated symbols undergoing several transforms and degradations. Nowadays, state-of-the-art recognition schemes yield performances far above ninety percent of recognition rates but the real challenge should not be to achieve the one hundred percent but should be centered in three different aspects. In the first place we should see if the proposed methods are real scalable in terms of the number of symbols to recognize. Secondly, it should be tested if the proposed methods could be applied to any symbol design or whether they are ad-hoc conceived to recognize a specific dataset and tackle with a specific source of noise. And finally and most important, we should consider if these methods can recognize symbols present in complete drawings without previous segmentation. In this direction, the concept of spotting graphical symbols within graphic-rich documents has been introduced by Tombre and Lamiroy in [TL03]. Five years later, the authors present in [TL08] some achievements in this field by pointing several open challenges.

Generally speaking, the *Symbol Spotting* problem can be defined as the location of a set of regions of interest from a document image which are likely to contain an instance of a certain queried symbol without explicitly recognizing it. One of the main applications for symbol spotting methods is its use in large collections of documents. This particular application can be seen as a Content Based Image Retrieval (CBIR) application but having some particularities. The main difference is that standard

---

<sup>5</sup> Seventh IAPR International Workshop on Graphics Recognition, GREC07. Curitiba, Brazil. 20-21 September 2007.

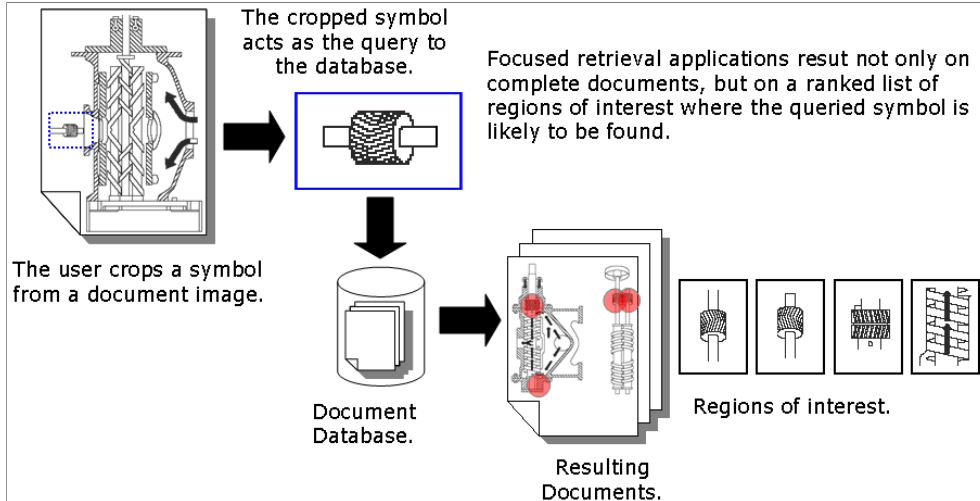
document retrieval approaches find atomic documents, leaving to the user the tasks of locating relevant information within the provided results. Whereas symbol spotting provides the user a more direct access to relevant information by returning a set of regions of interest which are sub-parts of the documents in the collection. Such applications which return passages of interest within documents instead of complete documents, are known as *Focused Retrieval* systems. The interested reader is referred to the recent review by Joty and Sadid-Al-Hasan [JSAH07] on the topic of focused retrieval.

To our best knowledge, in the workshops organized by the community of focused retrieval<sup>6</sup>, no works dealing with graphics have ever been proposed, and all the works are centered in the retrieval of textual passages from ASCII documents. Back to the image documents, there is an important difference between the spotting systems dealing with graphics and the ones dealing with word images. In the case of word spotting, usually a learning step is required and only a small subset of keyword queries is allowed. In the symbol spotting problem, the amount of items composing the symbol alphabet can increase indefinitely. In addition, the input of a spotting system is the user's query symbol which he wants to retrieve from the whole collection. Therefore, usually the spotting systems are queried by example. That is, the user segments a symbol he wants to retrieve from the document database and this cropped image acts as the input. This particularity reinforces the fact that spotting methods should not work for a specific set of model symbols nor have a learning stage where the relevant features describing a certain symbol are trained. The retrieval of the relevant zones should be done on-the-fly. Nevertheless, in the acquisition step, i.e. when a given document is added to the collection (which is a process that could be done off-line) several steps of primitive extraction and description can be computed. The desired output of the spotting methods is a ranked list of zones of interest likely to contain similar symbols to the queried one. That is, each result should have an associated confidence value depending on a certain similarity function between the query and the result. We can see an overview of the symbol spotting methodology applied to a focused retrieval application in Fig. 1.1.

In Document Image Analysis and Recognition and in Computer Vision in general, the relationship between recognition results and segmentation performance presents a common problem known as the Sayre paradox [Say73]. In order to achieve good recognition results the objects should be previously segmented, but to get a reliable segmentation, the objects should be previously recognized. To avoid such paradox, symbol spotting architectures do not use a preliminary segmentation step followed by a proper recognition method but are usually conceived to coarsely recognize and segment in a single step. We can appreciate in Fig. 1.2 our proposal of a general architecture for symbol spotting systems. Basically, three different levels can be identified. The first level aims to represent and compactly describe the primitives that compounds the graphical symbols. These features describing graphical symbols are then stored in a particular data structure. This data structure should be chosen carefully

---

<sup>6</sup>The first Workshop on Focused Retrieval <http://www.cs.otago.ac.nz/sigirfocus/> held in Amsterdam in 2007, and the second Workshop on Focused Retrieval <http://www.cs.otago.ac.nz/sigirfocus2008/> held in Singapore in 2008.



**Figure 1.1:** Symbol spotting applied to a focused retrieval application overview.

in order to provide efficient access to the symbol descriptors. During the querying process, this data structure is traversed and the locations within the document images where to find similar primitives than the queried ones are retrieved. A final validation stage determines which are the valid hypotheses where the queried symbol is likely to be found.

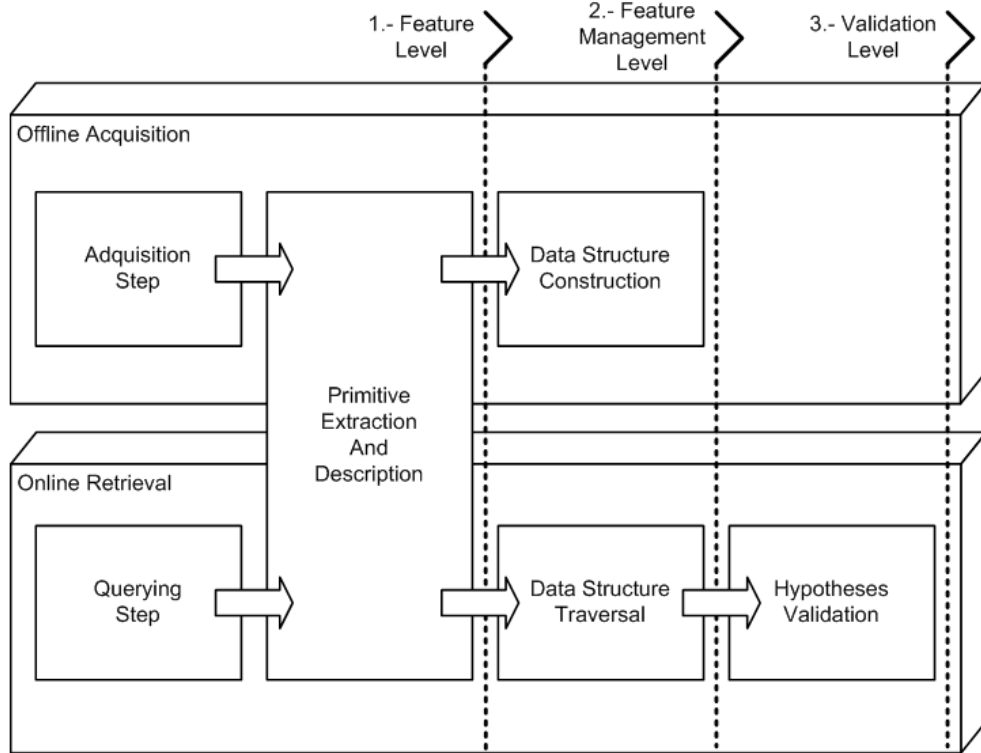
Summarizing, our work has been motivated by the specific problem of proposing a spotting methodology applied to a focused retrieval problem. The proposed methods should be able to locate and retrieve graphical content within a database of complete document images. From a methodological point of view, the main challenges stem from the nature of the queries, which are iconic queries instead of the ASCII strings used in the keyword-based searches. The fact of working with graphical entities raises several problems to tackle. The first important problem is how to compactly represent and describe symbols without a preliminary segmentation stage. Another important issue is the choice of the data structures allowing to efficiently retrieve graphical patterns by similarity.

### 1.3 Objectives and Contributions of this Thesis

#### Main Objective of this Work

The main objective of this work is to propose a symbol spotting methodology for locating graphic symbols within a collection of complete documents. The spotting method is formulated in terms of a search by similarity of all the primitives which compose the queried graphical symbol. Among the wide variety of possible symbols and graphic documents, we have basically focused our research on a framework dealing





**Figure 1.2:** General architecture of a symbol spotting system.

with technical line-drawings such as architectural floor-plans or electronic schemes.

To this end, the problem will be tackled from different points of view and this main objective can be detailed into the following points:

### 1. Testing Well-known Methods from the Computer Vision Field

Although symbol spotting has its own particularities, the problem of locating symbols in documents can be seen as a particular case of the object recognition problem from the Computer Vision field. Our first objective is to test if such well-known techniques can be applied to the problem of spotting graphic symbols. In this part of the work we describe graphical symbols by means of well-known photometric descriptors. We identify which are the limitations of those approaches in the particular scenario of spotting symbols in line-drawing collections.

The main contribution of this part of the thesis do not correspond to the recognition methodology, since we use off-the-shelf recognition methods, but the application of this kind of techniques to the graphics recognition domain.

## 2. Geometric and Structural Symbol Description Techniques

Since our work is mainly focused on technical line-drawings, the rest of the thesis is centered on the use of geometric and structural constraints to describe graphical symbols and graphic-rich documents. The primitives to extract and the description techniques to represent a graphical symbol are expressed and defined in the domain vectorial domain instead of working with the raw image format. The objective of this part of the thesis is to find a methodology to describe symbols aiming to cope with the different noise sources that we have to face in our framework. In this work we present three different proposals of vectorial primitives and the subsequent symbol description techniques:

- **Vectorial Signatures:** The use of signatures as a coarse description technique is usually used on spotting systems. Taking vectors as the primitives which compound a graphical symbol, a model of vectorial signature is proposed. Symbols are described by the occurrences of simple geometric configurations among segments.
- **String Representation of Polygons:** The second proposal to represent graphical symbols is the use of a higher-level entity than segments. In that case graphical symbols are described by a set of chains of adjacent segments grouped into polygon instances. These polygons are described as one-dimensional attributed strings, and the distance between two similar polygons is computed by using string edit operations.
- **Off-the-shelf Shape Descriptors Applied to Vectorial Primitives:** Finally, we study the use of several well-known shape descriptors applied to the vectorial primitives which compose a symbol. In this case the contribution is not the descriptors themselves but its use to represent vectorial symbols.

## 3. The Descriptors Organization

The second main research axis of this thesis is centered on how the primitives' descriptors can be organized in a data structure for posterior efficient access. The main objective of this part is to find a mechanism allowing graphical patterns to be used as indices so as to provide an efficient access to graphic information contained in large data corpora.

Along this thesis we present three different approaches, each one of them related to the previous description of symbols. These structures aim to organize by similarity all the extracted vectorial primitives from the documents in the collection. In focused retrieval applications, it is indispensable to avoid one-to-one matching when querying a certain graphical primitive by providing mechanisms which allow to search for graphical primitives by similarity.

#### 4. The Hypotheses Formulation

The last step of spotting architectures is the hypotheses formulation. Regions of interest where the queried symbol is likely to appear have to be generated with their associated confidence value. The main objective of this part, is to present validation schemes which aim to reduce the false alarms that may appear from the retrieval of primitives by similarity.

Inspired by the classical voting schemes where the hypotheses validation is done in terms of an accumulation of evidences, we present a validation scheme which aims to discard false alarms. In addition of the accumulation of evidences in terms of locations within a document where the query symbol can be found, we also propose a relational validation method, which also takes into account the spatial configuration and the structural relationships among the primitives which composes a graphical symbol.

#### 5. Performance Evaluation

Finally, one of the main concerns of the Graphics Recognition community is the generation of evaluation studies which aim to assess and compare the accuracy and robustness of the proposed methods. To our best knowledge, there has been very few attempts to describe a performance evaluation protocol for symbol spotting architectures applied to focused retrieval tasks.

Inspired by several works on the performance evaluation of Graphics Recognition methods and algorithms, and on the evaluation measures used in the Information Retrieval field, we propose a set of measures which aim to evaluate the performance of symbol spotting systems in terms of its localization and recognition abilities.

## 1.4 Organization

The rest of this thesis is organized in eight chapters and one appendix, structured in three main parts.

In chapter 2, the state of the art in symbol spotting is reviewed. Since symbol spotting is quite an emerging topic, the literature dealing with this problem is not vast. Some other works which are not directly related to the problem of spotting symbols, but which may be related to one of the three levels of the spotting architecture are presented. After a brief overview of the literature of symbol spotting, we organize this chapter in three differentiated parts, namely, the state of the art in symbol description techniques, in feature organization and in hypotheses validation.

## Part I

The first part of this thesis, is centered on the application of well-known methods of Computer Vision for recognizing objects in scenes, to the specific problem of spotting graphical symbols in documents.

- In chapter 3 we present a method for spotting symbols by using techniques from the Computer Vision field. As a running example, we present an application of logo spotting for a document categorization application. The method processes incoming document images such as invoices or receipts. The categorization of these document images is done in terms of the presence of a certain graphical entity detected without segmentation. The symbols are described by a set of local features computed by using the well-known methods of SIFT and shape context descriptors. The categorization of the documents is performed by the use of a bag-of-visual-words model. Spatial coherence is introduced by a voting scheme in order to reinforce the correct category hypotheses, aiming also to spot the logo inside the document image. Experiments which demonstrate the effectiveness of this system on a large set of real data are presented.

## Part II

The second part of this thesis is devoted to proposing spotting methods in a framework of line-drawing images. Therefore, it is centered on the use of geometrical and structural constraints as symbol description techniques.

- In chapter 4 we present a method to determine which symbols are probable to be found in technical drawings by the use of vectorial signatures as symbol descriptors. The proposed signature model is formulated in terms of geometric and structural constraints among segments, as parallelisms, straight angles, etc. After representing vectorized line drawings with attributed graphs, our approach works with a multi-scale representation of these graphs, retrieving the features that are expressive enough to create the signature. A window-based system aims to compute these signatures within complete documents identifying the zones of interest where a symbol is likely to appear.
- In chapter 5 we present a spotting method which uses a prototype-based search as the basis for the focused retrieval task. First, symbols are decomposed in primitives representing closed regions. These primitives are then encoded in terms of attributed strings. Second, the strings are organized in a lookup table so that the set median strings act as representative prototype of the clusters of similar primitives. This indexing data structure aims to efficiently retrieve the locations from the document collection where to find similar primitives than the queried ones. Finally, a voting scheme formulates hypotheses in the locations of the line drawing image where there is a high presence of regions similar to the queried ones, and therefore, a high probability to find the queried graphical symbol. The proposed approach has been proved to work even in the presence of noise and distortion introduced by the scanning and raster-to-vector processes.

- In chapter 6 we present an indexing method aiming to retrieve locations of interest where a query symbol is likely to be found. In order to foster the querying speed, a hashing technique is proposed which is able to retrieve primitives by similarity very efficiently. Vectorial primitives are coarsely encoded by well-known shape description methods providing a numerical description of the primitives. A relational indexing approach is presented in order to introduce some structural information of the symbols and provide an accurate hypotheses validation. Experimental results show the performance of the proposed approach.

### Part III

Finally, the third part of this thesis has just one chapter focused on the performance analysis of spotting methods.

- Chapter 7 is centered on the performance evaluation of spotting systems. Since symbol spotting systems and focused retrieval applications shall have the ability to recognize and locate graphical symbols in a single step, the measures to evaluate the performance of a symbol spotting system are defined in terms of recognition abilities, location accuracy and scalability. By testing the spotting method of chapter 6 we show that the proposed measures allow to determine the weaknesses and strengths of the analyzed method.

Finally, in chapter 8 we give some concluding remarks about this work, and we specify some possible future research lines on symbol spotting techniques.

Along this work, different symbolic databases have been used to perform the experiments. All these databases are explained in the appendix A. For each database, we detail the kind of symbols it contains and the distortions which have been introduced in the original elements. Some other characteristics for each database, as the number of elements, the number of primitives in the vectorial representation, their size, etc. are also detailed.



# Chapter 2

## State of the Art in Symbol Spotting

---

In this chapter we will review the related work to symbol spotting which has been proposed in the last years. We first present a review of the contributions from the Graphics Recognition community to the spotting problem. In a second part, we focus the attention on the different symbol description techniques and families we can find in the literature. Then, the existing data structures which aim to store the extracted descriptors and provide efficient access to them will be analyzed. We finally review which are the existing methods for hypotheses validation which can be used for spotting purposes.

---

### 2.1 Introduction

Generally speaking, the architecture of a symbol spotting system consists in the three main levels outlined in the previous chapter in Fig. 1.2. In the first level, the documents are decomposed in a set of primitives which are characterized by a descriptor capturing the most important cues. The second level is focused on how these descriptors are organized to be posteriorly consulted. Finally, the third level is in charge of validating the hypotheses arising from the matching between model and stored data. This third level shall provide the resulting list of locations where a queried symbol is likely to be found.

We organize this state of the art in four different parts. First we briefly review in section 2.2 the recent contributions of the Graphics Recognition community to the spotting problem. The subsequent three parts refer to each level of the general symbol spotting architecture. In section 2.3 we focus on the symbol descriptor categorization. Section 2.4 focuses on the organization and access to the stored descriptors and in section 2.5 we present the existing approaches for hypotheses validation. We finally summarize in section 2.6 which are the suitable approaches for spotting graphics.

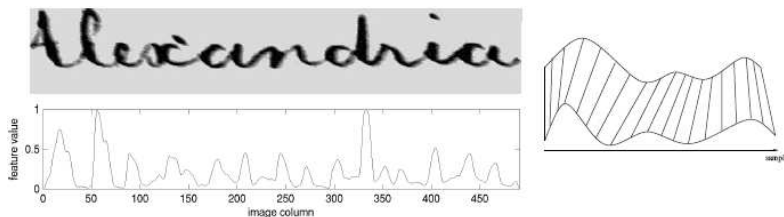
## 2.2 Spotting Graphical Elements

Among the Graphics Recognition community, a lot of efforts have been devoted in the last years to the problem of locating elements in document images. However, two different applications can be identified, namely locating words in textual image documents in the image domain or identifying regions likely to contain a certain symbol within graphics-rich documents. Although the problem is the same, the proposed methods are very different whether the focus of the application is centered on text or in graphics. Let us briefly review in the next sections the existing work on both word and symbol spotting.

### 2.2.1 Word Spotting

OCR engines benefit from the nature of alphanumeric information, i.e. text strings, are one-dimensional structures with underlying language models that facilitate the construction of dictionaries and indexing structures. Word spotting techniques take also advantage of this aspect and usually represent words as one-dimensional signals which will be further matched against the query word image. The main idea of these approaches is to represent keywords with shape signatures in terms of image features. The detection of the keyword in a document image is usually done by a crosscorrelation approach between the prototype signature and signatures extracted from the target document image.

Although using image features without word recognition, the information is still one-dimensional and it facilitates the use of some classical techniques used in speech recognition. Rath and Manmatha presented in [RM03, RM03] a method to spot handwritten words. They use the normalized projection profiles of segmented words as word signatures. These word signatures are seen as time series and are aligned using the Dynamic Time Warping (DTW) distance. We can see an example of such approach in Fig 2.1.



**Figure 2.1:** Word image signature for word spotting using the DTW distance (reprinted from [RM03]).

Kuo and Agazzi used in [KA94] another classical technique from the speech processing field. A Hidden Markov Model (HMM) is used to spot words in poorly printed documents. In this case, a learning step to train the HMM is needed. In addition, the features describing each word the user wants to query have to be learned previously.



Lladós and Sánchez proposed in [LS07] a keyword spotting method based on the shape context descriptor. Words are represented by a signature formulated in terms of the shape context descriptor and are encoded as bit vectors codewords. A voting strategy is presented to perform the retrieval of the zones of a given document containing a certain keyword.

Leydier et al. presented in [LLE07] a word spotting method in order to perform text searches in medieval manuscripts. The orientation of the gradients in a given zone of interest are taken as features describing the local structure of the strokes and the orientation of the characters' contours. A matching process is then proposed to identify and retrieve the locations where a given word is likely to be found. The experimental results show that the proposed method is tolerant to several kinds of noises as well as geometric distortions.

In [KGN07], Konidaris et al. presented another strategy for word spotting. In that case, the query is not an image but is an ASCII string typed by the user. Thus, the features which representing a word should be invariant enough to appear in both synthetic characters and extracted character from the ancient documents. The authors propose an hybrid approach by using the density of pixels in a given zone of the character and the projections of the upper and lower profile of the character. In order to improve the retrieval performance, a user's feedback procedure is also proposed.

Recently, Lu and Tan have proposed in [LT08] a very simple typewritten word coding which is useful enough to characterize documents. The proposed word code is based on character extremum points and horizontal cuts. Words are represented by simple digit sequences. Several similarity measures based on the frequency of the codes are defined to retrieve documents written in the same language or talking about similar topics.

Finally, Kise et al. addressed in [KTM02] another interesting aspect of the word spotting problem. Since they focus their approach in Japanese documents, they found that a word can be formed by several Kanji characters. Locating a query word within a document is then done by analyzing the characters density distribution within a document image. The same idea can be applied to other languages when we want to spot not only a single word, but we want to perform what is known as passage retrieval.

One of the weak points we find in almost all the methods presented in the existing literature, is that most of the approaches take advantage of the layout knowledge. By assuming that the entities of the document images follow a certain spatial structure, they are able to segment words and take them as atomic elements. To our best knowledge there are very few methods which can deal with the document image as a whole without a specific word segmentation step. This is a strong limitation of these approaches since the performance of these methods will always be strongly dependent on the performance of the previous word segmentation. We believe that rather than using crosscorrelation approaches, the use of some indexing structure pointing to the locations where the queried word is likely to appear would be much more interesting

for spotting purposes. As we will see, some symbol spotting methods are based on this idea.

### 2.2.2 Symbol Spotting

The main idea of symbol spotting is to describe symbols by a very coarse descriptor to foster the querying speed rather than the recognition rates. Even if symbol spotting is still an emerging topic, several works facing the problem can be found.

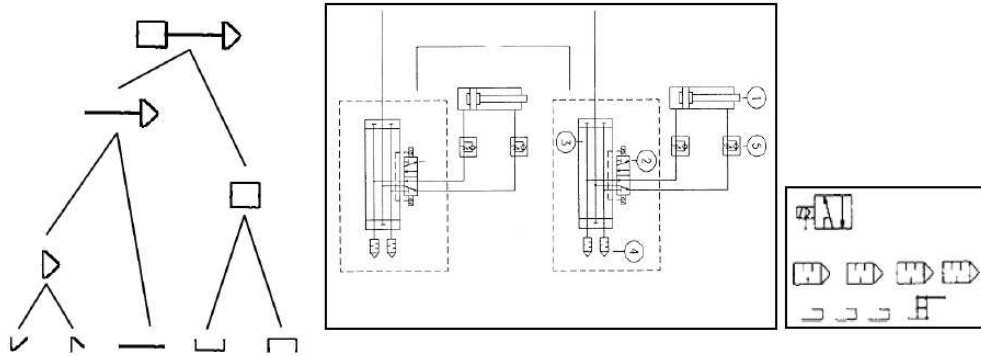
Müller and Rigoll proposed in [MR00] one of the first approaches we can identify as symbol spotting. By using a grid of fixed size the technical drawing images are partitioned. Each small cell acts then as an input in a two-dimensional HMM trained to identify the locations where a symbol from the model database is likely to be found. The main advantage the system presents is that symbols can be spotted even if they appear in a cluttered environment. However, the fact that the recognizer must be trained with the model symbols entails a loss of flexibility to the presented method.

On the other hand, some techniques work with a previous ad-hoc rough segmentation, as presented in [TWT03]. In that case, an algorithm of text/graphics separation is applied in order to separate symbols from the text and the background. In [TW04, TWZ04], the symbols which are linked to a network are segmented by analyzing the junction points of the skeleton image by a loop extraction process. After these ad-hoc segmentations, global numeric shape descriptors are computed at each location and compared against the training set of pixel features extracted from model symbols. As most of the word spotting methods, in this case, when querying a certain object, a set of segmentations are proposed. A descriptor is computed sequentially for each sub-image and a distance metric decides whether if it is the searched item or not. The one-to-one matching is a clear limitation of such approaches which won't be a feasible solution to adopt when facing large collections. In addition, the ad-hoc segmentations are only useful for a restricted set of documents which makes the method not scalable to other application domains.

Other techniques as in [MB96, LMV01, BHA05, LAT07, QRB08] rely on a graph based representation of the document images. These methods focus on a structural definition of the graphical symbols. Subgraph isomorphism techniques are then proposed to locate and recognize graphical symbols with a single step. However these approaches do not seem suitable when facing large collection of data since graph matching schemes are computationally expensive.

Realizing that the computational cost has to be taken into account, several works as [VS94, DL04, ZW07] are centered on computing symbol signatures in some regions of interest of the document image. These regions of interest can come from a sliding window or be defined in terms of interest points. Obviously, these methods are quicker than graph matching or sequential search, but make the assumption that the symbols always fall into a region of interest. In addition, symbol signatures are usually highly affected by noise or occlusions.

Zuwala and Tabbone presented in [ZT06, Zuw06] an approach to find symbols in graphical documents which is based on a hierarchical definition of the symbols. They



**Figure 2.2:** Dendrogram representation for spotting symbols in technical drawings (reprinted from [ZT06]).

propose the use of a dendrogram structure which aims to hierarchically decompose a symbol. A symbol is represented by its composing subparts splitted at the junction points. These subparts are merged according to a measure of density building the dendrogram structure. Each subpart is described by an off-the-shelf shape descriptor. The dendrogram can be subsequently traversed in order to retrieve the regions of interest of a line drawing where the queried symbol is likely to appear. We can see an example on the use of a dendrogram representation for symbols appearing in technical drawings and the obtained spotting results in Fig 2.2. The authors proposed in [TZ07] an enhancement of the traversal step, which by the use of indexing strategies allowed to reduce the retrieval time.

Finally, in some domains, graphical objects can be annotated by text labels. In these cases, the spotting mechanism could manage textual queries to provide graphical results as presented by Lorenz and Monagan in [LM95]. Najman et al. present in [NGB01] a method to locate the legend in technical documents. The text contained in the legend can be posteriorly used to extract graphical areas annotated by these text strings, as presented in [SM99] by Syeda-Mahmood. In our work we do not consider textual annotations and thus the spotting method only manages graphical entities.

We can found in Table 2.1 a summary of the state-of-the-art symbol spotting approaches. Our feeling, as in the case of word spotting, is that indexing mechanisms and voting schemes are very useful when trying to not only recognize a graphical object but when trying to locate and recognize at the same time. Spotting methods which do not use indexing structures may discriminate zones of interest from a document image, but can hardly be transferred to a real focused retrieval application dealing with large collections of document images. Let us focus in the next section on the problem of how can we describe graphical symbols.

**Table 2.1:** State-of-the-art symbol spotting approaches.

Family	Method	Pros.	Cons.
2D HMM	[MR00]	Segmentation-free.	Needs training.
Pixel Features	[TWT03] [TW04] [TWZ04]	Robust symbol description.	Ad-hoc previous segmentation.
Graph-based	[MB96] [LMV01] [BHA05] [LAT07] [QRB08]	Simultaneous symbol segmentation and recognition.	Computationally expensive.
Symbol Signatures	[VS94] [DL04] [ZW07]	Compact and simple symbol description.	Performance decreases if the symbol could no be perfectly isolated.
Hierarchical Symbol Representation	[ZT06] [Zuw06] [TZ07]	Linear matching is avoided by using an indexing technique.	Dendrogram structure is strongly dependent on the merging criterion.
Textual queries	[LM95] [SM99] [NGB01]	More robust since it is easier to recognize characters than symbols.	Only applicable when textual information is present.

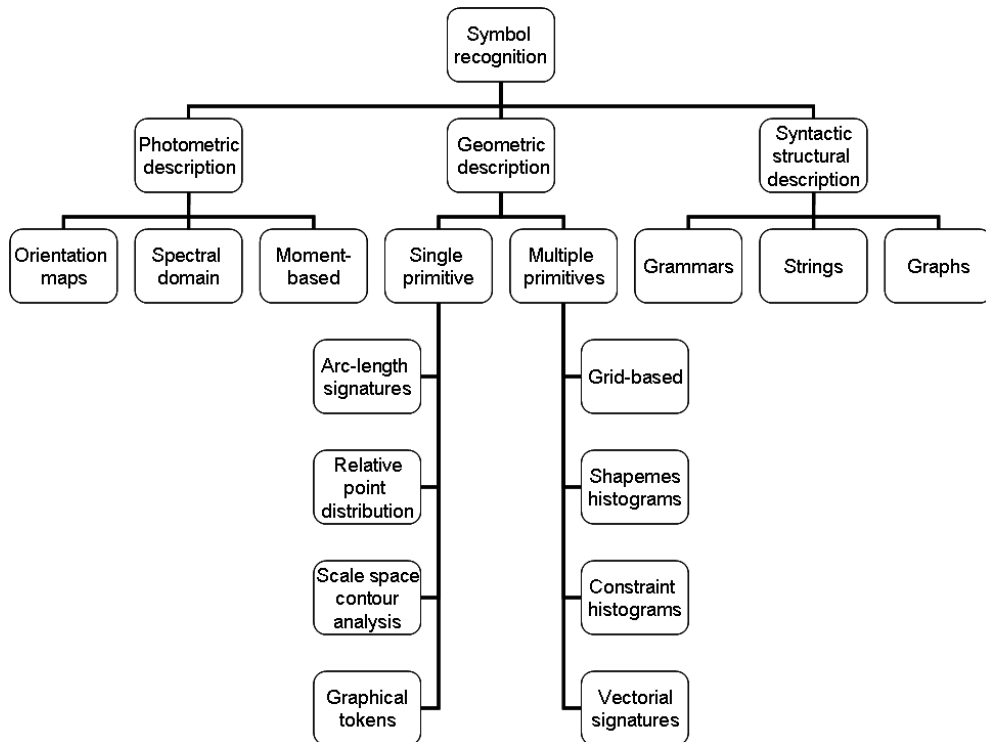
## 2.3 Symbol Description

Symbol Recognition is at heart of many of the Graphics Recognition applications. As pointed out in the state of the art review of Lladós et al. in [LVS02], because the wide range of different graphic documents, each of them containing its particular symbols, it is not easy to find a precise definition of what a symbol is. In the context of graphic-rich documents, symbols can be defined as the graphical entities which are meaningful in a specific domain and which are the minimum constituents that convey the information.

From this definition, we can see that there is a large variety of entities that can be considered symbols. Symbols can be from simple 2-D binary shapes composed of line segments as the case of the entities found on engineering or architectural documents to complex sets of gray-level or even color sub-shapes, as in the case of trademarks or logos.

This vast and heterogenous nature of symbols provoke that when facing the problem of describing and recognizing symbols, the proposed methods that can be found in the literature can rely on different primitives and visual cues to describe a symbol depending on the application to face. In the different reviews of description techniques each author propose a different taxonomy to cluster the methods following different criteria. For instance Mehtre et al. base their classification of shape description techniques presented in [MKL97] on whether the methods described the shapes from the previously extracted boundary of the objects or if they are region-based and use all the internal pixels to describe a shape. A more recent review of shape description was proposed by Zhang and Lu in [ZL04]. In that case, the authors add another

criteria to cluster the existing methods. Besides the contour-based or region-based nature of the systems, they propose to look if they are structural or global. This sub-class is based on whether the shape is represented as a whole or represented by segments/sections. Lladós et al. presented in [LVS02] a state of the art on symbol recognition techniques, clustering the existing methods not only by techniques nature but also by their intended applications. In our work, we propose to cluster the description techniques in three different categories clearly defined by the different visual cues which the methods aim to encode. In the first place, the *photometric description* of symbols describe the graphic objects in terms of the intensity of its pixels. It encodes thus several visual cues at the same time, as the shape of the object, the color, its texture, etc. On the other hand, a *geometric description* of symbols is only centered on the analysis of the shape as basic visual cue. Finally, the *syntactic and structural description* of symbols aims to represent the structure of a set of geometric primitives by defining relationships among them. Obviously, some methods in the literature are difficult to classify following this taxonomy since they use a combined strategy, or because they may be understood as belonging to different categories at the same time. We can find the whole hierarchy of the classification in the diagram shown in Fig. 2.3.



**Figure 2.3:** Classification of symbol representation and description techniques.

### 2.3.1 Photometric Description

The main interest of this kind of approaches to describe a graphical symbol is that the photometric description encode several visual cues at the same time. This kind of descriptions are suitable when we have to face with complex symbols that could be hardly described by only using the shape information. The problem of logo recognition is one of the application examples where a photometric description is suitable.

Bagdanov et al. present in [BBB07] a method focused on the detection of trademarks appearing in real images. In order to describe those symbols, in that work, the authors use the SIFT descriptor to match the trademark models against video frames. The SIFT descriptor, presented by Lowe in [Low99, Low04], basically characterize the local edge distribution around a given interest point by analyzing the intensity gradients in a patch surrounding the previously extracted keypoint having a certain scale and orientation. The feature descriptor is computed as a set of orientation histograms on a grid of  $4 \times 4$  neighborhoods. These histograms are computed relative to the keypoint orientation in order to achieve invariance to rotations. In addition, the magnitude and orientation of the gradients are computed from the Gaussian scale space image closest in scale to the keypoint's scale. The contribution of each pixel is weighted by the gradient magnitude, and by a Gaussian with a  $\sigma$  value proportional to the scale of the keypoint. Histograms contain 8 bins each, and each descriptor contains an array of 16 histograms around the keypoint. This leads to a SIFT feature vector with  $4 \times 4 \times 8 = 128$  elements. The SIFT descriptor has been widely used in Computer Vision for several applications as object recognition or robotics related problems as SLAM (Simultaneous Localization and Mapping). It could be very useful to describe complex graphic symbols as logos, but it loses effectiveness to represent simpler symbol designs.



**Figure 2.4:** Example of the generic Fourier descriptor. (a) An original logo image; (b) polar-raster sampled image plotted in Cartesian space; (c) Fourier spectra of (b); (reprinted from [ZL02]).

From another point of view, there is a family of photometric descriptors which base the symbol representation in the spectral domain. The analysis of images in the spectral domain overcome the problem of noise sensitivity. Within this family we can find some works focused on the application of such descriptors for symbol recognition. For instance, the Generic Fourier Descriptor (GFD) presented by Zhang and Lu in [ZL02] is used to recognize a set of trademarks. In this work, the raster images of logos (as the one shown in Fig. 2.4) are transformed from the Cartesian to the polar space and then a two-dimensional Fourier transform is applied to obtain the symbol description. Another example of spectral descriptors is the Fourier-Mellin transform. After a polar representation of the image, the angular parameter is expressed by the

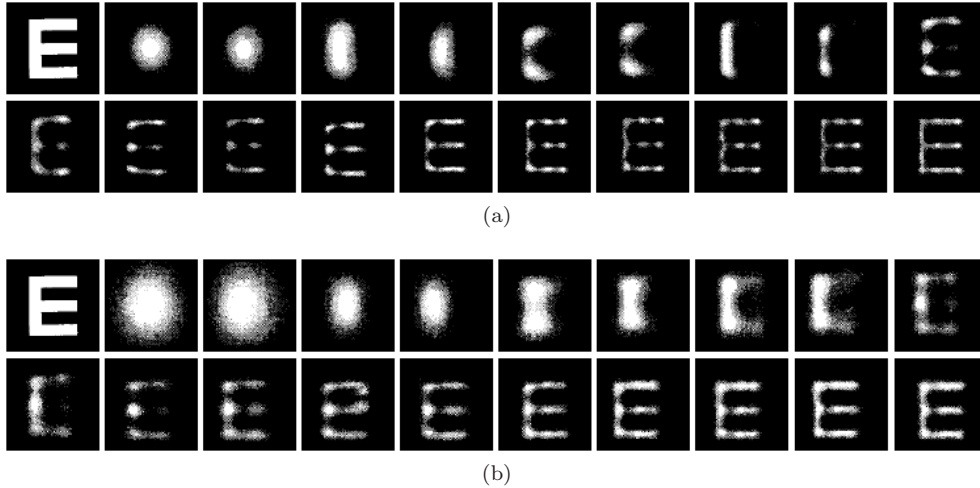
coefficients of the Fourier transform whereas the Mellin transform is applied to the radial parameter. In [AOC00], Adam et al. present a method allowing the classification of multi-oriented and multi-scaled characters appearing in technical documents. They base their set of invariants on the Fourier-Mellin transform, and are able to deal even with connected characters without a previous segmentation step.

Another family of photometric descriptors are the ones based on moments. As presented in Teh and Chin's review [TC88], moments have been utilized as pattern features in a number of applications to achieve invariant recognition of two-dimensional image patterns. Let us briefly review some of the moment-based descriptors which can be applied to the description of graphical symbols. Hu first introduced in [Hu62] a set of moment invariants by using nonlinear combinations of geometric moments. Those invariants have the properties of being invariant under image translation, scaling, and rotation. All these properties make Hu's invariants a suitable symbol descriptor. For instance, in [CKL93], Cheng et al. presented a symbol recognition system focused on the recognition of previously segmented electrical symbols. After a normalization, a symbol is described by a feature vector representing the six geometric moment invariants computed with respect to the symbol centroid. From the theory of orthogonal polynomials, Zernike moments have been introduced in [Tea80]. By projecting the symbol image to a vectorial space defined by a set of orthogonal polynomials named Zernike polynomials, the Zernike moments are obtained. Independent moment invariants are then easily constructed in an arbitrarily high order. As an application example, Khotanzad and Hong use in [KH90] the Zernike moments to describe a small set of upper case letters affected by several transformations and distortions. They show that Zernike features compare favorably with Hu's geometric moment invariants. In addition, the Zernike moments have the ability to reconstruct the graphical symbol from its description, which in some applications may result useful. Finally, another orthogonal moments are the Legendre moments, which make use of Legendre polynomials. In [CRM04], a descriptor based on an enhancement of the Legendre moments is presented to recognize Chinese characters ongoing several transformations. We can see an example of the ability to describe and reconstruct shapes of both Zernike and Legendre moments in Fig. 2.5. Usually, moment invariants are good descriptors since they are easy to compute and besides describing the intensity of the pixels, they also have a relation with geometric properties as center of gravity or axes of inertia.

We can found in Table 2.2 a summary of the state-of-the-art photometric symbol descriptors. Let us focus in the next section on the geometric descriptors.

### 2.3.2 Geometric Description

Geometric description techniques are primitive-based methods encoding basically the shape as the most important visual cue to describe graphical symbols. Symbols are broken down into lower level graphical primitives and are then described in terms of these primitives. The usual extracted primitives from the symbols are: contours, closed regions (loops), connected components, skeletons, etc. Within this family, we will differentiate between methods which are only able to describe one primitive, or



**Figure 2.5:** Example of the orthogonal moments for reconstruction of the letter *E* with from the second-order moments (a total of six moments) through up to 20th-order moment (a total of 231 moments). (a) Zernike moments; (b) Legendre moments; (reprinted from [TC88]).

methods which can be used to describe a whole symbol in terms of all these composing primitives.

### Single Primitive Description

A great variety of simple shape descriptors coping with geometric characteristics exist in the literature. Those descriptors are very easy to compute but are usually poorly discriminant. They can be used as a the first stage of the selection process among the shapes likely to be good candidates. Most of these simple descriptors are computed over a contour primitive, but can also be used to describe the skeleton or a region. Among the whole variety of simple shape descriptors we can cite a few. The area and the perimeter of the shape under analysis can be used as a coarse filter in applications where invariance to scale is not needed. The diameters of the circles with the same area or perimeter than the shape can also be used as simple descriptors, but, again the scale invariance is not achieved. Usually, for segmentation purposes, the orthogonal projections of the shape following the  $x$  and  $y$  axes are used to describe whether a shape is present or not. To have invariance to rotation, Feret's diameters are used. The Feret's diameters are the maximal and minimal orthogonal projections of the shape on a line. In order to achieve invariance to scale, usually some ratios among simple features are used. The eccentricity, aspect-ratio or Feret's ratio characterizes the dimensionality of the shape and is computed as the ratio between the maximum and minimum Feret's diameters. The area-perimeter ratio which is computed as  $4\pi A(X)/(P(X)^2)$  (being  $A(X)$  and  $P(X)$  the area and the perimeter of the shape  $X$  respectively) characterizes deviations of the shape from a circular form. For a disc it is equal to 1 while for all other shapes it is less than 1. The convexity ratio is



**Table 2.2:** State-of-the-art photometric symbol descriptors.

Name	Application to Symbol Description	Notes
SIFT	[BBB07]	Invariance to affine transforms and illumination changes. Set of orientation histograms computed over previously extracted keypoints.
GFD	[ZL02]	Applied to segmented symbols. 2D Fourier transform of the polar image.
Fourier-Mellin	[AOC00]	Can be applied to non-segmented symbols.
Hu's invariants	[CKL93]	Nonlinear combination of the lower order moments. Invariant to similarity transforms but not too much discriminative.
Zernike	[KH90]	Orthogonal moments. Allow a reconstruction of the shape. Robust to noise.
Legendre	[CRM04]	Orthogonal moments. Allow a reconstruction of the shape. Less robust than Zernike.

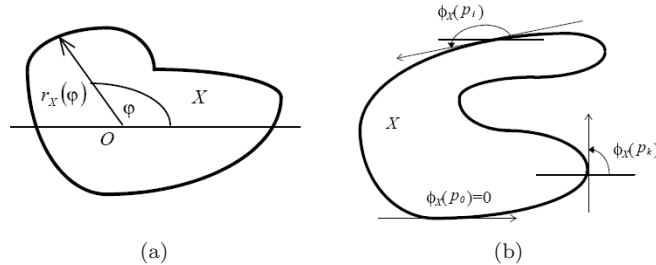
defined as the ratio between the area of the shape and the area of its convex-hull. It characterizes deviations from convexity.

Obviously, not all these simple descriptors have the desired invariance to rotation or scale, but most of them can be easily normalized to achieve such invariance. In addition to these simple descriptors, we can find in the literature several arc-length-based signatures. These signatures represent a shape by a one-dimensional function derived from its contour. From the variety of arc-length signatures, we can cite:

- The radius-vector function,  $r_x(\varphi)$  is the distance from a reference point  $O$  in the interior of the shape  $X$  to the contour in the direction of the  $\varphi$ -ray where  $0 \leq \varphi \leq 2\pi$ .
- The tangent-angle function  $\phi_x(p)$  characterizes the changes of direction of the points of the contour. The tangent angle at some point is measured relative to the tangent angle at the initial point.

We can find in Fig. 2.6 an illustration on how these contour signatures are computed. These signatures are also normalized to achieve invariance to translation and scale. Invariance to rotation is obtained by considering the function as periodic and analyzing a circular permutation of the signature. In addition to the high matching cost, contour signatures are sensitive to noise, and slight changes in the boundary can cause large errors in matching.

Another family of descriptors are the ones working at different scales. The scale space representation of a given shape is created by tracking the positions of interest points (protrusions and inflections) in a shape boundary filtered by a Gaussian filter of different width  $\sigma$ . As the  $\sigma$  value increases, little inflections are eliminated from the contour and the shape becomes more and more smooth. The inflection points

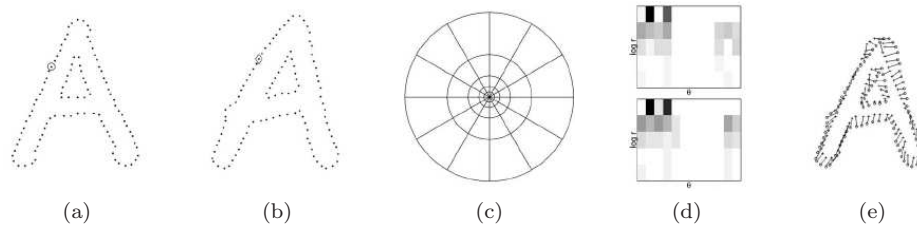


**Figure 2.6:** Computation of the arc-length-based signatures. (a) The radius-vector function; (b) the tangent-angle function; (reprinted from [Kin97]).

of the shape under analysis that remain present in the representation are expected to be significant object characteristics. Mokhtarian et al. present in [MAK96] the Curvature scale space (CSS) signature. The peaks from the curvature scale space contour map are extracted and used as to match two shapes under analysis. The CSS signature is tolerant noise and changes in the boundary since it bases its representation at different detail scales. However, since the matching process tries to find the best match between the contour branches of the CSS signature by applying shifts and different scales to achieve invariance to scale and rotation, the matching process proves to be very expensive.

As another example of geometric descriptor for single primitives, we can cite the Shape Context (SC) descriptor presented by Belongie et al. in [BMP02]. The shape context descriptor allow measuring shape similarity by recovering point correspondences between the two shapes. Given a set of points from a symbol (e.g. interest points extracted from a set of detected edge elements), the shape context captures the relative distribution of points in the plane relative to each point on the shape. Specifically, a histogram using log-polar coordinates which counts the number of points inside each bin is constructed. The descriptor offers a compact representation of the distribution of points relative to each selected point. An example of the shape context descriptor to match shapes can be seen in Fig. 2.7. Translational invariance come naturally to shape context. Scale invariance is obtained by normalizing all radial distances by the mean distance between all the point pairs in the shape. In order to provide rotation invariance in shape contexts, angles at each point are measured relative to the direction of the tangent at that point. Shape contexts are empirically demonstrated to be robust to deformations and noise. The shape context descriptor has been tested on different datasets. It has been used to recognize handwritten digits, to retrieve silhouettes by similarity and even to retrieve logos. In [MS05], Mikolajczyk and Schmid, proposed to enhance the shape context descriptor by weighting the point contribution to the histogram with the gradient magnitude, and to add orientation information to the histogram besides point locations. This enhancement makes that the shape context descriptor may also be classified into the photometric description techniques.

Describing graphical symbols by geometric descriptors coping with a single primi-

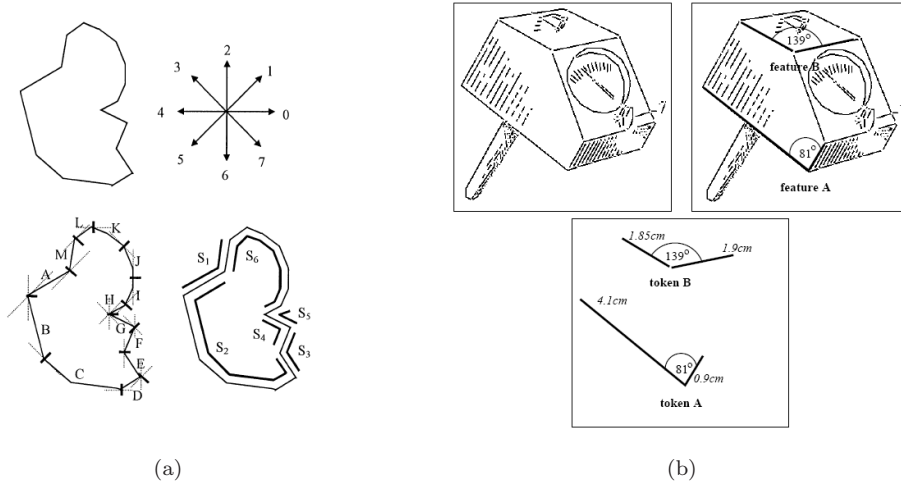


**Figure 2.7:** Example of the shape context descriptor for shape matching. (a) and (b) Original shapes to match with sampled edge points; (c) diagram of the log-polar histogram bins used in computing the shape contexts; (d) shape contexts obtained from one point of (a) and one point of (b); (e) correspondences found using bipartite matching for the two shapes (a) and (b); (reprinted from [BMP02]).

tive can be very useful when the symbols are non-isolated and other entities than the symbol may appear. These methods may also be very useful when the symbols can be affected by occlusions. As a last example of geometric description for single primitives we can mention what we call a description based on graphical tokens. Given a primitive (usually a contour or a skeleton of a symbol), it is partitioned into small graphical entities which can be described by very simple attributes. For example, Berretti et al. present in [BBP00] a system which partitions a contour into a set of tokens by partitioning the shape at minima of the curvature function. Each token  $\tau_i$  is described through the features  $(m_i, \theta_i)$  representing the curvature of the token and its orientation with respect to a reference system. Stein and Medioni propose in [SM92] to polygonally approximate a shape and then to partition this representation into sets of adjacent segments named super-segments. Those chains of consecutive segments are then represented by several attributes as the lengths, angles, orientation and eccentricity of the token. Nishida presents in [Nis02] a simpler yet effective approach. He proposes to apply a quantized-directional codes to the approximated contour and to characterize the tokens by a tuple representing the angular span and the direction of the segment (Fig. 2.8a). Lorenz and Monagan in [LM95], propose another set of simple tokens to represent graphical entities (Fig. 2.8b). To describe regular structures, parallel segments and junctions are taken as tokens and represented by attributes as length ratios and angles. To cope with irregular structures, chains of adjacent segments are taken as more complex tokens and are encoded by using the first six harmonics of the Fourier approximation of the segments chain. Those simple descriptions of symbols allow to cope with distorted symbols and with occlusions. However, as the symbol to recognize is composed by several tokens, usually, in order to match two different symbols, an algorithm of bipartite graph matching has to be used.

### Description of Several Primitives

A first example of geometric description of symbols which can handle several primitives at the same time can be grid-based method proposed by Lu and Sajjanhar in



**Figure 2.8:** Examples of graphical tokens-based description. The symbol primitives are partitioned into small graphical entities which are described by very simple attributes. (a) Quantized-directional codes (reprinted from [Nis02]); (b) length and angular tokens (reprinted from [LM95]).

[LS99]. This descriptor is inspired on the classic photometric descriptor known as Zoning [Bok92], but adapted to work with primitives, thus encoding geometric constraints among them. After a primitive extraction step, as contours or skeletons, the symbol is normalized for rotation and scale. The symbol is scaled into a fixed size rectangle, shifted to the upper left of this rectangle and rotated so the major axis  $F_{max}(X)$  of the symbol is horizontal. Then, the symbol is mapped on a grid of fixed cell size. Subsequently, the grid is scanned and a binary value is assigned to the cells depending on whether the number of points in the cell are greater than or less than a predetermined threshold. A unique binary number is obtained as the symbol descriptor. Although its simplicity, this kind of simple description is very dependent on the normalization step, and may not tolerate well slight distortions.

Inspired by the shape context descriptor described above, Mori et al. present in [MBM01] the shapeme histogram descriptor. This approach computes the shape context descriptor for all the interest points extracted from a symbol and uses vector quantization in the space of shape contexts. Vector quantization involves a clustering step of the shape context feature vectors and then represents these feature vectors by the index of the cluster that it belongs to. These clusters are called shapemes. By this means, we obtain a single descriptor for a symbol, no matter how many primitives it has. Each symbol is represented by a collection of shapemes by a histogram. The matching of two symbols is done by finding the nearest neighbors in the space of histograms of shapemes.

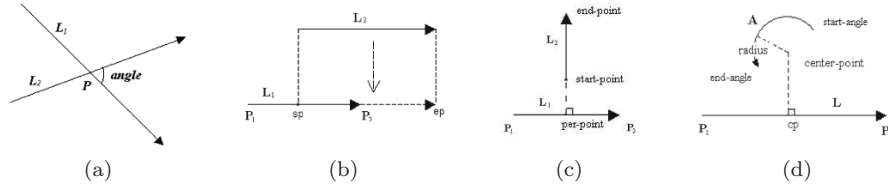
Yang presents in [Yan05] a symbol descriptor which as the shape context descriptor also captures the relative distribution of points from a symbol. However, the Pixel

Level Constraint Histogram (PLCH) descriptor encodes all the complete symbol no matter how many primitives (skeleton branches in that case) compose the symbol. The descriptor represents geometric constraints between every pair of points from the skeleton in reference to a third point. At each point from the symbol we compute a histogram depicting how the other points configure surrounding this point. Length ratios and angles are computed for each pair of points by using one point as reference. Using an equal bin partition and an accumulation space, two matrices (one for the length information and other for the angle information) of fixed dimensions are obtained. These matrices are used as the shape descriptor. The distance between two symbols is then defined as the sum of differences between the model and the test matrices. The tests, using the data from the symbol recognition contest held in the Fifth IAPR International Workshop on Graphics Recognition (*GREC 2003*) [VD04], give good recognition rates under diverse drawbacks such as degradation, distortion, rotation and scaling. As the descriptor focuses on geometric constraints among points, the rotation and scale-invariance are guaranteed. However this method can only work with segmented symbols and its computational complexity may become very high ( $\mathcal{O}(n^3)$ ), since all the triplet of pixels of the skeleton image are considered.

Another family of methods to describe graphical symbols using geometric information is the approaches based on vectorial signatures. This description technique is best suited for applications dealing with symbols arising from line-drawings as electronic diagrams or architectural floor plans where the primitives are of vectorial nature. The primitives representing the symbols are the segments extracted from a polygonal approximation of the contour or the skeleton of the symbol. The signatures are defined as a set of elementary features, containing intrinsically a discrimination potential. Huet and Hancock present in [HH99] a simple and compact histogram representation which combines geometrical and structural information for line-patterns. The attributes which are taken into account to build the signature are computed between pairs of line segments. These pairwise geometric attributes are the relative orientation between pairs of line segments, length ratios, distances and projections. This representation can be effectively used to index into a large database according to shape similarity. Based on the work by Etemadi et al. [ESM91], Dosch and Lladós present in [DL04] a method for symbol discrimination by using vectorial signatures. The method starts by a study on basic relationship between pairs of lines. Several main relations are thus enumerated: collinearity, parallelism and intersections. For each of these relations, some extensions are considered, like overlapping for parallelism, or the kind of intersection point. The number and the type of the relations found in a particular zone will form the signature. In [WZY07] Liu et al. present a similar approach. In that case, symbols are also represented by the occurrences of intersections among segments, parallelism and perpendicularities. Each of those features are attributed with certain parameters as angles, length ratios or directions. We can see an illustration of the considered geometric constraints which built the proposed signature in Fig. 2.9. These approaches present a very compact representation of graphical symbols having enough discriminative power to be used as a basis for spotting systems. However, the main drawback of such signatures is they may be very sensitive to slight changes of the primitives.

**Table 2.3:** State-of-the-art geometric symbol descriptors.

Name	Application to Symbol Description	Primitives	Notes
Simple ratios	[NSS02]	Single	Very simple to compute. Low discriminant power.
Arc-length Signatures	[TR03]	Single	Sensitives to slight boundary deformations.
CSS	[MAK96]	Single	Scale space analysis of a single contour. Tracking of the inflection points. Robust to boundary noise.
Shape Context	[BMP02]	Single	Compact representation of distribution of points relative to a reference point. Invariant to similarity transforms.
Graphical Tokens	[SM92], [LM95], [BBP00], [Nis02]	Single	Partition of graphical primitives into simpler entities. Those tokens are described by simple geometric attributes. Very useful in case of occlusions.
Grid-based	[LS99]	Multiple	Provide a binary description of the symbol's shape. Very simple representation but dependent on a previous normalization step to achieve invariance to similarity transforms.
Shapeme	[MBM01]	Multiple	Vector quantization on the shape contexts of a symbol. Obtains a single feature vector for a symbol.
PLCH	[Yan05]	Multiple	Represents geometric constraints between every pair of points from the skeleton in reference to a third point. Invariant to similarity transforms and robust to noise.
Vector Signatures	[HH99], [DL04], [WZY07]	Multiple	Compact representation of vectorial symbols. Invariant to similarity transforms but very sensitive to noise at vector level.



**Figure 2.9:** Geometric constraints taken as features to build a vectorial signature. (a) Intersection of segments; (b) parallelism; (c) perpendicularity; (d) constraints regarding arcs and circles; (reprinted from [WZY07]).

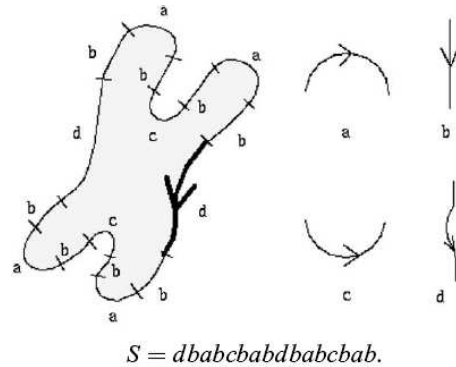
We can find in Table 2.3 a summary of the state-of-the-art geometric symbol descriptors. Let us focus in the next section on the syntactic and structural description family.

### 2.3.3 Syntactic and Structural Description

Finally, the syntactic and structural description approaches are focused on the structure of the symbol under analysis. Symbols are first decomposed in basic primitives which may be represented by any description presented above. The syntactic and structural descriptors aim then to define the relationships among those primitives. Whereas in syntactic description we offer a rule based description of the symbols, in structural description, the recognition of a given symbol is performed by comparing its symbolic representation against a predefined model of the symbol under analysis.

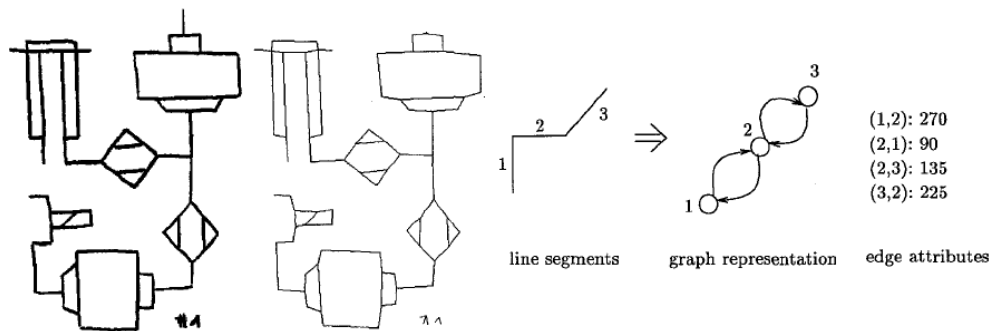
As introduced by Fu in [Fu74], the syntactic approach to pattern recognition provides a capability for describing a large set of complex patterns by using small sets of simple pattern primitives and of grammatical rules. The application of grammars to the problem of symbol description has been widely used over the years since this application is especially well suited to model description through syntactic rules. Terminal elements of the grammars will correspond to the basic primitives composing a graphical symbol and the non-terminal elements will describe the production rules. Syntactic analyzers are built from these grammars in order to group the basic primitives following the production rules and in order to finally recognize graphical symbols. We can find in the literature many kinds of grammars, from the linear ones as the PDL-grammars presented in [Sha69] or the adjacency grammars used by Mas et al. in [MJS08], to more complex grammatical structures as the graph grammars presented by Bunke in [Bun82]. However, the syntactic approaches present the problem that rule based description schemes are very affected by noisy data. Since the recognition of the symbols is done in terms of the rules of composition of primitives, slight perturbations on the terminal elements may provoke that the production rules can not be applied, and then the symbol can not be recognized.

As a first example of structural descriptors, we focus on the string representation of symbols. Symbols are represented by an ordered set of primitives which are encoded as a one-dimensional string. These descriptors codify in which order we expect to find



**Figure 2.10:** String representation of closed contours (reprinted from [Fu74]).

the primitives that compose a given symbol. For instance, Fu proposed in [Fu74] to represent chromosome shapes by a chain of boundary segments forming codewords (see Fig. 2.10). In this kind of approaches, the similarity measure between two string representations of a symbol will be computed by using the string edit operations proposed by Wagner and Fischer in [WF74]. Tsay and Tsai in [TT89] or Wolfson in [Wol90] use the string edit operations applied to the recognition of polygons. Another commonly used method for transforming shapes into one-dimensional strings is the use of the chain codes presented by Freeman in [Fre61]. In that case, shapes are described by a sequence of unit-size line segments with a given set of orientations.



**Figure 2.11:** Attribute relational graph for symbol description. Nodes represent line segments and are attributed by their length while edges represent an adjacency relationship and are attributed by the formed angle between the two segments (reprinted from [MB96]).

The most common structural representation of symbols is the Attribute Relational Graph (ARG). Graphical primitives are extracted from the symbols and a graph is built representing the structural relationships among those primitives. Messmer and Bunke proposed in [MB96] a symbol recognition framework based on an ARG representation. The primitives taken into account are the line segments that arise from



a polygonal approximation of an engineering drawing. An ARG is constructed by taking the segments as the nodes of the graph and the edges represent that two segments are adjacent. Both nodes and edges have associated attributes. The length of the segment is stored in the nodes whereas the angle between two segments is stored in the corresponding edge. We can see an example of such graphs in Fig. 2.11. As another example, Lladós et al. present in [LMV01] another approach by using an ARG. In this case, the authors use higher level primitives than segments. Closed regions are identified from the symbol prototype and are used as the nodes of the graph. The nodes of two adjacent regions of the symbol are linked through an edge of the graph. The nodes of the region graph are attributed by the string representation of the boundary of the region. The edges are attributed by the shared string of the two adjacent regions. Length and orientation attributes are also added to this graph representation. The use of attributed graphs for representing symbols has the main advantage that we can have a very complete description of the symbols. Primitives can be described by photometric or geometric descriptors and these descriptions can be the attributes of the nodes. In addition, the relationships among those primitives are represented by the edges, codifying structural information. In the general case of graph-based symbol description, the recognition of the symbols has to be done by the use of a sub-graph isomorphism. For each symbol, a prototype of its ideal shape is built as an attributed graph. An input symbol is recognized by means of the matching between the representation of the symbol and the symbol prototype. The main drawback of such powerful representation is that the sub-graph isomorphism algorithms are extremely expensive in computation time since they it is an NP-complete problem as stated in [GJ79].

In order to avoid the complex step of matching two graph representations, several approaches can be found. For instance, Franco et al. [FOL04] use the minimum spanning tree as a simplification of a graph. By connecting all the pixels constituting the object under the constraint to define the shortest path the shape topology is captured. A template matching algorithm which uses minimum spanning trees as symbol representation is presented.

We can find in Table 2.4 a summary of the state-of-the-art syntactic and structural symbol description approaches. Let us focus in the next section on the problem of how to organize the descriptors to provide an efficient access to the information.

## 2.4 Descriptors Organization and Access

In the problem of recognizing graphics appearing within document images, the basic paradigm involves a matching step between the features extracted from the model graphical symbol and the features extracted from the document images. To be able to recognize and to locate elements in documents, the descriptors should be stored in a data structure and clustered by similarity. Once the user formulates a query in terms of a symbol, we have to retrieve by an efficient mechanism the locations within the document images where similar symbols to the queried one appear. As pointed out by Califano and Mohan in [CM94], since all feature combinations may have to be

**Table 2.4:** State-of-the-art syntactic and structural symbol descriptors.

Name	Application to Symbol Description	Notes
Grammars	[Sha69], [MJS08], [Bun82]	Rule-based description. Performance highly affected by noise in primitives.
Strings	[Fre61], [Wol90]	One-dimensional representation of symbols. The structural information is the order followed by the primitives. Similarity measure defined in terms of edit operations.
Graph-based	[MB96], [LMV01], [FOL04]	Prototype-based description. Very powerful tool to describe symbols. Extremely expensive in computation time.

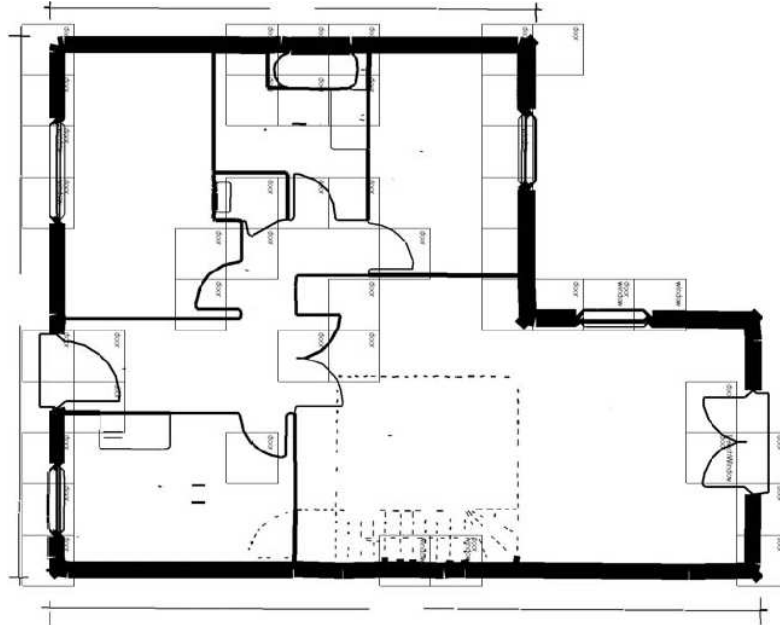
explored, brute-force matching is equivalent to exponential search. In focused retrieval applications with large databases, these costs may become unaffordable. The choice of a data structure providing efficient access to the descriptors is crucial to the final performance of the system. Let us briefly overview in this section which approaches can be found in the literature.

### 2.4.1 Sequential Access

In the first place, we can find a family of spotting methods which work with a sequential access to the symbol descriptors, i.e. the descriptors are stored in a list or a similar sequential data structure. In those methods, regions of interest where the symbols are likely to be found are extracted by some means. A symbol descriptor is computed afterwards for each of these regions of interest. When the user wants to retrieve the zones of the image collection having similar description than the queried symbol, one-to-one matching has to be computed in a sequential way. The complexity of searching similar descriptors by using data structures with sequential access is  $\mathcal{O}(N)$ , with  $N$  being the number of segmented regions of interest for the entire collection. Even if the use of sequential structures has several important drawbacks, it is the most used approaches in the literature dealing with symbol spotting.

As application examples we can mention the work by Tabbone et al. [TWT03], where an algorithm of text/graphics separation is applied in order to separate symbols from the text and the background of technical documents. Each connected component is represented by a photometric descriptor computed over the area defined by the bounding box of the connected component. Subsequently, each descriptor is sequentially compared against all the model descriptors and if the distance between two descriptors is small enough, the interest region is labelled as containing a certain symbol. Such approaches are obviously very dependent on the segmentation phase.

To avoid the dependence on a previous segmentation method, some approaches as [MR00] or [DL04] use a grid partition of the document or a sliding window approach to compute the descriptors all over the documents. As in the previous method, each



**Figure 2.12:** Symbol spotting results by using sliding windows. (reprinted from [DL04]).

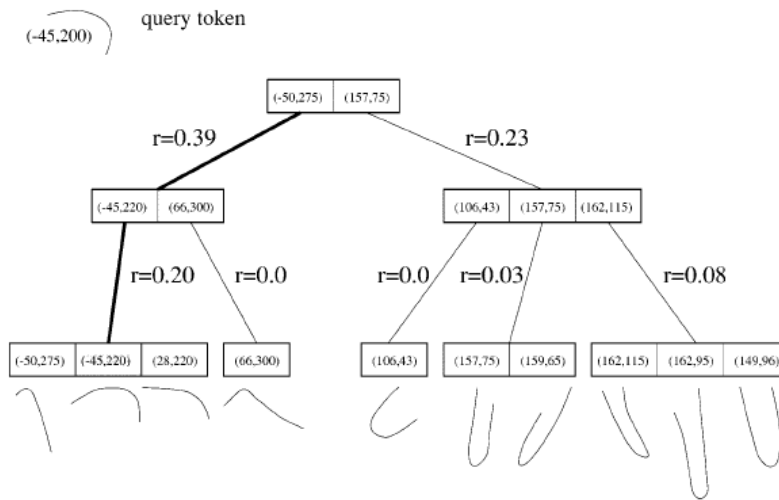
descriptor arising from a window is sequentially compared against all the model descriptors. If the descriptor matches one of the model's descriptions, the window is labelled as containing a certain symbol. We can see in Fig. 2.12 the result of spotting symbols by using a sliding window approach. In those cases, the amount of descriptors to compute and the number of distances among descriptors is dramatically increased.

The spotting methodologies which use a sequential access to the descriptors present several drawbacks. On the one hand the access to the descriptors is not efficient, leading to exponential search when all the combinations of descriptors have to be tested. On the other hand, they are hardly scalable to a large number of documents to consider or a larger number of symbol models.

## 2.4.2 Hierarchical Organization

In order to provide a more efficient search by similarity in the description space, we can find a number of methods which work with a hierarchical representation of descriptors. These methods use data structures such as trees, dendrograms, lattices or graphs. Hierarchical data structures are based on the principle of recursive decomposition. They are attractive because they are compact and depending on the nature of the data they save space as well as time and also facilitate operations such as search. The search by similarity is done by a traversal of the data structure that usually can be done in logarithmic time with respect of the number of clusters involved in the structure.

Decades of research in the data mining field have resulted in a great variety of hierarchical data structures that aim retrieval by similarity. The interested reader is referred to Gaede and Günther's [GG98] comprehensive survey on multidimensional access methods. As examples, we can for instance cite the *K-D-B-trees* [Rob81], which partition the universe and associate disjoint subspaces with tree nodes in the same level. The *LSD-trees* [HSW89], guarantee that the produced data structure is in addition a balanced tree being much more efficient in the traversal step.



**Figure 2.13:** Hierarchical organization of descriptors by using an M-tree structure. Traversing the structure allows a nearest neighbor search in logarithmic time (reprinted from [BBP00]).

As application examples we can mention Lowe's work [Low99] which uses a *k-d* tree structure [Ben75] to organize the instances of the SIFT descriptor. Berretti et. al. [BBP00] divide in their work contour primitives into diverse tokens which are subsequently stored in an *M-tree* indexing structure [CPZ97]. We can see an example of the obtained *M-tree* structure in Fig. 2.13. From another point of view, Punitha and Guru [PG08] use a *B-tree* [BM76] to represent the spatial organization of previously recognized primitives.

Within the symbol recognition field, we can for instance cite the work of Zuwala and Tabbone [ZT06], which use a dendrogram to hierarchically represent the primitives which compound the graphical symbols present in technical documents. The dendrograms are tree structures which are used to illustrate the arrangement of the clusters produced by a clustering algorithm. Following a similar idea, Guillas et al. use in [GBO06] a concept lattice to hierarchically organize symbol descriptors. The navigation of the concept lattice is done in a similar way than a decision tree.

Finally, graphs can also be used to store information and provide some kind of hierarchical organization of data. For example, in [WZY07] graphs representing symbols are reduced to a spanning tree which are posteriorly traversed to identify symbols

within technical documents. Messmer and Bunke propose in [MB96] to build a network of common subgraphs patterns. This network is then traversed by applying graph isomorphisms. Ah-Soon and Tombre propose in [AST01] to build a graph of geometric constraints which are then used to recognize symbols appearing in line-drawings.

Structures aiming hierarchical organization of information based on the principle of recursive decomposition can be very useful for the specific application of symbol spotting. Thousands of feature vectors may arise from the documents and in the querying step a similarity search has to be done in an efficient way. However, trees can grow arbitrarily deep or wide and usually the efficiency of the traversal step is very dependent on the tree topology. Balancing algorithms can be applied to maintain the structure usability but are very costly to apply.

### 2.4.3 Prototype-based Search

Another kind of techniques conceived to avoid brute force matching are the ones based on a prototype-based search. Although these kind of approaches are not very common in the data mining field, in the particular case of pattern recognition they have been used in several applications to provide efficient access to clusters of patterns by similarity.

In prototype-based search, we are given a set of distorted samples of the same pattern and want to infer a representative model. In this context, the median concept turns out to be very useful. Given a set of similar patterns, a representative of this set having the smallest sum of distances to all the patterns in the set can be computed. If we want to retrieve similar patterns than a certain query, by using a prototype-based search the retrieval by similarity is done efficiently since only the distances between the query pattern and the representative of a cluster of similar patterns has to be computed. The fact of avoiding a brute-force distance computation allows a fast pattern retrieval by similarity.

The computation of the median of a given set is straightforward when the feature vectors used as descriptors are numeric, however it is more complex to extend this concept to the symbolic domain. We can find in the literature several approaches which aim to compute the median of a set of symbolic representations. Recently, Ferrer et al. present in [FVS09] a method to compute the median of a set of graphs. Jiang et al. review in [JMB01] the possible applications of the median graphs. Obviously, due to the extreme cost of computing the matching between graphs, prototype-based search are very efficient methods to provide access the information.

### 2.4.4 Hashing Approaches

Finally, the other widely used data structure is the look-up table which aims to access to the information immediately without any structure traversal step. For instance, grid files [NHS84] are a bucket method which superposes a  $n$ -dimensional grid on the universe and a directory (build with the definition of a hash function) associates

cells with bucket's indices. When using hashing techniques, the search operations can theoretically reach  $\mathcal{O}(1)$  time with well chosen values and hashes. To perform a search by similarity by using such structures, the hash function must be seen as a clustering function which can assign the same index to similar shapes.

Multidimensional hashing methods partition the space into hypercubes of known size and group all the records contained in the same hypercube into a bucket. To identify the bucket to which a certain query belongs, the index of the query is automatically computed using a hash function (performing one-dimensional partitions) and the resulting bucket is obtained. In the specific case of shape retrieval, given a primitive, a feature vector is computed using one of the presented descriptors. A hash function establish a quantization criteria to apply to each dimension of the feature vector to limit the index parameters to a finite number of discrete values.

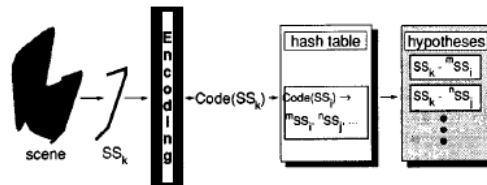


Figure 2.14: Using hash tables for descriptor organization [SM92]).

Califano and Mohan use in [CM94] a look-up table mechanism to replace the runtime computation of one-to-one matching with a simpler lookup operation. The speed gain can be significant since retrieving a value from this structure is faster than traversing a tree structure and much faster than a sequential comparison. Stein and Medioni propose in [SM92] a similar approach to retrieve by similarity subparts of a shape. As we can appreciate in Fig. 2.14, the subparts composing a shape are encoded by a hash function resulting in the bucket index of the indexing structure. The same hash function is applied in the querying step, and all the instances of similar primitives stored in the bucket identified by the resulting index are retrieved.

Another classical example of the use of such structures is geometric hashing method introduced by Lamdan and Wolfson in [LW88]. The geometric hashing approach aims to match geometric features against a database of such features. Geometric hashing encode the model information in a pre-processing step and store it in a hash table. During the recognition phase, the method accesses the previously constructed hash table, indexing the geometric features extracted from the scene for matching with candidate models. By the use of geometric hashing the search of all models and their features is obviated. The simplest features one can use are the points coordinates forming a shape. Normalizing the shape scaling, rotating and translating it taking as basis a reference vector will give the position of hash table bins to store the shape entry. The major disadvantage of the method is that the same subset has to be chosen for the model image than the previously acquired images.

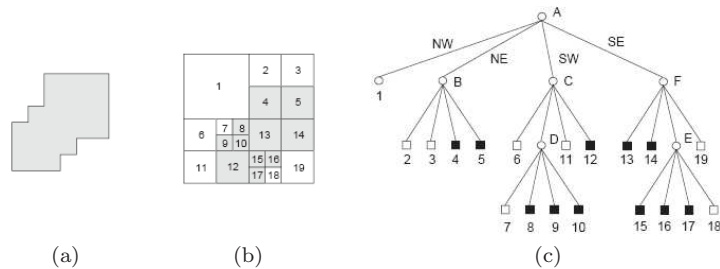
In the data mining field, the main drawback of hashing techniques are the collisions. Given two different entries to store in the database, the database system has

to guarantee that the hash function used to index such entries do not assign the same key-index to them. If such thing happens, it would provoke the data to be lost. To overcome this problem expensive re-hashing algorithms have to be applied once a collision is detected. In the specific case of shape retrieval by similarity, collisions are not a problem but the basis of the indexing strategy. Given two similar (but not equal) primitives, they are represented by a compact feature vector. Hopefully, if the two primitives have similar shape, the two feature vectors will be two nearby points in a  $n$ -dimensional space. The hash function has to guarantee that both points fall into the same bucket to have stored in a single entry all the similar primitives.

### 2.4.5 Spatial Access Methods

So far, we have only focused on the *point access methods*, i.e. indexing structures which can only handle information represented by  $n$ -dimensional points. However, another family of indexing structures exist, aiming to manage polygons and high dimensional polyhedra, which is named the *spatial access methods*.

Point access methods can not be directly applicable to databases containing objects with spatial extension. Spatial data consists of objects made up of points, lines, regions, rectangles, surfaces, volumes, etc. The spatial access methods can be seen as a joint shape description and feature organization. The spatial access methods can handle such data and are finding increasing use in applications in urban planning, geographic information systems (GIS), etc. The interested reader is referred to the book on spatial access methods by Samet [Sam90].



**Figure 2.15:** Quadtree representation of a shape. (a) Sample region; (b) its maximal blocks from the array representation; (d) its Quadtree representation; (reprinted from [Sam90]).

In the spatial access methods class we can find the Quadtree structures, illustrated in Fig. 2.15, which divide a two dimensional space by recursively partitioning it into four quadrants or regions. The  $R$ -trees [Gut84], which represents a hierarchy of nested  $n$ -dimensional intervals storing minimum bounding boxes in leaf nodes. An improved structure are the  $R^*$ -trees [BKS90], which try to minimize the overlap between bucket regions, minimize the perimeter of the leaf regions and maximize the storage utilization.  $SS$ -trees [WJ96] use spheres instead of rectangular regions,  $SR$ -trees [KS97] combine both  $R^*$ -trees and  $SS$ -trees and store the intersection between

spheres and rectangles. Finally *P*-trees [Jag90] manage polygon-shape containers instead of intervals.

Even if such data structures may be very helpful in the context of symbol spotting, our work is focused on the use of symbol descriptors as a basis for data representation. In this thesis we will only work with point access methods organizing feature vectors describing graphic objects.

### 2.4.6 Curse of Dimensionality

The curse of dimensionality is a term coined by Bellman [Bel57] to describe the problem caused by the addition of extra dimensions to a space which provokes an exponential increase in volume. This volume increase usually results in a performance degradation. Weber et al. argue in [WSB98] that indexing techniques reduce to sequential search for ten or higher dimensions. However, in the case of symbol spotting, it is difficult that we suffer this problem.

The study by Korn et al. showed in [KPF01] that the feeling that nearest neighbor search is hopeless in high dimensions due to the curse of dimensionality may be overpessimistic. Real data sets disobey the assumption that the data is uniformly distributed since they typically are skewed and exhibit intrinsic dimensionalities that are much lower than their embedding dimension due to subtle dependencies between attributes.

In addition, for spotting purposes high-dimensional descriptors are not the best suited. Usually, high-dimensional descriptors are more robust to noise and transforms and are more reliable than simpler ones. Obviously, in the case of isolated symbol recognition, it is desirable to have this robustness and accuracy despite the volume explosion. However, in the case of symbol spotting, we are more interested in the efficiency of the retrieval by similarity step than the final accuracy of the recognition task. Usually compact representations are best suited for spotting purposes despite the discriminative power loss. Therefore, for spotting applications low-dimensional descriptors are usually chosen.

## 2.5 Hypotheses Validation

In the retrieval stage, the result of the traversal of the data structure is a set primitives similar to the ones which compounds the searched symbol. The document locations accumulating several primitives are hypothetic locations where it is likely to found the symbol under a certain pose. These hypotheses have to be validated in a final phase of the retrieval process.

We can find two different approaches to face the hypotheses validation problem. The first one focuses on the feature vectors arising from the description phase, and if this descriptors can or can not be a correct symbol. On the other hand, there are some other approaches which will focus on geometric and spatial relationships among primitives to finally validate if a zone is likely to contain a certain symbol.



The first family usually be focused on a statistical analysis of the features, whereas the second family is based on a voting strategy and on the accumulation of little evidences to solve the pose estimation problem.

### 2.5.1 Statistical Validation

One of the most common approaches to validate if a zone contains or not a symbol is based on the study of statistical measures with the help of a probabilistic classifier. The features obtained by a symbol descriptor are seen as points in a  $n$ -dimensional space and the classifiers which are previously trained with a supervised learning step, are able to identify which is the class the symbols belongs to. Within this family of approaches, a lot of different classifiers are used to the problem of classifying symbol instances. One of the most common approaches are the Bayesian classifiers, which for instance are used in [TÓD90] to recognize handwritten symbols. The Support Vector Machines (SVN), which are used in [HN04] to recognize sketched symbols described by Zernike moments. The modelling of neural networks as classifiers is also another option, as the method applied to musical symbols recognition presented in [SCC02].

From another point of view, there are some other validation methods which are based on the bag-of-words (BoW) model. These approaches use a frequency vector of features to decide whether a region can contain a symbol or not. The principle of the bag-of-words model relies on a document representation as a vector of features where each feature has an assigned frequency. Bag-of-words approaches have been used over the years for text document classification, as for instance in [ADW94], but the analogy to the bag-of-visual-words can be derived to classify images as in [SRE05]. The approaches based on bag-of-words models have the advantage that the hypotheses validation is done without any spatial information being very simple to implement a quick to use. In [BHA05] Barbu et al. present a method which applies the bag-of-words model to the symbol recognition problem. However, instead of building the vocabulary from a photometric description of the symbols, they propose a bag-of-graphs model where structural descriptors act as words.

Although high recognition rates can be obtained with statistical validation the main drawback these approaches present is that they are dependent on a learning stage. In order to have a good performance, we need a lot of training samples to feed the classifier. In the particular case of symbol spotting we can not use such a priori knowledge nor have an immense sample set of every symbol we want to query. Since spotting approaches are intended to be a query-by-example, usually a statistical validation for symbol spotting approaches is out of question.

### 2.5.2 Voting Strategies and Alignment

On the other hand, there exist other validation approaches which do not focus on the features arising from the description phase, but on testing if the spatial organization of features in a certain location agrees with the expected topology.

Usually, these approaches are focused on some kind of voting strategy as the Generalized Hough Transform (GHT) presented by Ballard in [Bal81]. The problem of

finding a query object inside an image is transformed into the problem identifying accumulation points in a parameter space. A transformation function maps spatially sparse shapes in the image space to compact regions in the parameter space. The parameter space is divided into buckets. Then, every query descriptor votes in this space according to transformations provided from the matchings with the database descriptors. A high density of votes in a bucket indicates a high probability of detecting the object with its corresponding transformations. For example, in [LG96] Lamiroy and Gros extended the geometric hashing method with a Hough-like voting strategy to validate the hypotheses in an object recognition application.

Depending on the nature of the symbols to retrieve, the hypotheses validation can be seen as a registration problem. Some approaches validate the coherence of the symbol retrieval using geometric alignment techniques that put in correspondence the original information of the query symbol with the information of the retrieved zones of interest. Some affine transformations can be inferred to align the information of the model object and the retrieved results. Classical techniques use spatial distance between the contours of both images, but other characteristics as the gradient information can be used as shown in [HU87]. Other techniques as B-splines or snakes can also be used for elastic shape matching, as in the approach presented by Del Bimbo and Pala in [BP97]. However, these alignment techniques based on deformable template matching are hardly applicable to symbols where the extracted primitives are not their contour.

## 2.6 Conclusions and Discussion

In order to summarize this state-of-the-art chapter, let us recall which are the most suitable methods to apply to the symbol spotting problem in each of the three levels.

Regarding the description phase, we should select one of the photometric descriptors if our application has to deal with complex symbols as logos, which may have information in several visual cues as color, shape, texture, etc. since the descriptors from this family have the ability to encode all this information. For simpler symbols designs, as the ones appearing in line-drawings, geometric descriptors are the most suitable methods to compactly represent the primitives' shape. We should carefully select the appropriate descriptor depending on if the symbols can be represented in an accurate fashion by a single primitive as the contour or, on the other hand, if we need several primitives to describe a single symbol instance. Finally, syntactic and structural descriptors are a powerful tool in the context of symbol description, but present some drawbacks in the context of symbol spotting. Syntactic approaches are very sensitive to noise and need a definition of the rule set, which is a strong burden for the approach flexibility. Structural techniques should be carefully applied due to the strong time constraints which have to face spotting approaches.

Another factor which is important to take into account is that for spotting purposes, it is not essential to look for the descriptor which provides the more accurate description and the better recognition results. Usually, a simpler description able

to coarsely discriminate symbols would be a better choice rather than a complex descriptor with high accuracy.

Once the suitable description technique has been chosen, we have to think how we should organize all the information arising from the document collection to provide an efficient search access. Obviously, the approaches which follow a sequential access to the descriptors, are simple to design, but its application to large databases is not realistic. Indexing mechanisms, whether from the hierarchical category or the hashing techniques should be adopted to access to the data. We believe that in the particular case of spotting, hashing techniques are a better choice, since we avoid traversal steps and costly balancing algorithms. However, a comparative study should be further described in order to really determine which are the strengths and the weaknesses of both approaches in this application.

We strongly believe that the use of low-dimensional descriptors is highly recommended in spotting applications in order to avoid the curse of dimensionality. As we previously mentioned, the use of simpler descriptors will cause an accuracy loss and an increase of false positives, however these two phenomena are not a limitation in the case of spotting, since high recognition rates are not needed.

Finally, regarding the hypotheses validation step, our feeling is that voting strategies are more recommendable than statistical validation schemes for a spotting application. Voting strategies do not need a learning stage which is an advantage for scalability reasons. In addition, some voting schemes as the Hough transform or some works inspired by the geometric hashing are formulated in terms of spatial organization of primitives. The use of a geometric descriptor and such voting schemes provide a combined geometrical definition and structural validation.



## Part I

# On the Use of Photometric Descriptors for Symbol Spotting



# Chapter 3

## Symbol Spotting for Document Categorization

---

In this chapter we present a method for spotting symbols in document images by using a photometric description of symbols. As running example we present an application of logo spotting. The presented method use a bag-of-words model in order to perform a categorization of document images such as invoices or receipts. The hypotheses validation is done in terms of spatial coherence by the use of a Hough-like voting scheme. Experiments which demonstrate the effectiveness of this system on a large set of real data are presented at the end of the chapter.

---

### 3.1 Introduction and Related Work

The problem of locating symbols within document images can be seen as a particular case of the object recognition problem from the Computer Vision field. In this first part of the thesis, we want to test if some well-known techniques from the object recognition field can be applied to the specific case of symbol spotting. Instead of a focused retrieval application, we propose an application of detecting logos in document images such as invoices, receipts, etc. for document categorization.

Companies deal with large amounts of paper documents in daily workflows. Incoming mail is received and has to be forwarded to the correspondent addressee. A study on the invoice processing in several German companies [KAD04] revealed that in average the cost of manually process (opening, sorting, internal delivery, data typing, archiving) these incoming documents is about 9€ per invoice. These costs represent an important quantity of money if we consider the amount of documents received by a big company at the end of the day.

Several systems intended to automatically process incoming documents have been designed over the years. As an example, Viola et al. presented in [VRL04] a system aiming to automatically enrout incoming faxes to the correspondent recipient.

However, most of the existing systems only process typewritten information making the assumption that the recipient information is printed in the document image. In many cases, graphic elements present in the documents convey a lot of important information. For instance, if a company receives a document containing the logo of a bank, usually this document should be forwarded to the accounting department, whereas if the document contains the logo of a computer supplier, it is quite probable that the document should be addressed to the IT department. The categorization of documents may also have other applications besides the automatic rerouting. For instance it is helpful organize documents and providing efficient access to all the documents coming from a certain supplier. The recognition of such graphic elements can help to introduce contextual information to overcome the semantic gap between the simple recognition of characters and the derived actions to perform brought by the document understanding. In this chapter we use the presence of graphical symbols (logos) to categorize the class of the incoming documents.

Many contributions exist in the Graphics Recognition literature that deal with logo recognition and retrieval, e.g. the recent work on trademark recognition from Wei et al. [WLC09]. However they just focus on isolated or pre-segmented graphic images which are affected by synthetic noise and deformation sources. As noted in [VDF08], one of the big challenges for the next years for the Graphics Recognition community is the localization/recognition of graphic symbols appearing in complete documents without any previous segmentation. To our best knowledge, in the literature, only Zhu and Doerman addressed in [ZD07] the problem of logo spotting by means of a cascade of classifiers. We propose in this chapter a method which aims to categorize documents and to detect graphical logos in a single step. The main contribution of this chapter is the use of well-known strategies of the Computer Vision field to this particular kind of images. State-of-the-art photometric descriptors are used to characterize graphical symbols and a bag-of-visual-words approach is presented to categorize the documents. This kind of approaches are commonly used in object recognition as in [SRE05] and image classification applications. To our best knowledge, very few works have been proposed in the literature using this kind of descriptors to the domain of document images. Due to the binary nature of the document images, usually, photometric descriptors are not well suited for document analysis applications. However, the fact that the recent proposed descriptors work at several scales and blur the image, makes possible its use on binary images. The bag-of-visual-words is an analogy to the Computer Vision domain of the classic bag-of-words model, where a text is represented by an unordered set of words. In that case, an image is represented by collection of image patches. By the combination of photometric descriptors and a bag-of-visual-words model, we propose a segmentation-free recognition method which do not rely on a learning step but uses a single instance of logo models so as to benefit the scalability of the method.

The presented application, however, differs a little bit of the main objective of this thesis. By means of spotting graphical elements, we want to build an indexing mechanism which aims to query a large collection of documents and perform focused retrieval tasks. Whereas object recognition methods rely on an off-line learning of the models to search for, indexing methods should be queried by example without any



training step. Object recognition methods have an off-line stage where a classifier is trained with several examples of the features extracted from the models to recognize and, usually, a set of negative examples to be considered as non-objects. Once the classifier is trained, the images are given as input of the system, and the regions of interest of this image where any of the trained models appear are retrieved. On the other hand, indexing mechanisms have also an off-line step in which the documents are acquired and some features are extracted and organized, but, the input of the system is one single instance of the model to retrieve. The main difference of such applications stems for the training stage and the nature of the input, even if both applications can perform spotting and focused retrieval tasks.

The remainder of this chapter is structured as follows: the next section presents an overview of the proposed method. In section 3.3, we detail the detection procedure from the feature extraction to the bag-of-words model used to categorize the documents. Section 3.4 focus on the addition of a set of spatial coherence rules which aim to refine the results and moreover, to perform logo spotting in addition to the categorization. Section 3.5 presents the experimental setup by using a large set of real documents. Finally, the conclusions and a short discussion can be found in section 3.6.

## 3.2 Outline of the Approach

Our document categorization method is based on the presence of graphical logos in the incoming documents. This application is as a particular case of the problem of object recognition but has certain particularities. First of all, the documents are in binary format and are affected by the noise arising from the different acquisition systems. Since photometric descriptors are used to process gray-level (or even color) images, usually, when trying to codify a binary images we obtain poorly discriminative feature vectors. This may cause that the presence of false alarms increases. Secondly, the object recognition methods usually rely on a costly learning stage where a classifier is trained with multiple instances of the objects to recognize. In our application, in order to benefit the scalability of the method, no learning stage is involved and a single instance of the logos to locate is needed. Generally speaking, the presented method has a structure like the one proposed by Sivic et al. in [SRE05], where a bag-of-words model is translated to the visual domain by the use of photometric descriptors over interest points.

We can see an overview of the presented method in Fig. 3.1. The extracted local features from a document are matched against the codeword dictionary and an accumulator is used in order to decide to which category the queried document belongs. Let us further detail in the next sections the followed steps.

## 3.3 Document Categorization by Logo Detection

The document categorization and the logo detection is performed by using a bag-of-words model of visual words. These visual words are defined in terms of local features

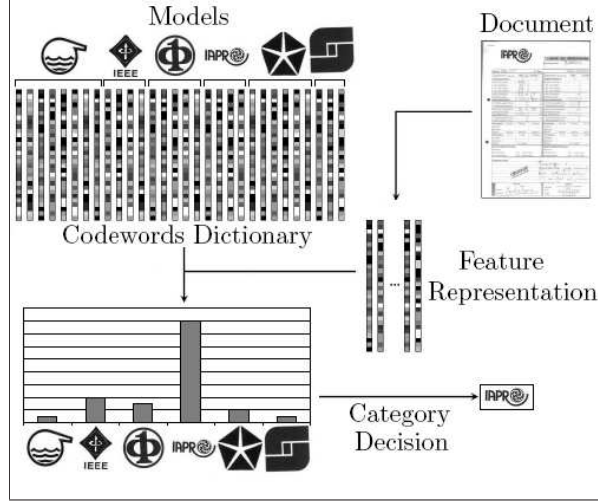


Figure 3.1: Overview of the proposed document categorization method.

extracted from a photometric descriptor. Let us first detail how these features are extracted and computed, and then focus on the bag-of-words model.

### 3.3.1 Feature Extraction and Description

Our method is inspired on the work presented by Bagdanov et al. in [BBB07], focused on the recognition of trademarks in real images. In that work, the authors use SIFT features to match trademark models against video frames. We use a similar matching approach, whereas our aim is to categorize and to use several different logos as models. Logos are represented by a photometric descriptor applied to a set of previously extracted key-points.

#### Interest Point Detection: Harris-Laplace Detector

The interest points are detected by using the Harris-Laplace detector presented by Mikolajczyk and Schmid in [MS04]. This algorithm extracts points with high curvatures (as corners or junctions) and automatically selects the scale of the region where to compute the photometric descriptor. Let us briefly review how this detection algorithm works.

The corner detector proposed by Harris and Stephens in [HS88] is based on the second moment matrix. This matrix is then adapted to scale changes to make it independent of the image resolution. The scale-adapted second moment matrix is defined by:

$$\mu(x, \sigma_I, \sigma_D) = \sigma_D^2 g(\sigma_I) \times \begin{bmatrix} L_x^2(x, \sigma_D) & L_x L_y(x, \sigma_D) \\ L_x L_y(x, \sigma_D) & L_y^2(x, \sigma_D) \end{bmatrix} \quad (3.1)$$

where  $L_a$  is the derivative computed in the  $a$  direction. The local derivatives are computed with Gaussian kernels of size  $\sigma_D$ . The derivatives are then averaged in the neighborhood of the point by smoothing with a Gaussian window of size  $\sigma_I$ . The Harris measure is then defined in terms of the trace and the determinant of this second moment matrix as:

$$M_c = \det(\mu(x, \sigma_I, \sigma_D)) - \kappa \text{trace}^2(\mu(x, \sigma_I, \sigma_D)) \quad (3.2)$$

where local maxima of  $M_c$  determine the location of interest points, with  $\kappa$  being a tunable sensitivity parameter. The Harris-Laplace detector uses the scale-adapted Harris function from eq. 3.2 to localize points in scale-space. The scale-space representation of the Harris function is built for pre-selected scales  $\sigma_n = \xi^n \sigma_0$  where  $\xi$  is experimentally set to 1.4. The matrix  $\mu(x, \sigma_n)$  is computed with the integration scale  $\sigma_I = \sigma_n$  and the local scale  $\sigma_D = s\sigma_n$  with a experimentally set parameter  $s = 0.7$ . For each point an iterative algorithm that detects the location and the scale of interest points is applied. The extrema over scale of the Laplacian-of-Gaussian, eq. 3.3, are used to select the scale of interest points by rejecting the points for which the LoG response does not attain any extremum or which response is below a certain threshold.

$$|\text{LoG}(x, \sigma_n)| = \sigma_n^2 |L_{xx}(x, \sigma_n) + L_{yy}(x, \sigma_n)| \quad (3.3)$$

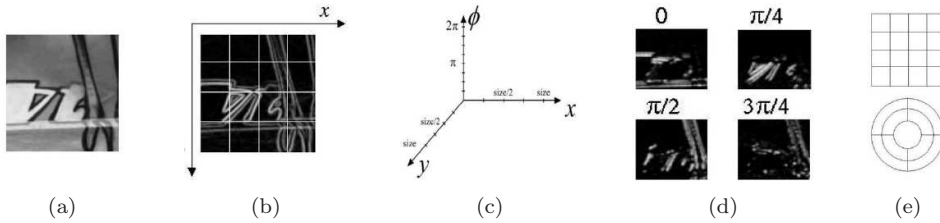
### Interest Point Description: SIFT and Shape Context

After the interest points are detected in an image, a photometric descriptor has to be applied to each region of interest defined by these key-points. In our experiments, we use and compare the performance of two different photometric descriptors. On the one hand we use the SIFT features and, on the other hand, we use the shape context descriptor<sup>1</sup>. As we will see in the experimental results section, each descriptor has its own strengths and weaknesses. Both descriptors are computed with the code provided by Mikolajczyk et al.<sup>2</sup>.

SIFT descriptors, presented by Lowe in [Low99, Low04], are computed for normalized image patches arising from the key-point detection stage. The descriptor is a histogram of gradient locations and orientations. The locations are quantized into a  $4 \times 4$  location grid and the gradient angles are quantized into eight predefined orientations. The resulting descriptor has 128 dimensions. Each orientation plane represents the gradient magnitude corresponding to a given orientation. In order to

<sup>1</sup>Note that in the chapter 2 we classified the shape context descriptor as a geometric descriptor since it copes with spatial arrangement of points. However, the enhancement of this descriptor proposed by Mikolajczyk and Schmid in [MS05] which takes into account not only point locations but also gradient magnitudes and orientations makes that this modified version should be considered as belonging to the photometric class.

<sup>2</sup><http://www.robots.ox.ac.uk/~vgg/research/affine/index.html>



**Figure 3.2:** SIFT and shape context descriptors. (a) Detected region; (b) gradient image and location grid; (c) dimensions of the histogram; (d) four of eight orientation planes; (e) cartesian and the log-polar location grids; (reprinted from [MS05]).

obtain illumination invariance, the descriptor is normalized by the square root of the sum of squared components.

The shape context descriptor implementation, based on the original presented by Belongie et al. in [BMP02], is similar to the SIFT descriptor, but is based on edges. Shape context is a histogram of edge point locations and orientations. Edges are extracted by the Canny detector. Location is quantized into nine bins of a log-polar coordinate system and orientation quantized into four bins. A 36 dimensional descriptor is therefore obtained. In addition, the point contribution to the histogram is weighted with the gradient magnitude.

We can see in Fig. 3.2 an example of the computation of the SIFT and the shape context descriptor. The gradient image is quantized in a location and orientation grid. Depending on which descriptor we use, the location grid has a cartesian or log-polar representation.

Let us see in the next section how we formally describe logos with the above presented key-point detection and description methods, and how similar logos can be matched.

### 3.3.2 Logo Representation and Matching

A given logo  $S_i$  is represented by its  $n_i$  interest points extracted from the Harris-Laplace detector. Each of these key-points are then described by a feature vector arising from a photometric descriptor. A logo instance is thus formally represented as:

$$S_i = \{(x_k, y_k, s_k, F_k)\}, \text{ for } k \in \{1 \dots n_i\} \quad (3.4)$$

where  $x_k$  and  $y_k$  are the x- and y-position, and  $s_k$  the scale of the  $k$ th key-point.  $F_k$  corresponds to the photometric description of the region represented by the key-point. An individual key-point  $k$  of the logo  $S_i$  will be denoted as  $S_i^k$ . The same notation applies when the key-points and the description vectors are computed over a complete document  $D_j$ . The matching between a key-point from the complete document and the ones of the logo model is computed by using the two first nearest neighbors:

$$\begin{aligned}
N_1(S_i, D_j^q) &= \min_k (F_q - F_k) \\
N_2(S_i, D_j^q) &= \min_{k \neq N_1(S_i, D_j^q)} (F_q - F_k)
\end{aligned} \tag{3.5}$$

Then the matching score is determined as the ratio between these two neighbors:

$$M(S_i, D_j^q) = \frac{N_1(S_i, D_j^q)}{N_2(S_i, D_j^q)} \tag{3.6}$$

If the matching score  $M$  is lower than a certain threshold  $t$  this means that the key-point is representative enough to be considered. By setting a quite conservative threshold ( $t = 0.6$  in our experiments) we guarantee that the appearance of false positives is minimized since only really relevant matches are considered as so. We can appreciate in Fig. 3.3 an example of the feature extraction and matching between a model and a document. However, for categorization purposes, we can not directly apply this matching procedure between the query document and all the model logos we consider. We use instead a bag-of-words model which have reached successful results for topic categorization. Let us describe in the next section how we adapt this model to the visual domain.

### 3.3.3 Bag-of-visual-words

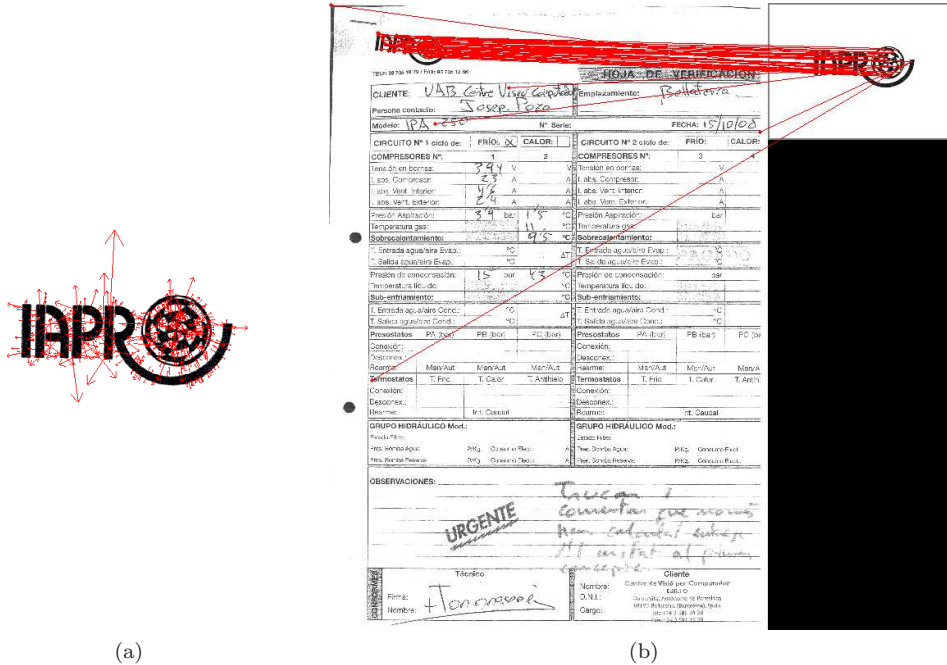
The bag-of-visual-words is an analogy to the Computer Vision domain of the classic bag-of-words model, where a text is represented by an unordered set of words. In that case, an image is represented by collection of image patches. In our particular case, given a set of logo models considered as different categories, we extract all the feature vectors  $F_k^i$  from them. Each feature vector is associated to its corresponding logo model  $S_i$ . By joining all the feature vectors from all the logos, we obtain the codeword dictionary  $W = [F_1^1, F_2^1 \dots F_k^i]$ . This dictionary is computed off-line from all the model logo database. Given a query document  $D_j$ , all the feature vectors  $D_j^q$  are used as indexes and matched against the codewords of the dictionary  $W$ . The matching function  $M_q$  returns the index  $i$  corresponding to the logo class of the matched feature vector  $F_k^i$  as follows:

$$M_q = \{i | M(W, D_j^q) < t\} \tag{3.7}$$

Finally, the determination of whether a document contains a logo is done by using by accumulating hypotheses of document categories in a histogram  $H$ .

$$\begin{aligned}
H[M_q] &= 0 \text{ at initialization,} \\
H[M_q] &= H[M_q] + 1, \text{ for } q \in \{1 \dots n_j\}
\end{aligned} \tag{3.8}$$

In the original bag-of-words model, a given document is categorized in terms of the frequency of appearance of certain words. For each document category we

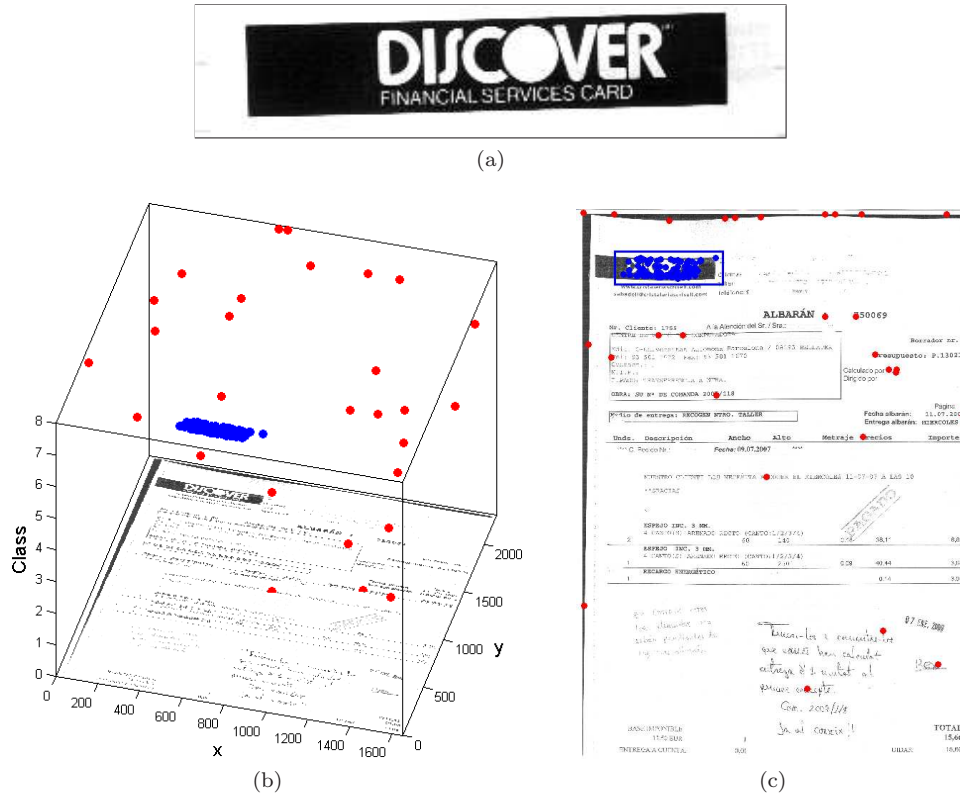


**Figure 3.3:** Matching logos in documents with the SIFT features. (a) SIFT features computed over an isolated logo model; (b) feature matching between the model and the complete document.

have a histogram of frequencies, and in order to determine which is the category an incoming document belongs to, distances among its histogram and all the model histograms must be computed. We face here a slightly different problem. If in a given document we have several appearances of parts of a given logo, we shall consider that it is probable that the document contains this logo. The document category is thus determined by searching the maximum  $m$  of the accumulator  $H$  after normalizing each accumulation cell with the total number  $n_i$  of features of the corresponding logo  $k$ . If the value of  $m$  is less than a threshold  $T$ , which has been experimentally set, we consider that the document do not contain any logo and is categorized in a rejection class.

### 3.4 Introducing Spatial Density for Logo Spotting

Whereas bag-of-words models have been very successful in the text domain, the analogy to visual words for image categorization has an important drawback. Bag-of-words models completely ignore the spatial relationship among features. Even if this drawback in the text domain is overcome due to the important impact of few keywords, in the image domain it is an important burden since the spatial layout among



**Figure 3.4:** Introducing spatial density information to spot logos. (a) Logo model; (b) parameter space; (c) spotted region of interest.

features has similar importance as the feature description itself.

It has been shown that the spatial organization of photometric descriptors computed from key-points is a powerful tool to recognize objects in scenes and to index images in terms of their contents as for instance the work of Mikolajczyk and Schmid presented in [MS01]. In the document analysis field, Nakai et al. introduced in [NKI05, NKI06] a method to retrieve document images acquired with a camera from a large image database using the arrangement of invariants computed over extracted feature points. The results are promising in terms of accuracy, time and scalability.

To overcome this drawback we use a simple yet effective voting scheme to guarantee that the spatial organization of features maintain certain coherence by introducing a density factor. Before contributing to the accumulator  $H$ , we get rid of the all the feature points of a same category  $i$  that are isolated in space. A Hough-like approach is used to transform the matched key-points from the image domain to a three-dimensional parameter space in order to cluster reliable model hypotheses that agree upon a particular model pose. The three-dimensional parameter space is built from the  $x$ - and  $y$ -locations of the matched key-points and a third dimension  $i$  which

represents the logo class. This parameter space is quantized and the problem of finding coherent locations is transformed in the problem of finding a maxima in this parameter space. By this means, only clusters of key-points which belong to the same category and which are close in space are considered. As we can see in Fig. 3.4, all the false alarms when matching key-points are eliminated. The red dots are inconsistent hypotheses and the blue dots maintain a certain spatial coherence and are taken as likely hypotheses. The bounding-box of likely hypotheses are returned to the user as the zones of the document image where the logo should be found. The presented method, given a document is able to in a single step categorize it in a certain class and return the zone of the document which contain the logo.

## 3.5 Experiments

To provide a realistic evaluation of the proposed method we used a large document collection. The collection consists in 1000 real document images which were sent by fax and then scanned. These images correspond to several kinds of documents such as invoices, letters, receipts, forms, etc. They contain both typewritten and handwritten text. Graphical elements such as logos, stamps, tables, etc. are also present in most of these documents. Typical dimensions of documents are near  $2500 \times 3500$  pixels with varying resolutions. All the images were scanned in binary format by using the built-in thresholding method of the scanner. Ground-truth of the entire collection was manually created identifying 18 different logo classes appearing in nearly 180 images, the rest of document images do not contain any logo and are used to test if the presented method is also able to reject these documents.

### 3.5.1 Evaluation Methodology

We will base our performance evaluation on how well the categorization of the documents is done. The performance of categorization methods is usually evaluated by confusion matrices to see if the systems under evaluation confuse two classes, mislabelling one as another. In addition, the true positive rate ( $TPR$ ) and false positive rate ( $FPR$ ) are used as evaluation measures in order to compare the performance among different methods. These ratios are derived from the contingency table and defined in terms of the amount of true positives ( $TP$ ), false positives ( $FP$ ), true negatives ( $TN$ ) and false negatives ( $FN$ ):

$$TPR = \frac{TP}{(TP + FN)} ; \quad FPR = \frac{FP}{(FP + TN)} \quad (3.9)$$

The  $TPR$  ratio measures the effectiveness of the system in retrieving the relevant items. Whereas the  $FPR$  ratio measures the probability that a non-relevant document is retrieved by the query. In our experiments we use the  $TPR$  ratio to summarize the correct categorization of documents containing a given logo. The  $FPR$  is used to measure the amount of documents that do not contain any logo which are incorrectly identified as belonging to a certain class.



### 3.5.2 Performance Comparison

We can appreciate in Fig. 3.5 the obtained confusion matrices after running the whole experimental categorization. We can appreciate some differences between the use of SIFT features and the shape context descriptor. For example, when using shape context, a lot of documents are incorrectly classified as class 8 (shown in row 8), or the documents corresponding to class 17 are usually misclassified in other document categories (shown in column 17). These misclassifications leads the overall *TPR* shown in Table 3.1 to be lower when using the shape context descriptors than when using SIFT features. On the other hand, when we test the documents that do not contain any logo and should be categorized in the rejection class, the SIFT features perform worst than the shape context descriptor, as shows the *FPR* of Table 3.1. In addition, the computational complexity when using SIFT is higher due to the highest number of dimensions of the feature vectors than when using the shape context descriptor, resulting in a higher querying time.

**Table 3.1:** Evaluation measures for the document categorization experiment.

Descriptor	<i>TPR</i> (%)	<i>FPR</i> (%)	Time (secs.)
SIFT	92.2	1	3.25
SC	81.6	0.3	1.34

In conclusion, the use of SIFT features should be preferred in applications where it is important to correctly identify the incoming documents, no matter if false alarms (documents which do not contain any logo) are present. On the other hand, if for the intended application it is preferable to minimize the false alarms even if we reject or misclassify some documents, or, if we want a faster method, the shape context descriptors should be considered.

## 3.6 Conclusions and Discussion

In this chapter we have presented a method for spotting logos in document images by using a photometric description of symbols. The use of a bag-of-words model reformulated to manage feature vectors arising from photometric descriptors combined with a Hough-like voting approach to guarantee the spatial and density coherence, aim to spot logos inside the document image and, in addition, to determine the category of the queried document. The presented experiments demonstrate the effectiveness of the method on a large set of real document images.

The presented application in this part, although it can be understood as a symbol spotting application, has been inspired by the characteristics of the object recognition methods from the Computer Vision field. The spotting of logos by means of a bag-of-words model applied to incoming documents, is useful for categorization purposes, but not for indexing or browsing of a large collection of documents. The main application of the rest of the thesis is the focused retrieval in a collection of line-drawing images rather than the object recognition one.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
1	100			2,44														7,69	
2		100																	
3			100																
4				92,69			14,29											3,85	
5					90														
6						90,9													
7							71,42				50								
8								100											
9				4,87					80										
10					10					100									
11											50								
12												100							
13													100						
14														80					
15						9,1			20						100		3,85		
16														20		100			
17							14,29											84,61	
18																			100

(a)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	100			2,44														3,84
2		100																
3			100															
4				85,36									16,67			10	3,84	
5					60													
6						100								40			3,84	
7							57,15											
8				7,32	10		28,57	100	20						7,14	10	7,7	
9				4,88					80				16,67					11,12
10					10					100								
11											0							
12											50	100			7,14		3,84	
13													66,66					
14														60				
15															85,72			
16																80	3,84	
17							14,28				50						69,26	
18					20												3,84	88,88

(b)

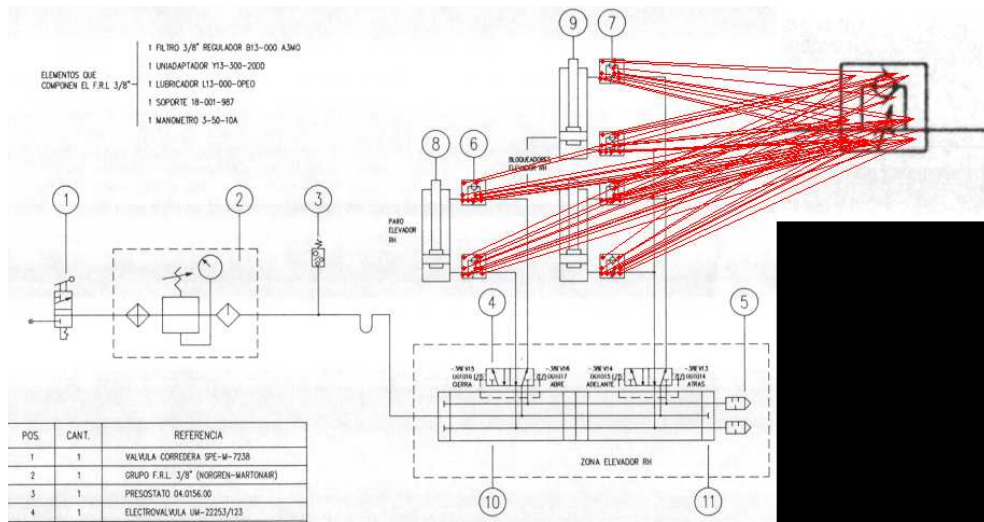
**Figure 3.5:** Confusion matrices for the document categorization experiment. (a) Using SIFT features; (b) using the shape context descriptor.

In the following part of this thesis, we will focus on the use of geometric and structural description techniques for the representation of graphical symbols rather than using photometric descriptors. Although photometric descriptors yield good recognition results and can be used as a basis for symbol recognition and matching applications, they have several limitations in the context of spotting graphical symbols from line-drawings.

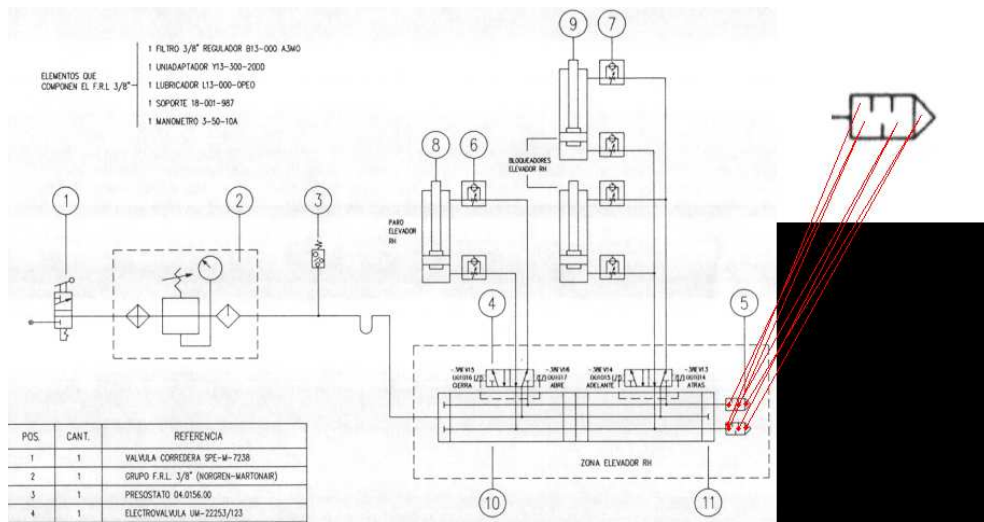
The first conclusion that can come to our mind is that photometric descriptors encode several visual cues at the same time, but the line-drawing images are usually binary and the symbols appearing in it are usually just made by line segments. Since the only discriminative visual cue to recognize such symbols is the shape, it seems more natural to use a geometric or structural description to really cope with the useful information. However, as we can appreciate in Fig. 3.6, the results of matching line-drawn symbols by a photometric descriptor (the SIFT features in this case), are not bad at all. We can notice nevertheless, that for simpler symbol designs, e.g. Fig. 3.6b, very few key-points are matched since the description is not discriminative enough. This factor can be problematic when the images in the collection are affected by some noise, and the symbol can be completely lost if these few key-points can not be matched against the model. The main factor provoking the discriminative power loss, is that in the case of line-drawings, the presence of a corner or a junction is not so relevant as in the case of real images (or the logos in the document analysis context). The information conveyed by the gradient magnitudes and orientations is not really discriminant in this particular context.

In addition, there is another limitation on the use of photometric descriptors in the context of spotting symbols for indexing a large collection of line-drawings. Usually, photometric descriptors tend to be high-dimensional. The SIFT descriptor has 128 dimensions whereas the adaptation of the shape context descriptor has 36 dimensions. This high-dimensionality helps to be discriminant enough to recognize objects in real images, but hinders the possibility to build indices over the high-dimensional description space. Even if Califano and Mohan claim in [CM94] that multidimensional indexing performs better than when using smaller spaces, the curse of dimensionality affects such high-dimensional spaces. In order to reduce the impact of the curse of dimensionality, when trying to index such descriptors, a step of reduction of dimensions as PCA as proposed by Ke and Sukthankar in [KS04], should be studied. Since geometric and structural description techniques are based on a previous primitive extraction, they tend to have lower dimensionalities than photometric descriptors which work at pixel level.

Let us see in the next part of this thesis three different approaches for symbol spotting in line-drawings which are based in a geometric and structural description of the symbols.



(a)



(b)

Figure 3.6: Matching symbols in line-drawings with the SIFT features.

## Part II

# On the Use of Geometric and Structural Constraints for Symbol Spotting



# Chapter 4

## Vectorial Signatures for Symbol Recognition and Spotting

---

In this chapter we present a method to determine which symbols are probable to be found in technical drawings by the use of vectorial signatures as symbol descriptors. The proposed signature model is formulated in terms of geometric and structural constraints among segments, as parallelisms, straight angles, etc. After representing vectorized line drawings with attributed graphs, our approach works with a multi-scale representation of these graphs, retrieving the features that are expressive enough to create the signature.

---

### 4.1 Introduction and Related Work

Since in the context of recognizing and locating graphical symbols from line-drawing images, the most important visual cue to describe graphical elements is the shape, a geometric and structural description of primitives seems the most natural choice. In order to apply such description techniques, a primitive extraction step is needed. Graphical symbols are broken down into lower level graphical primitives such as contours, loops, connected components, skeletons, etc. In the field of symbol discrimination, probably the approach which has gained most attention is the use of vectorial signatures as the description technique aiming to represent the graphical symbols. Vectorial signatures are geometric symbol descriptors which compactly encode the symbol in terms of particular geometric constraints among line primitives. The total amount of occurrences of each constrain forms the final signature. In order to compute the descriptors, the images must be processed in order to be broken down into segments by the use of a raster-to-vector conversion algorithm. Since spotting techniques are intended to coarsely recognize symbols, these particular descriptors are conceived as being very compact and having enough discriminative power to, at least, identify most of the zones of interest of a document image where a given symbol is likely to appear. Let us briefly overview the related work on the use of signatures

for symbol description and focused retrieval.

One of the first vector-based signatures has been proposed by Ventura and Schettini in [VS94]. In their work, the authors propose a signature for recognizing symbols from the architectural and electronic fields. First, line-drawing images are pre-processed by a thin/thick line separation algorithm, and then, the thin lines are polygonally approximated by straight segments. From the segments composing a symbol, they extract a number of geometric features as the number of segments intersecting in one point, the angles among these segments, their lengths, etc. Thick structures are described by their area, orientation and second order geometric moments. All these features are combined to create the signature which aims to describe the vectorized symbols. In order to make the signature more reliable, two values are added to each feature: a tolerance threshold and a weight. In the recognition step, the signature of the symbol to recognize is compared with all the model signatures. The distance among features is computed dependent on the tolerance threshold and normalized by the corresponding weight. A global threshold finally determines whether the query symbol matches a certain model. Results show efficient recognition, but this approach has the strong limitation that it has been conceived to just recognize isolated or pre-segmented graphical symbols.

Recently, Zhang and Wenyin presented in [ZW07] another model of vectorial signature for symbol description. Starting from the assumption that the symbols are in vectorial forms, primitive-pair relationships are recorded and employed to create the signature which is subsequently used as descriptor. Besides the basic relationships among segments, the authors propose a set of measures in order to describe several relationships among primitives having different nature, i.e. relationships among segments and arcs, segments and circles, etc. The proposed descriptor however, can also only be used to recognize isolated symbols.

Usually, in order to have a powerful representation of the vectorial symbols to easily compute the signatures, an attributed graph is used. We can find several examples in the literature. Coustaty et al. in [CGV08] or Qureshi et al. in [QRC07] use a graph representation of the symbols. In the nodes of the graph, primitives are stored and edges encode a certain geometric relationship among pairs of primitives. The use of the adjacency matrix of the attributed graph as descriptor has been widely used. However, despite the representative power of this structure, the proposed signatures are only tested in a symbol recognition framework working with pre-segmented instances of the graphic elements to recognize.

Inspired by the work of Ventura and Schettini and using some of the geometric features conceived to describe line-patterns presented by Etemadi et al. in [ESM91], Dosch and Lladós proposed in [DL04] another signature model. In addition to the description itself, the authors proposed a windowing methodology in order to be able to discriminate the regions of interest within a line-drawing where a given symbol is likely to be found. The signature of a graphic element is defined as a set of elementary features, containing intrinsically a discrimination potential. The method starts by a study on basic relationship between pairs of lines. Several main relations are thus enumerated: collinearity, parallelism and intersections. For each of these relations,



some extensions are considered, like overlapping for parallelism, or the kind of intersection point. The number and the type of the relations found in a particular zone will form the signature. The zones of interest are built from a decomposition of the image in several non overlapping tiles. The graphical primitives are then stored in these buckets, and relationships are only computed between the primitives of a bucket and the primitives of its neighboring buckets. The results show that this simple implementation can discriminate the learned symbols. A lot of false alarms are however present, especially with symbols not present in the library. Symbols containing arcs often lead to some non relevant signatures. But the main drawback is the fixed bucket partition of the image, that makes the method not really scale invariant, and causes a lack of flexibility.

Besides the fact that most of the approaches of vectorial signatures are focused on the application of symbol recognition and not on symbol spotting, we find that most of these approaches present another important drawback. Since vectorial signatures describe symbols in terms of geometric and structural constraints among sets of primitives, the inclusion of errors in the process of primitive extraction may provoke large variations in the signature, thus entailing a severe loss of discriminative power. We propose a signature model inspired in the work presented by Huang in [Hua97], where the main primitives describing a symbol are not just straight segments but are more complex sub-shapes composing a symbol. Inspired on the work of Dosch and Lladós [DL04] we also propose a window-based methodology allowing to compute different signatures within a whole graphic document, permitting to locate symbols appearing within a complete document image.

The remainder of this chapter is structured as follows: the next section presents the pre-processing step which aims to transform from the raw images acquired with the scanner to a vectorial format by introducing some state-of-the-art methods we use to achieve a raster-to-vector conversion. In section 4.3 our vectorial signature model is presented. Subsequently, in section 4.4 we define the window-based methodology which allows the computation of signatures within complete graphic documents. Section 4.5 presents the experimental results. Finally, the conclusions and a short discussion can be found in section 4.6.

## 4.2 Pre-processing Step: Raster-to-vector Conversion

In the part II of this thesis we focus on the particular application of spotting symbols in line-drawings by means of geometric and structural description techniques. Both description families work with primitives such as line-segments, arcs, etc. thus they need a previous step of conversion from the raw image to the primitive domain. We present in this section the pre-processing algorithms we use to convert the raw images to the vectorial format. Basically, we follow three steps. First, gray-scale images are denoised and binarized. In a second step, the skeletons of the foreground components are extracted. Finally, these skeletons are polygonally approximated.

Two different approaches to reach a raster-to-vector conversion can be found in the literature. Some methods are based on the combination of a skeletonization algorithm, followed by some kind of polygonal approximation. On the other hand, there exist another family of approaches which are not based on a recursive thinning but on contour following. These algorithms do not compute the skeleton but extract paths which are equidistant from contour lines, and approximate two parallel contour lines by segments. Although both families have their advantages and drawbacks, we decide to use a skeleton-based vectorization. The interested reader can find in [TASD00] a review on which are the most suitable methods to build a raster-to-vector system.

In this thesis, we work with the QGAR<sup>1</sup> implementation of the raster-to-vector conversion. In the QGAR library, the raster-to-vector conversion is based on Trier and Taxt's binarization, the (3,4)-Distance Transform skeletonization, and Rosin and West's polygonal approximation algorithm. Let us review in the following subsections these three methods.

#### 4.2.1 Document Binarization

First, grayscale images should be denoised using simple operations based on morphological operations. When working with scanned documents, the inherent noise and distortions as warping, paper folds, paper stains, etc. arising from these processes have to be faced. The interested reader is referred to Loce and Dougherty's review [LD97] of some simple existing techniques for digital acquired document enhancement and restoration.

After the document beautification stage, the grayscale line-drawing images should be transformed into binary format. A lot of well-known binarization methods exist, the interested reader is referred to the recent benchmarking study of binarization methods of Ntirogiannis et al. [NGP08]. We choose to use the approach of Trier and Taxt [TT95]. This binarization method was conceived to treat document images and yields good results. The interested reader can find a recent comparative study on the performance of different binarization techniques in [NGP08].

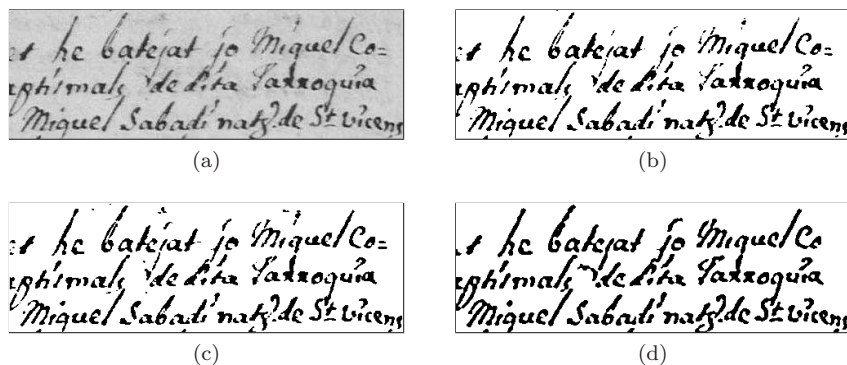
Trier and Taxt's method is based on the method by White and Rohrer [WR83] where a gradient-like operator is used to achieve a three level label image. Pixels with activity below a manually threshold  $T_A$  are labelled '0'. Then if the Laplacian edge operator of the pixel is positive, the pixel is labelled '+', otherwise '-'. The idea is that in a sequence of labels, edges are identified as '-+' or '+-' transitions and object pixels are assumed to be '+' and '0' labels between a '-+' and '+-' pairs.

Trier and Taxt improved this method by three modifications. First smoothing the input image with a  $5 \times 5$  mean filter in order to remove some noise. Then a print pixel identification is done in order to delete the false positives corresponding to noise blobs that are still present in the background area. The constraint of the original method, namely that '+' marked regions should be surrounded by '-' pixels to be labelled as print, is not sufficient criterion to remove the false print objects. For each '0' marked

---

<sup>1</sup><http://www.qgar.org>

region the number of ‘-’ and ‘+’ labels that are 8-connected is counted and the pixel is labelled print only if the number of ‘+’ pixels is in majority. Finally a postprocessing step is proposed to remove false print objects. The average gradient value at the edge of each printed object is computed. Objects having an average gradient below a threshold  $T_P$  are labelled as misclassified, and are removed. In Fig. 4.1 we can see the results of binarizing an old document by several methods. Both Otsu’s and Niblack’s well-known methods to binarize images leave some misclassified regions, whereas Trier and Taxt’s method perform a good binarization of document images.

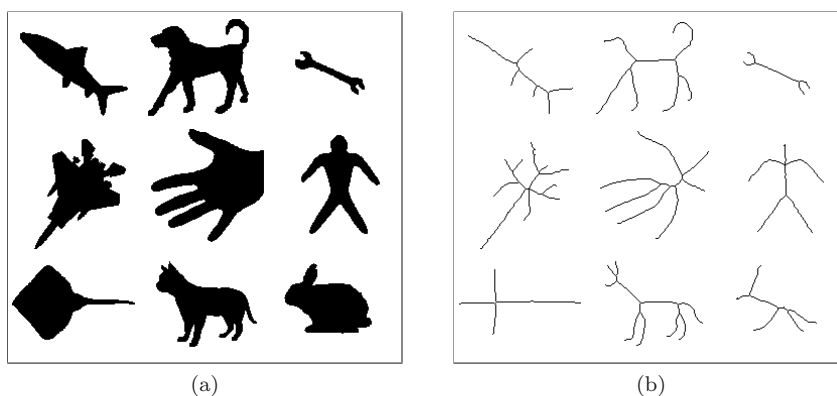


**Figure 4.1:** Binarization of an old handwritten document. (a) Original image; (b) Otsu’s binarization; (c) Niblack’s binarization; (d) Trier and Taxt’s binarization.

#### 4.2.2 Skeletonization

Sanniti di Baja proposed in [Baj94] a skeletonization method which is not based in thinning operations but on the analysis of the (3,4)-Distance Transform of the binary image. Each pixel of the shape is labelled with its distance to the contour. Each pixel  $p$  can be interpreted as the center of a disc, which includes all the pixels whose distance from  $p$  is less than the label of  $p$ . A disc  $D_p$  not completely included in the disc  $D_q$  centered on any neighbor  $q$  of  $p$  is called a maximal disc. The skeleton of a shape will include all the centers of maximal discs of the (3,4)-Distance Transform, except for those whose removal is indispensable to allow the skeleton to be a unit wide set. On the skeleton, the pixels can be classified into *end points*, *normal points* and *branch points*, by taking into account the number of components of neighbors not belonging to the skeleton. This method does not require the iterated application of topology preserving removal operations, and does not need checking a condition specifically tailored to end point detection, since end points are automatically identified when the maximal centers are found. Skeletal pixels found on the distance transform can be classified as “parallelwise detectable” and “sequentially detectable” skeletal pixels. Parallelwise detectable pixels can be directly identified on the distance transform, due to the structure of their neighbors. This is the case of the maximal discs. Sequentially detectable skeletal pixels can be found only after some of their neighbors with smaller

labels have been identified and marked as skeletal pixels. Sequentially detectable pixels are necessary to link to each other the components of parallelwise detectable skeletal pixels. After the detection of skeletal pixels a raster scan inspection is done to reduce to unit width and to fill the holes. Then a pruning and beautification step is proposed to erase some non significant pixels of the skeleton. An example of the obtained skeletons is shown in Fig. 4.2. The computational cost of this method is modest and independent of the thickness of the pattern to be skeletonized.



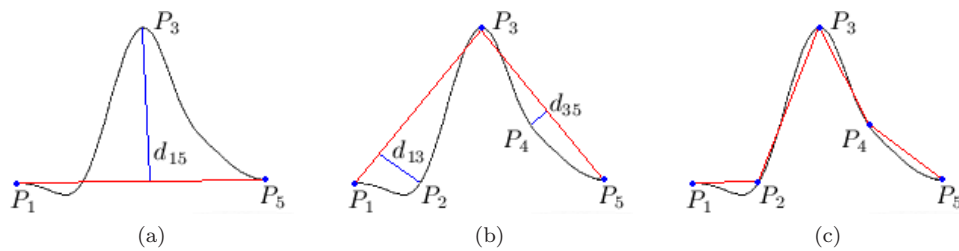
**Figure 4.2:** (3,4)-Distance-based skeletonization. (a) Original images; (b) obtained skeletons.

### 4.2.3 Polygonal Approximation

The last step of the raster-to-vector conversion is to approximate skeleton images by segments. Rosin and West proposed in [RW89] a method of segmenting curves in images into a combination of circular arcs and lines. The method is an extension of the algorithm proposed by Lowe in [Low87]. Lowe's algorithm segments each curve by recursively splitting it at the maximum deviation from the approximating straight line. At each level a decision is made looking if whether the single straight line is better than the representation at a lower level consisting of two or more approximating straight lines. The measure of goodness of fit is termed the "significance" and is defined as the ratio of maximum deviation from the straight line to the length of the straight line.

The attractive property of this algorithm is that no thresholds are used to control the accuracy of the resulting representation. This is controlled by the significance values that can be regarded as the error between the curve and the straight line description weighted by the length of the straight line. Thus long straight lines are regarded as being a better representation even though the error can be greater. This introduces scale invariance such that a contour of different scales will have the same or similar description.

A list  $L_{ij}$  of skeleton pixels is hypothesized as being a straight line passing through its end points  $P_i$  and  $P_j$ , the point  $P_n$  corresponding at the point of maximum deviation  $d_{ij}$  to the straight line segments the list  $L_{ij}$  in two lists  $L_{in}$  and  $L_{nj}$  and the process is repeated recursively on each of the two lists. The recursive process is stopped when a line segment is smaller than four pixels long or the deviation is less than three pixels. The result of the recursive process is a multilevel tree where the description of the list of skeleton pixels at each level is a finer approximation of the level above. The tree is then traversed back up to the root. At each level, if any of the line segments passed up from the lower level are more significant than the line segment at the current level, they are retained and passed up to the next higher level as candidate line segments. If this is not the case, the line segment at the current level is passed up. In Fig. 4.3 we can see an example of the first steps of the algorithm in approximating a curve by a set of straight lines.



**Figure 4.3:** Three levels of the straight line approximation. (a) First iteration; (b) second iteration; (c) third iteration.

The significance measure is the ratio of the maximum deviation divided by the line segment length. Thus, the lower the significance value, the more significant the line. The procedure is weighted in favor of long line segments. The longer the line is, the greater the deviations that will be tolerated. This algorithm produces a high quality, general purpose polygonal approximation. No arbitrary error threshold is required. Instead, the most appropriate values are chosen dynamically throughout the procedure.

### 4.3 A Vectorial Signature for Symbol Description

Starting from a vectorial representation of the documents, we propose in this section a model of vectorial signature aiming to describe symbols in terms of geometric constraints. In order to have a powerful representation of the vectorial symbols to easily compute the signatures, an attributed graph is used. The nodes of the attributed graph represent the segments of the symbol and graph edges represent spatial relationships between segments. Let us formally define a graph  $G$  in the next subsection.

### 4.3.1 Representing Symbols by Attributed Graphs

Starting from a vectorial representation of the symbols from a line-drawing, we represent these symbols with a graph  $G$  defined as follows:

**Definition 4.1** An **attributed graph** is denoted as  $G = (V, E, \mu, \nu)$  where  $V$  is the set of nodes representing the segments of the symbols and  $E$  is the set of edges representing the spatial relationships among them. A **subgraph** of  $G$  containing the nodes  $s_i, \dots, s_j$  is denoted as  $G_{\{s_i, \dots, s_j\}}$ .  $\Sigma_V$  and  $\Sigma_E$  are a set of **symbolic labels**, and the functions  $\mu : V \rightarrow \Sigma_V$  and  $\nu : E \rightarrow \Sigma_E$  assign a label to each node and each edge.  $\Sigma_V = [\theta_{s_i}, \rho_{s_i}]$  contains the information of each segment  $s_i$  according to a polar representation.  $\Sigma_E = \{L, T, P, 1, 0\}$  represents the different kind of spatial relationships between a pair of segments. The possible relationships between segments are:

1.  $L$  represents a straight angle between a pair of adjacent segments.
2.  $T$  represents a straight angle between a pair of non-adjacent segments.
3.  $P$  represents two parallel segments.
4.  $1$  represents two adjacent segments.
5.  $0$  represents a non expressive relation between two segments.

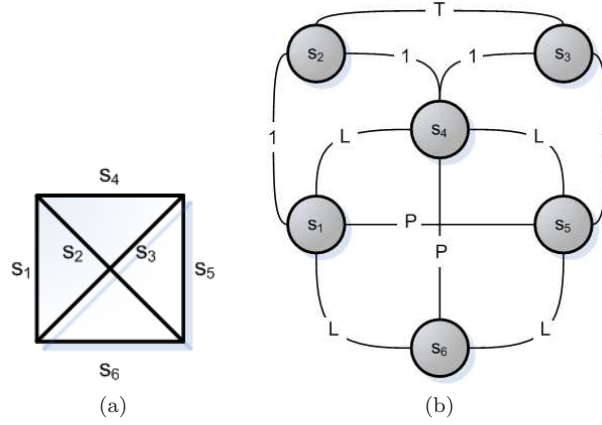
We define a signature in terms of a hierarchical decomposition of symbols. Following the idea presented by Huang in [Hua97], a symbol can be described by the number of occurrences of particular sub-shapes. In our proposal, these expressive sub-shapes are extracted from the analysis of the adjacency matrix. Following a combinatorial approach on the number of subgraph nodes sub-shapes as squares, triangles, parallelisms, etc. are taken into account. In Fig. 4.4, we show a graph<sup>2</sup> of a simple symbol. Let us see in the next subsection how the signatures are built from the analysis of the adjacency matrix.

### 4.3.2 Building the Vectorial Signature

Starting from the analysis of the adjacency matrix, we propose a combinatorial approach which aims to extract particular subshapes which compose a symbol. For all the segments, all the subgraphs formed by at least two nodes, and a maximum of four nodes are analyzed to search some representative shapes. Being  $n$  the number of segments, eq. 4.1 gives the number of subgraphs to analyze.

$$\#G_{\{\dots\}} = \sum_{k=2}^4 C_n^k = \sum_{k=2}^4 \frac{n!}{(n-k)! \times k!} \quad (4.1)$$

<sup>2</sup>The edges labelled with a 0 are not shown.



**Figure 4.4:** Attributed graph representation of graphical symbols. (a) Graphical symbol; (b) its graph representation.

For each subgraph, we work with its adjacency matrix. The matrix  $M_G$  is in fact only computed once for all the segments, and then, when we want to focus to a subgraph, a group of rows and columns of this matrix is selected. Notice that in most cases the relations between segments could be extracted in the vectorization process. In the extraction of these constraints, each comparison has associated a threshold value in order to be more tolerant. For the simple shape of Fig. 4.4, we can see below in 4.2, its corresponding adjacency matrix  $M_G$ .

$$M_G = \begin{pmatrix} s_1 & 1 & 0 & L & P & L \\ 1 & s_2 & T & 1 & 0 & 0 \\ 0 & T & s_3 & 1 & 1 & 0 \\ L & 1 & 1 & s_4 & L & P \\ P & 0 & 1 & L & s_5 & L \\ L & 0 & 0 & P & L & s_6 \end{pmatrix} \quad (4.2)$$

From this matrix, we examine all the possible combinations of sub-matrices taking four, three and two of the six possible nodes. Hence three different levels are considered. For all the sub-matrices representing the subgraphs, normally the analysis of one single row can determine the shape that it encodes.

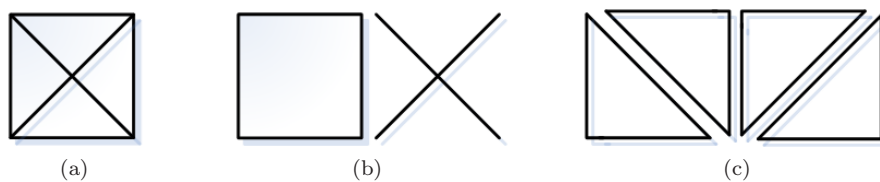
The vectorial signature of a symbol is then defined as a 40-dimensional vector  $V_S$  where in each position we have the number of occurrences of a particular geometric configuration of segments forming a given sub-shape. We have defined a set of 30 geometric configurations among different number of segments which can be efficiently extracted by analyzing the adjacency matrix. We can see some examples the sub-shapes taken into account to build the signature in Table 4.1. In order to provide a more accurate description of the symbols, some additional information as the length-ratios and the distance-ratios are added in the last 10 positions of the signature. Since

**Table 4.1:** Examples of sub-shapes composing the vectorial signature.

Level	Considered sub-shapes	Level	Considered sub-shapes
4 nodes		3 nodes	△
4 nodes	≡	3 nodes	∨
4 nodes	□	2 nodes	
3 nodes		2 nodes	+
3 nodes	≡	2 nodes	↗

these measure features can take values from 0 to 1, this space is split into five bins where the occurrences of these geometric ratios among segments are accumulated.

It may seem redundant to store the information for multiple levels of the subgraph, since if in the level of four nodes we find a square, it is obvious that we will find two parallelisms and four straight angles in the level of two nodes. But this redundancy helps to detach completely all the multiple shapes in the drawing. For instance, a square with a cross inside can be seen as a square and a straight angle, or it can be seen as a set of triangles (see example in Fig. 4.5). This redundancy helps to be more error tolerant and to store all the geometric configurations of all the multiples sub-shapes of the drawing. The occurrences of each sub-shape are accumulated to build the vectorial signature.



**Figure 4.5:** Sub-shapes extracted from a symbol at different levels. (a) Original symbol; (b) symbol detached at level four and at level two; (c) symbol detached at level three.

Once the the signatures of the model symbols are extracted, in the querying step we can compare the obtained signature with the model signatures and associate a confidence value to each correspondence between the original symbol and all the model symbols depending on a distance function. The used distance function is the chi squared distance computed as follows:



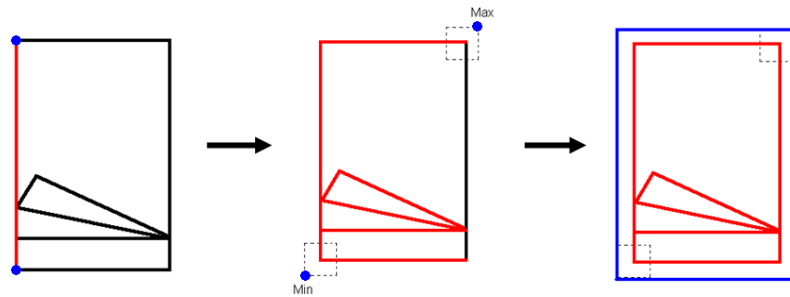
$$C_{ij} = \frac{1}{2} \sum_{k=1}^K \frac{(V_i[k] - V_j[k])^2}{V_i[k] + V_j[k]} \quad (4.3)$$

where  $V_i$  and  $V_j$  are the vectorial signatures of two symbols  $i$  and  $j$ , and  $K$  the length of the vectorial signature (40 in our experimental setup).

However, as we commented in the introduction section 4.1, the approaches based on vectorial signatures have the drawback that can not be straightforwardly used to locate a given symbol within a complete document. The spotting approaches based on vectorial signatures need a previous segmentation stage. Inspired by the work on symbol discrimination in complete documents presented by Dosch and Lladós in [DL04], we present in the next section a window-based strategy aiming to compute the vectorial signatures within documents.

## 4.4 Sequential Access to Signatures: Defining Regions of Interest

When working with complete drawings, the usual approach is to divide the drawing into windows of fixed size which frame every symbol. In each zone of interest, a signature is computed and compared against the set of model signatures. However, these approaches lack of flexibility and may be quite sensitive to the scale of the documents. We propose to use a more dynamic approach, where the windows are built depending on the original line-drawing.



**Figure 4.6:** Computing a region of interest from a reference segment. In the second step, all the adjacent segments to the reference segment are considered. A bounding-box is obtained from the minimum and maximum coordinates.

Regions of interest are computed from the maximum and minimum coordinates of several adjacent segments. So, the size of the regions of interest is variable. Also, a first filter of area and aspect-ratio can be easily implemented in order to delete some non relevant symbols as for example the walls in the architectural field or the wiring connections in electronic diagrams. Formally, for each node  $n_{s_i}$  of  $G$  (segment in the drawing) we build a list of all the nodes connected to  $n_{s_i}$  by an edge. Having a list of all the endpoints of the adjacent segments to reference segment, we get the maximum

and minimum coordinates of the endpoints that will construct a framing window of these segments. We can see an example on how to build the regions of interest in Fig. 4.6. As in most cases of technical drawings the symbols have a low eccentricity, its bounding-box are square-shaped and this kind of windows frame them. But, as the windows are based on the connection of the segments, the efficiency decreases if the symbols are disconnected or overlapped.

Moreover, in the vectorization step, more problems may happen: small vectors can appear due to noise, straight lines can be split into several collinear vectors, the arcs are approximated by polylines, some neighboring lines in the drawings are not adjacent in the vectorial representation because of gaps, dashed lines appear as a set of small segments instead of one unique instance, etc. To solve this kind of problems, the best results are reached when we work with a lower resolution of the drawing to calculate the windows. This sub-sampling step reduces local distortions in the vectorial representation but preserving the most salient geometrical properties.

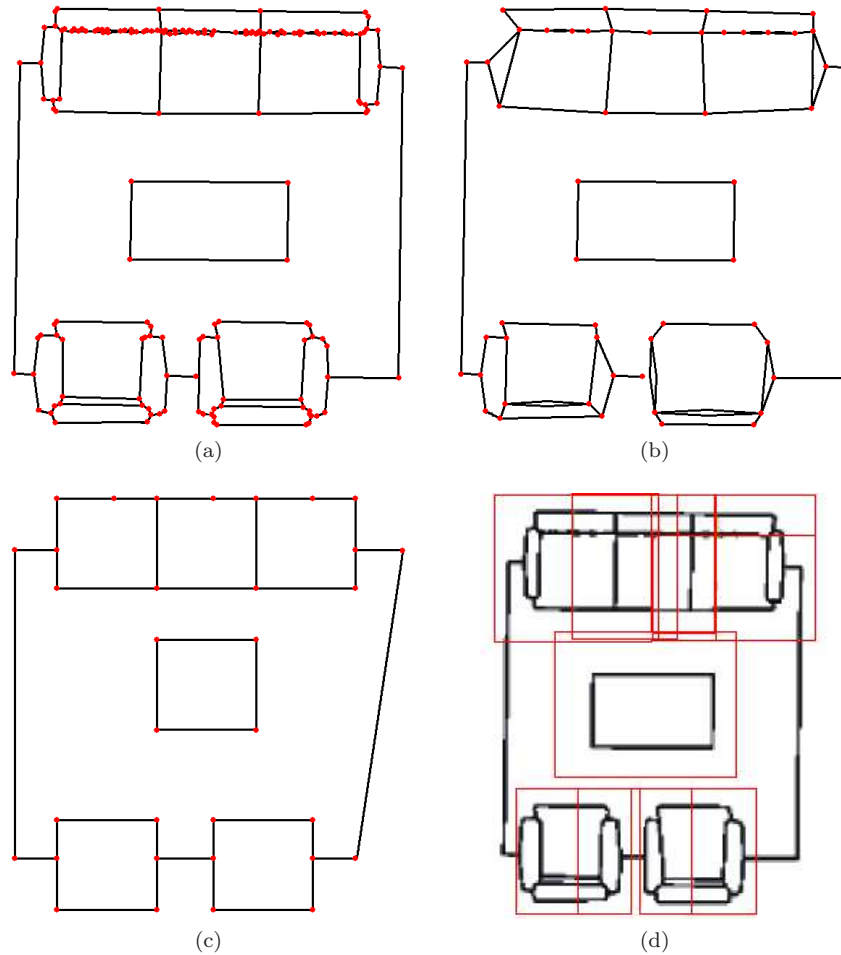
First, a contraction of the normalized graph is done, merging the adjacent nodes having a lower distance than a threshold  $thr$ . This graph contraction by distance allows to reduce the jaggedness of some collinear straight segments. Then, applying eq. 4.4 to each node coordinate we get a lower resolution graph. With this representation with decreased resolution, the problems of the gaps, or the split segments are solved. Every endpoint is sampled for each step of  $m$ , so the minor errors are corrected.

$$\begin{aligned} x &= m \times \text{round}\left(\frac{x}{m}\right) \\ y &= m \times \text{round}\left(\frac{y}{m}\right) \end{aligned} \quad (4.4)$$

Experimentally, in Fig. 4.7a the graph has 154 nodes because an horizontal line has been split in the vectorization process. When the graph contraction by distance is done (with a threshold value  $thr = 0.06$ ) Fig. 4.7b, we get a graph with 52 nodes which lines are crooked due to the node contraction, and with the decreased resolution graph (with  $m = 35$ ) Fig. 4.7c we have to face up to only 33 nodes. Finally, we can appreciate in Fig. 4.7d the resulting extracted windows where to compute the vectorial signature.

This change of resolution can cause some errors, for example some lines which are almost horizontal or vertical can be represented with a very different slope. But these errors do not interfere with the obtained windows, since they continue to frame the symbols. Notice that these lowest resolution images will only be used to calculate the regions of interest, not for the spotting process. Since each segment proposes a region of interest, there is no problem if one of the segments of a symbol gives a mistaken window.

Let us present in the next section the experimental results.











































**Figure 4.7:** Obtaining regions of interest from a low-resolution representation of line-drawings. (a) Original drawing; (b) graph contraction by distance; (c) low resolution representation; (d) obtained windows in the document image.

## 4.5 Experimental Results

Our experimental framework consist of two different scenarios. First we test the performance of the vectorial signatures to recognize and classify isolated symbols. Secondly, we have used the method for symbol spotting in a small set of real architectural drawings and we will show some qualitative results.

The first tests were done using the *GREC-SEG* database, which is detailed in appendix A. This database contains a selection of symbols from the GREC2005 database which does not contain arcs. For each model symbol, we have applied three levels of synthetical distortion and twenty instances at each level have been generated. The

**Table 4.2:** Results of the recognition of GREC-SEG database (1).

Symbol	TPR (%)				Symbol	TPR (%)			
	$r = 5$	$r = 10$	$r = 15$	Total		$r = 5$	$r = 10$	$r = 15$	Total
001 	100	100	100	100	002 	100	100	100	100
003 	95	70	45	70	005 	100	100	100	100
007 	100	100	95	98,3	008 	100	100	100	100
011 	100	100	100	100	012 	100	100	100	100
013 	100	100	70	90	014 	90	60	25	58,3
015 	100	100	25	75	018 	100	95	60	85
020 	100	100	80	93,3	023 	100	100	95	98,3
027 	100	100	100	100	028 	100	85	70	85
029 	100	90	65	85	030 	100	100	100	100
031 	100	100	85	95	032 	100	100	60	86,6
033 	100	95	70	88,3	034 	100	100	100	100
037 	100	100	100	100	041 	100	100	100	100
042 	100	100	100	100	043 	100	100	100	100
044 	100	85	40	75	045 	100	100	100	100
048 	100	100	100	100	051 	100	100	100	100
052 	100	100	100	100	053 	100	100	60	86,6
054 	100	85	55	80	055 	100	100	100	100
057 	100	100	100	100	058 	100	100	95	98,3
059 	100	100	100	100	060 	100	100	100	100
062 	100	75	50	75	063 	100	100	100	100

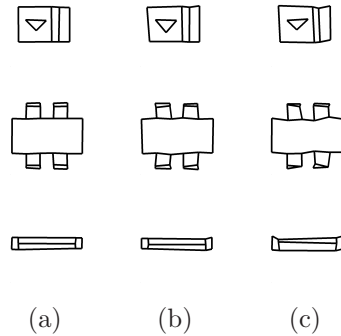
symbols are represented with a graph, where the nodes represent the segments endpoints. Each node from the graph is randomly shifted within a predefined radius  $r$ . As the symbols are represented with a graph, the connectivity is not lost. We can see an example of these distortions in Fig. 4.8.

We can see the recognition results in Tables 4.2 and 4.3 expressed in terms of the

**Table 4.3:** Results of the recognition of GREC-SEG database (2).

Symbol	TPR (%)				Symbol	TPR (%)			
	$r = 5$	$r = 10$	$r = 15$	Total		$r = 5$	$r = 10$	$r = 15$	Total
065	100	100	70	90	068	100	100	85	95
069	100	100	100	100	072	100	100	100	100
074	100	100	100	100	078	100	90	40	76,6
079	100	100	100	100	084	50	10	10	23,3
085	100	100	100	100	088	100	100	75	91,6
091	100	100	100	100	093	100	100	100	100
094	40	50	80	56,6	098	100	80	60	80
104	100	100	100	100	106	100	100	100	100
107	100	100	100	100	108	100	100	100	100
110	100	100	100	100	111	100	85	60	81,6
113	100	90	40	76,6	114	100	65	30	65
115	100	100	100	100	120	100	100	100	100
121	100	100	100	100	126	100	100	100	100
127	100	100	100	100	128	100	100	100	100
130	100	85	30	71,6	132	100	100	100	100
133	100	100	100	100	134	100	100	95	98,3
136	100	100	90	96,6	137	100	100	95	98,3
138	100	100	100	100	143	100	100	100	100
144	100	100	85	95	145	100	95	95	96,6
147	100	100	100	100	Total	97,79	92,58	79,25	91,18

True Positive Rate (*TPR*). We can see that the method uses to yield good results when applying a low degradation of symbols. Most of the symbols of of the GREC-SEG database are “square-shaped” and the computed signatures are discriminative enough. But, when a symbol is composed of tiny little segments, and not very connected, the results are worst. Let us analyze some problematic symbols. Symbol 014 and 015 are very thin, but composed by an expressive sub-shape really discriminative,

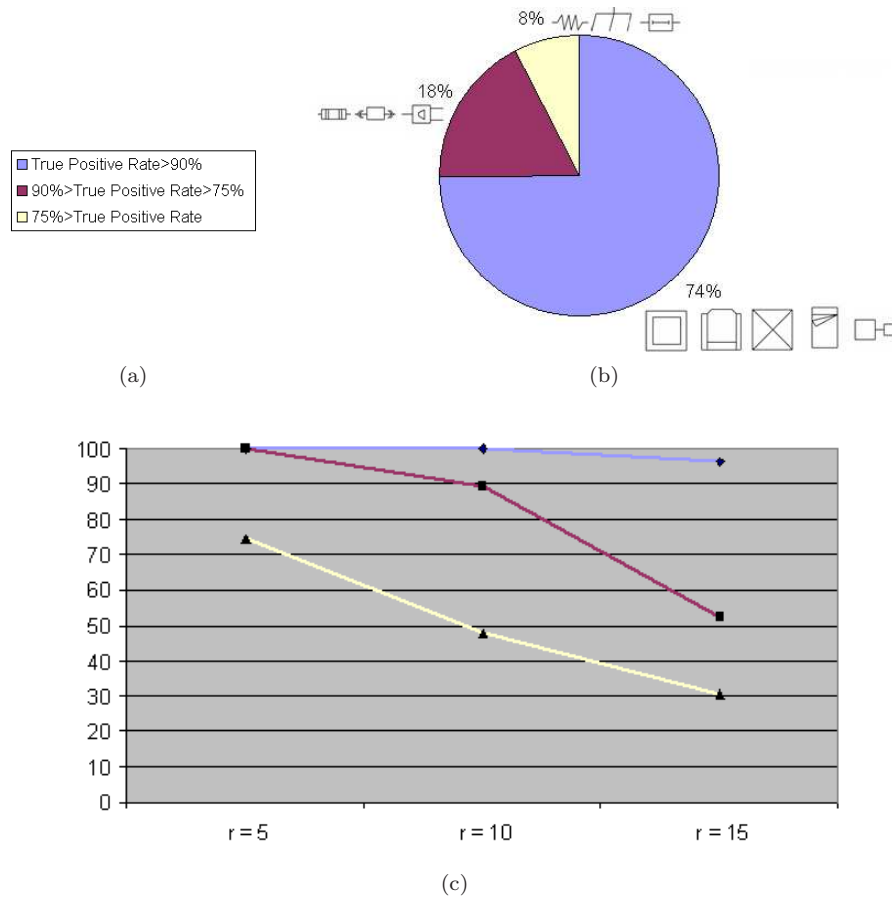


**Figure 4.8:** Example of synthetical distortion from the GREC-SEG dataset. (a)  $r = 5$ ; (b)  $r = 10$ ; (c)  $r = 15$ .

a square. These two symbols give good results when applying low degradation, 90% and 100% of recognition respectively, but their recognition rate falls to 25% in both cases when applying a huge geometric deformation. As these symbols are composed by very short segments, a higher deformation distorts too much the little segments and damages the performance. On the other hand, symbols having segments which are not very connected between them give also bad results. For example symbols 003, 054, 114 or 130. As the deformation model guarantees the connectivity between segments, deforming the graph representation of the symbol, not the symbol itself, when the symbol is composed by non connected segments, these segments are more affected by the deformation than the connected segments, that, in some way, share the deformation between them. Notice that the recognition performance of symbol 094 evolves inversely to the expected way. Symbol 094 does not have very expressive sub-shapes, only parallelisms and adjacency can be found. As vectorial signatures encode the presence of salient geometric features, when a symbol is composed of few sub-shapes, the recognition performance is very low. When applying a higher geometric noise most of these sub-shapes are not preserved, but as the model of distortion guarantees the connectivity, this symbol is recognized better at high distortion levels than at the lower ones.

Finally, in order to have an idea on whether the symbol design can affect the performance of the recognition abilities of the signature model and to test if there are some symbol designs which are more sensitive to distortions, we present in Fig. 4.9 some indicators on the recognition performance. We classified the symbols in the GREC-SEG database into three different classes dependent on their average true positive rate. We can see that a 74% of the symbols of this database attain an average true positive rate greater than 90%. A second symbol family can be identified. A 18% of the symbols in the database has an average true positive rate ranging between 75 and 90%. Finally, an 8% of the symbols has an average true positive rate below the 75%. As we can appreciate in the example, the symbols in this last group are formed by less discriminative sub-shapes, and thus, the description ability of the signature

is severely damaged. We can appreciate in Fig. 4.9c the different tolerance to the distortions for all the three classes of symbols and how simpler symbol designs are more affected when we increase the synthetic deformations.



**Figure 4.9:** Average true positive rates for three different symbol categories. (a) Three different symbol categories depending on their average recognition rate; (b) its distribution on the database; (c) average recognition rates per different symbol degradation levels.

In the second test, we tried out the vectorial signatures with real architectural drawings by using the windowing approach presented in section 4.4. Using more relaxed threshold values than when we are working with the database of isolated symbols, the symbols appearing within a complete document can be spotted. As we can see in Fig. 4.10, some false positives appear (red squares) and some symbols are still missed. False positives appear when a window does not correctly frame a symbol. The stairs which consist of a lot of segments give a lot of regions of interest where false positives appear, and the wrong segmentation of the tables makes that the part where

the chairs are drawn a sofa is spotted, because their representation is very close. When we use vectorial signatures in real drawings there are two factors that may cause the spotting results not to be so good. First of all, the symbols can be adjacent between them or to a wall, or the region of interest could not frame perfectly the symbol, in this case we face up to occlusions and additions of segments which distort too much the signature. On the other hand, in real drawings, the symbol design may be different of the learned model, so the learned features of a symbol could not appear in real drawings, in this case it is obvious the symbol can not be spotted, a semantic organization of different design instances for any symbol is necessary.

## 4.6 Conclusions and Discussion

In this chapter we have presented a vectorial signature model which is able to describe graphical symbols in terms of the occurrences of certain spatial configurations of segments. Since signatures are compact and yet effective symbol descriptors, they are very suitable to be used as the basis for a spotting approach. We have also presented a window-based segmentation system which aims to use the vectorial signatures to spot symbols appearing within complete documents.

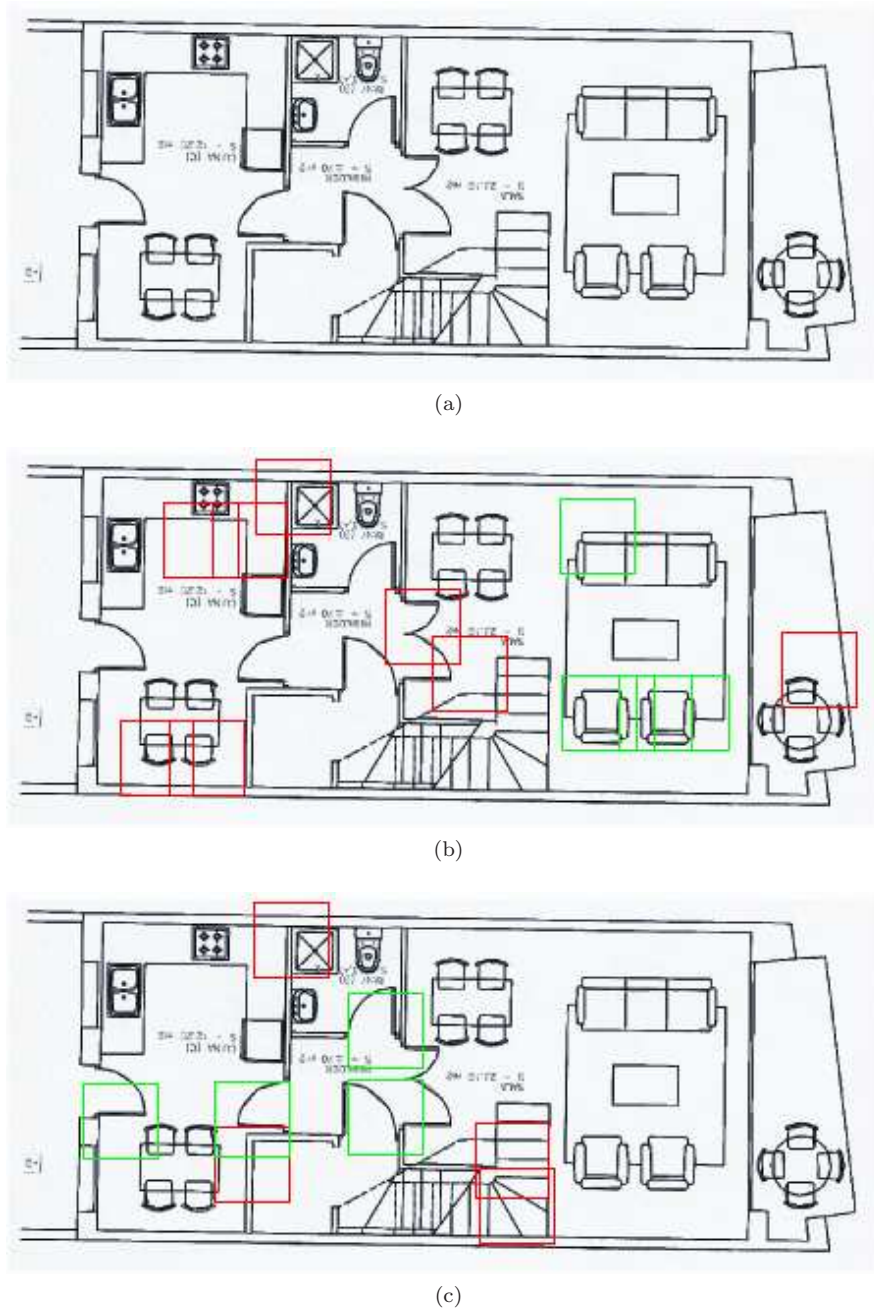
We can see that the symbol discrimination using vectorial signatures yields good results when we are working with the database of pre-segmented symbols and with symbols with synthetical distortion, which is a controlled framework. In real scanned architectural drawings, even if the symbols are usually well spotted, a lot of false positives appear. However, since the objective of spotting techniques is not to give a recognition of the symbol but in some way to index the drawing, the false positives problem is not so significant.

### 4.6.1 Limitations of the Vectorial Signatures

Even if the recognition performance of the signatures attains good levels, the proposed methods presents some important drawbacks to be used as a spotting method for indexing a document collection to be used by a focused retrieval application. Let us enumerate the most important drawbacks and let us see in the next two chapters how can we solve these problems.

Since the presence of the sub-shapes is determined from the analysis of the adjacency matrix, the method is not tolerant at all to segment fragmentation. If the number of segments composing a symbol does not correspond of the number of segments of the learned model the signature can be severely damaged. In addition, the connectivity of adjacent segments also must be guaranteed in order to achieve an acceptable performance. Even if this effects do not occur on synthetic data as the one of the GREC symbol recognition competitions, these two phenomena occur frequently in real document images treated with a raster-to-vector conversion. A more tolerant set of features in which base our signature must be find to tolerate this kind of errors. We introduce in the next chapter the notion of polylines instead of segments. We





**Figure 4.10:** Qualitative results of spotting symbols by using vectorial signatures. (a) Original drawing; (b) spotting the sofa symbol; (c) spotting the door symbol.

will also see how we can represent the expressive sub-shapes defined above at polyline level.

Moreover, the extracted sub-shapes from the adjacency matrix composing the vectorial signature, were ad-hoc defined to describe a particular family of graphical symbols. We saw in the experimental results section, that the recognition performance of the signature model was dependent on the symbol design. However, spotting approaches should be more scalable in terms of the nature of the documents, and not be conceived for a particular collection. We propose in the next chapters the use of more flexible description approaches which should perform similarly no matter which nature the line-drawing is.

In addition, the proposed signature model just captures geometric configurations formed by straight segments. It is not able to deal with circular primitives such as arcs, circles, ellipses, etc. This fact is a strong limitation since only symbols formed by lines can be considered. The methods proposed in the next chapters can deal with any polygonal shape.

Finally, the presented window-based system can be useful in applications with a predefined set of model symbols. Given an input line-drawing, regions of interest are extracted by the windowing approach and the signatures are sequentially computed and matched against the model database. However, this sequential approach is hardly useful when facing focused retrieval applications. In these cases, we can have large collections of documents and the user can query any symbol. We shall provide a more efficient access to the descriptors by using indexing data structures. We propose in the next chapters the use of particular data structures having graphical patterns as indices for accessing and navigating large collections of documents and be able to use spotting methods as the basis of a focused retrieval application.

# Chapter 5

## Symbol Spotting Through Prototype-based Search

---

In this chapter we present a method to determine which symbols are probable to be found in technical drawings by the use of a prototype-based search. First, symbols are decomposed in primitives representing closed regions. These primitives are then encoded in terms of attributed strings. Second, the strings are organized in a lookup table so that the set median strings act as representative prototype of the clusters of similar primitives. This indexing data structure aims to efficiently retrieve the locations from the document collection where to find similar primitives than the queried ones. Finally, a voting scheme formulates hypotheses in the locations of the line drawing image where there is a high presence of regions similar to the queried ones, and therefore, a high probability to find the queried graphical symbol. The proposed approach has been proved to work even in the presence of noise and distortion introduced by the scanning and raster-to-vector processes.

---

### 5.1 Introduction and Related Work

The vectorial signature model presented in the last chapter is only able to discriminate symbols if the query symbol has the same number of segments that the symbol present in the collection. Even if vectorial signatures yield good results on recognizing isolated symbols they present important weaknesses when trying to apply them in a focused retrieval application dealing with a collection of real vectorized line-drawings. Basically, one of the main problems to face is the noise and the segment fragmentation introduced by the raster-to-vector conversion process. In addition, the raster-to-vector algorithms used in this thesis do not detect arcs, but approximate them by a set of adjacent segments. The interested reader can find in the literature some works concerning the arc detection for vectorization algorithms as for instance in [Dor95, WD98]. The fact that the vectorization algorithm does not detect arcs make that symbols containing arcs and circles were not considered in the previous chapter.

In order to enhance the robustness of the spotting method, we propose to introduce a polyline approximation in the raster-to-vector algorithm as a post-processing step. Let us formally define the term polyline.

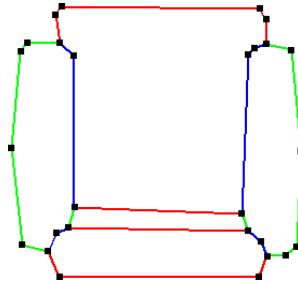
**Definition 5.1** Given a segment  $s_i = (x_1, y_1), (x_2, y_2)$ , the **adjacency**  $A(s_i) = (p, k)$  of  $s_i$  is defined as the number of segments incident in  $(x_1, y_1)$  for  $p$  and incident in  $(x_2, y_2)$  for  $k$ .

**Definition 5.2** Let  $s_1 \dots s_i \dots s_n$  be a set of sorted and adjacent segments where

$$\begin{aligned} A(s_1) &= (p, 1) \text{ with } p \neq 1 \\ A(s_i) &= (1, 1) \text{ with } i \in \{2, \dots, n-1\} \\ A(s_n) &= (1, k) \text{ with } k \neq 1 \end{aligned}$$

The **polyline**  $P_{s_1 \dots s_i \dots s_n}$  is the geometric shape considering the set of adjacent segments  $s_1 \dots s_i \dots s_n$  as an unique instance. A polyline start and end at the points were a segment  $s_n$  finishes and no other segment is adjacent to  $s_n$ , or at the points were more than two segments are adjacent.

When the segment approximation is done in the raster-to-vector step, a grouping of the adjacent segments is implemented. All the segments having  $A = (1, 1)$  form part of a polyline. Segments having  $A = (p, 1)$  or  $A = (1, k)$  with  $p, k \neq 1$  are the end-segments of a polyline. We can see an example of polyline decomposition in Fig. 5.1.



**Figure 5.1:** Polyline decomposition of a vectorized symbol. The vectorized symbol has 31 segments which are grouped into 12 polyline instances (displayed in different colors).

The main idea of treating multiple segments as an unique instance is that we should not have the restriction of the number of segments forming a symbol. Since we obtain more tolerance to the noise arising from the raster-to-vector conversion step, the jaggedness problem is not critical. The underlying problem is how to describe these polylines in terms of geometric constraints. Since polylines are ordered sets of adjacent segments, a polyline can be seen as a one-dimensional chain. We can find

some works in the literature which describe shapes by chains of adjacent segments. Let us briefly review a few.

Stein and Medioni proposed in [SM92, SM92] to describe objects by a polygonal approximation of their contours. The segments arising from the raster-to-vector conversion are then grouped into sets of adjacent segments named super-segments. These chains of consecutive segments are then described by a feature vector of geometrical attributes as the lengths or the angles of the segments, and the global orientation and eccentricity of the super-segment. In order to be tolerant to changes of the number of segments composing a super-segment, the authors propose to work at multiple cardinality scales. However, the fact of considering this cardinality scale implies an important growth of the number of feature vectors describing a shape, thus exponentially increasing the time of computing distances among shapes.

In order to provide more tolerance to changes in the number of segments composing a polyline, we can find in the literature some works which, starting from an attributed string representation of the shape, they use string edit operations to compute the distance between two strings representing two shapes. As examples, we can for instance cite the work of Wolfson presented in [Wol90], the methods of Bunke and Bühler [BB93], or the more recent work of Kaygin and Bulut [KB02]. In all these works, polygonally approximated contours are represented by attributed strings. Similar shapes are matched depending on the costs of the operations needed to edit and transform one string into the other. We use in this chapter this particular shape description and matching technique in order to be tolerant to changes in the number of segments composing a given polyline. Let us see which particular data structure we use in order to provide an efficient access to the stored descriptors.

One of the drawbacks of the method presented in the last chapter is that in order to retrieve similar symbols, all the matchings between the query descriptors and the ones of the line-drawings have to be computed. In order to face a focused retrieval application with a large document collection, we shall provide an efficient access to the descriptors by using indexing data structures. We propose in this chapter the use of a lookup table indexing structure aiming to retrieve primitives by similarity and drastically reducing the amount of comparisons to be computed. The main idea is to provide what we call prototype-based search. In prototype-based search, we are given a set of distorted samples of the same primitive and want to infer a representative model. In this context, the median concept turns out to be very useful. The use of the string matching algorithm to compute a similarity measure also allows the computation of the set median strings. Given a set of similar strings representing vectorial primitives, a representative of this set having the smallest sum of distances to all the strings in the set can be computed. These set median strings act as indexing keys of a lookup table. When performing a query, the retrieval by similarity of primitives is done efficiently since only the distances between the query primitive and the representative of a cluster of similar polylines has to be computed. The fact of avoiding a brute-force distance computation allows a fast primitive retrieval by similarity. By the use of a lookup table together with a Hough-like voting scheme, we propose in this chapter a framework able to spot graphical symbols within a document image collection.

The remainder of this chapter is structured as follows: the next section reviews the string matching theory and algorithms and provides details about our particular cost functions. Subsequently, in section 5.3 we detail the proposed prototype-based indexing framework allowing to spot graphical symbols within a document collection. Section 5.4 presents the experimental results. Finally, the conclusions and a short discussion can be found in section 5.5.

## 5.2 String Matching Theory and Algorithms

String edit distances were first defined by Wagner and Fischer in [WF74] to find out the minimum cost edit sequence to convert the string  $A$  into the string  $B$  using edit operations. Although the origin of the algorithm is spelling correction, it has been used for different purposes, and particularly as an approach to the problem of recognizing and classifying polygons. The problem is to define dissimilarity measures between polygons, and to find algorithms that compute these measures fast enough. The string matching-based approaches should be independent of the scale, translation and rotation of the polygons under analysis. Let us review the string matching theory and algorithms and subsequently provide the details about our particular cost functions to match polygons.

### 5.2.1 Definitions

Let us first introduce some basic notation and definitions of the basic string matching algorithm first proposed by Wagner and Fischer in [WF74].

**Definition 5.3** Let  $\Sigma$  be a set of elements called **symbols** and let  $\Sigma^*$  denote the set consisting of all finite **strings** over  $\Sigma$ . The length  $|A|$  of a string  $A \in \Sigma^*$  is the number of symbols in  $A$ . And let  $\Lambda$  denote the null string which has length 0.

**Definition 5.4** For a string  $A = a_1a_2\dots a_n \in \Sigma^*$ , a **cyclic shift** is a mapping  $\sigma : \Sigma^* \rightarrow \Sigma^*$  defined by  $\sigma(a_1a_2\dots a_n) = a_2a_3\dots a_na_1$ . For all  $k \in \mathbb{N}$ , let  $\sigma^k$  denote the composition of  $k$  cyclic shifts. Two strings  $A$  and  $\bar{A}$  will be called **equivalent** if  $A = \sigma^k(\bar{A})$ .

**Definition 5.5** An **edit operation**  $s$  is an ordered pair  $(a, b) \neq (\Lambda, \Lambda)$  of strings, each of a length less than or equal to 1, denoted by  $a \rightarrow b$ . An edit operation  $a \rightarrow b$  will be called an **insert** if  $a = \Lambda$ , a **delete operation** if  $b = \Lambda$ , and a **substitution operation** otherwise.

**Definition 5.6** A string  $B$  **results** from a string  $A$  by the edit operation  $s = (a \rightarrow b)$ , denoted by  $A \rightarrow B$  via  $s$ , if there are strings  $C$  and  $D$  such that  $A = CaD$  and  $B = CbD$ . An **edit sequence**  $S = s_1s_2\dots s_k$  is a sequence of edit operations. We say that  $S$  takes  $A$  to  $B$  if there are strings  $A_0, A_1, \dots, A_k$  such that  $A_0 = A, A_k = B$  and  $A_{i-1} \rightarrow A_i$  via  $s_i$  for all  $i \in \{1, 2, \dots, k\}$ .

**Definition 5.7** Let  $\gamma$  be a **cost function** that assigns a non-negative real number  $\gamma(s)$  to each edit operation. For an edit sequence  $S$ , we define the cost  $\gamma(S)$  as

$$\gamma(S) = \sum_{i=1}^k \gamma(s_i)$$

The **edit distance**  $\delta(A, B)$  from string  $A$  to string  $B$  is then defined as

$$\delta(A, B) = \min\{\gamma(S)\}$$

And the **edit distance**  $\delta([A], [B])$  of two cyclic strings  $[A]$  and  $[B]$  is given by

$$\delta([A], [B]) = \min\{\delta(\sigma^k(A), \sigma^l(B)) | k, l \in \mathbb{N}\}$$

Attributed string matching has been used for polygon matching in several applications. In order to avoid the segmentation inconsistencies due to the noisy images or distorted shapes, Tsay and Tsai introduced in [TT89] two other edit operations.

**Definition 5.8** The **split** operation is the result of splitting a symbol  $a_i$  into a sequence of  $k$  consecutive symbols, denoted as  $a_i \rightarrow a_{i1}a_{i2}\dots a_{ik}$ .

**Definition 5.9** The **merge** operation is the result of merging  $k$  consecutive symbols into a symbol, denoted as  $a_i a_{i+1} \dots a_{i+k-1} \rightarrow a'_i$ .

## 5.2.2 Linear String Matching

Let  $A$  and  $B$  be two strings over  $\Sigma^*$  of length  $n$  and  $m$  respectively. The Wagner and Fischer [WF74] algorithm takes  $\mathcal{O}(nm)$  time in find  $\delta(A, B)$  by determining a minimum weighted path in a weighted directed graph. Let  $D(i, j)$  denote the cost of a minimum weighted path from the vertex  $v(0, 0)$  to the vertex  $v(i, j)$ , so  $D(n, m) = \delta(A, B)$ . We can see below in algorithm 5.2.1 the details of linear string matching.

For the split and merge step, a window  $q \times q$  is needed. For each  $D(i, j)$  the cost of split and merge is considered as the minimum cost of all the possibilities in a region starting at the vertex  $v(i, j)$  and ending at  $v(i - q, j - q)$ . And the costs are computed as a sum of three costs. For example, in the merge case

$$\begin{aligned} \gamma(a_k a_{k+1} \dots a_{k+p} \rightarrow b_l b_{l+1} \dots b_{l+t}) &= \gamma(a_k a_{k+1} \dots a_{k+p} \rightarrow a') + \\ &\gamma(b_l b_{l+1} \dots b_{l+t} \rightarrow b') + \gamma(a' \rightarrow b') \end{aligned} \quad (5.1)$$

## 5.2.1: Linear string matching algorithm

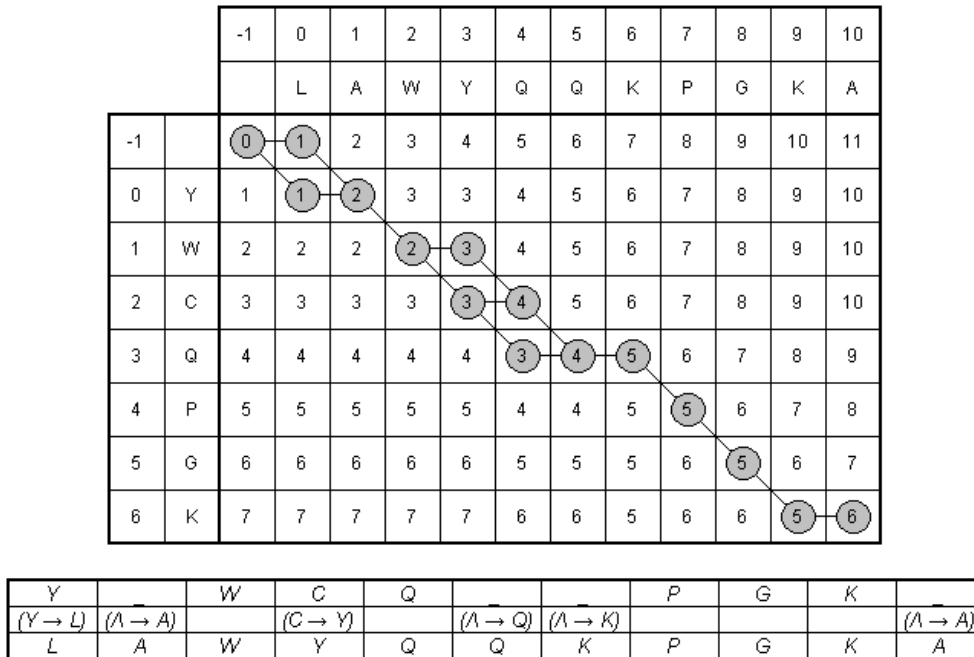
---

```

D(0,0) := 0;
for i := 1 to n do
  D(i,0) := D(i-1,0) + γ(ai → Λ);
end
for j := 1 to m do
  D(0,j) := D(0,j-1) + γ(Λ → bj);
end
for i := 1 to n do
  for j := 1 to m do
    D(i,j) := {
      D(i-1,j) + γ(ai → Λ)
      D(i,j-1) + γ(Λ → bj)
      D(i-j,j-i) + γ(ai → bj)
    }
  end
end

```

---



**Figure 5.2:** Example of the string matching algorithm. Edit operations and obtained cost to transform the string “YWCQPGK” into the string “LAWYQQKPGKA”.



### 5.2.3 Cyclic String Matching

Linear string matching can not tackle with strings having cyclic shifts since the computed paths starts always from a given initial symbol. A cyclic string matching procedure is needed in the case of cyclic strings.

Given two finite strings  $A$  and  $B$ , the cyclic string matching problem is the problem of determining  $\delta([A], [B])$  and an edit sequence realizing this cost.

Let  $BB = b_1b_2\dots b_mb_1b_2\dots b_m$  be the concatenation of  $B$  with itself. For all  $l \in \{1, 2, \dots, m\}$ , we can find a minimum cost edit sequence from  $A$  to  $\sigma^l(B)$  by determining a minimum weighted path from  $v(0, l)$  to  $v(n, m + l)$ .

Although the computation of only one path takes  $\mathcal{O}(nm)$  time, the computation of all these paths can be done in  $\mathcal{O}(nm \log m)$  time, since all the paths can be chosen such that two different paths never cross.

### 5.2.4 A String Matching Cost Function for Polygon Recognition

Visually, two chains of segments are similar if the length attributes and angles between consecutive segments can be aligned. In the literature on polygonal shape recognition, most approaches base the distance definition between two polygonal shapes on length and angle differences. For example, Arkin et al. used in [ACH91] the turning function which gives the angle between the counterclockwise tangent and the  $x$ -axis as a function of the arc length. Their results are in accordance with the intuitive notion of shape similarity.

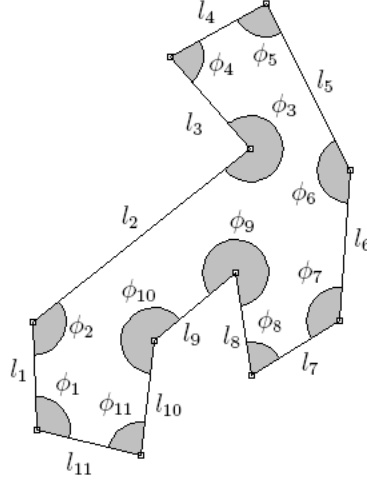
In order to use string matching for polygon recognition, we will use an attributed string matching. Starting from a polygonal approximation of the shape, we will use the segments as primitives, encoding them with a pair of numbers  $(l_i, \phi_i)$ , where  $l_i$  denotes the length of the segment  $s_i$  and  $\phi_i$  denotes the angle between  $s_i$  and  $s_{i-1}$  in the counterclockwise direction. We can appreciate an example on how these attributes are computed for a sample shape in Fig. 5.2.4

Let  $A$  and  $B$  be two chains of adjacent segments, represented as strings, with total lengths  $|A| = n$  and  $|B| = m$  and with respectively attributed string representations:

$$\begin{aligned} A &= (l_1^A; \phi_1^A) \dots (l_n^A; \phi_n^A) \quad \text{and,} \\ B &= (l_1^B; \phi_1^B) \dots (l_m^B; \phi_m^B) \end{aligned} \tag{5.2}$$

The costs functions for attributed string matching are as follows:

$$\begin{aligned} \gamma((l_i^A; \phi_i^A) \rightarrow (l_j^B; \phi_j^B)) &= \frac{|\phi_i^A - \phi_j^B|}{360} + \left| \frac{l_i^A}{|A|} - \frac{l_j^B}{|B|} \right| \\ \gamma(\lambda \rightarrow (l_j^B; \phi_j^B)) &= \frac{l_j^B}{|B|} \\ \gamma((l_i^A; \phi_i^A) \rightarrow \lambda) &= \frac{l_i^A}{|A|} \\ \gamma((l_{i,j}^A; \phi_{i,j}^A) \rightarrow (l_u^A; \phi_u^A)) &= \frac{(\sum_{k=i}^j l_k^A) - l_u^A}{\sum_{k=i}^j l_k^A} \end{aligned} \tag{5.3}$$



**Figure 5.3:** Attributed representation of a chain of adjacent segments.

which are the proposed cost functions inspired by the ones proposed by Tsay and Tsai in [TT89] where they use string matching for shape recognition. Maes proposed in [Mae91] to use a weighting factor in the length costs to compensate undesirable cost bias for angle differences. However, in our experiments we did not observe any improvement in adding such parameter. Finally, for the sake of simplicity, the previous operations are grouped by a block substitution using the merge operation. The total cost of substituting a whole sequence of symbols by another is computed as follows:

$$\begin{aligned} \gamma(A_{i,j} \rightarrow B_{k,l}) = \\ \gamma(A_{i,j} \rightarrow u) + \gamma(B_{k,l} \rightarrow v) + \gamma(u \rightarrow v) \end{aligned} \quad (5.4)$$

being  $u$  and  $v$  the segments starting at the initial point of  $A_i$  and  $B_k$  and ending at the final point of  $A_j$  and  $B_l$  respectively.

As all the length comparisons are weighted by the total perimeter of the chain of segments and the angles are computed relatively to the previous segment, the proposed string matching approach is rotation and translation invariant. In addition, the merge operation attributes low edit costs to primitives undergoing noisy transformations as the inherent segment fragmentation from the raster-to-vector process and aims to compare strings with different number of segments making the system tolerant to segment cardinality and to scale changes.

### 5.3 Spotting Method

Given a technical document, the main idea of this chapter is to organize clusters of similar string primitives in a lookup table. Each entry of this table is indexed by a

representative string allowing to use a graphical pattern as query and avoiding the computation of distances over all the stored primitives.

We divide the spotting method in four different parts. First primitives are extracted from the symbol instances in terms of strings attributed with geometric constraints among their segments. Then, the off-line step to build the indexing data structure to organize the primitive descriptors. Third, the on-line process of formulating a graphical query and the search in the data structure for similar primitives. Finally, the Hough-like voting scheme which spots the zones of interest where there is high probability to find the symbol. Let us further describe the above steps.

### 5.3.1 Symbol Representation in Terms of String Primitives

The first step to consider in any symbol recognition methods using geometric constraints as descriptors is the preprocessing step allowing to decompose into primitives the target documents as well as the queries. Let us briefly explain these pre-processing steps of primitive extraction.

We use in this chapter the same raster-to-vector algorithm implementation presented in the last chapter (see section 4.2) with a slight modification. Rather than representing symbols with a polygonal approximation based on a skeletonisation, our choice for this chapter is focused on computing a closed region labelling and extraction based on a connected component analysis. The contours of these closed regions are then polygonally approximated using the Rosin and West's [RW89] algorithm.

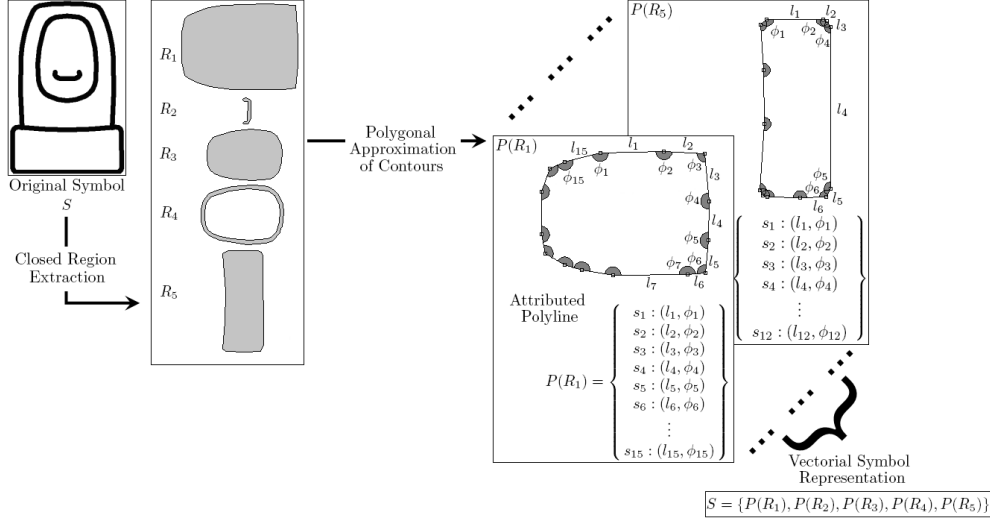
After computing the polygonal approximation of the contours of the closed regions, an association of chains of adjacent segments resulting in a polyline is done. These polylines are encoded as attributed strings, and used as primitives to describe the symbol to be recognized.

Formally, let  $R$  be the contour of a closed region which is polygonally approximated and represented by the chain of adjacent segments  $P(R) = \{s_1 \dots s_n\}$  consisting of  $n$  segments  $s_i$ . As we have seen in the previous subsection 5.2.4, each segment  $s_i$  is attributed with the tuple  $(l_i, \phi_i)$ , where  $l_i$  denotes the length of the segment  $s_i$  and  $\phi_i$  denotes the angle between  $s_i$  and the previous segment  $s_{i-1}$  in the counterclockwise direction. A symbol is then described in terms of its composing  $p$  region contours and denoted as  $S = \{P(R_1) \dots P(R_p)\}$ . We can see a graphical example in Fig. 5.4.

The distance between two polylines is computed by using the string matching algorithm and the particular cost functions defined in the previous section. The final distance between two symbols is then computed as the sum of distances among the corresponding primitives. Let us focus the next section on the primitive description organization and the indexing structure construction.

### 5.3.2 Off-line Lookup Table Construction

We propose in this section the use of an indexing structure which allows a fast primitive retrieval by similarity. The main idea is to cluster similar polylines into entries of



**Figure 5.4:** Symbol representation in terms of polygonal approximation of closed region contours. Each of these region contours are represented by strings attributed by length and angles.

a lookup table. The retrieval of primitives by similarity is done by a prototype-based search. Each entry of the lookup table is identified by a representative of a cluster of similar primitives. When querying this lookup table, a list of locations where to find similar primitives is obtained without computing the distance between the query and all the primitive instances, but just by computing the distance between the query and the cluster prototypes. In order to correctly identify the lookup table entries, a representative of the clusters of primitives has to be computed.

Each lookup table entry representing a cluster of similar strings appearing in the document collection, consists of two different items: a representative polyline of each cluster which acts as indexing key and the stored list of locations where we can find the polylines belonging to this cluster. If strings are used for representing the objects under consideration, then we are faced with the task of finding the median of a set of similar strings. Let us see how can we compute the representative string from a set of similar strings.

Generally speaking, the representative polyline of each cluster can be computed in two ways, namely the *mean string* or the *set median string*. As proposed in Sánchez et al. in [SLT02], the mean string  $M$  over a string cluster  $C = \{A_1 \dots A_n\}$  is defined as:

$$M = \arg \min_{M \in \Sigma^*} \left( \sum_{i=1}^n \delta(A_i, M) \right) \quad (5.5)$$

The mean string is computed as a new string that represents the average shape

among all the strings in the set. The main drawback of this approach is its computational cost which increases with a large number of shapes. On the other hand, the set median string  $\widetilde{M}$  is defined as the string in a given set minimizing the sum of distances to all the strings in the set. The set median string is defined as:

$$\widetilde{M} = \arg \min_{\widetilde{M} \in C} \left( \sum_{i=1}^n \delta(A_i, \widetilde{M}) \right) \quad (5.6)$$

In our case, we have experimentally verified that a set median string is useful enough to be used as index of a table entry. Besides, it is less expensive since we do not need to compute a new string being an exact shape average of the set, but to select it between the strings composing the cluster. Let us see how, from the set median string formalism, we can build an indexing structure aiming to retrieve by similarity stored primitive strings.

The lookup table is built as follows. For all the polylines  $P(R_i)$  appearing in a document of the collection, we store its location in the lookup table. In order to be able to query the indexing structure by similarity of graphical patterns, we should select a cluster of similar polylines where it belongs to. The selection of this cluster is done by applying the string matching algorithm proposed above. We select the cluster where the cost of editing the string  $P(R_i)$  to match the set median string  $\widetilde{P}_C$  of the cluster is lower than a threshold  $thr$ . Once we identified the corresponding cluster, we add  $P(R_i)$  to it. The set median string  $\widetilde{P}_C$  of the corresponding cluster is recomputed in order to keep offering a good cluster representative. If no cluster has a set median string similar to  $P(R_i)$ , then we define a new cluster having  $P(R_i)$  as representative. The set median strings act as indexing keys of the lookup table where at each entry a list of translation vectors  $\vec{v}_i = (x_i, y_i, d_i)$  are stored. Where  $(x_i, y_i)$  are the coordinates of the middle point of the polyline, and  $d_i$  identifies the corresponding document in the collection where  $P(R_i)$  appears. We can see the details in the algorithm 5.3.1:

---

5.3.1: Algorithm to build a LUT from a list of primitives.

---

```

for  $i = 0$  to  $length(R)$  do
  if  $LUT[P(R_i)]$  is not  $NULL$  then
     $LUT[P(R_i)].AddValue(\vec{v}_i)$ ;
     $LUT[P(R_i)].UpdateKey()$ ;
  end
  else
     $LUT[P(R_i)].CreateNewPos(\vec{v}_i)$ ;
  end
end

```

---

When applying the algorithm described above, the order followed to add polylines in the lookup table is important since the primitive clustering is done in an incremental way. However as in retrieval applications the user can add more and more documents

to the database at any time, we preferred to use an incremental primitive clustering than a classical classification method which would need a learning stage, making difficult to increase the data collection. In addition, the coarse primitive clustering offered by the lookup table is compensated by the use of a voting scheme.

Let us detail in the next subsection how we proceed to use the lookup table in order to retrieve the location of graphical primitives by similarity.

### 5.3.3 On-line Querying of Symbols: Activating Table Entries

Given a query symbol  $S = \{P(R_1) \dots P(R_p)\}$  and the lookup table containing  $q$  entries, a maximum of  $p$  table entries are activated resulting on the one hand in a list of locations where to find similar primitives and, on the other hand, in a confidence value depending on the similarity ratio between the query primitives and the prototypes representing these table entries. A table entry having as prototype a certain median string  $\widetilde{P}_j$  is activated depending on the following condition:

$$\begin{aligned} \delta(P(R_i), \widetilde{P}_j) < thr \\ \text{where } 1 \leq i \leq p \text{ and } 1 \leq j \leq q \end{aligned} \quad (5.7)$$

From the traversal of the lookup table, we obtain a list of locations where to find similar primitives than the ones that compose the query symbol. The zones of a document in the collection likely to contain the query symbol are the ones where we can find more accumulation of the symbol's primitives. By using a Hough-like voting scheme, we determine the zones of the collection where we have more accumulation of primitives by simply looking at maxima at the voting space. The locations where there is a presence of most of the polylines composing the symbol  $S$  form clusters of coherent votes. The presence of similar polylines in other locations of the line drawing provokes false positive votes which are scattered into the voting space. The accumulation of evidences is done in terms on the similarity between the query primitive and the prototype one, so the values of the votes to distribute in the parameter space are proportional to each  $\delta(P(R_i), \widetilde{P}_j)$ . Let us detail in the next section how we proceed to validate the location hypotheses.

### 5.3.4 Hough-like Voting Scheme to Validate Location Hypotheses

The voting space is a four-dimensional space  $(x, y, s, d)$  consisting of  $2D$  position coordinates, a scale ratio and an index of a given document in the collection. Given a query string  $P_q$  we accumulate votes in the translation coordinates  $\vec{v}_i = (x_i, y_i)$  of the corresponding line-drawing image  $d$ . The third dimension of this space represents the scale factor between the query polyline and the polylines stored in the lookup table. This voting scheme formulates hypotheses of spatial location, document instance, and scale of the queried symbol. The zones of the line-drawing where we find similar primitives at a similar scale that the ones that form the query symbol tend to accumulate more votes and thus to form clusters in the voting space. The problem of

finding zones where a symbol is likely to be found is then reduced to a local maxima localization problem in the voting space.

For the sake of simplicity the voting space is split into several bins,  $I_{(1,1,1)} \dots I_{(m,n,s)}$  named buckets for each document  $d$ . The bin size has to be related to the scale of the symbol so the different votes fall in nearby buckets. In our experiments the grid size has been empirically set and is determined in terms of the size of the original image. In our case,  $m$  and  $n$  are determined such as  $\max(m, n) = 128$ , preserving the aspect ratio of the original image. The scale dimension is sampled to  $s = 8$  possible buckets. The parameter  $d$  is directly the number of documents stored in the collection. Following a similar idea than the proposed by Lorenz and Monagan in [LM95] we use a voting method known in signal processing as anti-aliasing to relate  $(\vec{v}_i, s)$  to a set of  $I_j$  neighboring buckets, based on the Euclidean distance between the voting location and the discrete bin partition buckets. Each  $(\vec{v}_i, s)$  has the edit cost vote to distribute among its eight neighboring buckets, depending on their proximity.

Given a symbol  $S$ , the activation of the lookup table entries results on a list  $L = \{(\vec{v}_1, s_1) \dots (\vec{v}_n, s_n)\}$  of translation vectors. Being  $d((\vec{v}_i, s_i), I_j)$  the Euclidean distance between  $(\vec{v}_i, s_i)$  and one of the eight neighboring buckets  $I_j$  we define the value of the vote  $V(I_j)$  received in the bucket  $I_j$  is accumulated as:

$$V(I_j) = V(I_j) + \frac{w_1}{d((\vec{v}_i, s), I_j)} + \frac{w_2}{\delta(P(R_i), \overline{P_j})} \quad (5.8)$$

Where  $w_1$  and  $w_2$  weight the distance factor and the edit cost. We can see an example of the voting distribution scheme in Fig. 5.5.

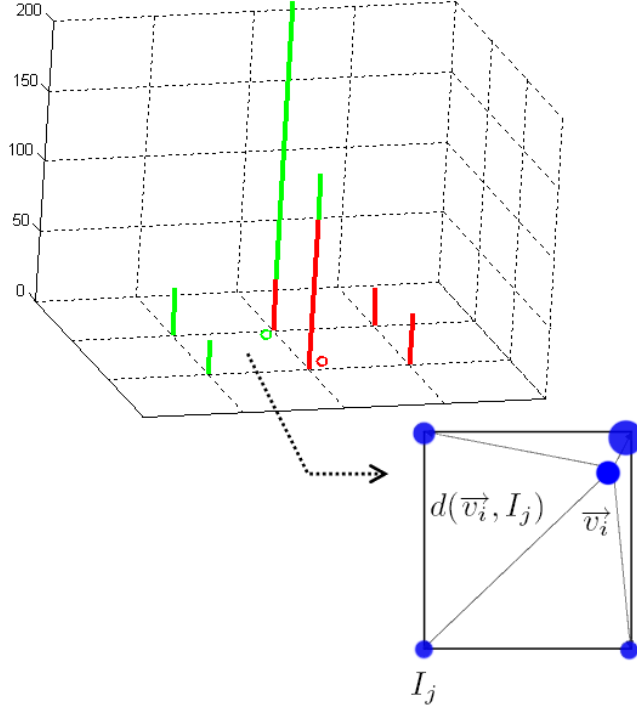
As we can see, the anti-aliasing method reduces the problem of working with a discrete grid where to distribute votes. Voting schemes are only efficient if a high number of votes fall in the right bin, so that the bin can be easily detected among the background noise. If some votes fall in the neighboring bins, the significance of the correct bin decreases. Since the votes are now distributed among nearby buckets, even if the locations of a symbol do not fit a unique bin, the votes of close buckets collaborate between them.

Since the intended application of the spotting methods is a focused retrieval process, given a query symbol, the top  $k$  zones of interest in terms of accumulated votes are returned to the user. The more primitives a symbol has, the more votes can be accumulated in a given zone. However, since only one query is done at the same time, there is no need to normalize the votes to retrieve the zones of interest.

Let us see in the next section the experimental results testing the performance of the proposed description technique, and the ability of locating and retrieving symbols within a collection of complete documents.

## 5.4 Experimental Results

In order to evaluate the proposed spotting methodology we present three different experiments. The first one only focuses on the string matching algorithm as a distance



**Figure 5.5:** Anti-aliasing method to cast votes. Even if two votes fall in different buckets due to the discretization, they still contribute to form coherent peaks in the desired values of the voting space.

measure between polygonal shapes. It aims to empirically determine a well suited threshold value *thr* which determines whether two strings are considered similar or not. The second experiment is designed to test if the proposed primitives are sufficiently discriminative to represent a graphical symbol. Finally, the third experiment tests the symbol spotting method by querying a document image database of real architectural floor-plans.

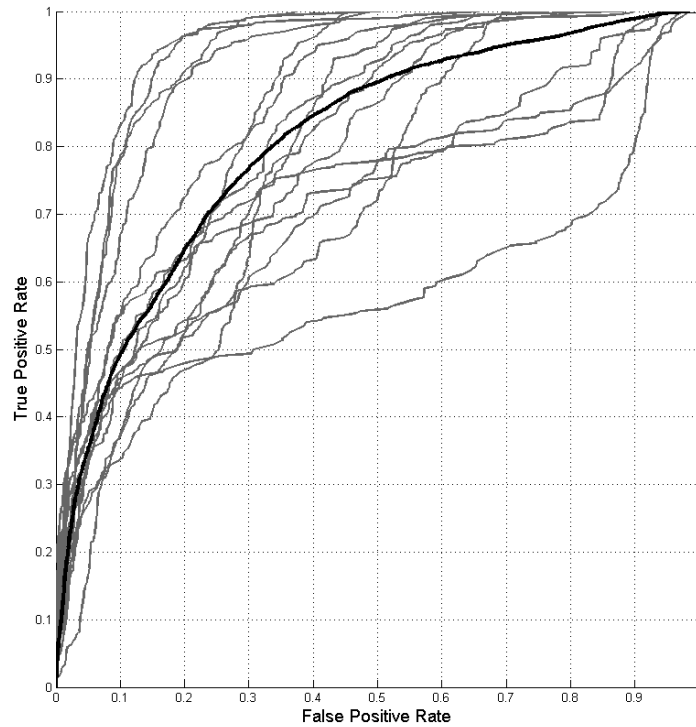
#### 5.4.1 Silhouette Shape Matching

The first experiment is designed to test the efficiency of the string matching algorithm as a shape descriptor. The algorithm is used as a distance between two shapes represented by a polygonally approximated contour. This experiment also aims to empirically determine a well suited value of the threshold *thr* which determines whether two polylines are similar or not. We used a subset of isolated silhouette shapes from the MPEG-7 core experiment described by Latecki et al. in [LLE00]. We call this polygonal shapes collection the MPEG-POLY database.

For each one of the 15 shape models, the noise model presented by Kanungo et



al. in [KHP93] is applied to generate 300 degraded images per class, which are then polygonally approximated. The fact of applying the noise model and then converting the images from raster to vector format, introduces a lot of variations in the number of segments approximating a given silhouette. All the details of this dataset can be checked in the appendix A. With all this dataset we run a classification experiment. The distance between each model and the vectorized shapes is computed by using the cyclic string matching algorithm with the proposed cost functions. These results are sorted by increasing distance to extract a Receiver Operating Characteristic (ROC) curve (the interested reader is referred to paper by Fawcett [Faw06] on ROC analysis), which plots the true positive rates against the false positive rates. These evaluation metrics are the same than we used previously to evaluate the performance of the document classification method of chapter 3. We can see how they are computed in eq. 3.9.



**Figure 5.6:** Receiver Operating Characteristic curve for the silhouette matching experiment. Average ROC curve is shown in black.

We can appreciate in Fig. 5.6 the tradeoff between the correctly classified items and the appearance of false positives. In our framework, as we use a voting scheme to accumulate evidences, we are more interested in achieving high true positive rates values rather than having low false positive rates. We can find in Table 5.1 the obtained false positives rates and thresholds for different true positives rates.

**Table 5.1:** Obtained false positive rates and decision threshold  $thr$  for several true positive rates.

$TPR$	$FPR$	$thr$
0.25	0.025	0.008
0.5	0.105	0.014
0.75	0.281	0.024
0.9	0.511	0.035

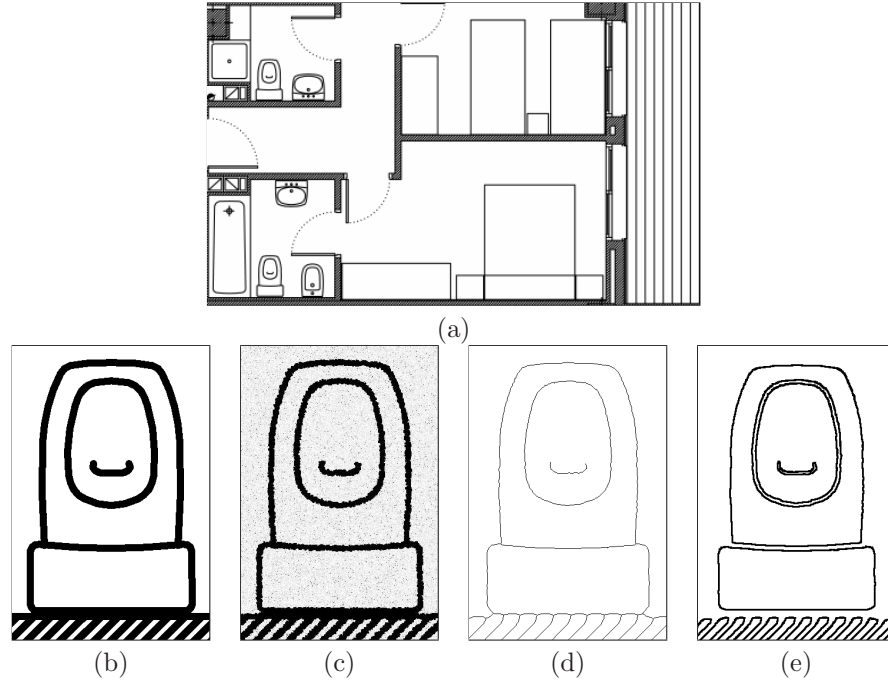
In the presented spotting method the lookup table offers a coarse clustering that is then refined by the use of a voting scheme. The presence of false positives in a lookup table entry is not a problem but we want to minimize the missed primitives. In our experiments, we used a  $thr$  value of 0.03 which guarantees about a 75% of correctly clustered shapes in a given lookup table entry. False positives appear but the voting strategy will hopefully discard them.

#### 5.4.2 Evaluation of the Contours as Primitives

The second test aims to see if the region contours are better primitives to represent a graphical symbol than the skeletons, which were the extracted features to be polygonally approximated in the last chapter. We compare the performance of the presented method by using both vectorization strategies, one computing the skeletons of the objects and then applying the Rosin and West’s algorithm, and the other which tries to approximate the contours of the closed regions extracted from the image. To carry this experiment, a real floor-plan has been degraded to build a collection of 500 synthetically distorted plans by using again the noise method of Kanungo et al. These distorted images are then polygonally approximated with both representations: contours and skeleton primitives. We can appreciate the differences between both primitives in Fig. 5.7.

In Fig. 5.8 we can see the obtained precision and recall graph (the interested reader is referred to van Rijsbergen book [vR79] on information retrieval) when querying a symbol. The graph shows that the presented primitives are more expressive than the use of skeletons since the spotting method using this representation outperforms the skeleton in all cases. In average there is a gain near a 17.5% of precision for the same recall values. Details are shown in Table 5.3, where we can see the number of false positives we have when requesting a certain number of the 500 possible solutions. In addition, using contours as primitives, in only 5 of the 500 images the queried symbol has been missed and using the skeleton we miss the symbol in 73 of the 500 images. This yields to a significant gain in the recall value when using contours instead of approximating the symbol skeleton.

Finally, we tested the method in a database of isolated symbols affected by vecotirnal noise. We can see in Fig 5.9 the obtained ROC curve for the matching experiment with the 150 isolated symbols from the GREC-POLY database (fully detailed in

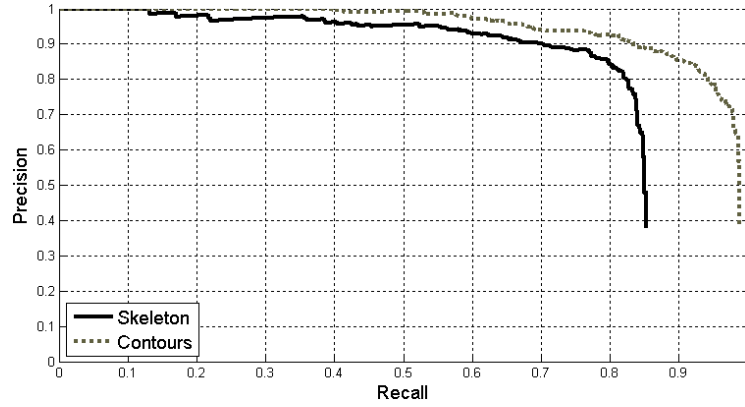


**Figure 5.7:** Symbol Primitive Representations. (a) Model floor-plan; (b) zoom of the toilet symbol; (c) degraded image; (d) symbol with skeleton primitives; (e) symbol with contour primitives.

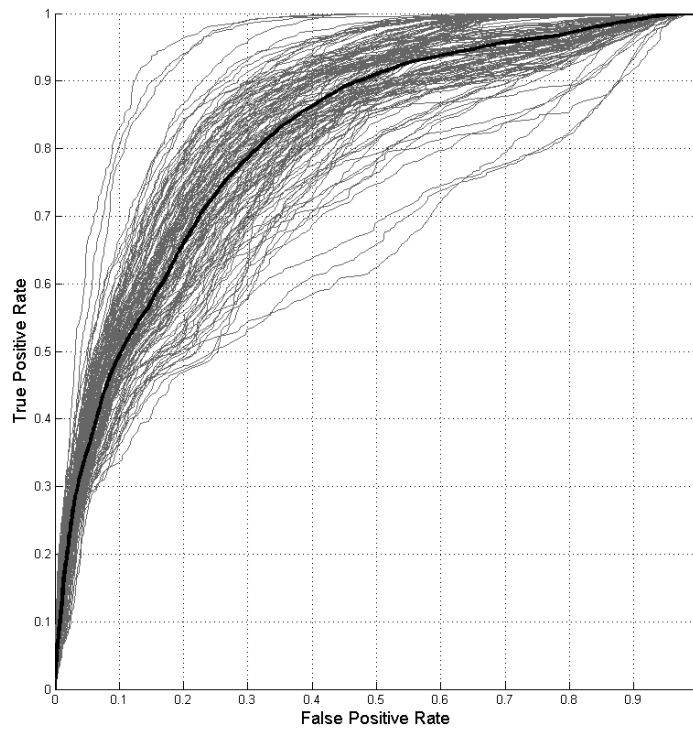
**Table 5.2:** Number of false positives when requesting a certain number of retrieved zones.

Primitives	Retrieved Items			
	200	300	400	475
False positives with Contours	6	7	29	159
False positives with Skeletons	73	76	89	273

appendix A). The experimental setup is the same than the silhouette matching experiment. The main difference is that we do not try to match individual shapes but all the primitives composing a given graphical symbol. When all the primitives from a symbol are matched against the model primitives, the symbol is then considered as recognized.



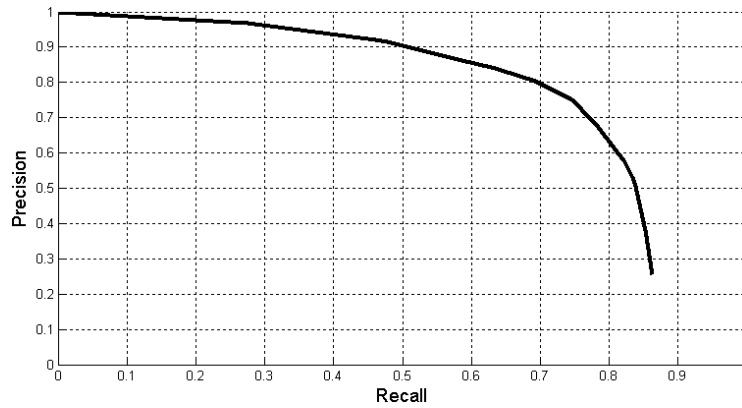
**Figure 5.8:** Precision and recall plot when spotting the toilet symbol shown in Fig. 5.7 using two different symbol primitives. The contours outperforms in both precision and recall the skeleton primitives.



**Figure 5.9:** Receiver Operating Characteristic curve for the symbol matching experiment. Average ROC curve of the 150 symbols is shown in black.

### 5.4.3 Symbol Spotting in a Document Database

Finally, we tested our method with a collection of ten real floor-plans and ten different symbols as queries. This is a subset of the FPLAN-POLY dataset detailed in appendix A. The query symbols appear in the floor-plans several times and are segmented by cropping a zone in the floor-plan image and vectorizing it. Each floor-plan has been polygonally approximated and ground-truthed. The database consist on approximately 14200 polylines which after the lookup construction result in near 320 table entries. The amount of distance computations is thus reduced by a factor of 45 with respect to a sequential access to all the primitives appearing in the collection. We can see in Fig. 5.10 the precision and recall plot resulting from spotting these symbols in the whole floor-plan database.



**Figure 5.10:** Precision and recall plot for symbol spotting in the document database.

In Table 5.3 we present a detailed set of measures to evaluate the performance of retrieval systems which aim to evaluate the spotting architecture. As we can see, the recall ratio is quite good. However there is an important number of false positives in the results which harm the precision value. The  $F$ -score is a composite measure which aims to rank the results. However, the most interesting point here is to notice the difference between the precision and the average precision  $AveP$  values. The average precision is a measure of quality which rewards the earliest return of relevant items. As we can see, even if in our experiments the precision values are quite low, the average precisions are significantly higher. That means that usually the false positives are ranked worst than the correct results, as we can also see in the qualitative results shown in Fig. 5.11. Finally, we also show the average time taken by our software prototype to spot a symbol per plan. It is remarkable that usually the symbols which are composed by common simple primitive shapes (circles, squares, etc.) are the ones which are more time consuming since the entries of the lookup are more populated and more hypotheses have to be considered. No significant differences due to the number of polylines composing a symbol can be appreciated.

**Table 5.3:** Detailed retrieval measures for each model symbol for the symbol spotting in a document database experiment.

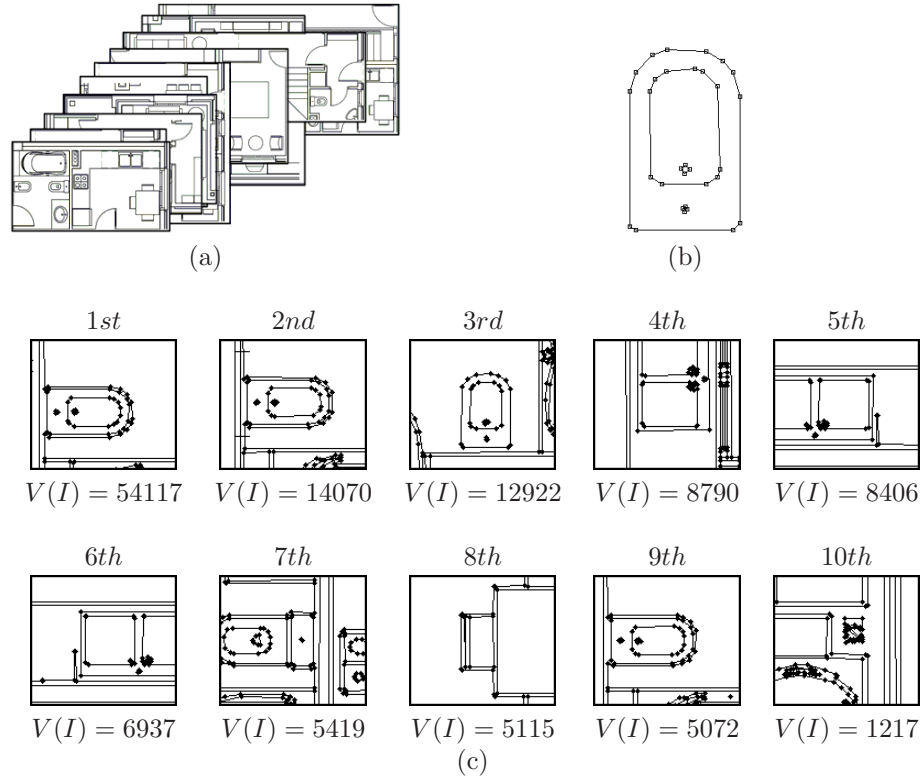
Symbol		Retrieval Measures				
Class	$p$	Precision (%)	Recall (%)	$F$ -score (%)	$AveP$ (%)	Time (secs./plan)
Bidet	4	30.8	100	47.1	87.5	0.76
Chair	5	36.8	100	53.8	83.3	0.64
Burners	9	5.1	100	9.6	59.1	1.09
Toilet	5	50	37.5	42.9	27.1	0.98
Toilet sink	5	30	100	46.2	68.7	1.89
Kitchen sink	5	11.8	50	19	33.3	1.16
Single sofa	4	37.5	100	54.6	100	0.43
Double sofa	6	15	75	25	65	0.22
Table	7	16.7	100	28.6	100	0.24
Tv set	4	20	100	33.3	95	0.12
AVERAGE	5.4	25.4	86.2	36	71.9	0.75

## 5.5 Conclusions and Discussion

In this chapter we have presented a method of symbol spotting and its use in a focused retrieval application from a collection of technical line-drawings. First a suitable symbol representation as a set of closed region contours and its codification with attributed strings has been presented. The distance definition using a cyclic string matching algorithm allows to tolerate the segment fragmentation problem. Then, a clustering of salient zones of interest and a voting method have been presented and tested to spot symbols in real technical line-drawings.

The experiments show that the representation and distance approaches are able to tackle with the inherent noise arising from the scanning process and the distortions introduced by the raster-to-vector algorithms. The presence of false positives is not a critical problem since the purpose of spotting methods is to find by a fast technique a coarse identification of zones where a given symbol appears. Finally, we can see that the use of voting strategies are of vital importance for spotting problems. To reach higher precision one can use better shape descriptors, however this also entails a complexity increment. The accumulation of evidences allows to work with a coarsely recognition in the indexing step.

There are still some aspects which should be further studied. The main concern is that the order followed to add polylines in the lookup table is important and in some cases could lead to some misclassifications. However for spotting applications where the user can add more and more documents at any time, the primitive clustering must be incremental. The use of incremental classifiers such as iPCA [AJL02] or iLDA [POK05] applied to primitive clustering should be studied. On the other hand, the presented matching approach can not cope with occlusions which will provoke the



**Figure 5.11:** Qualitative results for symbol spotting by cyclic string matching. (a) Vectorized floor-plan database; (b) query example; (d) ranked top ten results.

polylines to be broken. A partial matching algorithm as the one presented in [TVH05] by Tănase et al. could be helpful in such situations.

One of the main advantages of the proposed method regarding the vectorial signature approach, is the use of a prototype-based search. This indexing technique provides an efficient way to retrieve the locations of graphical patterns by similarity. Even if the implementation of the method is still a prototype and has not been optimized, the times to retrieve the occurrences of a symbol given in Table 5.3 are encouraging. However, depending on the applications, the number of table entries may increase drastically, and even the computation of the distance with the prototypes can be time consuming. The use of a hashing structures instead of lookup tables, can provide a more efficient access to the data without computing any distance with prototype primitives. We propose in the next chapter the use of this particular data structures aiming a faster primitive retrieval by similarity.

There is another drawback in the presented approach. since graphical symbols are composed of several primitives, querying a symbol consists in separately querying each of its primitives. The locations showing a higher accumulation of primitives

are the most plausible hypotheses to contain the queried symbol. The structural configuration of these primitives in that location is not taken into account. Most of the false alarms presented in the qualitative results do not contain any similar symbol than the query. But in those regions, usually there is a high presence of simple primitives, and these locations accumulate several votes. We present in the next chapter an indexing methodology aiming add structural information in the primitive queries. An enhanced voting scheme aiming to better validate the spotted locations is also presented in the next chapter.

Finally, our feeling is that, representing the primitives as attributed strings is a powerful description technique. The retrieval by similarity with this particular data representation has however an important burden compared with descriptors working with a feature vector description. Symbolic descriptions are meaningful but computationally expensive to match. Numeric-based descriptions are usually less expressive, but easier to match by just the definition of a distance. We will use in the next chapter several off-the-self numerical description techniques instead of symbolic ones, in order to foster the retrieval by similarity.



# Chapter 6

## A Relational Indexing Method for Symbol Spotting

---

In this chapter we present a method to retrieve from a collection of document images the regions of interest where a query symbol is likely to be found. In order to foster the querying speed, a hashing technique is proposed which is able to retrieve primitives by similarity very efficiently. Vectorial primitives are coarsely encoded by well-known shape description methods providing a numerical description of the primitives. A relational indexing approach is presented in order to introduce some structural information of the symbols and provide an accurate hypotheses validation. Experimental results show the performance of the proposed approach.

---

### 6.1 Introduction and Related Work

The use of a lookup table providing a prototype-based search of similar primitives, as presented in the last chapter, allow to avoid the computation of the similarity measure for all the primitives extracted from the collection. The use of such indexing structures aims to efficiently access and to retrieve graphic elements by similarity, and becomes a must when dealing with applications which have to face large collections of documents. In the particular use case presented in the last chapter, we achieved to reduce the amount of distance computations by almost a factor of 45 without missing an important number of symbols. However, there is still need to compute several hundreds of distances between descriptors. Even if this is not an important burden when working with numeric descriptors, it may be an important inconvenient when we use symbolic description of primitives as the attributed strings. We propose in this chapter to enhance the accessibility to the stored descriptors by two means. First, we will coarsely describe primitives by the use of well-known descriptors with low dimensionality. These descriptors result in a numeric feature vector. The distance among those descriptors is easily computed as the distance between two points in the  $n$ -dimensional description space. Second, this description space is efficiently organized

and accessed by the use of a hashing technique. The use of hashing techniques allow in ideal conditions to retrieve items by similarity with a complexity  $\mathcal{O}(1)$ . We can find in the literature many works which use such efficient indexing structures to organize and retrieve the primitive descriptors. Califano and Mohan used in [CM94] a hash table indexed by four-dimensional indices describing the geometric configuration of triplets of points extracted from a contour image in order to efficiently locate in an image the location of query objects. Stein and Medioni also used a Hash table in [SM92] in order to provide an efficient retrieval of similar portions of a contour described by a set of features extracted from a super-segment. Recently, Lladós and Sánchez proposed in [LS07] a binary codification of the shape context descriptor which is stored in an indexing structure aiming to efficiently retrieve the locations within a document image where a given typewritten word is likely to appear.

Moreover, there is another drawback in the previously presented method. Since graphical symbols are composed of several primitives, querying a symbol consisted in separately querying each of its primitives. The locations showing a higher accumulation of primitives were taken as the most plausible hypotheses to contain the queried symbol. This technique may lead to several false alarms since we are not checking which primitives appear in those zones and whether their spatial organization and their structural configuration is consistent with the query symbol design. We propose in this chapter an indexing methodology aiming add structural information in the primitive queries. We can find some works in the literature as the one proposed by Chang and Lee. in [CL91] or the one by Costa and Shapiro in [CS00], which are focused on the addition of structural information to the primitive querying process. We can call this kind of approaches relational indexing, since besides the fact of indexing primitive objects, these works try to index also their spatial relationships. An enhanced voting scheme aiming a better validation of the spotted locations is also presented in this chapter.

The remainder of this chapter is structured as follows: we first start by detailing how the symbols are represented in terms of a polygonal approximation of contours and a relational graph. Subsequently, in section 6.3 we present the off-the-shelf shape descriptors we have used in our experiments to coarsely describe and index the primitives by similarity. Even if some of the descriptors were conceived to describe images, they are reformulated to be applied to a set of polygonal primitives. Section 6.4 presents the indexing structure aiming to efficiently retrieve primitives and section 6.5 outlines how the relational indexing methodology works. In section 6.6 we present some qualitative results in using the proposed spotting architecture to retrieve locations of interest from a collection of line-drawing images. Finally, the conclusions and a short discussion can be found in section 6.7.

## 6.2 Description of Graphical Symbols in Terms of Vectorial Primitives

Recognition schemes rely on two basic steps namely primitive extraction and description. First, the primitive extraction step has to transform the image drawings

arising from the scanning process to a vector domain. Then, in the second step, such primitives have to be represented by a shape descriptor.

### 6.2.1 Vectorial Primitives

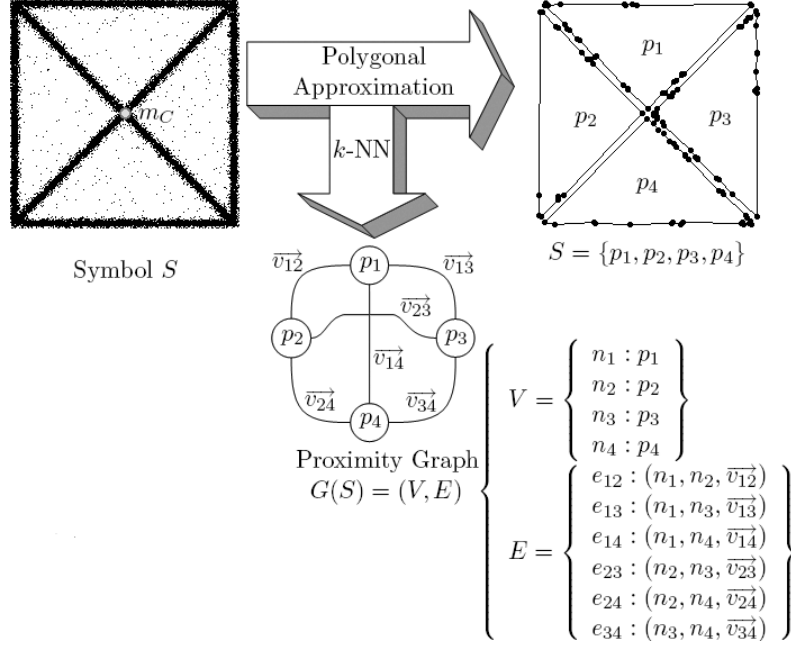
Graphical symbols are usually composed by the union of several simple sub-shapes. According to that, a symbol can be described in terms of the assembly of sub-shapes which composes it. The basic primitives we want to extract to represent a graphical symbol are these simple sub-shapes.

As our work is focused on the management of graphical data in vectorial format the documents which are in paper format need a digitalization process. We use in this chapter the same raster-to-vector process than in the previous chapter with just one particularity. Since we want to add relational information between primitives to the indexing framework, a graph representation of the symbols is also needed. The documents are scanned and de-noised by some simple morphological operations. The raster-to-vector algorithm proposed in [RW89] is then applied to these line-drawing images to obtain a vectorial representation of the documents. However, vectors as it, are not suitable to be used as primitives due to its instability in terms of artifacts, fragmentation, errors in junctions, etc. A higher level entity has to be used as primitive. Adjacent vectors are merged together into a polyline instance. These polylines represent then the sub-shapes conforming a given graphical symbol. In our method, we use the contour of the closed loops conforming a symbol as the primitives to polygonally approximate and to merge as single polylines.

Formally, let  $p = \{s_1 \dots s_n\}$  be a polyline consisting of  $n$  segments  $s_i$ . A symbol is represented in terms of its polylines representing loops and denoted as  $S = \{p_1 \dots p_m\}$ . The gravity center of the symbol is computed as the average of the gravity centers of each polyline, and it is denoted as  $m_C$ . The gravity center of the symbol will be used in the subsequent process of localization of the query symbol inside the line-drawing images. To represent the spatial organization of primitives which compounds a symbol, a proximity graph is constructed. Using the  $k$ -NN algorithm, each primitive is linked to its  $k$  nearest primitives by an edge of the graph  $G(S) = (V, E)$ . A node  $n_i \in V$  is attributed with the primitive  $p_i$ . An edge  $e \in E$  is denoted as  $e = (n_i, n_j, \vec{v}_{ij})$  where  $n_i$  and  $n_j$  are nodes of  $V$  and  $\vec{v}_{ij}$  is a vector representing the spatial relationship between the primitives  $p_i$  and  $p_j$ . This proximity graph is the basis of the proposed relational indexing technique.

We can appreciate in Fig. 6.1 how the different parts of a symbol are detached making the regions meaningful primitives, and how their spatial organization can describe a symbol.

Note that the same primitive representation and extraction is used for the complete documents in the acquisition step. A given document  $D$  is composed by a large number of polylines. A proximity graph  $G(D)$  is also computed to link nearby primitives and store their spatial relationship. Obviously, in this case we do not know which polylines compose a symbol, the graph just represents neighboring primitives.



**Figure 6.1:** Primitive symbol decomposition. A graphical symbol is decomposed in sub-shapes which are polygonally approximated. An attributed proximity graph is the basis for the relational indexing.

The polygonally approximated sub-shapes are used as the local components of a given symbol. To describe them, we apply at each primitive separately one of the off-the-shelf global numerical shape descriptors existing in the literature.

### 6.3 Off-the-shelf Shape Descriptors Applied to Vectorial Data

Formally speaking, given a symbol  $S = \{p_1 \dots p_m\}$  and a shape descriptor  $f$  defined over the space of primitives, after applying  $f$  to each primitive we will have in return a set of feature vectors  $f(p_i)$  for all  $i \in [1, m]$ . A symbol is then expressed by a set of feature vectors describing its conforming primitives. Let us briefly review in the next section the used shape descriptors.

Global numerical shape descriptors are formulated in terms of a compact representation of expressive invariant features describing a shape as a whole. The interested reader is referred to Zhang and Lu's [ZL04] review of shape representation and description techniques. In this section we will summarize the global shape descriptors used in our experiments. We make no claims about robustness of the chosen descriptors. Depending on the nature of the data better descriptors can be used. The point

here is only to test several shape descriptors seen as black-boxes which one can plug-in into the system. The selection of one or another shape descriptor is application dependent. For example, if we are interested in retrieve just correct symbols despite missing some positives, an accurate shape descriptor has to be chosen. On the other hand, if the user wants to retrieve all the instances of a given symbol without giving really importance to the presence of false positives, one must choose a simpler shape descriptor. Four shape descriptors with different accuracy are chosen here to test the behavior of the system.

Let us further overview the numerical shape descriptors used in our work. Firstly we introduce some basic notation. We consider an image  $I(x, y)$  containing an object shape  $O$  with area  $A$  and perimeter  $P$ . Its centroid is the point  $c = (\bar{x}, \bar{y})$ . The boundary  $B$  of the shape is polygonally approximated by a polyline  $p_O$  composed by a set of  $n$  adjacent segments  $s_i = \{(x_i, y_i), (x_{i+1}, y_{i+1})\}$ . A shape descriptor will result in a compact representation of the shape formulated in terms of a feature vector  $f(O)$ . Let us briefly introduce the well-known shape descriptors we use.

### 6.3.1 Moment Invariants

The central  $(p + q)$ th order moment for a digital image  $I(x, y)$  is expressed by

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I(x, y) \quad (6.1)$$

The use of the centroid  $c = (\bar{x}, \bar{y})$  allow to be invariant to translation. A normalization by the object area is used to achieve invariance to scale.

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^\gamma} \quad \text{where } \gamma = \frac{p+q}{2} + 1 \quad (6.2)$$

#### Boundary Moments

The geometric moments can also be computed among the contour of the object as introduced by Chen in [Che93] and by Sardana et al. in [SDI94] by using eq. 6.1 only for the pixels of the boundary of the object. In that case, a normalization by the object perimeter is used to achieve invariance to scale by using eq. 6.2 with  $\gamma = p + q + 1$ . By sampling the polygonal approximation we can use the boundary moments as geometric descriptors of the primitives.

#### Geometric Moments for Line Segments

When the contours of the objects are polygonally approximated, the geometric moments can be formulated for line segments as introduced by Lambert and Gao in [LG95, LG96]. Given a polygonally approximated shape composed of  $n$  segments, let us take  $a_i = (y_{i+1} - y_i)/(x_{i+1} - x_i)$  as the slope of the segment  $s_i$ . The line moments are then computed by

$$\begin{aligned} \mu_{pq} &= \sum_{i=1}^n D_i, \\ D_i &= \sqrt{1 + (a_i)^2} \cdot \sum_{k=0}^q \left\{ \binom{q}{k} a_i^k (y_i - a_i x_i)^{q-k} \cdot \frac{x_{i+1}^{p+k+1} - x_i^{p+k+1}}{p+k+1} \right\} \end{aligned} \quad (6.3)$$

And if the segment  $s_i$  is vertical, we use

$$D_i = x_i^p \cdot \frac{y_{i+1}^{q+1} - y_i^{q+1}}{q+1} \quad (6.4)$$

### Invariant Moments

To obtain invariance under translation, the centroid is used as in eq. 6.1. The normalization by the polyline length is used to obtain scaling invariance. Finally, invariance to rotation is achieved by using the set of seven functions proposed in [Hu62] involving moments up to third order.

$$\begin{aligned} \phi_1 &= \eta_{20} + \eta_{02} \\ \phi_2 &= (\eta_{20} - \eta_{02})^2 + (2\eta_{11})^2 \\ \phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\ \phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\ \phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\ &\quad (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ \phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\ \phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - \\ &\quad (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned} \quad (6.5)$$

Moment invariants can be normalized with to get the different invariants into similar numerical ranges. Usually we can use the logarithm as a coarse normalization:

$$\psi_1 = \log|\phi_i|, i \in [0...7] \quad (6.6)$$

Hupkens and de Clippeleir proposed in [HdC95] the following normalization of invariants to achieve a better robustness to noise.

$$\begin{aligned} \phi'_1 &= \phi_1 = \eta_{20} + \eta_{02} \\ \phi'_2 &= \phi_2 / \phi_1^2 \\ \phi'_3 &= \phi_3 / \phi_1^3 \\ \phi'_4 &= \phi_4 / \phi_1^3 \\ \phi'_5 &= \phi_5 / \phi_1^6 \\ \phi'_6 &= \phi_6 / \phi_1^4 \\ \phi'_7 &= \phi_7 / \phi_1^6 \end{aligned} \quad (6.7)$$

### 6.3.2 Simple Shape Description Ratios

The eccentricity, aspect-ratio or Feret's ratio of a given shape is the ratio of the length of the longest chord of the shape to the longest chord perpendicular to it. It can be computed by using the moments described in eq. 6.3 as

$$ecc = \frac{\mu_{20} + \mu_{02} + \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}}{\mu_{20} + \mu_{02} - \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}} \quad (6.8)$$

The circularity, or area-perimeter ratio of a shape is defined as how closely-packed the shape is. For a circle it is equal to 1, all other shapes have a circularity lesser than 1. It is computed as

$$circ = \frac{4\pi A}{P^2} \quad (6.9)$$

Obviously, there are many other shape ratios describing certain geometrical properties. The interested reader is referred to [Rus], [SS94]. In our case, we only use these two ratios as the feature vector describing a shape.

### 6.3.3 Fourier Descriptors

Given a polyline  $p_O$  which is the polygonal approximation of the boundary of a shape  $O$ , we use as a vectorial shape signature the central distance function computed as

$$r_i = \sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2} \quad \text{for } (x_i, y_i) \in p_O. \quad (6.10)$$

Zahn obtained in [ZR72] a Fourier descriptor of a shape, applying the Fourier transform on the signature representing the shape boundary. Sampling  $r_i$  to  $N = 2^n$  samples so the use of the FFT is possible, the feature vector of the Fourier descriptor is given by

$$f(O) = \left[ \frac{|F_1|}{|F_0|} \dots \frac{|F_{N/2}|}{|F_0|} \right] \quad (6.11)$$

where  $F_i$  corresponds to the  $i$ th component of the Fourier spectrum. Other shape signatures as curvature or complex coordinates can be used to compute the Fourier descriptor. The interested reader is referred to [KSP95].

In the case of graphical symbols, the shape descriptors above presented can be applied to each of the primitives of the symbol extracted as mentioned in section ???. Formally speaking, given a symbol  $S = \{p_1 \dots p_p\}$ , applying one of the presented descriptors will return a set of feature vectors  $f(p_i)$  for all  $i \in [1, p]$ . Let us study in the next section how to adapt classical indexing structures used in the databases field to index graphical symbols in a document database.

## 6.4 Multidimensional Hashing to Index Primitives

The previously presented methods for spotting symbols from a document database, present an important constraint. As the number of considered shape models is increased, the computational cost of the matching step can be unaffordable. As pointed in [CM94], in order to avoid a brute-force matching step, the use of indexing paradigms becomes necessary.

Among the wide taxonomy of indexing structures (cf. [GG98]), the *point access methods* are the ones which are more suitable for our purposes. Tree-based structures are frequently used in indexing mechanisms. Nevertheless, they suffer from several drawbacks. The querying process can be computationally expensive since the tree have to be traversed and in addition, tree balancing algorithms are needed to maintain an effective search performance. As in our case we want to foster the querying speed and we want a system where the data could be easily added at any moment, a multidimensional hashing technique has been selected instead of a tree-based one. In particular, we use a grid file structure, described in [NHS84], in order to index the vectorial primitives. Let us overview with more detail how multidimensional hashing methods work.

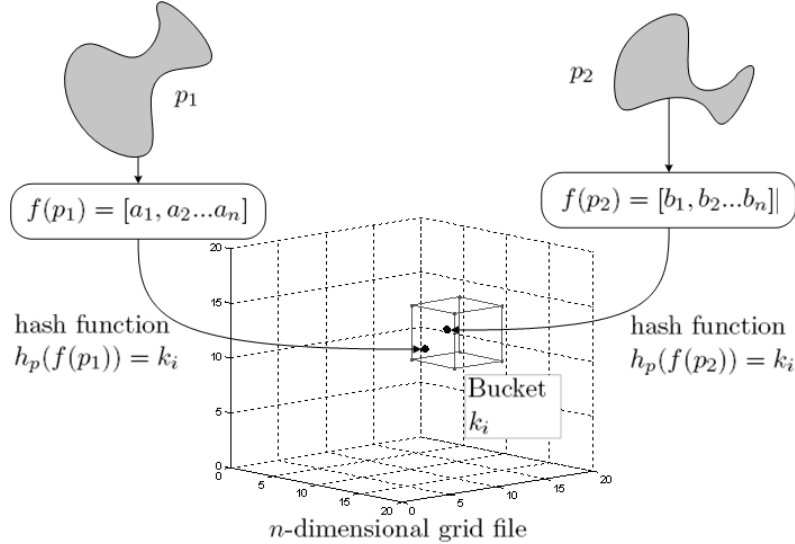
Multidimensional hashing methods partition the space into hypercubes of known size and group all the records contained in the same hypercube into a bucket. The buckets are uniquely identified by a *key-index* which aims a fast retrieval of all the data contained in the bucket. A hash function performing one-dimensional partitions, automatically computes the key-index of a given query to identify the bucket which it belongs to.

In our case, given a polyline, a feature vector is computed using one of the presented descriptors and then a hash function obtains the key-index. This hash function establish a quantization criterion to apply to each dimension of the feature vector to limit the key-index parameters to a finite number of discrete values. To avoid boundary effects, each primitive is stored into the two closest buckets in each dimension.

Usually, the main drawback of hashing techniques is the collisions. Given two different items to store in the database, we have to guarantee that the hash function used to index such items do not assign the same key-index to them. To overcome this problem expensive re-hashing algorithms are applied once a collision is detected. In our case, collisions are not a problem but the basis of our indexing strategy. Given two similar (but not equal) primitives, they are represented by a compact feature vector. Hopefully, if the two primitives have a similar shape, the two feature vectors will be two nearby points in the description  $n$ -dimensional space. The partition of this space by the grid file has to guarantee that both points fall into the same bucket (or at least to neighboring buckets) to have stored in a single entry all the similar primitives. This technique allows to have an efficient retrieval by similarity.

In Fig. 6.2 we can appreciate an overview of how the indexing mechanism works. Formally speaking, a symbol  $S = \{p_1 \dots p_m\}$  is described by a set of feature vectors  $f(p_i)$  for all  $i \in [1, m]$  arising from one of the descriptors presented above in section 6.3. A hash function  $h_p(f(p_i)) = k_i$  returns a key-index identifying a certain bucket in





**Figure 6.2:** The use of a grid file to index vectorial primitives. The hash function projects the feature vectors into key-indices. Two similar primitives are stored into the same bucket.

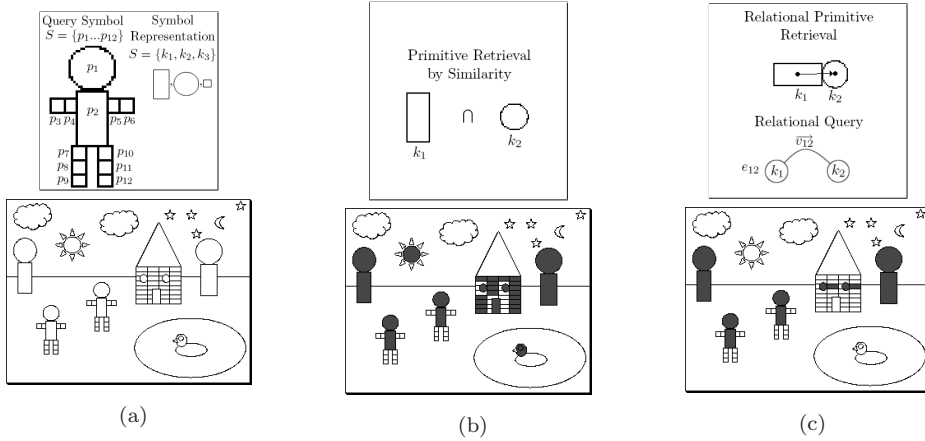
the  $n$ -dimensional indexing space. As the shape descriptors are invariant to similarity transformations and robust to noise, even if the input primitives are not completely equal, the whole procedure leads to the same bucket. The symbol  $S$  is then represented by the set of key-indices  $\{k_1 \dots k_k\}$  with  $k \leq m$  since all the similar primitives are represented by the same key-index.

In each bucket the information of the position in a three-dimensional space (i.e.  $(x, y)$  coordinates of the primitive gravity center appearing in a certain document  $d$  of the collection) of all the primitives in the document database having key-index  $k_i$  is stored. Summarizing this section, the proposed indexing methodology allows retrieve all the spatial locations where similar primitives than the queried one are likely to be found.

## 6.5 Relational Indexing and Hypotheses Validation

Since graphical symbols are composed of several primitives, indexing a symbol consists in separately indexing each of its primitives. This approach has a big drawback since the spatial coherence of the retrieved primitives is not taken into account. We present in this section a relational indexing algorithm to furnish the indexation methodology with spatial information. A voting scheme aiming to validate the spotted locations is also presented.

### 6.5.1 Relational Indexing



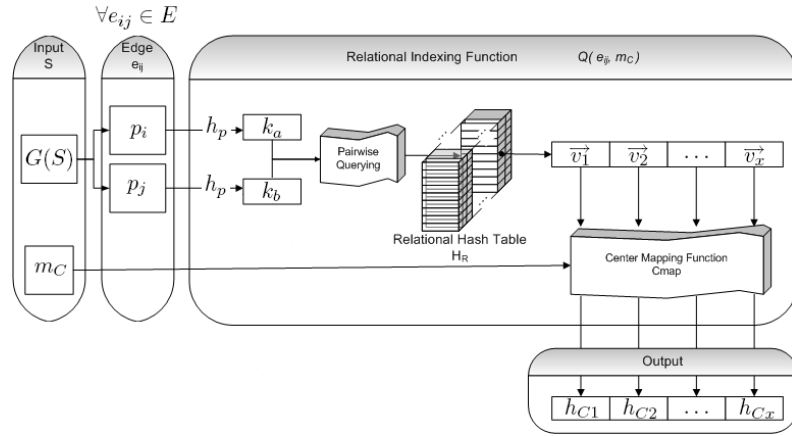
**Figure 6.3:** Relational indexing. For the sake of visibility, only two primitives  $p_1$  and  $p_2$  are queried. (a) Sample line-drawing and the query symbol; (b) results of retrieving a couple of primitives by similarity without taking into account the spatial information, the resulting primitives are highlighted in gray; (c) retrieving the same two primitives by using the relational indexing mechanism.

When considering large databases, many symbols may share a substantial part of primitives with many other. Bag-of-words models describe objects in terms of the presence of the primitives which compounds them, ignoring their spatial structure. Recently, a method to locate objects in images using a bag-of-words model has been proposed in [SRE05]. The large amount of features taken from interest points aim to discard spatial information. However, in our case, the presence in a given location of a set of primitives do not guarantee the presence of the searched symbol, since symbols are not usually composed by too many primitives. The geometrical configuration of these primitives is a crucial information to refine the zones of interest. Inspired by the work presented in [CS00], spatial relationships among primitives are also considered when indexing in order to obtain much more valid hypotheses.

Given a symbol represented by a set of primitives  $S = \{p_1 \dots p_m\}$ , the similar primitives appearing in a document can be retrieved by using the set of key-indices  $\{k_1 \dots k_k\}$ . To take into account the spatial configuration of those primitives, the proximity graph  $G(S)$  has to be used. The edges  $e_{ij} \in E$  represent the relationship between two primitives stored in the nodes  $n_i$  and  $n_j$ . These edges can be used to retrieve by similarity pairs of primitives agreeing with a certain spatial distribution. We can appreciate in Fig. 6.3 an example on the use of relational indexing.

To efficiently retrieve all the edges of a query symbol, a hash table  $H_R$  is used to store in memory the adjacency matrix of the proximity graphs. This hash table is indexed by pairs of primitives. The use of hash tables with multiple indices has been used over the years to store and guarantee an efficient access to sparse matrices, like

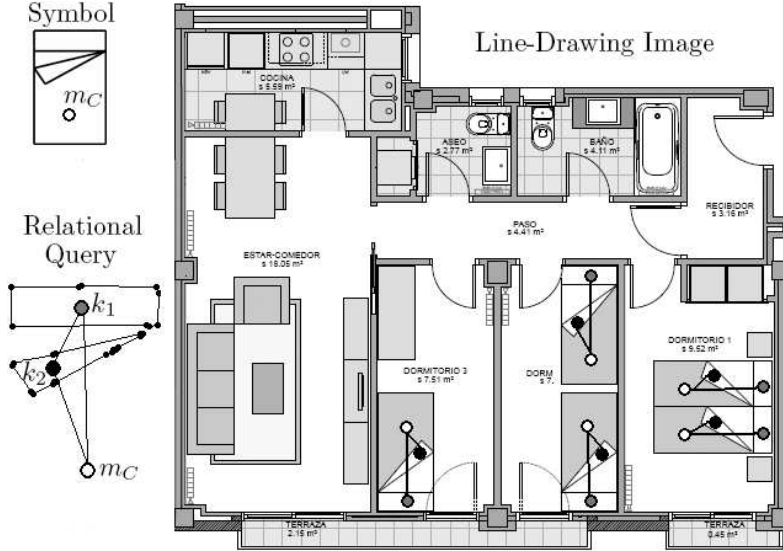
presented in [SMK72]. The entry of the table  $H_R[k_a, k_b]$  stores all the possible edges  $e_{ij}$  where the primitive stored in the node  $n_i$  is indexed by  $k_a$  and the primitive of the node  $n_j$  is indexed by  $k_b$ . In the acquisition step, for all the documents  $D$  in the collection, each graph  $G(D)$  is added to the table  $H_R$  so a spatial relationship between two given primitives can be efficiently retrieved from all the document collection.



**Figure 6.4:** Relational indexing architecture. Starting from the proximity graph, each edge performs a relational query based on the indices representing the primitives stored in the nodes. A list of vectors is retrieved corresponding to spatial relationships between primitives in target documents. A center mapping function transform these vectors into hypothetical centers where the symbol should be found.

When querying a given symbol, each edge of the graph is considered. A querying function  $Q(e_{ij}, m_C)$ , taking an edge and the center of the query symbol  $m_C$ , results in a list of hypothetical centers  $Lh_C = [h_{C1} \dots h_{Cx}]$  where to find the two primitives with a given pose. We can see in Fig. 6.4 how this function proceeds. The key-indices representing the primitives stored in the nodes are computed by using the hash function  $h_p$ . Both indices identify an entry of the hash table  $H_R$  storing a list of edges, and most importantly the corresponding vectors  $\vec{v}_{ij}^r$ . These vectors are the spatial distributions of the primitives appearing in the document database. A center mapping function  $Cmap(\vec{v}_i^r, m_C) = h_{C_i}$  applies a scale and rotation transform to the center  $m_C$  in order to find the pose of the hypothetical center  $h_{C_i}$  depending on the vector  $\vec{v}_i^r$ . We can see an example of the hypothetical center location in Fig. 6.5. Note that the center mapping process align the query edge to the retrieved edges in the line-drawing database, thus being invariant to scale and rotation transforms.

By applying the relational indexing function to each edge of the proximity graph of the query, the locations in the documents where we can really find the queried symbol, should appear several times in the hypothetical centers list. The use of a voting scheme reinforces these hypotheses and validates the possible locations.



**Figure 6.5:** Center mapping function to find the pose of the hypothetic centers given an edge of the relational query and the gravity center of the query symbol.

### 6.5.2 Voting Scheme

Following the idea of the Generalized Hough Transform (GHT) [Bal81], each of these centers accumulate votes. Applying the querying function to each edge of the graph from the query symbol, we accumulate evidences in the hypothetic centers in the stored documents where it is probable to find similar primitives with the same spatial organization than the query. In the voting space, the coherent votes tend to form salient peaks, the rest of votes will be scattered in different locations but not forming clusters. A simple ranking of these clusters result in the positions of the documents where it is more feasible to find the queried symbol.

The querying process leads to consider each pair of primitives of the queried symbol  $S = \{p_1 \dots p_m\}$ , implying  $C_2^m$  accesses to the hash table  $H_R$ . The number  $x$  of hypothetic centers where to cast votes is the same as how many position vectors are stored at each table entry. Obviously, the  $x$  value is directly related to the number of documents stored in the library. That results that for each query symbol we have

$$x \cdot C_2^m = x \cdot \binom{m}{2} = x \cdot \frac{m!}{2(m-2)!} \quad (6.12)$$

centers where to accumulate votes. The locations where the votes are casted are sorted and returned as the retrieved regions of interest. Note that no threshold is used to decide whether a symbol is present or not. Let us present in the next section some qualitative results of applying the presented relational indexing method.

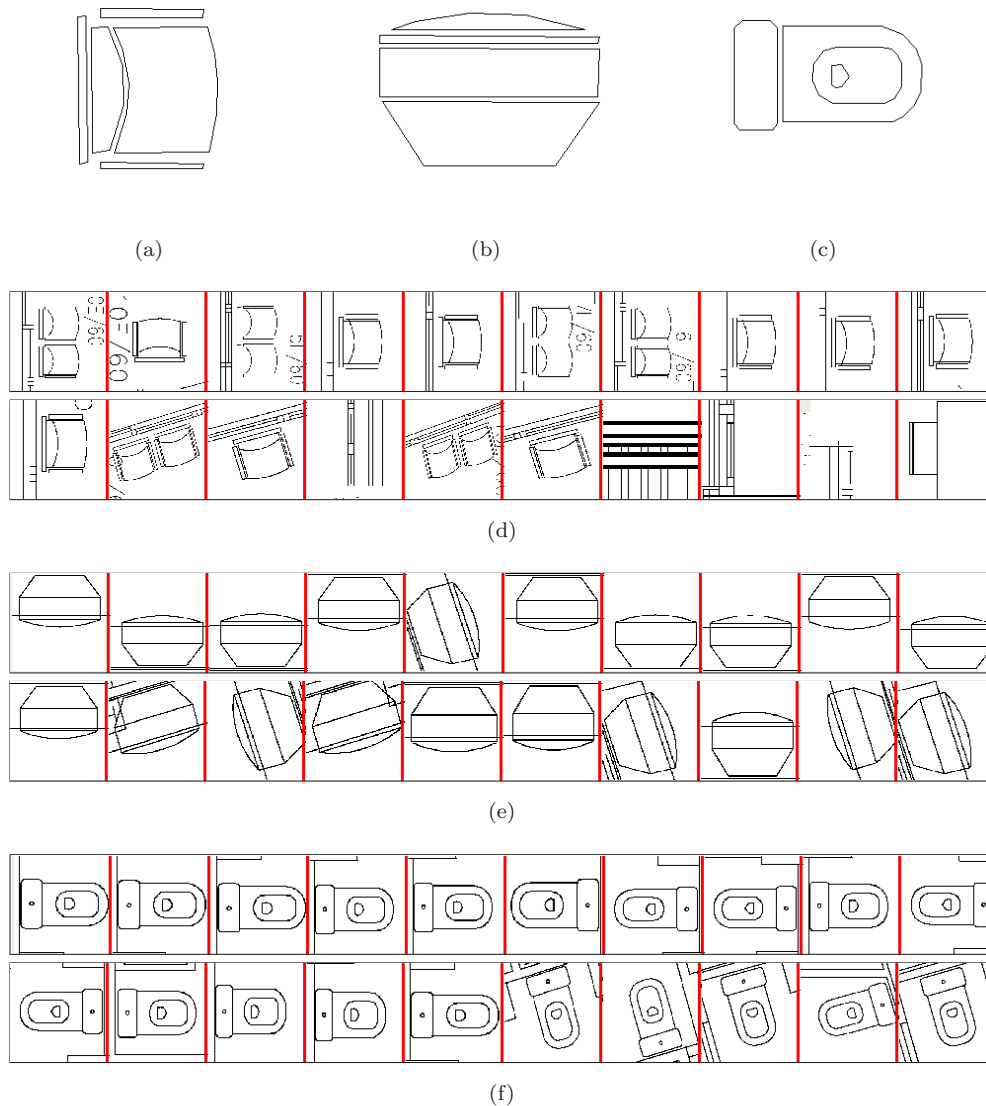
## 6.6 Experimental Results

To perform the experimental results we worked with a collection of architectural floor-plans consisting of 42 images (of  $3215 \times 2064$  pixels in average) arising from four different projects. This dataset is the FPLAN-POLY database, detailed in appendix A. These images are polygonally approximated resulting in a collection of vectorial documents. The symbols taken into account for these experiments are divided in 38 classes and we have in total 344 instances in the document images. In a single document image the average number of symbols is around 8 and it goes from 0 to 28 symbols. The models to query the document database are cropped from the document images, so they also contain vectorial distortions.

When querying a model symbol against the database, the convex hull of the activated polylines in the documents conform a set of regions of interest which are sorted by confidence value depending on the number of received votes. We can see in Figs. 6.6 and 6.7 the first twenty results of querying several symbols in the whole document collection when using the Fourier shape descriptor. As we can appreciate, most of the results correspond to the correct queried symbol, but obviously some areas of false positives appear. We observe two interesting phenomena, usually, two close symbols (i.e. burners in Fig.6.7f or chairs in Fig. 6.6d) are grouped in a single region of interest, on the other hand it is common to find that a symbol is well spotted but the returned region of interest is bigger than expected (i.e. the burners in Fig. 6.7f).

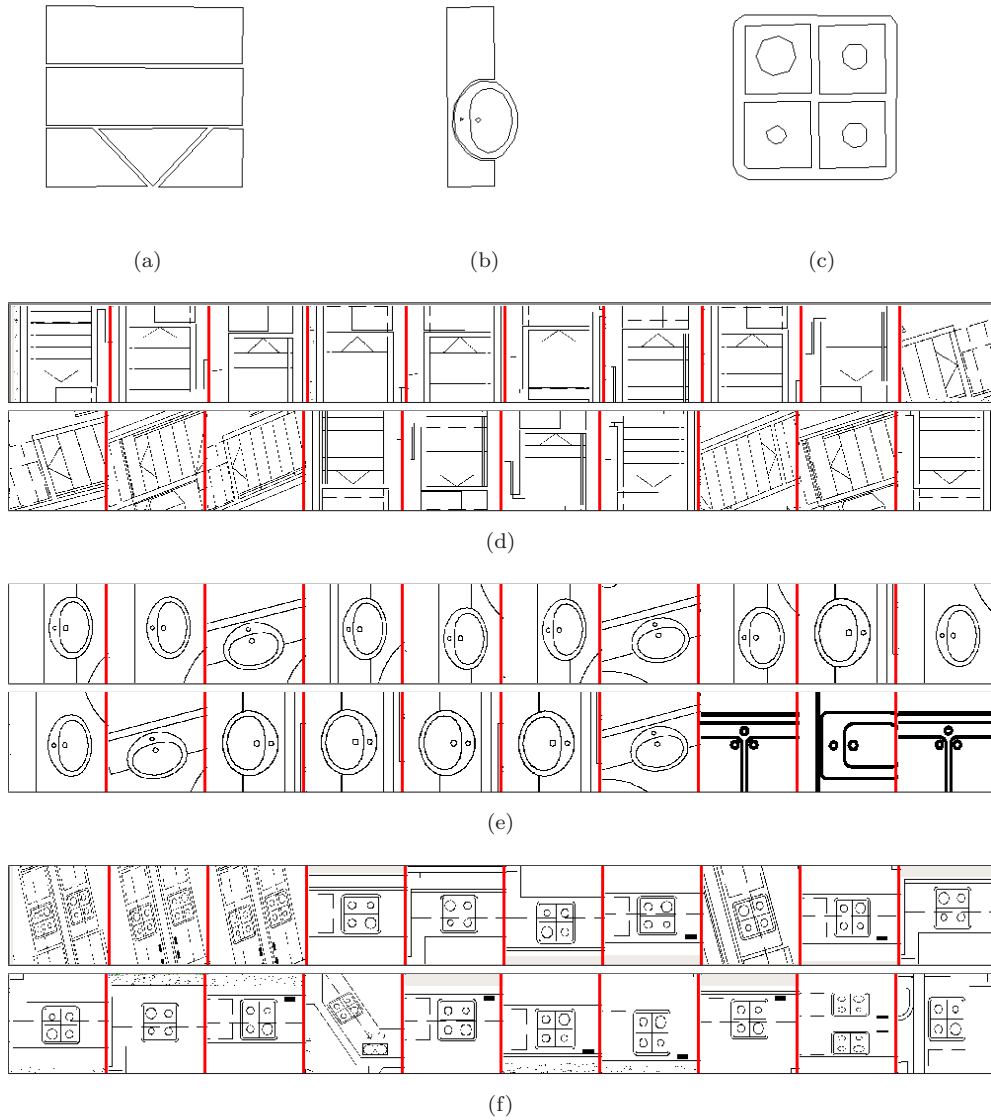
We consider that if the resulting polygons are able to overlap at least a certain percentage of the ground-truthed representation of a symbol, they can be considered as recognized. On the other hand, if the resulting polygons do not cover the ground-truth, the symbol should be considered as missed. Of course, as with all decisions implying a certain threshold, its value can be critical, and the system's evaluation can depend on it. The definition of this threshold is completely subjective as it depends on what the user considers a symbol as being detected or not. In our case, we consider a symbol as detected if it overlaps at least a 75% with the ground-truth area. We can see in Table 6.1 the total True Positive Rate (*TPR*) when applying the different shape descriptors and the average of False Positives (*FP*) regions obtained by all these methods. Notice that the time to retrieve a symbol from a document is highly related to the accuracy of the selected method. Methods having higher recognition rates expend more time in retrieving zones of interest since the table entries are more populated and the amount of false positives is also increased. On the other hand, the methods which have less recognition rate but also less false positives, are usually less computationally expensive.

However, in focused retrieval applications, there are some cases that the performance evaluation is not straightforward. Let us consider the example shown in Fig. 6.8. Given a document in the collection, we query one symbol which can be found twice in within the document. Instead of obtaining two different regions of interest framing the occurrences of this symbol, the system results in a single region framing both instances of the symbol. The two symbols were relatively close in space in the document, so it is understandable that the system just retrieves one big region of interest where the probability to find the query object is high enough. However,



**Figure 6.6:** Qualitative results of the relational indexing method (1). (a) Query symbol *chair*; (b) query symbol *TV set*; (c) query symbol *toilet*; (d),(e) and (f) first 20 retrieved regions when querying the symbols (a), (b) and (c) respectively.

the question on how to evaluate this result, is not easy to answer. Both symbols were retrieved, but the system fails to identify that there are two different instances. By returning just one region, its area is big enough to contain other graphic objects which are not the symbol, but it is hard to consider this result as a false alarm. We



**Figure 6.7:** Qualitative results of the relational indexing method (2). (a) Query symbol *stairs*; (b) query symbol *sink*; (c) query symbol *burners*; (d),(e) and (f) first 20 retrieved regions when querying the symbols (a), (b) and (c) respectively.

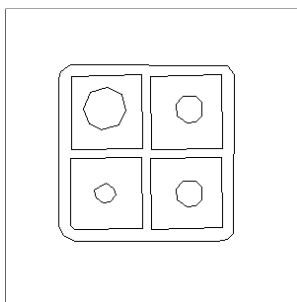
propose in the last part of this thesis a protocol for performance evaluation for symbol spotting and focused retrieval systems. In this part we will present the quantitative evaluation of the relational indexing method presented in this chapter.

**Table 6.1:** Recognition results of the relational indexing method.

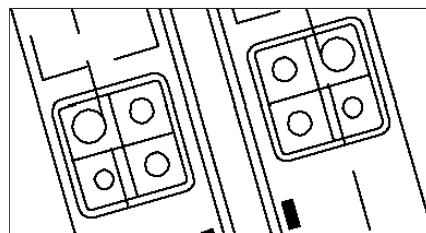
Description	<i>TPR</i> (%)	<i>FP</i>	Time (secs./plan)
Simple ratios	<b>93.62</b>	153.42	3.44
Hu's boundary moments	91.3	76.76	0.71
Line segment moments	55.62	63.89	<b>0.55</b>
Fourier descriptor	73.33	<b>58.76</b>	0.78



(a)



(b)



(c)

**Figure 6.8:** Illustration of a result which is difficult to evaluate. (a) Floorplan image in the collection; (b) queried symbol; (c) retrieved region.



## 6.7 Conclusions and Discussion

A relational indexing mechanism to spot symbols in a collection of line-drawing images in vectorial format has been presented. A first step of primitive extraction and description has been introduced in order to have a compact representation of the graphical symbols. These primitives are organized in an indexing structure aiming to retrieve by similarity all the primitives in the collection. A relational indexing mechanism has been presented in order to take into account not only the similarity of the primitives which compounds a symbol but also the spatial relationship among them. Finally a Hough-like voting scheme aims to validate the hypotheses where a symbol is likely to be found.

The qualitative results show good performance results. Most of the approaches in the literature always make a choice on using only structural information about the symbols or just numerical descriptions of a symbol. The presented approach use both structural and numerical information. The use of both information sources increases the robustness of the method. It also aims to use very simple descriptors with good results according to the user needs.

There is obviously still some room for improvements. By describing symbols by closed regions, we make the assumption that the symbols are composed by several loops. This may not be the case in certain graphic-rich documents. In such cases, another primitive extraction process should be considered.

In some application domains, as for instance in the case of complex electronic diagrams, some symbols share a substantial part of their design and only differ by slight details. Symbols may also be composed of other known and significant symbols. In this context, the proposed focused retrieval methodology might result in an important number of false alarms.



## Part III

# A Performance Evaluation Protocol for Symbol Spotting Systems



# Chapter 7

## Performance Evaluation of Symbol Spotting Systems

---

Symbol spotting systems are intended to retrieve regions of interest from a document image database where the queried symbol is likely to be found. They shall have the ability to recognize and locate graphical symbols in a single step. In this chapter we present a set of measures to evaluate the performance of a symbol spotting system in terms of recognition abilities, location accuracy and scalability. We show that the proposed measures allow to determine the weaknesses and strengths of different methods. In particular, we have evaluated in detail the spotting method presented in chapter 6.

---

### 7.1 Introduction

Performance evaluation methods are essential tools to understand and compare the behavior of algorithms and systems. A performance evaluation protocol should identify the strengths and weaknesses of the methods under test. The analysis of these strong points and drawbacks should determine which method is the most suitable for a certain use case and predict its behavior when using it in real applications with real data.

In the last years, performance evaluation has been a quite prolific research topic in the Document Image Analysis and Recognition field and in particular among the Graphics Recognition community. Several competitions focused on particular topics, namely, symbol recognition, layout analysis, text detection among others, have been organized in the major conferences and workshops of this field. We can also find a lot of contributions in the recent literature proposing evaluation techniques for different document image analysis applications. Performance evaluation frameworks have been proposed for evaluating low-level applications such as line and arc detection algorithms [WD97, WD98] or raster-to-vector systems [SST02]. However, in the

last years, frameworks aiming to evaluate higher level applications such as symbol recognition [VDW07] or layout analysis [AB07] have been proposed.

In this chapter, we propose a set of measures and methodologies to evaluate the performance of spotting systems. Although we mainly focus on the specific case of symbol spotting, these measures are also applicable to performance evaluation of other focused retrieval applications such as word spotting, or even object recognition in Computer Vision applications. Symbol spotting techniques should efficiently locate graphical symbols in document images without using full recognition methods. Such systems are intended to index large collections of document images in terms of the graphical symbols which appear in them. Given a graphical symbol as query, the system has to retrieve a ranked list of locations where the query symbol is likely to be found. Since spotting systems deal with recognition and segmentation at the same time, such abilities must be taken into account by the evaluation process. Segmentation errors must be punished as well as recognition mistakes.

As we illustrate in the review provided in section 7.2, there exist many approaches to measure the performance of different Graphics Recognition algorithms. However, in the particular case of symbol spotting, existing methods in the literature just provide measures based on binary decisions of *found / not found*. Along this thesis we have evaluated the proposed methods on these binary decisions. Based on a decision on whether to consider a symbol as being correctly located, we have presented all the results by giving information about the true positive rate (TPR) and the false positive rate (FPR) for the recognition and classification applications (chapters 3, 4, 5 and 6). We gave the results for the focused retrieval application in terms of the precision and recall of the binary decisions (chapter 5).

We develop the theory in this chapter that the performance of a symbol spotting system should be defined in terms of two components: the recognition and the location goodness. Starting from this hypotheses, the main contribution of this chapter is to propose a set of performance evaluation measures, based on the precision and recall concepts, to evaluate the performance of symbol spotting systems in terms of two criteria, namely recognition and location. In addition, a second contribution is to use the same formalism to evaluate a third quality criterion, the scalability under an increasing number of symbol prototypes. Most of the work found in the literature dealing with performance evaluation of Graphics Recognition systems is mainly focused on the computation of a score to allow an easy way to rank different methods. We strongly believe that the proposed measures can give a more accurate idea of the real behavior of the system under study than typical recognition rates.

The remainder of this chapter is organized as follows: We briefly overview in section 7.2 the work on performance evaluation for related areas such as retrieval systems, Graphics Recognition and Document Image Analysis applications. In section 7.3, we basically review the well-known measures of precision and recall typically used in retrieval evaluation and the measures we can derive from precision and recall. Section 7.4 outlines how these measures can be reformulated and applied to evaluate a spotting system in terms of retrieving regions of interest from a document image database. Section 7.5 shows a use case of such measures, evaluating the performance

of the symbol spotting method presented in chapter 6, which is based on a set of four different off-the-shelf shape descriptors. Finally, the conclusions and a short discussion can be found in section 7.6.

## 7.2 Related Work

Symbol spotting systems are intended to produce a ranked list of regions of interest cropped from the document images stored in the database where the queried symbol is likely to be found. Symbol spotting can thus be seen as a particular application within the Information Retrieval (*IR*) domain. Usually, retrieval systems are evaluated by *precision* and *recall* ratios which give an idea about the relevance and the completeness of the results (we will briefly review these measures in section 7.3). These basic measures can be enhanced considering many other indicators depending on the application. For instance, Lu et al. evaluate in [LSS07] a set of desktop search engines by deriving a set of ratios from precision and recall to indicate the abilities of the systems when incrementally retrieving documents. Müller et al. evaluate in [MMS01] content-based image retrieval systems, proposing some strategies to take into account the way the number of items stored in the collection affects the results and how user feedback can improve the response of such systems. Kang et al. evaluate in [KKL04] a text retrieval system which uses semantic indexing, focusing on the distribution and amount of key-indices used to index the database. Finally, we can find in [HWH07, NBM06] the performance analysis of some information retrieval systems having the information distributed in a peer-to-peer network (*P2PIR*), which takes into account the query response time, the network resources requirements and the tradeoff between distributed and centralized systems. As we can see, the coverage of information retrieval topic is so wide that even if researchers use similar indicators to evaluate the performance of their methods, no general evaluation framework can be defined. In our case we will also base our measures on the notions of precision and recall by adapting them to the recognition and location abilities that the spotting systems should present.

In the Document Image Analysis and more particularly the Graphics Recognition field, some work focused on spotting can be found. However all this work is evaluated by ad-hoc measures. For instance, Rath and Manmatha presented in [RM03] a system able to spot handwritten words in ancient documents. They evaluate their system with a score based only on the precision value. Marcus presented in [Mar92] an algorithm to spot spoken words in an audio signal. The evaluation is based on Receiver Operating Characteristics (*ROC*) graphs [Faw06] which are related to precision and recall measures. Tabbone and Zuwala present in [TZ07] a method to spot graphical symbols in a collection of electronic drawings. They base the evaluation of their method in precision and recall graphs. Finally, Valveny et al. present in [VDW07] a framework to evaluate symbol recognition methods envisaging a way to evaluate location and recognition of symbols by also using precision and recall measures. However, all these methods are computed on a binary retrieval notion: whether an item is considered retrieved or not. By these measures one can see the ability of the

system in retrieving relevant items and discarding negative ones, but these measures do not evaluate how well the system located the queried objects.

To avoid binary relevance labelling, our measurements are inspired in the techniques used to evaluate layout analysis systems. In fact, layout analysis shares some similarities with spotting in the sense that sub-regions from documents have to be labelled according to their content. Layout analysis competitions [AGB05, AGB07, AGK03] were held in last editions of the *ICDAR* conference. In these contests, the evaluation of the participants' methods was done according to the overlapping between regions of the results and the ground-truth. Two indicators introduced in [PC99] are used to formulate an entity detection measure from which an averaged segmentation measure is deducted to score the systems. Following the same idea, in the text detection competitions [Luc05, LPS05] held in last editions of *ICDAR*, precision and recall measures were computed in terms of overlapping between bounding-boxes of the ground-truth and the results. From the precision and recall numbers, a score was computed to rank the algorithm performance. However we believe that the use of a single evaluation score allow an easy ranking of the different systems, but hinders the understandability of their behavior and the performance prediction when using other type of datasets.

Finally, in the last symbol recognition competitions [AYS00, VD04, VD06] held in the *GREC* workshop editions, several symbol descriptors where evaluated. In that case, the performance is evaluated by the recognition rates the systems yield. In the last edition, other measures such as the homogeneity and the separability of the symbol classes in the description space have been introduced. We find very interesting the fact that the scalability of the systems is also tested. This test is performed looking how the performance of the systems evolve as the number of symbol classes to consider increases.

The measures we propose in this chapter are based on precision and recall, since it has been demonstrated to be a good way to evaluate recognition (or at least classification) and location at the same time. We formulate the precision and recall notions in terms of overlapping between retrieved areas and ground-truth. The presented measures and plots allow to assess the weaknesses and strengths of the methods in terms of recognition abilities and location accuracy. In addition we also present a methodology to extract a scalability measure from precision and recall to test if the methods can be used with a larger amount of classes. Let us first review the basic measures used to evaluate retrieval effectiveness.

### 7.3 An Overview on Measures to Evaluate Retrieval Effectiveness

In this section we review the basic measures provided in the literature used to evaluate the retrieval effectiveness. The measures outlined in this section will be reformulated in section 7.4 for the framework described in this work.



### 7.3.1 Precision and Recall

In the information retrieval field, most measures to evaluate effectiveness are based on a binary labelling of relevance of the items, namely whether each item is considered as relevant or non-relevant. In addition, these measures are also based on a binary retrieval notion, i.e. whether an item is retrieved or not.

Given a database consisting of a set of elements  $tot$ , and a query item  $i$  to retrieve from it, let us label as  $rel$  the set of relevant objects in the set and  $\overline{rel}$  the set of non-relevant items with regard to the query  $i$ . When querying this item to the database, we label as  $ret$  the set of retrieved elements and as  $\overline{ret}$  the set of elements from the database which were not retrieved. The retrieval matrix of Table 7.1 shows all the possibilities in terms of intersections between these sets.

**Table 7.1:** Retrieval Matrix.

	Relevant	Non-Relevant	TOTAL
Retrieved	$ ret \cap rel $	$ ret \cap \overline{rel} $	$ ret $
Not Retrieved	$ \overline{ret} \cap rel $	$ \overline{ret} \cap \overline{rel} $	$ \overline{ret} $
TOTAL	$ rel $	$ \overline{rel} $	$ tot $

The analysis of this table allows to define the well-known ratios of precision and recall (see van Rijsbergen's [vR79] book on Information Retrieval for more details) to evaluate the behavior of the information retrieval system which are computed as follows:

$$P = \frac{|ret \cap rel|}{|ret|}, \quad R = \frac{|ret \cap rel|}{|rel|} \quad (7.1)$$

For a given retrieval result, the *precision* measure  $P$  is defined as the ratio between the number of relevant retrieved items and the number of retrieved items. The precision measure measures the quality of the retrieval system in terms of the ability of the system to only include relevant items in the result. A hundred percent precision means that no false positive has been included in the system response. As the precision value decreases, the more non-relevant items are included in the results.

The *recall* ratio  $R$  is defined as the number of relevant retrieved items as a ratio to the total number of relevant items in the collection. It measures the effectiveness of the system in retrieving the relevant items. A hundred percent recall means that all the items labelled as relevant are retrieved and no one has been missed. As the recall value decreases, the more relevant items are missed by the system which wrongly considers them as non-relevant.

### 7.3.2 $P@n$ and $P(r)$

The precision and recall measures are computed on the whole set of items returned by the system. That is, they give information about the final performance of the system after processing a query and do not take into account the quality of ranking in the resulting list. Information retrieval systems return results ranked by a confidence value. The first retrieved items are the ones the system believes that are more likely to match the query. As the system provides more and more results, the probability to find non-relevant items increases.

Relevance ranking can be evaluated computing the precision at a given cut-off rank, considering only the  $n$  topmost results returned by the system. This measure is called precision at  $n$  or  $P@n$ . However, this measure presents the drawback that it does not give information about recall.

Let us define  $P(r)$  as the precision at a given recall cut-off, that is the precision at that point where recall has first reached the value  $r$ .

### 7.3.3 Precision and Recall Plots

The usual way to represent the stability of the system as the user requires more and more results is to plot precision and recall against each other. Such plots are computed stepwise retrieving at each step a given item while varying the decision threshold value over the confidence rate, i.e. computing  $P@n$  for the different values of  $n$  and plot this values against its associated recall.

These plots show the tradeoff between precision and recall. Buckland and Gey analyzed in [BG94] the relationship between both ratios concluding that they are inversely related, trying to increase one usually provokes the other to be reduced. Thus, when comparing several methods, the one yielding the higher values for both precision and recall will be the best. However, it is not always easy to assess which precision and recall plot corresponds to a better system.

### 7.3.4 Measures of Quality

Sometimes it is difficult to measure the effectiveness by a measure composed by more than a number. The difficulty in certain cases to assess which method is the best, has led to invest in some composite measures which are able to rank the methods under study according to a combination of precision and recall information. However, as claimed by vanRijsbergen in [vR79], usually these measures are rather ad-hoc and difficult to interpret.

Let us see a couple of composite measures which try to combine both precision and recall information in a single number.

### Average Precision

We can define the average precision *AveP* using each precision value after truncating at each relevant item in the ranked list resulting after a query. Average precision is one of the evaluation measures used by the *TRECVID*<sup>1</sup> community [SOK06].

For a given query, let  $r(n)$  be a binary function on the relevance of the  $n$ th item in the returned ranked list, we define the average precision as follows:

$$AveP = \frac{\sum_{n=1}^{|ret|} (P@n \times r(n))}{|rel|} \quad (7.2)$$

The average precision is a measure of quality which rewards the earliest return of relevant items. Retrieving all relevant items in the collection and ranking them perfectly will lead to an average precision of 1. The average precision can also be seen as the area under the precision and recall plot. However, average precision does not take into account the fact that a system returns non-relevant items after having reached a hundred percent recall (i.e. having returned all relevant items).

### F-score

Another classical composite measure is the *F*-score (see [HR05] for more details) which is the weighted harmonic mean of precision and recall, computed as follows:

$$F^\beta = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R} \quad (7.3)$$

Which for a value of  $\beta = 1$  is equivalent to Dice's coefficient (a well-known similarity measure between two sets  $X$  and  $Y$ ) defined as:

$$s = \frac{2|X \cap Y|}{|X| + |Y|} \quad (7.4)$$

Although there is some work like the one presented by Makhoul et al. in [MKS99] which point out some drawbacks of this measure, the *F*-score is widely used as a measure of merit in the information retrieval literature.

The *F*-score can also be computed at several recall cut-offs to evaluate the stability of a system's response. We re-formulate the *F*-score presented in eq. 7.3 for several recall values as:

$$F^\beta(r) = \frac{(1 + \beta^2) \times P(r) \times r}{(\beta^2 \times P(r)) + r} \text{ with } r \in [0, R] \quad (7.5)$$

We can see some examples on how  $F^1(r)$ -score evolves in Fig. 7.1 for several synthetic precision and recall plots. The better the system responds, the higher its values. As we can appreciate, the *F*-score heavily penalizes low values of precision or recall.

<sup>1</sup>TREC Video Retrieval Evaluation (<http://www-nlpir.nist.gov/projects/trecvid/>)

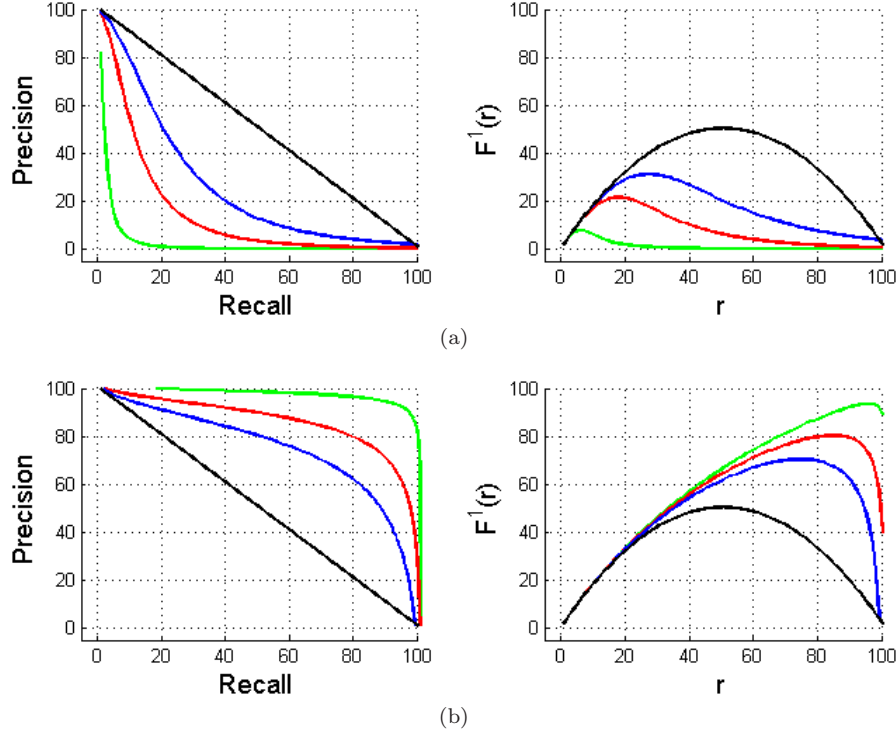


Figure 7.1:  $F^1(r)$ -score plots for different synthetic precision and recall plots.

### 7.3.5 Fall-Out and Generality

Let us finally introduce two more measures, one related to the non-relevant retrieved items and the other related to the dataset, which are computed as follows:

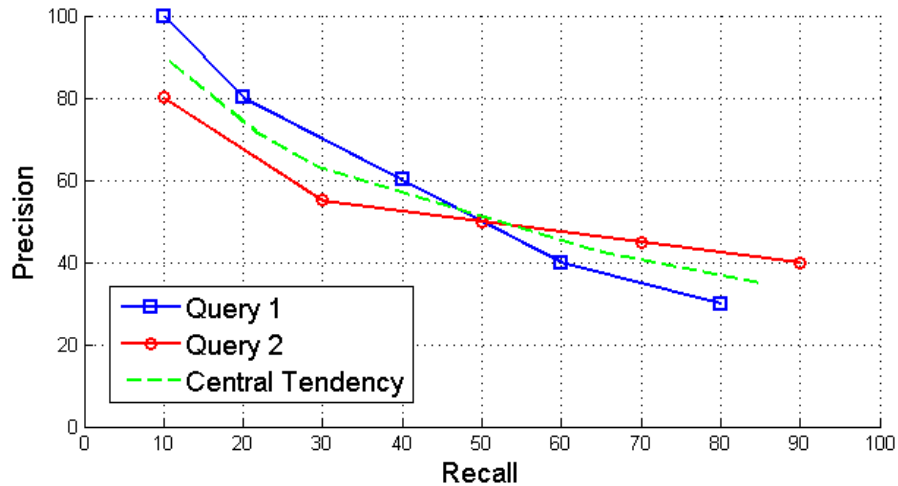
$$Fo = \frac{|ret \cap \overline{rel}|}{|rel|}, \quad G = \frac{|rel|}{|tot|} \quad (7.6)$$

The *fall-out* ratio  $Fo$  gives information about the number of non-relevant retrieved items in respect to the number of non-relevant items present in the collection. This measure is of special interest in unbalanced applications such as symbol spotting, where the amount of elements which are not relevant is much more larger than the relevant elements in the collection. Independent of the precision of a system, this measure should have low values to consider the behavior of the system good. Either because very few non-relevant items have been retrieved or because the number of non-relevant retrieved items is negligible in relation to the number of non-relevant items in the dataset. To evaluate the evolution of the systems response in terms of false positives usually the fall-out is plotted against recall. This plot is equivalent to the typical *ROC* graphs [Faw06], which are commonly used to evaluate the performance of classifiers. We can find in [DG06] a study of the relationship between precision-recall and *ROC* curves.

Finally, the *generality* ratio  $G$ , gives information about the collection dataset. It is computed as the number of relevant items in the entire collection for a certain query. It can be then averaged for all the considered queries in the experimental setup denoted as the *AveG* ratio. This ratio does not give any measure about the effectiveness of the retrieval itself, but complements the previous measures. As claimed by Huijsmans and Sebe in [HS05], when evaluating the performance of a retrieval system, this measure should be given to really understand the meaning of the values of precision, recall and fall-out.

### 7.3.6 Central Tendency of Precision and Recall

To evaluate a retrieval system, obviously many queries have to be performed. Each query under evaluation results in a precision and recall plot. To give an idea on well good the system responds, the retrieval results are averaged over these queries. The central tendency of several precision and recall plots are computed sampling individual curves at different points and averaging the samples. We can see an intuitive example of the central tendency in Fig. 7.2.



**Figure 7.2:** An example of computing the central tendency of precision and recall plots.

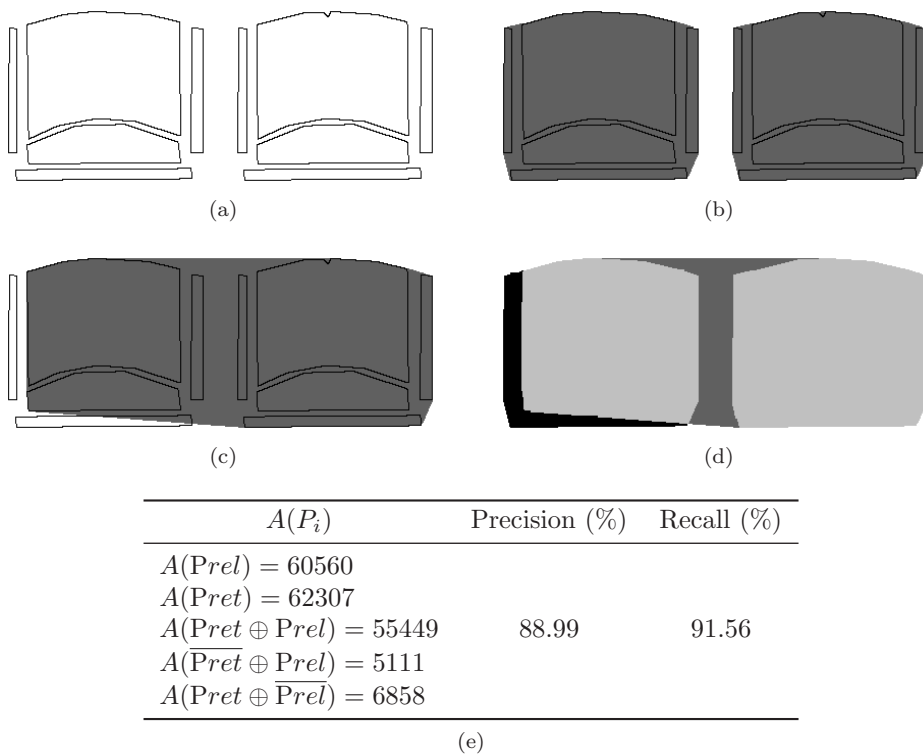
The same averaging technique is applied to fall-out versus recall plots and to  $F^\beta(r)$ -score plots.

## 7.4 Precision and Recall for Spotting Systems

Spotting systems are intended to perform both recognition and location at the same time, and thus, these abilities have to be evaluated together. Let us first propose a

formulation of the precision and recall measures to evaluate both concepts. To help the interpretation of precision and recall plots, we propose to use two more measures focused at symbol level which only consider a binary concept of retrieval. Finally, we propose a scalability test to check the systems ability to achieve similar behavior independent of the number of queried symbols.

#### 7.4.1 Precision and Recall of Regions of Interest



**Figure 7.3:** Overlapping between results and ground-truth. (a) Original image; (b) its ground-truth; (c) the result of a spotting system; (d) overlapping between results and ground-truth labelled according to  $\overline{Pret} \oplus Prel$  (light gray),  $\overline{Pret} \oplus \overline{Prel}$  (dark gray) or  $Pret \oplus \overline{Prel}$  (black); (e) detailed areas and obtained precision and recall.

To evaluate the performance of a spotting system we propose a set of measures inspired by both information retrieval and layout analysis. The idea is to merge both precision and recall measures with area overlapping rates. Precision and recall ratios provide information on the incremental accuracy of the retrieval process in terms of recognized items. On the other hand, the region overlapping between results and ground-truth data is used to evaluate the segmentation accuracy.

To compute the region overlapping between result and ground-truth, we define

for both data polygons representing regions of interest. The more accurate is the definition of the region of interest the more the evaluation is reliable. To define the region of interest where a symbol is located we use the convex-hull algorithm presented in [BDH96] of all the points belonging to the symbol. In the particular setup of chapter 6, the graphical symbols are defined by the contours of the closed regions composing a symbol, so the convex-hull of the contour pixels englobe the whole symbol. Convex-hulls define much more accurately the zones where a symbol is than bounding-boxes or ellipses. This representation can be extended to different formats of the data of the collection (bitmap or vectorial format) and to different symbol representations (internal pixels, skeleton, contours, segments, etc.).

Given a collection of graphical documents, we denote as  $P_{tot}$  the set of polygons representing the whole document image database. For any graphical symbol  $S$  to spot in the collection, we label as  $P_{rel}$  the ground-truth polygon set which is composed by all the polygons framing the locations where we find an instance of the symbol  $S$ . When spotting the symbol  $S$  in the document collection, we denote as  $P_{ret}$  the set of retrieved polygons. To match the results from the system to the ground-truth polygon set, we define the polygon set intersection operation  $P_k = P_i \oplus P_j$ , that given two polygon sets  $P_i$  and  $P_j$ , results in a set of polygons from the spatial overlapping of the polygons belonging to the different sets. To measure the total amount of polygon overlapping, we define the function  $A(P_i)$  as the sum of areas of all the polygons in the set  $P_i$ .

From the above sets and functions, precision and recall ratios of can thus be easily formulated in terms of areas of the overlapping between sets of polygons representing results and ground-truth as follows:

$$P_A = \frac{A(P_{ret} \oplus P_{rel})}{A(P_{ret})}, \quad R_A = \frac{A(P_{ret} \oplus P_{rel})}{A(P_{rel})} \quad (7.7)$$

We can see in Fig. 7.3 an example of ground-truthed symbols and a result from a spotting system. Some background region has been considered as forming part of the symbol. When we compute the overlapping between retrieved regions and relevant ones this false positive region is identified, resulting in a precision decrease. On the other hand some part of the symbol has been missed, this results to the recall value not reaching one hundred percent.

### 7.4.2 Measures of Quality, Fall-out and Generality

Analogously, the measures of quality  $AveP$  and  $F$ -score, and the ratios fall-out and generality can be expressed in terms of the area of the overlapping between polygon sets representing the ground-truth and the results from the spotting system.

We reformulate eq. 7.2 by using the area precision at  $n$  ( $P_A@n$ ). That is computing the area precision value after truncating the result list after each polygon having some overlapping with a polygon in the ground-truth. The average area precision is then computed as:

$$AveP_A = \frac{\sum_{n=1}^{|P_{ret}|} (P_A@n \times r(n))}{|P_{rel}|} \quad (7.8)$$

By using the area precision and area recall, we reformulate the  $F$ -score from eq. 7.3 as:

$$F_A^\beta = \frac{(1 + \beta^2) \times P_A \times R_A}{(\beta^2 \times P_A) + R_A} \quad (7.9)$$

and the use of the area precision at a certain area recall cut-off ( $P_A(r)$ ) aim to reformulate eq. 7.5 as:

$$F_A^\beta(r) = \frac{(1 + \beta^2) \times P_A(r) \times r}{(\beta^2 \times P_A(r)) + r} \text{ with } r \in [0, R_A] \quad (7.10)$$

Finally,  $\overline{Prel}$  being the complementary polygon set for the ground-truth we can reformulate the fall-out and the generality from eq. 7.6 as:

$$F_{oA} = \frac{A(\overline{Pret} \oplus \overline{Prel})}{A(\overline{Prel})}, \quad G_A = \frac{A(\overline{Prel})}{A(\overline{Ptot})} \quad (7.11)$$

### 7.4.3 Measures at Symbol Level

As pointed out by Lucas in [Luc05], sometimes precision and recall based measures are difficult to interpret. A precision of 70% could mean that all symbols were found with an accuracy of 70%, or, on the other hand, that only 70% of the symbols were correctly identified and the other 30% completely missed. A low precision value can be due to a low accuracy in the recognition or to a bad location due to over-segmenting. The recall value can be also affected by missed symbols or by under-segmentation.

To complement the precision and recall based measures, in our experiments we also provide two measures focusing on the recognition at symbol level. In this case we only consider a binary concept of retrieval. Whether a symbol is found or not. Let us consider one symbol  $S_i$  and its polygonal representation  $Prel_i$  from the ground-truth, it will be considered as recognized if:

$$A(Prel_i \oplus Pret) \geq thr * A(Prel_i) \quad (7.12)$$

That is, if the resulting polygons are able to overlap at least a certain percentage of the ground-truthed representation of a symbol, this symbol is considered as recognized. On the other hand, if the resulting polygons do not cover the ground-truth, the symbol is considered as missed. Of course, as with all decisions implying a certain threshold, its value can be critical, and the system's evaluation can depend on it. Its definition is completely subjective as it depends on what the user considers a symbol as being detected or not. The important thing here is that this value is provided when evaluating a system, so as the readers can easily interpret the meaning of the evaluation results. In our case, we consider a symbol as detected if it overlaps at least a 75% with the ground-truth area.

At symbol level, we derive the *recognition rate* of the spotting system under study. In addition, if one of the polygons  $Pret_j$  in the resulting set does not overlap with any recognized symbol, it is considered a *false positive*. For all the possible queries,



the average of false positives  $AveFP$  is computed. These two measures help to better interpret the values of precision and recall.

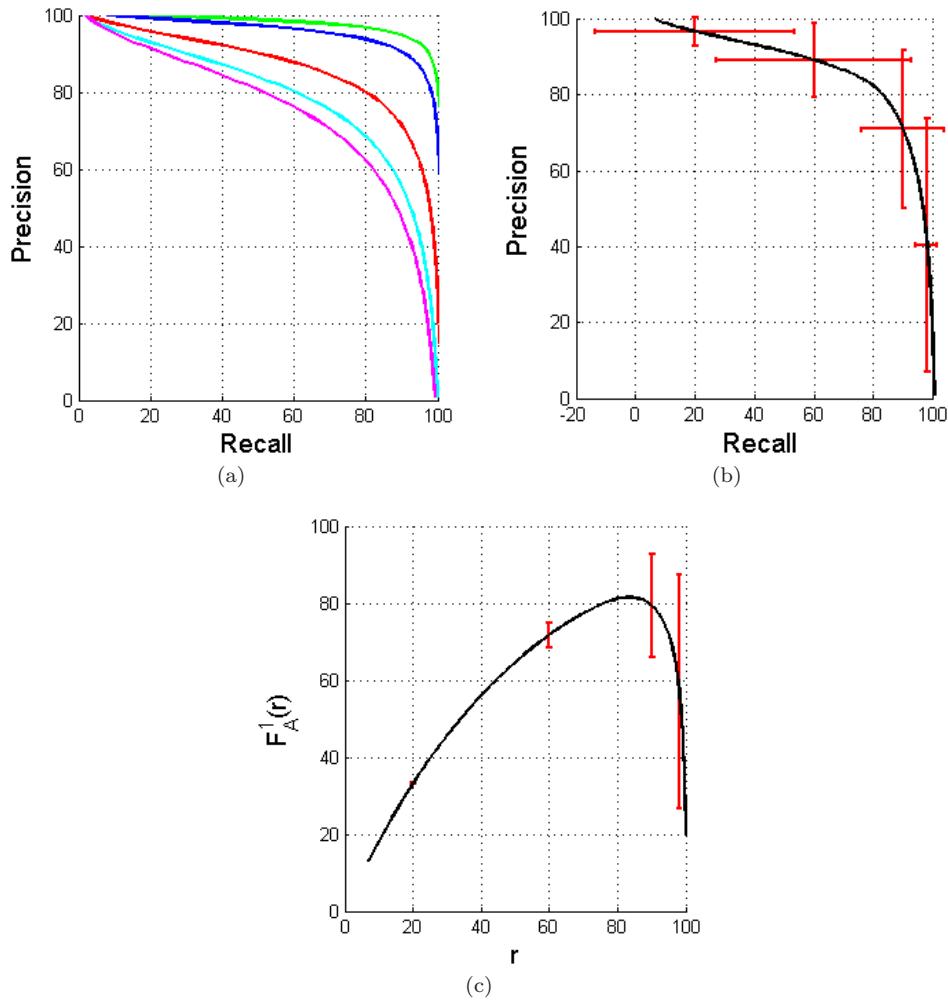
Notice that the recognition rate is expressed as a percentage of the total number of symbols in the ground-truth and can be used as a measure of quality by itself, but the false positives are not normalized and are given in absolute values. This is due to the fact that we can not define the negative set in terms of symbol items in the dataset. The false positive average can only be expressed in absolute values and used to compare methods between them.

#### 7.4.4 Scalability Test

Finally, one of the main interests for spotting systems is that a system has to be applicable to a large data corpora. To test the scalability of the system, i.e. its ability to achieve similar behavior independently from the number of queried symbols, we propose a measure to evaluate the scalability of the systems under study.

A scalable system has to yield similar responses no matter what the number of model classes taken into account is. We can measure the scalability of a system in terms of its variance in both precision and recall. Let us consider the synthetic example of Fig. 7.4a which is highly damaged by the addition of new classes. Let us define  $std_R$  and  $std_P$  the standard deviations in precision and recall for a certain sampling of the precision and recall plot. We can see in Fig. 7.4b the central tendency of all precision and recall plots with error bars following the vertical and horizontal axis to check the effect in both precision and recall measures when considering more and more classes. The greater the deviation is, the worst the system tolerates changes in the class number, thus the system can be considered as less scalable. To allow an easier interpretation, the standard deviation can be computed for the  $F_A^\beta(r)$ -score plots (as shown in Fig. 7.4c) having now a single variance measure instead of having one for precision and one for recall. To compare the scalability between different methods, both the mean  $\overline{std}$  of all the samples of the standard deviation and the maximum  $\max(std)$  of all the samples of the standard deviation are given as variance measures.

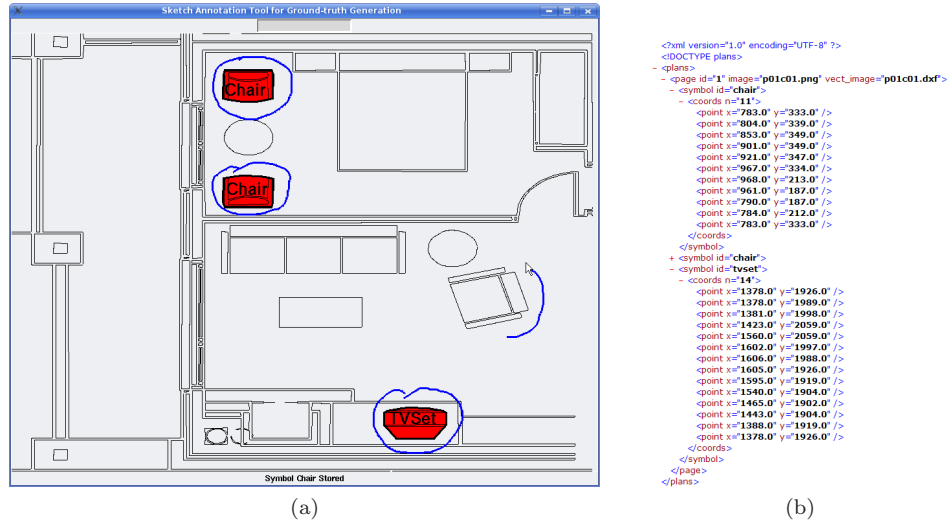
On the other hand, the performance of a spotting system not only is affected by the increasing number of considered models but also is dependent on the size of the document collection. To appreciate how the system degrades with the expansion of the dataset we propose to work at symbol level. Recognition rates and false alarms are given to illustrate the performance variability in relation to the size of the database. These measures help to predict how the performance of a system will be affected by the inclusion of more documents in the database. However, increasing the database size has an important drawback. When adding new documents in the database implicitly we can be adding new graphical symbols contained in these new documents. As a consequence of that the number of model symbols has also to be increased along with the dataset size.



**Figure 7.4:** Scalability test example. (a) Synthetic precision and recall plots; (b) averaged precision and recall plot with standard deviations in recall and precision; (c) averaged  $F_A^1(r)$ -score plot with associated standard deviations.

## 7.5 Evaluating a Symbol Spotting System

In this section, to show an example of application of the presented evaluation framework, we tested a symbol spotting architecture. We first explain the ground-truthing process, then we briefly detail the spotting system and the used dataset and finally we provide the evaluation results for this architecture.



**Figure 7.5:** Sketching annotation tool for ground-truth generation. (a) Graphical interface; (b) the generated ground-truth XML file.

### 7.5.1 Ground-truthing

First, an annotation tool has been developed to build the ground-truth. The user can select graphical entities in the document images roughly segmenting them using a sketching application. All the contour pixels falling inside the delimited zone of interest are taken as being part of the symbol. If a given connected component has more pixels outside the zone of interest than inside, it is considered as being part of the background. This basic annotation tool works fine with architectural drawings where the symbols are usually not extensively connected with background elements. For other kinds of documents, e.g. electronic diagrams or geographical maps, the annotation tool should be enhanced in order to provide a trusted ground-truth. For all the foreground pixels, we compute the convex-hull as presented in [BDH96] as the minimum area of interest which contains the symbol. Once the region of interest is shown, the user can modify it using certain control points and label them by their content. We can see a screen-shot of the sketching application in Fig. 7.5a. The use of convex-hulls as the ground-truth primitive may be inadequate for some spotting systems. The inclusion of noisy pixels in the spotting results may provoke considerable deviations of the convex-hull from the one defined in the ground-truth. However, the presented evaluation measures can be easily adapted to other choices of ground-truth primitives. From coarser to more refined primitives we can select for instance to use bounding-boxes, ellipses, isothetic polygons, quad-trees, etc. as ground-truth primitives. In all these cases, the computation of the overlap between ground-truth and automatically extracted primitives is straightforward.

As the user labels the regions containing the graphical symbols, an XML file is constructed to store the information about the whole library. Following the same file structure used for page layout ground-truth presented in [AKB06], the convex-hull

coordinates and the symbol category as well as other information about the document are organized in the XML file we can see in Fig. 7.5b.

As claimed in [LN01], creating a ground-truth for graphic documents is not always straightforward due to ambiguous cases or subjectivity issues. For example, in the architectural field, each architect tends to use its own symbol designs to represent a furniture element. Whereas human observers have no difficulty in clustering these elements despite the design differences, it is usually impossible for a spotting system to be able to identify different designs as the same object. In the process of ground-truth building, we tried to avoid such problems but we believe that the use of a collaborative framework as proposed in [VDW07] would enhance a lot the quality and the accuracy of this ground-truth. To avoid subjective decisions on the ground-truthing process, synthetic ground-truth can be generated for graphic rich documents, as recently presented in [DPV08]. Such tools which synthetically generate ground-truthed data present several interesting advantages. Subjective decisions are avoided since no human interaction is needed, thus providing an error-free labelling of graphical items. In addition, we have complete control on the number of items in the collection and the number of symbols which have to appear in each document, making the scalability tests much more easy and reliable. However, nowadays the data generated by these methods still appears quite artificial and the use of real data (when possible) should be preferred.

### 7.5.2 Spotting Methods Under Test

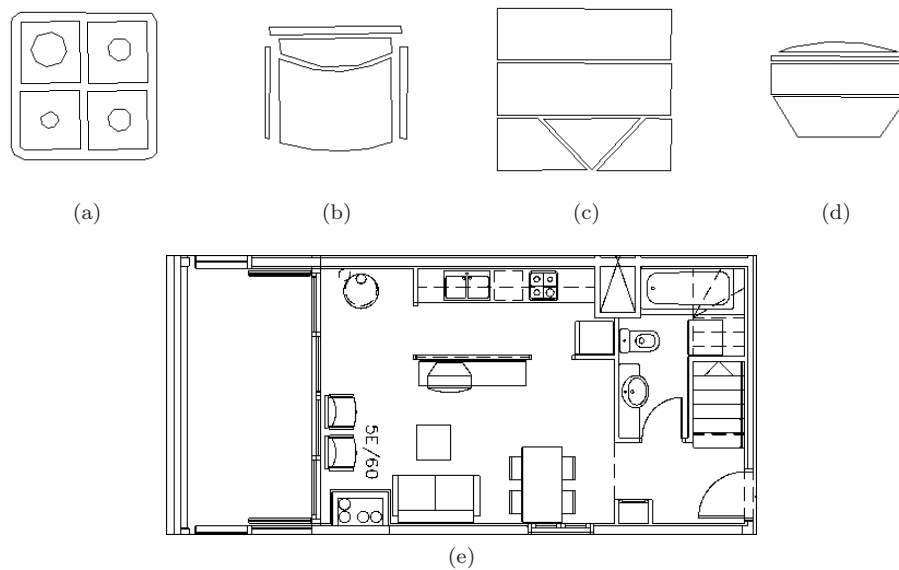
The symbol spotting architecture we use to test the evaluation measures is based on the relational indexing scheme presented in chapter 6. Summarizing, symbols are decomposed in basic primitives which are subsequently described by a geometric symbol description technique. The feature vectors arising from the description are indexed with a hashing technique. When querying this hash table, structural information is added by means of a relational indexing technique. That is, that only similar primitives sharing the same spatial relationship are retrieved. One of the most important points of the system is the way the graphical primitives are described to be indexed. We tested four off-the-shelf geometric symbol descriptors described below.

- **Method a:** uses a set of simple ratios described in [SS94] such as the eccentricity or the non-circularity as shape descriptors. These rough descriptors are formulated from the shape contour of the symbol's primitives. It is expected that the use of such simple shape description can only discriminate very dissimilar shapes; the system should result in a lot of false alarms but should be tolerant to distortions and thus retrieve almost all the instances of the queried symbol.
- **Method b:** uses Hu's geometric invariants [Hu62] to describe contours. These invariants are known as good shape descriptors. The expected performance is to have good spotting rates in all aspects.

- **Method c:** is based on a reformulation of the previous one. Geometric moments can be formulated for polygonally approximated contours [LG96] which are taken as primitives. In this case, the use of simpler primitives should result to smaller tolerance to distortions.
- **Method d:** uses the Fourier transform to compactly represent a curvature signature computed over the shape contour. This descriptor is detailed in [KSP95]. This is also a good shape descriptor and the systems performance is expected to be good in all aspects.

Note that we do not want to perform an exhaustive evaluation of shape descriptors or primitives. These methods have been chosen because of their different nature and to test if the proposed evaluation measures really determine the strong and weak points of each method. As the descriptors are well-known among the Graphics Recognition community, it is easy to assess whether the results correspond to the expected behaviors.

### 7.5.3 The FPLAN-POLY Dataset



**Figure 7.6:** Symbol models and an example of a document in the database. (a) Burner symbol; (b) chair symbol; (c) stairs symbol; (d) TV set symbol; (e) sample document.

The dataset is a collection of architectural floorplans consisting of 42 images (of  $3215 \times 2064$  pixels in average) arising from four different projects. Any given furniture symbol appears in several images in the database. The symbols taken into account

for these experiments are divided into 38 classes and we have in total 344 instances in the document images. In a single document image the average of symbols is around 8 ranging from 0 to 28 symbols. The models to query the document database are cropped from the document images. We can see in Fig. 7.6 some examples of model symbols as well as a sample document from the database. More details of this dataset are given in the appendix A.

#### 7.5.4 Evaluation

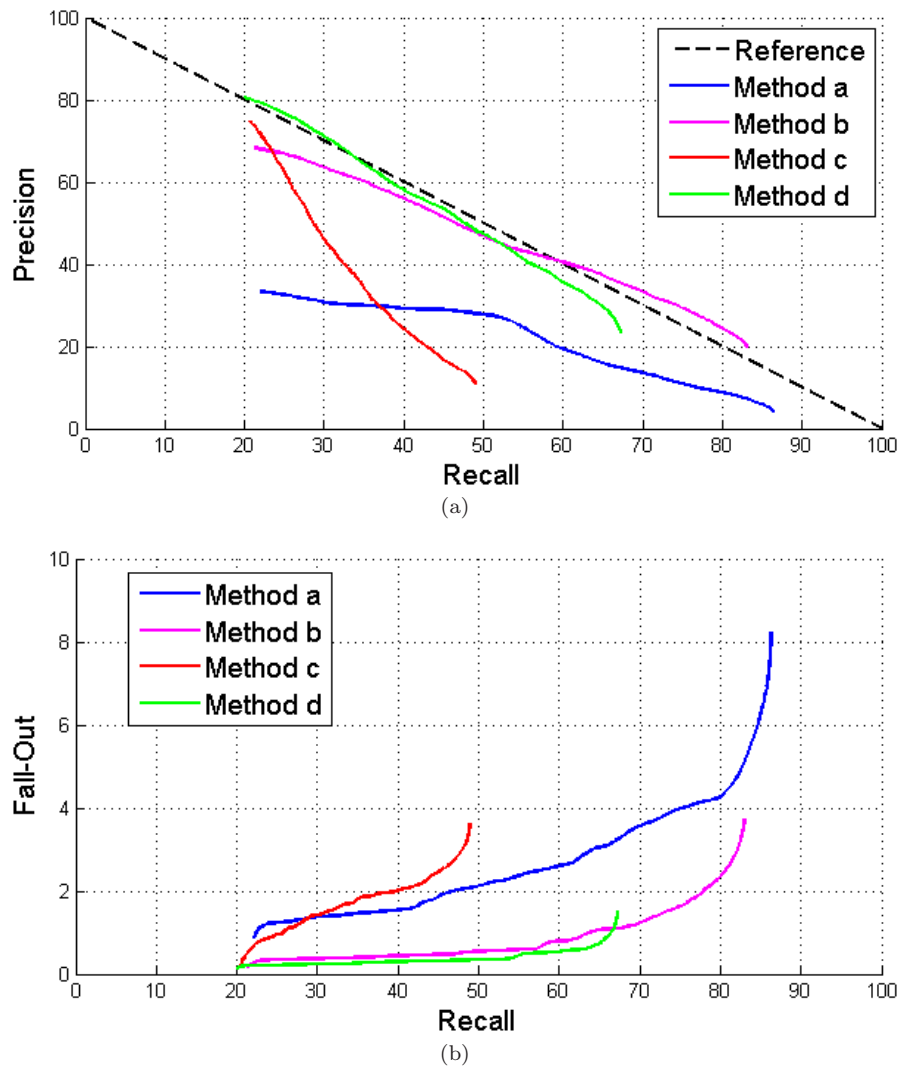


Figure 7.7: (a) Precision versus recall; (b) fall-out versus recall.

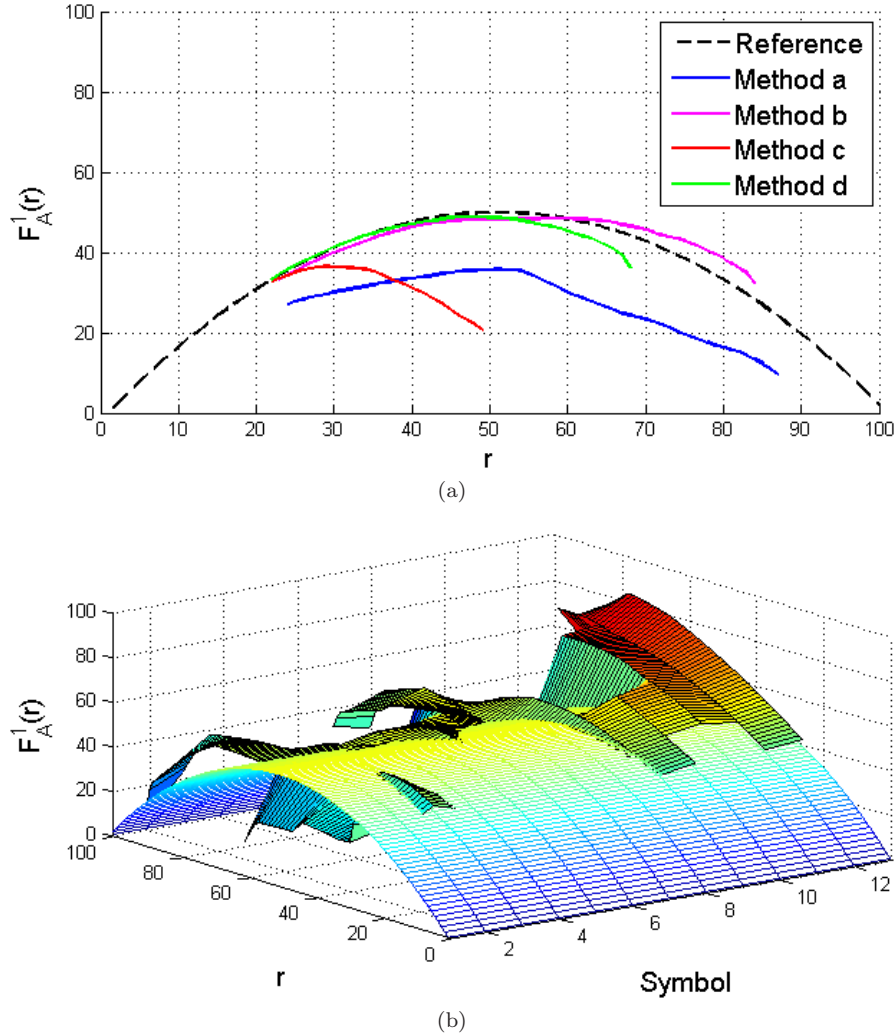
We first present the plots showing precision versus recall and fall-out versus recall in Fig. 7.7 for all the four spotting methods under evaluation using the whole collection of documents. Methods *b* and *d* show an acceptable tradeoff between precision and recall as expected. Method *d* misses much more symbols than method *b* but gives a significantly smaller amount of false positives. Method *a* yields good recall values, i.e. it succeeds in retrieving most of the symbols in the document database but has a poor precision due to the high amount of false positives. Finally, method *d* shows good precision values at early recall stages but quickly falls missing more than half of the symbols in the dataset. The proposed measures aim to stress the expected good behavior of methods *b* and *d* and to point out the simplicity of method *a* and the lack of tolerance of method *c*.

**Table 7.2:** Measures of quality.

Method	$AveP_A$	$F_A^1$ -score	Rec. rate (%)	$AveFP$	$AveG_A$ (%)
<i>a</i>	20.08	6.87	<b>93.62</b>	153.42	
<i>b</i>	39.77	<b>23.34</b>	91.3	76.76	0.16
<i>c</i>	23.69	12.57	55.62	63.89	
<i>d</i>	<b>41.99</b>	21.45	73.33	<b>58.76</b>	

We can appreciate in Fig. 7.8a the  $F_A^1(r)$ -score plots. In this graph we can see again the clear dominance of methods *b* and *d* over the other two. As the  $F$ -score combines both precision and recall, the methods which fail in one of those measures are clearly demoted in the overall evaluation. Method *a* starts with a low precision value while the precision of method *c* quickly falls stopping at a 50% recall. Those two methods are clearly at disadvantage as expected. In Fig. 7.8b we can see how we can use the  $F_A^1(r)$ -score plots to visually check the variance of performance of a given method depending on the symbol the user queries.

In Table 7.2 we can see the measures of quality for all the methods. As the average precision  $AveP_A$  measure does not take into account the recall, the method *d* is ranked as the best. On the other hand,  $F_A^1$ -score gives the best for method *b*. The measures working at symbol level, which are intended to evaluate only the recognition task, are consistent with the results shown in Fig. 7.7b. The amount of recognized symbols is related to the recall value, which ranks the methods in the order *a,b,d* and *c*, in terms of the amount of correctly retrieved elements. On the other hand, the average of false positives is related to the fall-off ratio, ranking the methods in the order *d,c,b* and *a*, in terms of the false alarms present in the results. Finally, the averaged generality gives an idea of the proportion between relevant and total elements in the dataset. These last measures aim to interpret the precision, recall and fall-out values. For spotting applications it is typical to have an extremely low generality measure, since usually the documents in the collections will have much more background objects than foreground ones. This low generality explains the low precision values in both



**Figure 7.8:** (a)  $F_A^1(r)$ -score plot for all methods under test; (b)  $F_A^1(r)$ -score plot depending on the queried symbol for method *b*.

precision and recall plots and in the average precision  $AveP_A$  indicator.

Finally, the scalability test results are shown in Figs. 7.9 and 7.10. Several sets of symbol classes are considered ranging from only 5 to 35 possible symbols to query. We randomly selected  $n$  symbols from the dataset and computed the average precision and recall for these queries. This experiment has been repeated 100 times for the sake of stability and the averaged curves are presented in Figs 7.9a and 7.10a. First, we notice in the Figs. 7.9b and 7.10b that the changes in the number of classes affect different properties depending on the method. The recall of method *a* drastically decreases when introducing more and more symbol classes, whereas the precision of



**Table 7.3:** Scalability test details.

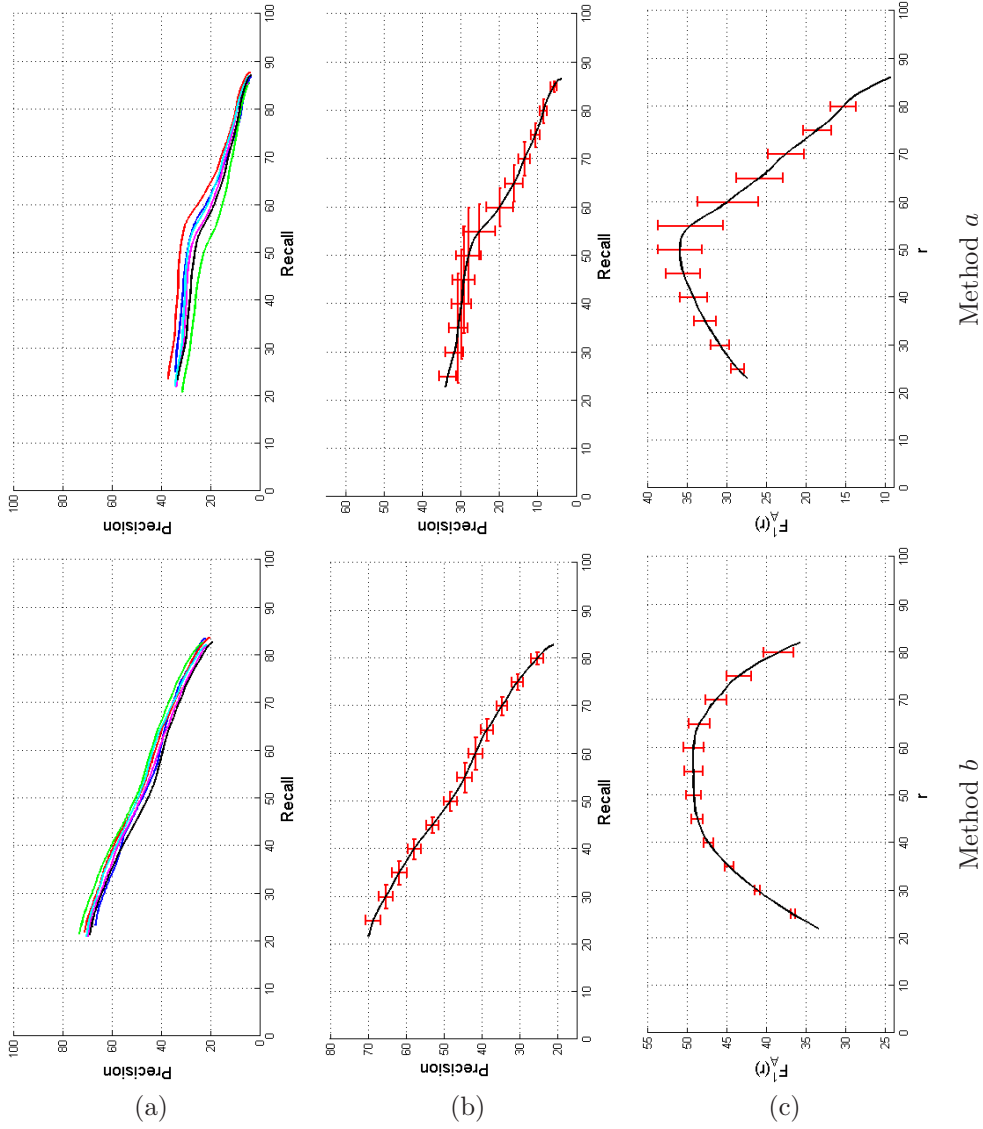
Method	Recall		Precision		F-score	
	$\max(std_R)$	$\overline{std_R}$	$\max(std_P)$	$\overline{std_P}$	$\max(std_F)$	$\overline{std_F}$
<i>a</i>	11.37	6.06	4.2	2.32	4.09	2.2
<i>b</i>	3.38	2.22	<b>2.01</b>	<b>1.74</b>	1.84	0.98
<i>c</i>	<b>1.6</b>	<b>1.02</b>	2.64	2.18	2.17	1.21
<i>d</i>	2.66	1.96	3.44	2.46	<b>1.09</b>	<b>0.85</b>

method *c* suffers much more than the recall. On the other hand, methods *b* and *d* seem to be equally affected by changes in scale in both precision and recall. From Figs. 7.9c and 7.10c we can see how the variations in the  $F_A^1(r)$ -score space are good indicators of the scalability of the methods under study. We can see in Table 7.3 the quality indicators for scalability tests. We present the mean of the standard deviations and its maximums. Again, methods *b* and *d* show much more scalability than methods *c* and *a* when looking at the composite measure. Finally, Figs. 7.11a and 7.11b show the scalability test at symbol level when increasing both the number of models and the dataset size. As we can appreciate, the recognition rates vary slightly whereas the number of false alarms is exponentially increased in all the cases along with the dataset size.

From these results we can conclude that methods *b* and *d* seem to be much better than the other two. Method *b* should be chosen when we desire to retrieve as much symbols as possible, and on the other hand method *d* is suitable if we want to reduce the amount of false positives. Method *a* should only be chosen if the presence of false positives is not a problem and the user prioritizes finding all the positive symbols despite of the presence of false positives. However its performance seems to be affected by the number of considered symbols. Finally, method *c* is only suitable if we are interested in retrieving positive symbols at the first positions of the ranked retrieved locations even if we completely miss the rest of the symbols. Methods *b* and *d* also tolerate well changes in the number of considered classes and should be considered when facing applications involving a large amount of data. On the other hand, the strong points of methods *a* and *c* are compromised when introducing more and more symbol classes. All these conclusions are in accordance with the expected behavior of the studied methods, showing that the proposed evaluation protocol emphasizes the expected strengths and weaknesses of the methods under study.

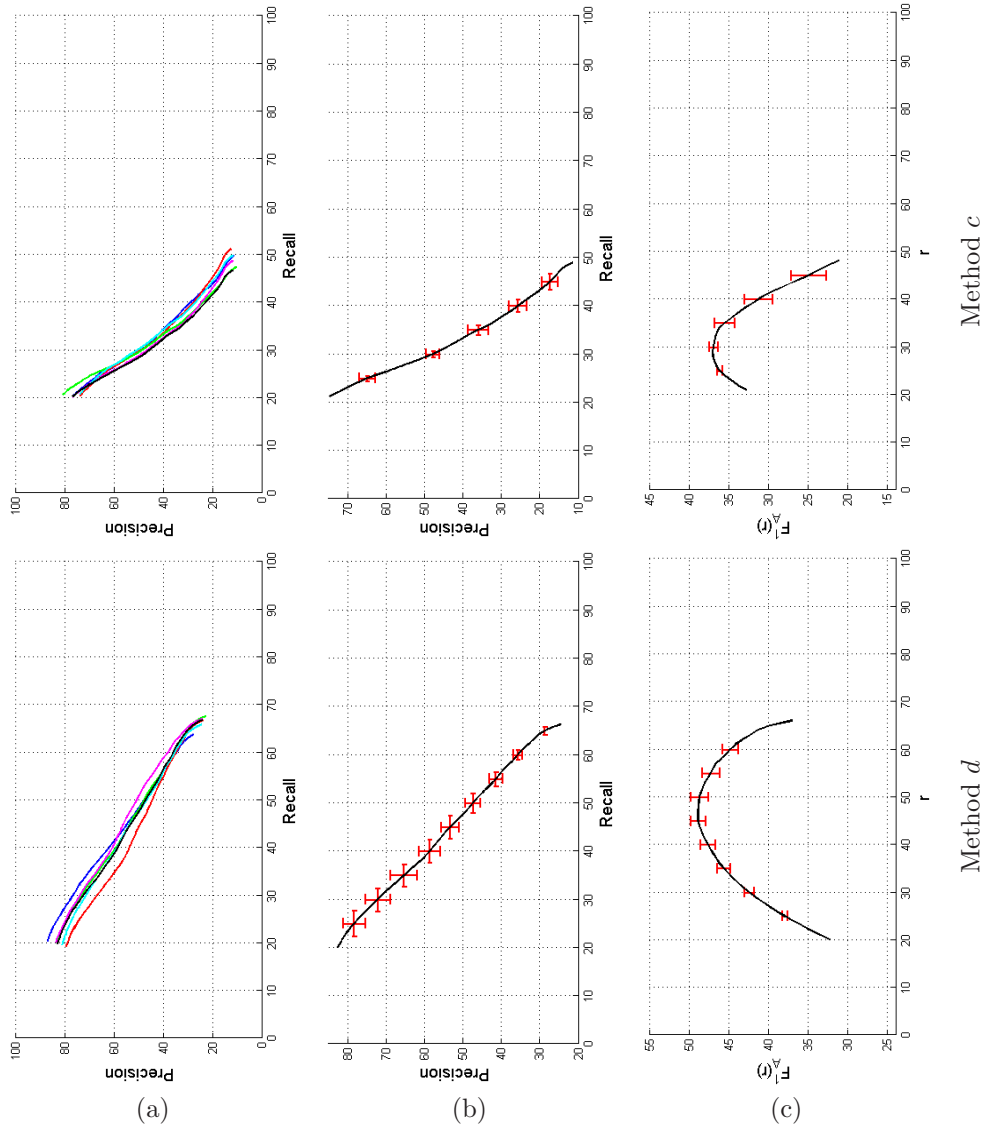
## 7.6 Conclusions and Discussion

Times where algorithms were tested with a small set of data are over. Nowadays, it is necessary the use of standard reference ground-truth and performance evaluation protocols. The Graphics Recognition community is one of the most healthy communities within the Pattern Recognition field regarding this aspect. A lot of works and efforts



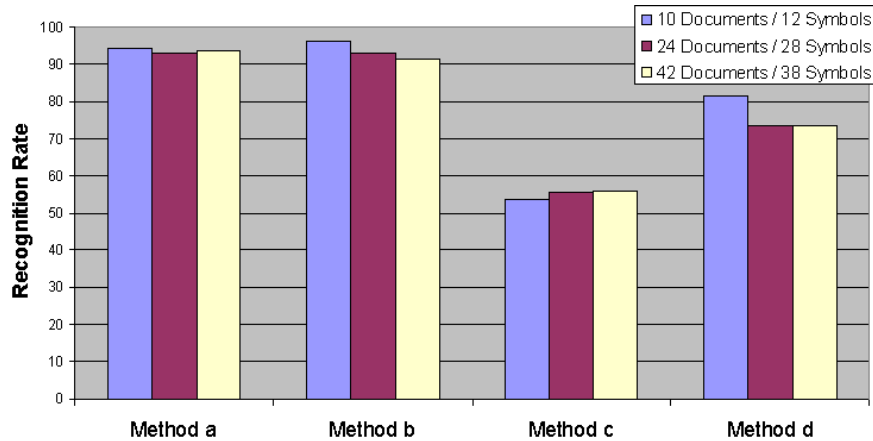
**Figure 7.9:** Scalability tests for the two first methods. (a) Precision recall plots for several amount of symbol classes; (b) averaged precision and recall plot with standard deviations following vertical (precision) and horizontal (recall) axis; (c) averaged  $F_A^1(r)$ -score plot with associated standard deviations.

are centered in proposing evaluation methods which aim to track the progress in a certain specific problem. As far as we know, the works focused on symbol spotting always have been evaluated by an ad-hoc set of measures. We hope that the proposal of the performance evaluation protocol presented in this part of the thesis can be used to evaluate other spotting methods.

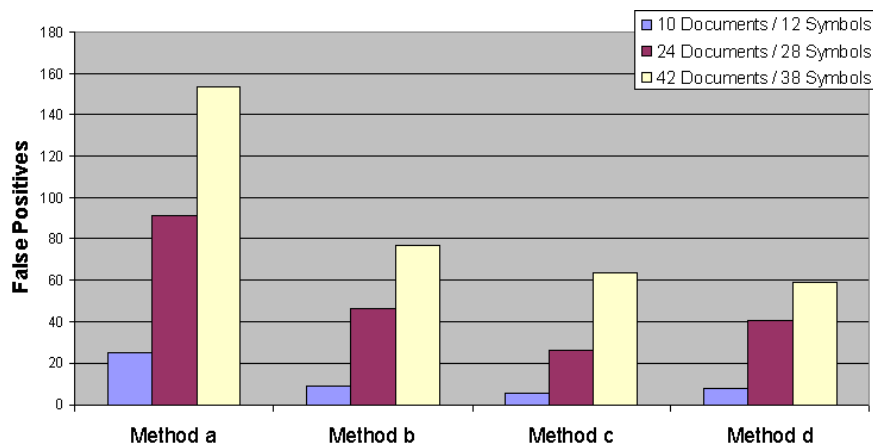


**Figure 7.10:** Scalability tests for the two last methods. (a) Precision recall plots for several amount of symbol classes; (b) averaged precision and recall plot with standard deviations following vertical (precision) and horizontal (recall) axis; (c) averaged  $F_A^1(r)$ -score plot with associated standard deviations.

One of the main problems of evaluating spotting methods is that we do not have any public dataset of real documents to test the proposed methods. Nowadays, the only available ground-truthed dataset which can be used to test spotting and focused retrieval of graphics is the one proposed by Delalandre et al. in [DPV08]. The main problem of this dataset is that it is composed only by synthetical generated documents



(a)



(b)

**Figure 7.11:** Scalability tests at symbol level. (a) Recognition rates; (b) false positives.

which do not yet seem realistic. We preferred to evaluate our work on a set of real documents.

One of the main criticisms of using precision and recall to evaluate the performance of classification and location tasks is that it is sometimes difficult to really assess the behavior of the system under study. As claimed in [Luc05], a low precision value can be due to a low accuracy in the recognition or to a bad localization due to over-segmenting. In addition, as pointed out in [WJ06], the amount of overlap between polygons seems not to be a perceptively valid measure of quality. Quality indicators as the  $F$ -score have been also questioned, in [MKS99] it is argued that this measure makes the systems look like they are much better than they really are.

We believe that the presented measures are able to evaluate well the behavior of symbol spotting and focused retrieval systems, emphasizing their strong and weak points, and their tolerance to changes in scale. Precision is sometimes hard to interpret or does not provide perceptively good indicators, but the point of a spotting system is to retrieve zones of interest of document images, and the presented measures aim to measure the system's ability to do this task. Quality indicators aim to rank the methods according to certain ability, so even if the numbers by themselves do not have an accurate absolute meaning they are useful to compare methods between them. Finally, precision and recall are enhanced by measures working only at symbol level and the generality factor which helps to interpret the meaning of the plots. As shown in the evaluation section, the results obtained by using the proposed evaluation protocol are consistent with the ratios working at symbol recognition level, and most importantly, emphasizes the expected strengths and weaknesses of the methods under study.



# Chapter 8

## Conclusions

---

In this chapter we summarize the contributions of this dissertation to symbol spotting problem and in particular to the application of focused retrieval of graphical symbols from collections of line-drawing images. We also present a discussion and the limitations of the presented approaches. We finally point some possible lines of continuation on the field of symbol spotting and some improvements of the proposed methods which should be further studied.

---

### 8.1 Summary of the Contributions

In this thesis we have introduced a complete framework for symbol spotting, and in particular for a focused retrieval application. As explained in chapter 1, our work has been motivated by the specific problem of proposing a spotting methodology able to locate and retrieve graphical content within a database of complete document images. A lot of interest is made worldwide for mass digitization of document collections and their storage in digital libraries. It results in digital repositories rich in information if they are semantically accessible. Although such semantic access has been improved a lot for textual queries, iconic access is still in early stages, especially when dealing with documents rich in graphical information like technical documents. This is the starting hypothesis of the work developed in this thesis. From a methodological point of view, the main challenges stem from the nature of the queries, which have to be iconic queries instead of the ASCII strings used in the keyword-based searches. In addition of the nature of the queries, the retrieval of the relevant zones should be done on-the-fly. In our framework, the system is queried by example, that is, the user segments a symbol he wants to retrieve from the document database and this cropped image acts as the input. This particularity reinforces the fact that the proposed spotting methods are not meant to work for a specific set of model symbols nor have a learning stage where the relevant features describing a certain symbol can be trained. The use of data structures having graphical patterns as indices so as to provide an efficient access to the graphic information contained in large data corpora

is a must in focused retrieval applications.

We have identified three different levels when conceiving a spotting architecture. The first level aims to represent and compactly describe the primitives that compounds the graphical symbols. In the second level, these features describing graphical symbols are then organized in a particular data structure. This data structure should be chosen carefully in order to provide efficient access to the symbol descriptors. During the querying process, this data structure is traversed and the locations within the document images where to find similar primitives than the queried ones are retrieved. The third level consists in a validation stage to determine which are the valid hypotheses where the queried symbol is likely to be found. Along this thesis, we have made some contributions in each of the three stages. Let us briefly summarize these contributions:

- **Extraction of Vectorial Primitives:** The part II of the thesis has been focused on the use of geometric and structural description techniques aiming to describe the graphical symbols. This family of descriptors, need a previous step of primitive extraction. Since graphical symbols are usually composed by the union of several simple sub-shapes, the basic primitives we have extracted to represent a graphical symbol are these simple sub-shapes. In chapter 4 the polygonally approximated skeletons of the shapes have been taken as the basic primitives representing a symbol. A symbol has been then represented by a set of line segments. Since this representation is quite unstable, in the chapters 5 and 6 a higher level entity has been used as primitive. Adjacent vectors have been merged together into a polyline instance. We have used the contour of the closed loops conforming a symbol as the primitives to polygonally approximate and to merge as single polylines. In chapter 5 we have proven that this primitive representation is more robust and representative than the use of the line segments arising from a vectorization of the skeleton.
- **Geometric Description of Symbols:** In order to describe these extracted primitives, three different methods have been presented in the second part of the thesis. In chapter 4 we have proposed a signature model which is formulated in terms of geometric and structural constraints among vectorial primitives, such as parallelisms, straight angles, etc. After representing vectorized line drawings with attributed graphs, our approach encode the features that are expressive enough to create the signature. The proposed description technique is simple yet effective to discriminate graphical symbols. In chapter 5, chains of adjacent segments have been described by an attributed string formalism. In this chapter, we have used a symbolic description instead of a numeric one. Distances between two primitives have been computed by following a string matching algorithm with a particular cost functions. Finally, in chapter 6, we have proposed to coarsely describe vectorial primitives by a reformulation of several off-the-shelf shape descriptors in order to apply them in the vectorial domain.
- **Efficient Access to Huge Amounts of Descriptors:** Once primitives are extracted and described, we shall organize all the data extracted from the document collection in order to provide an efficient access to it. In chapter 4 we



have worked with a window-based algorithm, so the access to the descriptors has been done in a sequential way. In chapter 5 we have proposed the use of a lookup table allowing a prototype-based search. By clustering primitives by similarity, this indexing data structure has aimed to efficiently retrieve the locations from the document collection where to find similar primitives than the queried ones. In order to try to reduce even more the complexity of accessing the data, we have proposed in chapter 6 the use of indexing structures based on the idea of multidimensional hashing. In particular we have used a grid file structure to organize the descriptor space.

- **Hypotheses Validation:** The last of the three levels in the spotting architectures is the validation of hypotheses. Since from the other levels we have obtained spatial locations where to find similar primitives, those locations should be validated. Along this work we have based our validation steps on the idea to build a Hough-like voting scheme which has aimed to validate the locations where several hypotheses were present. The main contribution within this part has been mainly introduced in chapter 6, where spatial relationships among retrieved primitives have been also introduced as a validation criterion, allowing a more robust identification of the zones of interest.
- **Photometric Descriptors for Symbol Spotting:** Although we have centered our research on a focused retrieval application dealing with line-drawing images, and in this context, a geometric and structural approach seemed the most convenient, we have also worked on another application on the part I of this thesis. An application of document categorization via logo spotting has been presented, and well-known photometric descriptors and matching techniques from the computer vision field have been tested. These kind of description techniques have been rarely used in the Graphics Recognition field, due to the bi-level nature of document images, and yet we have shown its good performance even though the application have to face binary and noised document images.
- **Performance Evaluation Protocol:** Finally, in part III, we have presented a set of measures to evaluate the performance of symbol spotting systems in terms of recognition abilities, location accuracy and scalability. We have shown that the proposed measures allowed to determine the weaknesses and strengths of different methods. In particular, we have evaluated in detail the spotting method presented in chapter 6. Although within the Graphics Recognition community there is an important interest in the research of the performance evaluation topic, to our best knowledge, no framework for evaluating the performance of spotting applications has been proposed in the past.

## 8.2 Discussion

In this thesis we have made some contributions about symbol spotting methods and the particular application of such methods for a focused retrieval system from a collection of line-drawings. Along the second part of the thesis we presented three different

symbol spotting methods which base the description of primitives in a set of geometric and structural constraints. The three methods should be seen as an evolution from the most limited method to a more general and applicable in real situations one.

Although it reaches good recognition results, our first method, presented in chapter 4, presents several important limitations when trying to apply it for spotting symbols in large collections. In order that the vectorial signatures reach good recognition results, we have to make the assumption that the number of segments composing a symbol will maintain stable. When facing real data arising from a raster-to-vector conversion step, this assumption is too strong, and the performance of the spotting system drops. In addition, although the presented method is able to spot symbols, it can not be used as a focused retrieval application. The main cause is the use of window-based systems. Windowing methods provoke a sequential access to the data and are obviously not well suited for large collections.

The use of prototype-based search as the one presented in chapter 5 is clearly a better choice than a sequential access to the data. In a retrieval by similarity framework, this kind of indexing structures allow to drastically reduce the amount of distance computations. However, the computation of the representatives from a given cluster is not always straightforward. The decision on whether two primitives should be considered as similar can be somehow subjective. In our experiments we used the MPEG-7 silhouette database in order to experimentally set up this kind of decision thresholds. However we believe that the performance of such systems might be enhanced by the use of more complex classification and clustering algorithms.

Our feeling is that one of the right directions to follow in spotting-related problems for the next years is the use of coarser descriptors rather than accurate descriptions techniques. We have shown that the combination of coarse description and relational validation, i.e. combining numeric and structural description techniques, yields very good results. In particular, we have proven in chapter 6 that there is no need for high-dimensional descriptors for spotting purposes, and with really simple shape descriptors, we can reach acceptable performances when combining those descriptions with relational information. Obviously, depending on the intended final application, the word “acceptable” may adopt several meanings. As we have seen in part III, if the user of the final application is interested in retrieving the most of the relevant portions of images from the collection, no matter the number of false alarms, a simpler description should be used. If the user is more interested in a better precision without caring the fact the system misses symbols, then we should start using more and more complex and fine description techniques. However, we strongly believe that for most of applications, the use of low-dimensional descriptors (in our experiments of the chapter 6, we use a maximum of seven-dimensional feature vectors) is enough. The choice of such low-dimensional feature vectors avoids the so-called curse of dimensionality and provides an efficient access to the data. The good results obtained by such simple description techniques are also favored by the inclusion of relational and structural information of the graphical symbols. The use of a joint local numerical description and structural analysis contributes to obtain an important discriminative power. However structural information should be added carefully since the analysis of complex structural relationships (entailing comparisons in the graph domain) can

not be managed on the context of symbol spotting due to its huge complexity.

One of the critical assumptions that we made along this thesis is that the graphical symbols can be well represented by a particular primitives, the region contours. Obviously, not in all the cases the symbols are formed by closed loops, and such proposed primitives can not be used. However, the presented architecture and framework remains valid no matter which kind of primitive representation we chose and no matter which appropriate description we select.

Finally, we would like to mention the importance that has the use of a performance evaluation protocol. Times where algorithms were tested with a small set of data are over. Nowadays, it is necessary the use of standard reference ground-truth and performance evaluation protocols. The Graphics Recognition community is one of the most healthy communities within the Pattern Recognition field regarding this aspect. A lot of works and efforts are centered in proposing evaluation methods which aim to track the progress in a certain specific problem. As far as we know, the works focused on symbol spotting always have been evaluated by an ad-hoc set of measures. We hope that the proposal of the performance evaluation protocol presented in part III can be used to evaluate other spotting methods and helps to track the progress on this topic as well as to identify the strengths and weaknesses of the proposed methods. However, one of the main problems is that we do not have any public dataset of real documents to test the proposed methods. Nowadays, the only available ground-truthed dataset which can be used to test spotting and focused retrieval of graphics is the one proposed by Delalandre et al. in [DPV08]. The main problem of this dataset is that it is composed only by synthetical generated documents which do not yet seem realistic, however it is the only one available and the community related to spotting applications should start using it.

### 8.3 Open Challenges

Since symbol spotting is quite a novel problem, we are convinced that there is still a lot of room for improvements and some open challenges.

First of all, the scalability of the proposed methods should be better checked by analyzing other kind of technical documents such as electronic diagrams or mechanical schemes. In addition, even if we have used quite large databases, the methods should be tested on massive data collections in order to assess their transference to real systems.

We would also like to further investigate on the use of a different architecture that the one proposed in the introduction. We would like to use some of the spatial access methods presented in [Sam90] for spotting purposes. These data structures should be able to describe and organize symbols in a single step and might be very useful for focused retrieval applications. They have been used in related problems as GIS system querying for long time.

Another possible research line that we would like to further investigate is the use of the proposed techniques to the retrieval of other elements besides graphical symbols.

To our best knowledge, most of the word spotting techniques existing in the literature work with a previous segmentation of words and with a learning stage where features from the words are extracted and trained. In the existing word spotting methods, indexing strategies are rarely used and the access to the data is often sequential. We would like to apply the indexing mechanisms and the on-the-fly retrieval framework to the topic of retrieving typewritten or even handwritten words in documents.

More generally speaking, the focused retrieval problem dealing with non-textual queries is and will be one of the major topics of interest of the computer vision and data mining communities. The possibility of formulate queries in an abstract level (semantic querying), i.e. graphics appearing in images, sounds being pronounced in speeches, etc. will be one of the major breakthroughs of the next years. Nowadays we are starting to see the first applications that are able to deal with massive data collections. As an example, Google recently released a *beta* version of its *Google Similar Images*<sup>1</sup> search. It allows the user to search for similar images using pictures rather than words. The similarity between images takes into account spatial information, color, shapes, texture, etc. and the results are quite impressive. Although from its behavior it is pretty clear that the software is getting clues from words associated with images, the results are very promising. The similarity computation and the indexation is all done off-line and the interface still does not perform real-time image analysis. The images the user would like to search for can not be uploaded or sketched, but need to be previously indexed.

Finally, another interesting topic which should be further studied is the use of sketch queries instead of a query-by-example paradigm for spotting and focused retrieval applications. In this context, users could roughly sketch the symbol to search into the collection of graphical documents, thus enhancing the usability of the system. Obviously, if the queries are sketched by the users, a distortion model has to be introduced in order to be able to define whether a symbol in a document is similar to the sketch or not, and aiming to tolerate the inherent distortions of the sketches. The use of the Gestalt laws of perceptual organization might be helpful in such applications.

---

<sup>1</sup><http://similar-images.googlelabs.com/>

# Appendix A

## *Databases*

---

Along this work, several databases have been used, namely, the *GREC database* of graphical symbols, the *MPEG database* of silhouette shapes, and the *floorplan database* which is a selection of several real floorplans. In this appendix we will explain in detail all these databases. For each one we will detail the kind of graphical data that composes the database, its vectorial representation and which kind of deformations and distortions are applied to the data.

---

### A.1 GREC 2005 Database

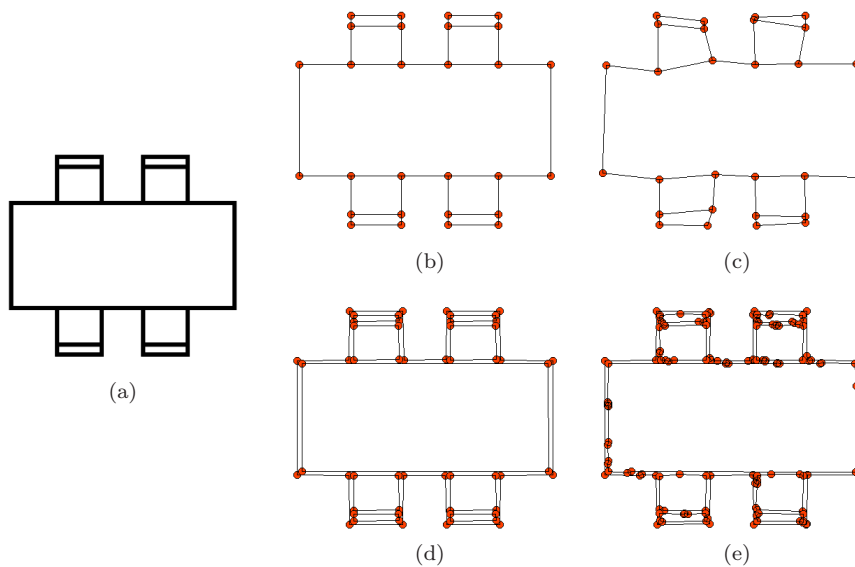
The GREC database is composed by a set of graphical symbols coming from different technical fields. It was originally created for the symbol recognition contests held in the past GREC workshops. The results of these competitions are summarized in the following communications [VD04, VD06, VDF08].

The main goal of the contest is to provide a framework for the evaluation of different methods for symbol recognition in graphic documents. This framework is intended to be general and flexible so that it can be used to evaluate a wide range of symbol recognition methods. The contest is based on a pre-defined set of symbols (tables A.1 to A.3). Using this set of symbols, different tests were generated, consisting of several images of each symbol with increasing levels of degradation and distortion in both bitmap images and vectorial representations.

Based on the complete collection of 150 model symbols we have used in our experiments two variations of the GREC database. The first variation only involves the symbols formed by straight lines. A distortion model is used to introduce noise at the location of the endpoints by preserving the connectivity and the number of segments which compose a symbol. The second variation is applied to all 150 models. A degradation model is applied to the bitmap images which are then polygonally approximated. In this variation, the number of polylines composing a symbol is guaranteed to be constant, but these polylines can be composed by different number of

segments.

### A.1.1 Variation GREC-SEG



**Figure A.1:** Example of variations for the GREC database. (a) Bitmap model; (b) GREC-SEG vectorial model; (c) GREC-SEG with endpoint distortion ( $r = 15$ ); (d) GREC-POLY vectorial model; (e) GREC-POLY with vectorial distortion.

In this variation we have used a subset of 80 different symbols of the original GREC database. The used symbols are the ones which are only composed by straight lines and have no arcs. For each class 60 distorted instances have been generated by using the following operations.

The vectorial representation of each symbol is represented by an attributed graph where the nodes represent points and the edges a segment between two endpoints. Each node from the graph is randomly shifted within a predefined radius  $r$ . Three different levels of distortion are generated with values of  $r$  equal to 5, 10 and 15, respectively. At each level, 20 different instances of the symbol are generated. Notice that the graph representation aim to maintain the connectivity and the number of segments which compose the graphical symbol. In this variation, the symbols are represented by segments stored in VEC format<sup>1</sup>. Fig. A.1c shows an example of this vectorial distortion. Some complementary characteristics of this variation can be seen in Table A.4.

<sup>1</sup>The vectorial file format used in the symbol recognition contests

Table A.1: GREC 2005 Database (1).













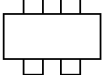










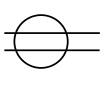




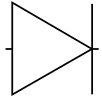
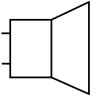
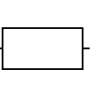
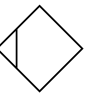
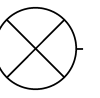

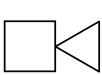






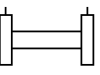
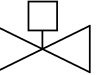
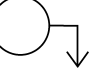
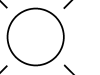


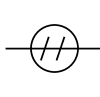
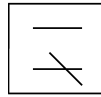
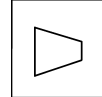
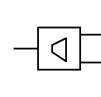
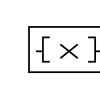
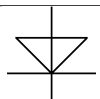
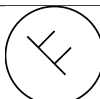
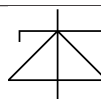
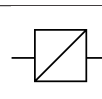
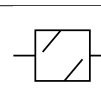
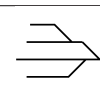
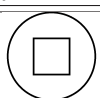
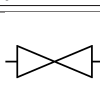
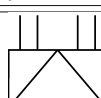
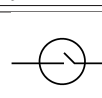
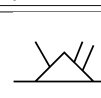
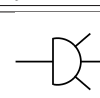
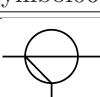
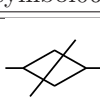
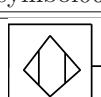
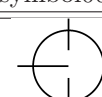
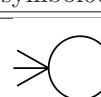
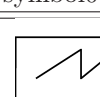

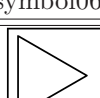
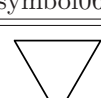

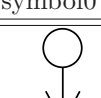
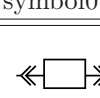
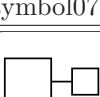
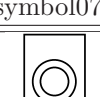
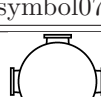
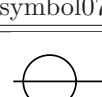
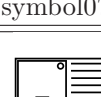
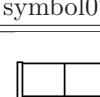
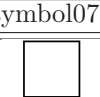
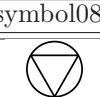
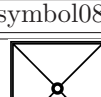
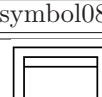
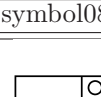
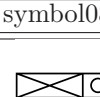
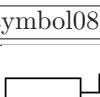
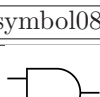
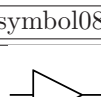
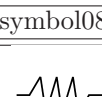
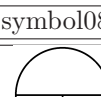
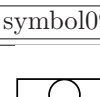
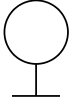
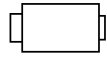


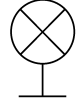
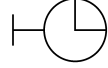
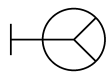




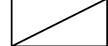
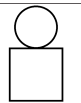
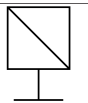

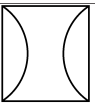

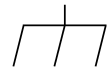
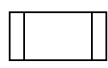





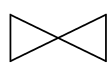
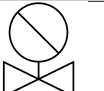




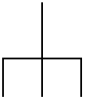
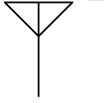

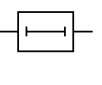
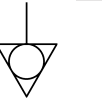

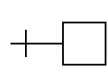
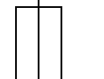

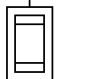
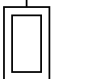
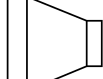
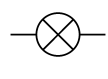
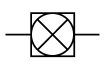
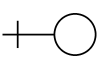

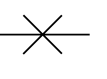
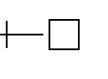
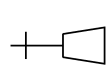

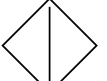
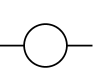


					
symbol001	symbol002	symbol003	symbol004	symbol005	symbol006
					
symbol007	symbol008	symbol009	symbol010	symbol011	symbol012
					
symbol013	symbol014	symbol015	symbol016	symbol017	symbol018
					
symbol019	symbol020	symbol021	symbol022	symbol023	symbol024
					
symbol025	symbol026	symbol027	symbol028	symbol029	symbol030
					
symbol031	symbol032	symbol033	symbol034	symbol035	symbol036
					
symbol037	symbol038	symbol039	symbol040	symbol041	symbol042
					
symbol043	symbol044	symbol045	symbol046	symbol047	symbol048

Table A.2: GREC 2005 Database (2).

					
symbol049	symbol050	symbol051	symbol052	symbol053	symbol054
					
symbol055	symbol056	symbol057	symbol058	symbol059	symbol060
					
symbol061	symbol062	symbol063	symbol064	symbol065	symbol066
					
symbol067	symbol068	symbol069	symbol070	symbol071	symbol072
					
symbol073	symbol074	symbol075	symbol076	symbol077	symbol078
					
symbol079	symbol080	symbol081	symbol082	symbol083	symbol084
					
symbol085	symbol086	symbol087	symbol088	symbol089	symbol090
					
symbol091	symbol092	symbol093	symbol094	symbol095	symbol096



**Table A.3:** GREC 2005 Database (3).

					
symbol097	symbol098	symbol099	symbol100	symbol101	symbol102
					
symbol103	symbol104	symbol105	symbol106	symbol107	symbol108
					
symbol109	symbol110	symbol111	symbol112	symbol113	symbol114
					
symbol115	symbol116	symbol117	symbol118	symbol119	symbol120
					
symbol121	symbol122	symbol123	symbol124	symbol125	symbol126
					
symbol127	symbol128	symbol129	symbol130	symbol131	symbol132
					
symbol133	symbol134	symbol135	symbol136	symbol137	symbol138
					
symbol139	symbol140	symbol141	symbol142	symbol143	symbol144
					
symbol145	symbol146	symbol147	symbol148	symbol149	symbol150

**Table A.4:** Some characteristics of GREC-SEG dataset.

Property	Value
<i>Number of classes</i>	80
<i>Total number of elements</i>	4,800 (60 elements/class)
<i>Max. number of segments in a symbol</i>	36
<i>Min. number of segments in a symbol</i>	3
<i>Mean number of segments in a symbol</i>	10.2

### A.1.2 Variation GREC-POLY

In this variation we have used all the 150 model symbols of the original GREC database. For each class 300 distorted instances have been generated by using the following operations.

The bitmap images are degraded by using the method presented by Kanungo *et al.* in [KHP93] to simulate the noise introduced by the scanning process. Some simple morphological operations are applied to these degraded images to get rid of the background noise. A connected component analysis is applied to label the closed regions and to extract the internal and external contours composing a symbol. These distorted contours are then polygonally approximated by using the Rosin and West algorithm introduced in [RW89]. In this variation, the symbols are composed of several polylines each one composed by a set of adjacent segments. The number of polylines which composes a symbol is constant for a given class, but the number of segments of these polylines is affected by the distortion model and varies from an instance to another. We store these symbols in DXF format. Fig. A.1e shows an example of this vectorial distortion. Some complementary characteristics of this variation can be seen in Table A.5.

## A.2 MPEG Database

The MPEG database is composed by simple pre-segmented shapes defined by their outer closed contours. It is the database used in the MPEG-7 Core Experiment CE-Shape-1 (described in [LLE00]) which aims to evaluate the performance of several 2D shape descriptors. We have adapted a subset of this database to build a shape database in vectorial format.

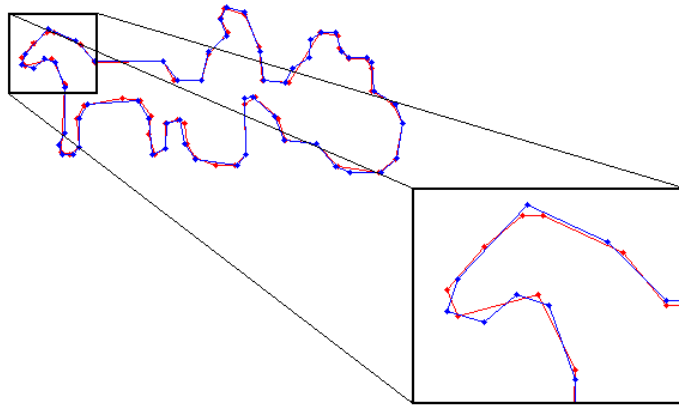
### A.2.1 Variation MPEG-POLY

The silhouettes of the MPEG database may be affected by several distortions such as change of view point or non-rigid object motion. As dealing with such strong deformation was not on the scope of our work, we selected a subset of 15 shape

**Table A.5:** Some characteristics of GREC-POLY dataset.

Property	Value
<i>Number of classes</i>	150
<i>Total number of elements</i>	45,000 (300 elements/class)
<i>Max. number of polylines in a symbol</i>	16
<i>Min. number of polylines in a symbol</i>	1
<i>Mean number of polylines in a symbol</i>	3.9
<i>Max. number of segments in a symbol</i>	264
<i>Min. number of segments in a symbol</i>	11
<i>Mean number of segments in a symbol</i>	73.7

classes (20 elements per class) which are only affected by slight changes in shape. We can see in Tables A.6 and A.7 some examples of the 15 shape classes.

**Figure A.2:** Example of the distortions of the MPEG-POLY database.

For each contour image we applied the same distortion model than in the GREC-POLY variation. The noise model proposed by Kanungo *et al.* is applied to degrade each image. The background noise is cleaned so the distortion only affects the shape contours. These degraded contours are then polygonally approximated. For each image we generate 300 distorted vectorial shapes. We can see in Fig. A.2 an example of the resulting polylines after the degradation process is applied to the same instance of the carriage class. Note that as the database is composed by closed contours, the resulting vectorial shape consist only on a single closed polyline. These vectorial shapes

Table A.6: MPEG Database (1).



























































































					
Bottle class.					
					
Brick class.					
					
Car class.					
					
Carriage class.					
					
Cellular phone class.					
					
Children class.					
					
Face class.					
					
Flatfish class.					

Table A.7: MPEG Database (2).

					
Fountain class.					
					
Key class.					
					
Pencil class.					
					
Personal car class.					
					
Teddy class.					
					
Truck class.					
					
Watch class.					

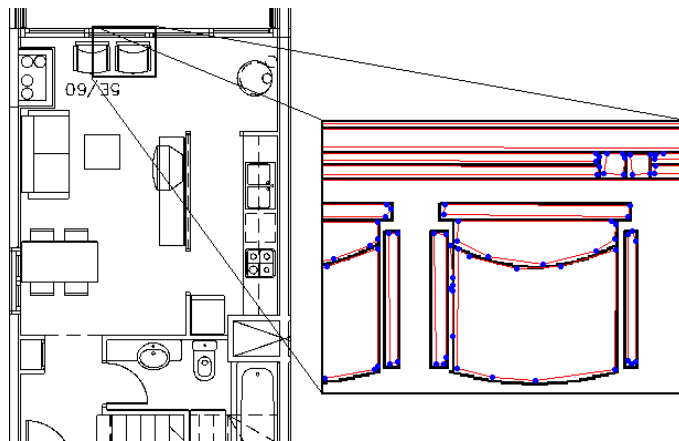
are stored in DXF format. Some complementary characteristics of this variation can be seen in Table A.8.

**Table A.8:** Some characteristics of MPEG-POLY dataset.

Property	Value
<i>Number of classes</i>	15
<i>Number of instances</i>	300 (20 instances/class)
<i>Total number of elements</i>	90,000 (300 elements/instance)
<i>Max. number of segments in a shape</i>	91
<i>Min. number of segments in a shape</i>	7
<i>Mean number of segments in a shape</i>	34.1

### A.3 FPLAN-POLY Database

The FPLAN-POLY database consist of a collection of 42 real floorplans in both DWG and PDF format. The floorplans come from four different projects designed by the same architect. Nevertheless, the same symbol design can only be found in floorplans from the same project. These originals have been ground-truthed and 388 symbols instances from 38 different symbol classes have been labelled.



**Figure A.3:** Example of the distortions of the FPLAN-POLY database.

To simulate the scanning acquisition process, we applied to each plan the same strategy from the GREC-VECT and MPEG-VECT variations. The floorplan images

are degraded via the Kanungo noise and after a cleaning process, they are vectorized to obtain the DXF files. Each floorplan image has been degraded 50 times to have a large database. We can see in Fig. A.3 the results of this distortion process. We find in Table A.9 the complementary characteristics of this database.

**Table A.9:** Some characteristics of FPLAN-POLY dataset.

<b>Property</b>	<b>Value</b>
<i>Number of real floorplans</i>	42
<i>Number of distorted floorplans</i>	2,100 (50 instances/floorplan)
<i>Number of symbol classes</i>	38
<i>Mean number of symbols per floorplan</i>	8.2
<i>Max. number of polylines in a floorplan</i>	3,710
<i>Min. number of polylines in a floorplan</i>	35
<i>Mean. number of polylines in a floorplan</i>	972.3
<i>Mean. number of segments conforming a polyline</i>	3.2





# List of Publications

This dissertation has led to the following communications:

## Journal Papers

- M. Rusiñol, Agnès Borràs and J. Lladós. Relational Indexing of Vectorial Primitives for Symbol Spotting in Line-Drawing Images. Submitted to *Pattern Recognition Letters*.
- M. Rusiñol, J. Lladós and G. Sánchez. Symbol Spotting in Vectorized Technical Drawings Through a Lookup Table of Region Strings. *Pattern Analysis & Applications*, 2009. doi:10.1007/s10044-009-0161-2.
- M. Rusiñol and J. Lladós. A Performance Evaluation Protocol for Symbol Spotting Systems in Terms of Recognition and Location Indices. *International Journal on Document Analysis and Recognition*, 2009. doi:10.1007/s10032-009-0083-y.

## Conference Contributions

- M. Rusiñol, K. Bertet, J.M. Ogier and J. Lladós. Symbol Recognition by Using a Concept Lattice of Graphical Patterns. To appear in *The Eight International Workshop on Graphics Recognition, GREC09*, 2009.
- M. Rusiñol and J. Lladós. Logo Spotting by a Bag-of-words Approach for Document Categorization. To appear in *The Tenth International Conference on Document Analysis and Recognition, ICDAR09*, 2009.
- D. Karatzas, M. Rusiñol, C. Antens and M. Ferrer. Segmentation Robust to the Vignette Effect for Machine Vision Systems. In *Proceedings of the Nineteenth International Conference on Pattern Recognition, ICPR08*, 2008. doi:10.1109/ICPR.2008.4760957.
- M. Rusiñol and J. Lladós. Word and Symbol Spotting Using Spatial Organization of Local Descriptors. In *Proceedings of the Eighth IAPR Workshop on Document Analysis Systems, DAS08*, pages 489–496, 2008. doi: 10.1109/DAS.2008.24.

- M. Rusiñol and J. Lladós. A Region-Based Hashing Approach for Symbol Spotting in Technical Documents. In *Graphics Recognition. Recent Advances and New Opportunities*, volume 5046 of *Lecture Notes on Computer Science*, pages 321–328, 2008. doi: 10.1007/978-3-540-88188-9\_11.
- M. Rusiñol, J. Lladós and P. Dosch. Camera-Based Graphical Symbol Detection. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 884–888, 2007. doi: 10.1109/ICDAR.2007.76.
- M. Rusiñol, P. Dosch and J. Lladós. Boundary Shape Recognition Using Accumulated Length and Angle Information. In *Proceedings of the Third Iberian Conference on Pattern Recognition and Image Analysis, IBPRIA07*, volume 4478 of *Lecture Notes on Computer Science*, pages 210–217, 2007. doi: 10.1007/978-3-540-72849-8\_27.
- M. Rusiñol and J. Lladós. Symbol Spotting in Technical Drawings Using Vectorial Signatures. In *Graphics Recognition. Ten Years Review and Future Perspectives*, volume 3926 of *Lecture Notes on Computer Science*, pages 35–46, 2006. doi: 10.1007/11767978\_4.

## Internal Workshops & Technical Reports

- M. Rusiñol and J. Lladós. Relational indexing of local descriptors to spot graphical objects in wiring diagrams. In *Proceedings of the Third CVC Internal Workshop on the Progress of Research & Development: Current Challenges in Computer Vision, CVCRD08*, pages 98–103, 2008.
- M. Rusiñol and J. Lladós. Graphical Symbol Spotting in Technical Documents Using a Region Hash Table. In *Proceedings of the Second CVC Workshop, Computer Vision: Advances in Research & Development, CVCRD07*, pages 13–17, 2007.
- M. Rusiñol. A Model of Vectorial Signatures in terms of Expressive Sub-Shapes: Symbol Indexation in Technical Documents. *CVC Technical Report* Num. 94, 2006.
- M. Rusiñol and J. Lladós. Querying Symbols in Technical Documents by an Indexing Lookup Table of Polylines. In *Proceedings of the First CVC Internal Workshop on the Progress of Research & Development, CVCRD06*, pages 140–145, 2006.

# Bibliography

- [AOC00] S. Adam, J.M. Ogier, C. Cariou, R. Mullot, J. Labiche, and J. Gardes. Symbol and Character Recognition: Application to Engineering Drawings. *International Journal on Document Analysis and Recognition*, 3(2):89–101, 2000. doi: 10.1007/s100320000033.
- [AST01] C. Ah-Soon and K. Tombre. Architectural symbol recognition using a network of constraints. *Pattern Recognition Letters*, 22(2):231–248, 2001. doi: 10.1016/S0167-8655(00)00091-X.
- [AYS00] S. Aksoy, M. Ye, M.L. Schaaf, M. Song, Y. Wang, R.M. Haralick, J.M. Parker, J. Pivovarov, D. Royko, S. Changming, and G. Farneback. Algorithm Performance Contest. In *Proceedings of the Fifteenth International Conference on Pattern Recognition, ICPR00*, pages 870–876, 2000. doi: 10.1109/ICPR.2000.903054.
- [AB07] A. Antonacopoulos and D. Bridson. Performance Analysis Framework for Layout Analysis Methods. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 1258–1262, 2007. doi: 10.1109/ICDAR.2007.4377117.
- [AGB05] A. Antonacopoulos, B. Gatos, and D. Bridson. ICDAR 2005 Page Segmentation Competition. In *Proceedings of the Eighth International Conference on Document Analysis and Recognition, ICDAR05*, pages 75–79, 2005. doi: 10.1109/ICDAR.2005.184.
- [AGB07] A. Antonacopoulos, B. Gatos, and D. Bridson. ICDAR 2007 Page Segmentation Competition. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 1279–1283, 2007. doi: 10.1109/ICDAR.2007.203.
- [AGK03] A. Antonacopoulos, B. Gatos, and D. Karatzas. ICDAR 2003 Page Segmentation Competition. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition, ICDAR03*, pages 688–692, 2003. doi: 10.1109/ICDAR.2003.1227750.
- [AKB06] A. Antonacopoulos, D. Karatzas, and D. Bridson. Ground Truth for Layout Analysis Performance Evaluation. In *Document Analysis Systems*,

- DAS06*, volume 3872 of *Lecture Notes on Computer Science*, pages 302–311. 2006. doi: 10.1007/11669487\_27.
- [ADW94] C. Apté, F. Damerau, and S.M. Weiss. Towards Language Independent Automated Learning of Text Categorization Models. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 23–30, 1994.
- [ACH91] E.M. Arkin, L.P. Chew, D.P. Huttenlocher, K. Kedem, and J.S.B. Mitchell. An Efficiently Computable Metric for Comparing Polygonal Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):209–216, 1991. doi: 0.1109/34.75509.
- [AJL02] M. Artáč, M. Jogan, and A. Leonardis. Incremental PCA for On-line Visual Learning and Recognition. In *Proceedings of the International Conference on Pattern Recognition, ICPR02*, pages 781–784, 2002. doi: 10.1109/ICPR.2002.1048133.
- [BBB07] A.D. Bagdanov, L. Ballan, M. Bertini, and A. Del Bimbo. Trademark Matching and Retrieval in Sports Video Databases. In *Proceedings of the International Workshop on Multimedia Information Retrieval*, pages 79–86, 2007. doi: 10.1145/1290082.1290096.
- [Baj94] G.S. Di Baja. Well-shaped, Stable, and Reversible Skeletons from the (3,4)-Distance Transform. *Journal of Visual Communications and Image Representation*, 5(1):107–115, 1994.
- [Bal81] D.H. Ballard. Generalizing the Hough Transform to Detect Arbitrary Shapes. *Pattern Recognition*, 13(2):111–122, 1981.
- [BDH96] C.B. Barber, D.P. Dobkin, and H. Huhdanpaa. The Quickhull Algorithm for Convex Hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996. doi: 10.1145/235815.235821.
- [BHA05] E. Barbu, P. Héroux, S. Adam, and E. Trupin. Using Bags of Symbols for Automatic Indexing of Graphical Document Image Databases. In *Graphics Recognition. Ten Years Review and Future Perspectives*, volume 3926 of *Lecture Notes on Computer Science*, pages 195–205. 2005. doi: 10.1007/11767978\_18.
- [BM76] R. Bayer and J.K. Metzger. On the Encipherment of Search Trees and Random Access Files. *ACM Transactions on Database Systems*, 1(1):37–52, 10.1145/320434.320445 1976.
- [BKS90] N. Beckmann, H.P. Kriegel, R. Schneider, and B. Seeger. The R\*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 322–331, 1990. doi: 10.1145/93597.98741.
- [Bel57] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

- [BMP02] S. Belongie, J. Malik, and J. Puzicha. Shape Matching and Object Recognition Using Shape Contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002. doi: 10.1109/34.993558.
- [Ben75] J.L. Bentley. Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517, 1975. doi: 10.1145/361002.361007.
- [BBP00] S. Berretti, A. Del Bimbo, and P. Pala. Retrieval by Shape Similarity with Perceptual Distance and Effective Indexing. *IEEE Transactions on Multimedia*, 2(4):225–239, 2000. doi: 10.1109/6046.890058.
- [BB08] R. Bertolami and H. Bunke. Hidden Markov Model-based Ensemble Methods for Offline Handwritten Text Line Recognition. *Pattern Recognition*, 41(11):3452–3460, 2008. doi: 10.1016/j.patcog.2008.04.003.
- [BP97] A. Del Bimbo and P. Pala. Visual Image Retrieval by Elastic Matching of User Sketches. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):121–132, 1997. doi: 10.1109/34.574790.
- [Bok92] M. Bokser. Omnidocument Technologies. In *Proceedings of the IEEE*, volume 80, pages 1066–1078, 1992. doi: 10.1109/5.156470.
- [BG94] M. Buckland and F. Gey. The Relationship between Recall and Precision. *Journal of the American Society for Information Science*, 45(1):12–19, 1994. doi: 10.1002/(SICI)1097-4571(199401)45:1;12::AID-ASI2;3.0.CO;2-L.
- [BvKS07] M. Bulacu, R. van Koert, L. Shomaker, and T. van der Zant. Layout Analysis of Handwritten Historical Documents for Searching the Archive of the Cabinet of the Dutch Queen. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 357–361, 2007. doi: 10.1109/ICDAR.2007.4378732.
- [Bun82] H. Bunke. Attributed Programmed Graph Grammars and their Application to Schematic Diagram Interpretation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(6):574–582, 1982. doi: 10.1109/TPAMI.1982.4767310.
- [BB93] H. Bunke and U. Bühler. Applications of Approximate String Matching to 2D Shape Recognition. *Pattern Recognition*, 26(12):1797–1812, 1993. doi: 10.1016/0031-3203(93)90177-X.
- [CM94] A. Califano and R. Mohan. Multidimensional Indexing for Recognizing Visual Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4):373–392, 1994. doi: 10.1109/34.277591.
- [CL91] C.C. Chang and S.Y. Lee. Retrieval of Similar Pictures on Pictorial Databases. *Pattern Recognition*, 24(7):675–681, 1991. doi: 10.1016/0031-3203(91)90034-3.

- [Che93] C.C. Chen. Improved Moment Invariants for Shape Discrimination. *Pattern Recognition*, 26(5):683–686, 1993. doi: 10.1016/0031-3203(93)90121-C.
- [CKL93] T. Cheng, J. Khan, H. Liu, and D.Y.Y. Yun. A Symbol Recognition System. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition, ICDAR03*, pages 918–921, 1993. doi: 10.1109/ICDAR.1993.395587.
- [CRM04] C.W. Chong, P. Raveendran, and R. Mukundan. Translation and Scale Invariants of Legendre Moments. *Pattern Recognition*, 37(1):119–129, 2004. doi: 10.1016/j.patcog.2003.06.003.
- [CPZ97] P. Ciaccia, M. Patella, and P. Zezula. M-Tree: An Efficient Access Method for Similarity Search in Metric Spaces. In *Proceedings of the Twenty-third International Conference on Very Large Data Bases, VLDB97*, pages 426–435, 1997.
- [CS00] M.S. Costa and L.G. Shapiro. 3D Object Recognition and Pose with Relational Indexing. *Computer Vision and Image Understanding*, 79(3):364–407, 2000. doi: 10.1006/cviu.2000.0865.
- [CCL07] B. Coüason, J. Camillerapp, and I. Leplumey. Access by Content to Handwritten Archive Documents: Generic Document Recognition Method and Platform for Annotations. *International Journal on Document Analysis and Recognition*, 9(2–4):223–242, 2007. doi: 10.1007/s10032-007-0044-2.
- [CGV08] M. Coustaty, S. Guillas, M. Visani, K. Bertet, and J.M. Ogier. On the Joint Use of a Structural Signature and a Galois Lattice Classifier for Symbol Recognition. In *Graphics Recognition. Recent Advances and New Opportunities*, volume 5046 of *Lecture Notes on Computer Science*, pages 61–70. 2008. doi: 10.1007/978-3-540-88188-9\_7.
- [DG06] J. Davis and M. Goadrich. The Relationship between Precision-Recall and ROC Curves. In *Proceedings of the Twenty-Third International Conference on Machine Learning*, pages 233–240, 2006. doi: 10.1145/1143844.1143874.
- [DPV08] M. Delalandre, T. Pridmore, E. Valveny, H. Locteau, and E. Trupin. Building Synthetic Graphical Documents for Performance Evaluation. In *Graphics Recognition. Recent Advances and New Opportunities*, volume 5046 of *Lecture Notes on Computer Science*, pages 288–298. Springer Verlag, 2008. doi: 10.1007/978-3-540-88188-9\_27.
- [Dor95] D. Dori. Vector-based Arc Segmentation in the Machine Drawing Understanding System Environment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1057–1068, 1995. doi: 10.1109/34.473231.

- [DL04] P. Dosch and J. Lladós. Vectorial Signatures for Symbol Discrimination. In *Graphics Recognition: Recent Advances and Perspectives*, volume 3088 of *Lecture Notes on Computer Science*, pages 154–165. 2004. doi: 10.1007/b99011.
- [EFM07] F. Esposito, S. Ferilli, N. Di Mauro, and T.M.A. Basile. Incremental Learning of First Order Logic Theories for the Automatic Annotations of Web Documents. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 1093–1097, 2007. doi: 10.1109/ICDAR.2007.4377084.
- [ESM91] A. Etemadi, J.P. Schmidt, G. Matas, J. Illinworth, and J.V. Kittler. Low-Level Grouping of Straight Line Segments. In *Proceedings of the British Machine Vision Conference, BMVC91*, pages 119–126, 1991.
- [Faw06] T. Fawcett. An Introduction to ROC Analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. doi: 10.1016/j.patrec.2005.10.010.
- [FVS09] M. Ferrer, E. Valveny, and F. Serratos. Median Graph: A New Exact Algorithm Using a Distance Based on the Maximum Common Subgraph. *Pattern Recognition Letters*, 30(5):579–588, 2009. doi: 10.1016/j.patrec.2008.12.014.
- [FOL04] P. Franco, J.M. Ogier, P. Loonis, and R. Mullot. A Topological Measure for Image Object Recognition. In *Graphics Recognition: Recent Advances and Perspectives*, volume 3088 of *Lecture Notes on Computer Science*, pages 279–290. 2004. doi: 10.1007/b99011.
- [Fre61] H. Freeman. On the Encoding of Arbitrary Geometric Configurations. *IRE Transactions on Electronic Computers*, 2:260–268, 1961.
- [Fu74] K.S. Fu. *Syntactic Methods in Pattern Recognition*. Academic Press, New York, 1974.
- [GG98] V. Gaede and O. Günther. Multidimensional Access Methods. *ACM Computing Surveys*, 30(2):170–231, 1998. doi: 10.1145/280277.280279.
- [GJ79] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W.H. Freeman, 1979.
- [GN93] H. Gish and K. Ng. A Segmental Speech Model with Applications to Word Spotting. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP93*, pages 447–450, 1993. doi: 10.1109/ICASSP.1993.319337.
- [GBO06] S. Guillas, K. Bertet, and J.M. Ogier. A Generic Description of the Concept Lattices Classifier: Application to Symbol Recognition. In *Graphics Recognition. Ten Years Review and Future Perspectives*, volume 3926 of *Lecture Notes in Computer Science*, pages 47–60. 2006. doi: 10.1007/11767978\_5.

- [Gut84] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 47–57, 1984. doi: 10.1145/602259.602266.
- [HS88] C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Proceedings of the Alvey Vision Conference*, pages 147–151, 1988.
- [HSW89] A. Henrich, H.W. Six, and P. Widmayer. The LSD-Tree: Spatial Access to Multidimensional Point and Non Point Objects. In *Proceedings of the Fifteenth International Conference on Very Large Databases, VLDB89*, pages 45–53, 1989.
- [HWH07] F. Holz, H.F. Witschel, G. Heinrich, G. Heyer, and S. Teresniak. An Evaluation Framework for Semantic Search in P2P Networks. In *Proceedings of the Seventh International Workshop on Innovative Internet Community Systems, I2CS07*, 2007.
- [HR05] G. Hripcsak and A.S. Rothschild. Agreement, the F-Measure, and Reliability in Information Retrieval. *Journal of the American Medical Informatics Association*, 12(3):296–298, 2005. doi: 10.1197/jamia.M1733.
- [HN04] H. Hse and A.R. Newton. Sketched Symbol Recognition Using Zernike Moments. In *Proceedings of the Seventeenth International Conference on Pattern Recognition, ICPR04*, pages 367–370, 2004. doi: 10.1109/ICPR.2004.1334128.
- [Hu62] M. Hu. Visual Pattern Recognition by Moment Invariants. *IRE Transactions on Information Theory*, 8:179–187, 1962.
- [Hua97] P.W. Huang. Indexing Pictures by Key Objects for Large-scale Image Databases. *Pattern Recognition*, 30(7):1229–1237, 1997. doi: 10.1016/S0031-3203(96)00143-4.
- [HH99] B. Huet and E.R. Hancock. Line Pattern Retrieval Using Relational Histograms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1363–1370, 1999. doi: 10.1109/34.817414.
- [HS05] D.P. Huijsmans and N. Sebe. How to Complete Performance Graphs in Content-Based Image Retrieval: Add Generality and Normalize Scope. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2):245–251, 2005. doi: 10.1109/TPAMI.2005.30.
- [HdC95] T.M. Hupkens and J. de Clippeleir. Noise and intensity invariant moments. *Pattern Recognition Letters*, 16(4):371–376, 1995. doi: 10.1016/0167-8655(94)00110-O.
- [HU87] D.P. Huttenlocher and S. Ullman. Object Recognition Using Alignment. In *Proceedings of the First International Conference on Computer Vision, ICCV87*, pages 102–111, 1987.



- [Jag90] H.V. Jagadish. Spatial Search with Polyhedra. In *Proceedings of the Sixth International Conference on Data Engineering, ICDE90*, pages 311–319, 1990. doi: 10.1109/ICDE.1990.113483.
- [JMB01] X. Jiang, A. Munger, and H. Bunke. On Median Graphs: Properties, Algorithms, and Applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1144–1151, 2001. doi: 10.1109/34.954604.
- [JFJ95] G.J.F. Jones, J.T. Foote, K.S. Jones, and S.J. Young. Video Mail Retrieval: The Effect of Word Spotting Accuracy on Precision. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP95*, pages 309–312, 1995. doi: 10.1109/ICASSP.1995.479535.
- [JSAH07] S.R. Joty and S. Sadid-Al-Hasan. Advances in Focused Retrieval: A General Review. In *Proceedings of the Tenth International Conference on Computer and Information Technology, ICCIT07*, pages 1–5, 2007. doi: 10.1109/ICCITECHN.2007.4579357.
- [KKL04] B.Y. Kang, H.J. Kim, and S.J. Lee. Performance Analysis of Semantic Indexing in Text Retrieval. In *Computational Linguistics and Intelligent Text Processing*, volume 2945 of *Lecture Notes on Computer Science*, pages 433–436. 2004. doi: 10.1007/b95558.
- [KHP93] T. Kanungo, R.M. Haralick, and I. Philips. Global and Local Document Degradation Models. In *Proceedings of the Second International Conference on Document Analysis and Recognition, ICDAR93*, pages 730–734, 1993. doi: 10.1109/ICDAR.1993.395633.
- [KS97] N. Katayama and S. Satoh. The SR-Tree: An Indexing Structure for High-Dimensional Nearest Neighbor Queries. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 369–380, 1997. doi: 10.1002/(SICI)1520-684X(19980615)29:6;59::AID-SCJ6;3.0.CO;2-K.
- [KSP95] H. Kauppinen, T. Seppänen, and M. Pietikäinen. An Experimental Comparison of Autoregressive and Fourier-based Descriptors in 2D Shape Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:201–207, 1995. doi: 10.1109/34.368168.
- [KB02] S. Kaygin and M.M. Bulut. Shape Recognition Using Attributed String Matching with Polygon Vertices as the Primitives. *Pattern Recognition Letters*, 23(1–3):287–294, 2002. doi: 10.1016/S0167-8655(01)00111-8.
- [KS04] Y. Ke and R. Sukthankar. PCA-SIFT: A More Distinctive Representation for Local Image Descriptor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR04*, pages 506–513, 2004. doi: 10.1109/CVPR.2004.1315206.

- [KH90] A. Khotanzad and Y.H. Hong. Invariant Image Recognition by Zernike Moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):489–497, 1990. doi: 10.1109/34.55109.
- [Kin97] V. Kindratenko. *Development and Application of Image Analysis Techniques for Identification and Classification of Microscopic Particles*. PhD thesis, University of Antwerp, Belgium, 1997.
- [KTM02] K. Kise, M. Tsujino, and K. Matsumoto. Spotting Where to Read on Pages - Retrieval of Relevant Parts from Page Images. In *Document Analysis Systems V*, volume 2423 of *Lecture Notes on Computer Science*, pages 388–399. 2002. doi: 10.1007/3-540-45869-7\_43.
- [KAD04] B. Klein, S. Agne, and A. Dengel. Results of a Study on Invoice-Reading Systems in Germany. In *Document Analysis Systems VI*, volume 3163 of *Lecture Notes on Computer Science*, pages 451–462. 2004. doi: 10.1007/b100557.
- [KGN07] T. Konidaris, B. Gatos, K. Ntzios, I. Pratikakis, S. Theodoridis, and S.J. Perantonis. Keyword-guided Word Spotting in Historical Printed Documents Using Synthetic Data and User Feedback. *International Journal on Document Analysis and Recognition*, 9(2–4):167–177, 2007. doi: 10.1007/s10032-007-0042-4.
- [KPF01] F. Korn, B.U. Pagel, and C. Faloustos. On the “Dimensionality Curse” and the “Self-similarity Blessing”. *IEEE Transactions on Knowledge and Data Engineering*, 13(1):96–111, 2001. doi: 10.1109/69.908983.
- [KA94] S.S. Kuo and O.E. Agazzi. Keyword Spotting in Poorly Printed Documents using Pseudo 2D Hidden Markov Models. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 16(8):842–848, 1994. doi: 10.1109/34.308482.
- [LG95] G. Lambert and H. Gao. Line Moments and Invariants for Real Time Processing of Vectorized Contour Data. In *Image Analysis and Processing*, volume 974 of *Lecture Notes on Computer Science*, pages 347–352. 1995.
- [LG96] G. Lambert and H. Gao. Discrimination Properties of Invariants Using the Line Moments of Vectorized Contours. In *Proceedings of the Thirteenth International Conference on Pattern Recognition, ICPR96*, pages 735–739, 1996. doi: 10.1109/ICPR.1996.546920.
- [LW88] Y. Lamdan and H.J. Wolfson. Geometric Hashing: A General and Efficient Model-based Recognition Scheme. In *Proceedings of the Second International Conference on Computer Vision, ICCV88*, pages 238–249, 1988.
- [LG96] B. Lamiroy and P. Gros. Rapid Object Indexing and Recognition Using Enhanced Geometric Hashing. In *Computer Vision, ECCV*, volume 1064 of *Lecture Notes on Computer Science*, pages 59–70. 1996. doi: 10.1007/BFb0015523.

- [LLE00] L.J. Latecki, R. Lakämper, and T. Eckhardt. Shape Descriptors for Non-Rigid Shapes with a Single Closed Contour. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR00*, pages 424–429, 2000. doi: 10.1109/CVPR.2000.855850.
- [LLE07] Y. Leydier, F. Lebourgeois, and H. Emptoz. Text Search for Medieval Manuscript Images. *Pattern Recognition*, 40(12):3552–3567, 2007. doi: 10.1016/j.patcog.2007.04.024.
- [LD08] X. Liu and D. Doerman. Mobile Retriever: Access to Digital Documents from their Physical Source. *International Journal on Document Analysis and Recognition*, 11(1):19–27, 2008. doi: 10.1007/s10032-008-0066-4.
- [LMV01] J. Lladós, E. Martí, and J. Villanueva. Symbol Recognition by Error-Tolerant Subgraph Matching between Region Adjacency Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1137–1143, 2001. doi: 10.1109/34.954603.
- [LS07] J. Lladós and G. Sánchez. Indexing Historical Documents by Word Shape Signatures. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 362–366, 2007. doi: 10.1109/ICDAR.2007.4378733.
- [LVS02] J. Lladós, E. Valveny, G. Sánchez, and E. Martí. Symbol Recognition: Current Advances and Perspectives. In *Graphics Recognition Algorithms and Applications*, volume 2390 of *Lecture Notes on Computer Science*, pages 104–127. 2002. doi: 10.1007/3-540-45868-9\_9.
- [LD97] R.P. Loce and E.R. Dougherty. *Enhancement and Restoration of Digital Documents: Statistical Design of Nonlinear Algorithms*. Society of Photo-Optical Instrumentation Engineers (SPIE), Bellingham USA, 1997.
- [LAT07] H. Locteau, S. Adam, E. Trupin, J. Labiche, and P. Héroux. Symbol Spotting Using Full Visibility Graph Representation. In *Proceedings of the Seventh International Workshop on Graphics Recognition, GREC07*, 2007.
- [LN01] D. Lopresti and G. Nagy. Issues in Ground-Truthing Graphic Documents. In *Graphics Recognition Algorithms and Applications*, volume 2390 of *Lecture Notes on Computer Science*, pages 46–66. 2001. doi: 10.1007/3-540-45868-9\_5.
- [LM95] O. Lorenz and G. Monagan. A Retrieval System for Graphical Documents. In *Proceedings of the Fourth Symposium on Document Analysis and Information Retrieval, SDAIR95*, pages 291–300, 1995.
- [Low87] D.G. Lowe. Three-dimensional Object Recognition from Single Two-dimensional Images. *Artificial Intelligence*, 31(3):355–395, 1987. doi: 10.1016/0004-3702(87)90070-1.

- [Low99] D.G. Lowe. Object Recognition from Local Scale-Invariant Features. In *Proceedings of the Seventh International Conference on Computer Vision, ICCV99*, pages 1150–1157, 1999. doi: 10.1109/ICCV.1999.790410.
- [Low04] D.G. Lowe. Distinctive Image Features from Scale-invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. doi: 10.1023/B:VISI.0000029664.99615.94.
- [LSS07] C.T. Lu, M. Shukla, S.H. Subramanya, and Y. Wu. Performance Evaluation of Desktop Search Engines. In *Proceedings of the IEEE International Conference on Information Reuse and Integration, IRI07*, pages 110–115, 2007. doi: 10.1109/IRI.2007.4296606.
- [LS99] G. Lu and A. Sajjanhar. Region-based Shape Representation and Similarity Measure Suitable for Content-based Image Retrieval. *Multimedia Systems*, 7(2):165–174, 1999. doi: 10.1007/s005300050119.
- [LT08] S. Lu and C.L. Tan. Retrieval of Machine-Printed Latin Documents Through Word Shape Coding. *Pattern Recognition*, 41(5):1816–1826, 2008. doi: 10.1016/j.patcog.2007.10.017.
- [Luc05] S.M. Lucas. ICDAR 2005 Text Locating Competition Results. In *Proceedings of the Eighth International Conference on Document Analysis and Recognition, ICDAR05*, pages 80–84, 2005. doi: 10.1109/ICDAR.2005.231.
- [LPS05] S.M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, K. Ashida, H. Nagai, M. Okamoto, H. Yamamoto, H. Miyao, J.M. Zhu, W.W. Ou, C. Wolf, J.M. Jolion, L. Todoran, M. Worring, and X. Lin. ICDAR 2003 Robust Reading Competitions: Entries, Results, and Future Directions. *International Journal on Document Analysis and Recognition*, 7(2):105–122, 2005. doi: 10.1007/s10032-004-0134-3.
- [Mae91] M. Maes. Polygonal Shape Recognition Using String-matching Techniques. *Pattern Recognition*, 24(5):433–440, 1991. doi: 10.1016/0031-3203(91)90056-B.
- [MKS99] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel. Performance Measures for Information Extraction. In *Proceedings of DARPA Broadcast News Workshop*, 1999.
- [Mar92] J.N. Marcus. A Novel Algorithm for HMM Word Spotting Performance Evaluation and Error Analysis. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP92*, pages 89–92, 1992. doi: 10.1109/ICASSP.1992.226113.
- [MJS08] J. Mas, J.A. Jorge, G. Sánchez, and J. Lladós. Representing and Parsing Sketched Symbols Using Adjacency Grammars and a Grid-directed Parser. In *Graphics Recognition. Recent Advances and New Opportunities*, volume 5046 of *Lecture Notes on Computer Science*, pages 169–180. 2008. doi: 10.1007/978-3-540-88188-9\_17.

- [MKL97] B.M. Mehtre, M.S. Kankanhalli, and W.F. Lee. Shape Measures for Content Based Image Retrieval: A Comparison. *Information Processing & Management*, 33(3):319–337, 1997. doi: 10.1016/S0306-4573(96)00069-6.
- [MB96] B.T. Messmer and H. Bunke. Automatic Learning and Recognition of Graphical Symbols in Engineering Drawings. In *Graphics Recognition Methods and Applications*, volume 1072 of *Lecture Notes on Computer Science*, pages 123–134. 1996. doi: 10.1007/3-540-61226-2\_11.
- [MS01] K. Mikolajczyk and C. Schmid. Indexing Based on Scale Invariant Interest Points. In *Proceedings of the Eight IEEE International Conference on Computer Vision, ICCV*, pages 525–531, 2001. doi: 10.1109/ICCV.2001.937561.
- [MS04] K. Mikolajczyk and C. Schmid. Scale & Affine Invariant Interest Point Detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004. doi: 10.1023/B:VISI.0000027790.02288.f2.
- [MS05] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005. doi: 10.1109/TPAMI.2005.188.
- [MAK96] F. Mokhtarian, S. Abbasi, and J. Kittler. Robust and Efficient Shape Indexing Through Curvature Scale Space. In *Proceedings of the British Machine Vision Conference, BMV96*, pages 53–62, 1996.
- [MBM01] G. Mori, S. Belongie, and J. Malik. Shape Contexts Enable Efficient Retrieval of Similar Shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR01*, pages 723–730, 2001. doi: 10.1109/CVPR.2001.990547.
- [MMS01] H. Müller, W. Müller, D.M.G. Squire, S. Marchand-Maillet, and T. Pun. Performance Evaluation in Content-Based Image Retrieval: Overview and Proposals. *Pattern Recognition Letters*, 22(5):593–601, 2001. doi: 10.1016/S0167-8655(00)00118-5.
- [MR00] S. Müller and G. Rigoll. Engineering Drawing Database Retrieval Using Statistical Pattern Spotting Techniques. In *Graphics Recognition Recent Advances*, volume 1941 of *Lecture Notes on Computer Science*, pages 246–255. 2000. doi: 10.1007/3-540-40953-X\_21.
- [NGB01] L. Najman, O. Gibot, and M. Barbey. Automatic Title Block Location in Technical Drawings. In *Proceedings of the Fourth International Workshop on Graphics Recognition, GREC01*, pages 19–26, 2001.
- [NKI05] T. Nakai, K. Kise, and M. Iwamura. Camera-based Document Image Retrieval as Voting for Partial Signatures of Projective Invariants. In *Proceedings of the Eight International Conference on Document Analysis and Recognition, ICDAR*, pages 379–383, 2005. doi: 10.1109/ICDAR.2005.64.

- [NKI06] T. Nakai, K. Kise, and M. Iwamura. Use of Affine Invariants in Locally Likely Arrangement Hashing for Camera-based Document Image Retrieval. In *Document Analysis Systems VII*, volume 3872 of *Lecture Notes on Computer Science*, pages 541–552. 2006. doi: 10.1007/11669487\_48.
- [NSS02] J. Neumann, H. Samet, and A. Soffer. Integration of Local and Global Shape Analysis for Logo Classification. *Pattern Recognition Letters*, 23(12):1449–1457, 2002. doi: 10.1016/S0167-8655(02)00105-8.
- [NBM06] T. Neumann, M. Bender, S. Michel, and G. Weikum. A Reproducible Benchmark for P2P Retrieval. In *Proceedings of the First International Workshop on Performance and Evaluation of Data Management Systems, ExpDB06*, pages 1–8, 2006.
- [NHS84] J. Nievergelt, H. Hinterberger, and K.C. Sevcik. The Grid File: An Adaptable, Symmetric Multikey File Structure. *ACM Transactions on Database Systems*, 9(1):38–71, 1984. doi: 10.1007/3-540-10885-8\_45.
- [Nis02] H. Nishida. Structural Feature Indexing for Retrieval of Partially Visible Shapes. *Pattern Recognition*, 35:55–67, 2002. doi: 10.1016/S0031-3203(01)00042-5.
- [NGP08] K. Ntirogiannis, B. Gatos, and I. Pratikakis. An Objective Evaluation Methodology for Document Image Binarization Techniques. In *Proceedings of The Eighth IAPR International Workshop on Document Analysis Systems, DAS08*, pages 217–224, 2008. doi: 10.1109/DAS.2008.41.
- [POK05] S. Pang, S. Ozawa, and N. Kasabov. Incremental Linear Discriminant Analysis for Classification of Data Streams. *IEEE Transactions on Systems, Man and Cybernetics*, 35(5):905–914, 10.1109/TSMCB.2005.847744 2005.
- [PC99] I.T. Phillips and A.K. Chhabra. Empirical Performance Evaluation of Graphics Recognition Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9):849–870, 1999. doi: 10.1109/34.790427.
- [PG08] P. Punitha and D.S. Guru. Symbolic Image Indexing and Retrieval by Spatial Similarity: An Approach Based on B-Tree. *Pattern Recognition*, 41:2068–2085, 2008. doi: 10.1016/j.patcog.2007.09.012.
- [QRB08] R.L. Qureshi, J.Y. Ramel, D. Barret, and H. Cardot. Spotting Symbols in Line Drawing Images Using Graph Representations. In *Graphics Recognition. Recent Advances and New Opportunities*, volume 5046 of *Lecture Notes on Computer Science*, pages 91–103. 2008. doi: 10.1007/978-3-540-88188-9\_10.
- [QRC07] R.L. Qureshi, J.Y. Ramel, H. Cardot, and P. Mukherji. Combination of Symbolic and Statistical Features for Symbols Recognition. In *Proceedings of the International Conference on Signal Processing, Communications and Networking, ICSCN07*, pages 477–482, 2007. doi: 10.1109/ICSCN.2007.350784.

- [RM03] T.M. Rath and R. Manmatha. Features for Word Spotting in Historical Manuscripts. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition, ICDAR03*, pages 218–222, 2003. doi: 10.1109/ICDAR.2003.1227662.
- [RM03] T.M. Rath and R. Manmatha. Word Image Matching Using Dynamic Time Warping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR03*, pages 521–527, 2003. doi: 10.1109/CVPR.2003.1211511.
- [Rob81] J.T. Robinson. The K-D-B-Tree: A Search Structure for Large Multi-dimensional Dynamic Indexes. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 10–18, 1981. doi: 10.1145/582318.582321.
- [RJN93] J.R. Rohlicek, P. Jeanrenaud, K. Ng, H. Gish, B. Musicus, and M. Siu. Phonetic Training and Language Modeling for Word Spotting. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP93*, pages 459–462, 1993. doi: 10.1109/ICASSP.1993.319340.
- [RW89] P.L. Rosin and G.A.W. West. Segmentation of Edges into Lines and Arcs. *Image and Vision Computing*, 7(2):109–114, 1989. doi: 10.1016/0262-8856(89)90004-8.
- [Rus] J.C. Russ. *The Image Processing Handbook*. CRC Press, Boca Raton.
- [SMVG08] S.P. Saldarriaga, E. Morin, and C. Viard-Gaudin. Categorization of Online Handwritten Documents. In *Proceedings of The Eighth IAPR International Workshop on Document Analysis Systems, DAS08*, pages 95–102, 2008. doi: 10.1109/DAS.2008.45.
- [Sam90] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, 1990.
- [SLT02] G. Sánchez, J. Lladós, and K. Tombre. A Mean String Algorithm to Compute the Average Among a Set of 2D Shapes. *Pattern Recognition Letters*, 23:203–213, 2002. doi: 10.1016/S0167-8655(01)00122-2.
- [SDI94] H.K. Sardana, M.F. Daemi, and M.K. Ibrahim. Global Description of Edge Patterns Using Moments. *Pattern Recognition*, 27(1):109–118, 1994. doi: 10.1016/0031-3203(94)90021-3.
- [Say73] K.M. Sayre. Machine Recognition of Handwritten Words: A Project Report. *Pattern Recognition*, 5(3):213–228, 1973. doi: 10.1016/0031-3203(73)90044-7.
- [Sha69] A.C. Shaw. A Formal Picture Description Scheme as a Basis for Picture Processing Systems. *InfoControl*, 14(1):9–52, 1969.

- [SRE05] J. Sivic, B.C. Russell, A.A. Efros, A. Zisserman, and W.T. Freeman. Discovering Objects and their Localization in Images. In *Proceedings of the Tenth International Conference on Computer Vision, ICCV05*, pages 370–377, 2005. doi: 10.1109/ICCV.2005.77.
- [SOK06] A.F. Smeaton, P. Over, and W. Kraaij. Evaluation Campaigns and TRECVID. In *Proceedings of the Eighth ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, 2006. doi: 10.1145/1178677.1178722.
- [SMK72] O.J.M. Smith, K. Makani, and L. Krishna. Sparse Solutions Using Hash Storage. *IEEE Transactions on Power Apparatus and Systems*, 91(4):1396–1404, 1972. doi: 10.1109/TPAS.1972.293271.
- [SST02] J. Song, F. Su, C.L. Tai, and S. Cai. An Object-Oriented Progressive-Simplification-Based Vectorization System for Engineering Drawings: Model, Algorithm, and Performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1048–1060, 2002. doi: 10.1109/TPAMI.2002.1023802.
- [SM92] F. Stein and G. Medioni. Structural Indexing: Efficient 2D Object Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(12):1198–1204, 1992. doi: 10.1109/34.177385.
- [SM92] F. Stein and G. Medioni. Structural Indexing: Efficient 3D Object Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):125–145, 1992. doi: 10.1109/34.121785.
- [SS94] D. Stoyan and H. Stoyan. *Fractals, Random Shapes and Point Fields (Methods of Geometrical Statistics)*. John Wiley & Sons, Chichester, 1994.
- [SCC02] M.C. Su, H.H. Chen, and W.C. Cheng. A Neural-network-based Approach to Optical Symbol Recognition. *Neural Processing Letters*, 15(2):117–135, 2002. doi: 10.1023/A:1015288717988.
- [SM99] T. Syeda-Mahmood. Indexing of Technical Line Drawing Databases. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(8):737–751, 1999. doi: 10.1109/34.784287.
- [TW04] S. Tabbone and L. Wendling. Recognition of Symbols in Grey Level Line-Drawings from an Adaptation of the Radon Transform. In *Proceedings of the Seventeenth International Conference on Pattern Recognition, ICPR04*, pages 570–573, 2004. doi: 10.1109/ICPR.2004.1334310.
- [TWT03] S. Tabbone, L. Wendling, and K. Tombre. Matching of Graphical Symbols in Line-Drawing Images Using Angular Signature Information. *International Journal on Document Analysis and Recognition*, 6(2):115–125, 2003. doi: 10.1007/s10032-003-0105-0.



- [TWZ04] S. Tabbone, L. Wendling, and D. Zuwala. A Hybrid Approach to Detect Graphical Symbols in Documents. In *Document Analysis Systems VI, DAS04*, volume 3163 of *Lecture Notes on Computer Science*, pages 342–353. 2004. doi: 10.1007/b100557.
- [TZ07] S. Tabbone and D. Zuwala. An Indexing Method for Graphical Documents. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 789–793, 2007. doi: 10.1109/ICDAR.2007.4377023.
- [TVH05] M. Tănase, R.C. Veltkamp, and H. Haverkort. Multiple Polyline to Polygon Matching. In *Algorithms and Computation*, volume 3872 of *Lecture Notes on Computer Science*, pages 60–70. 2005. doi: 10.1007/11602613.8.
- [TR03] E. Tapia and R. Rojas. Recognition of On-line Handwritten Mathematical Formulas in the E-chalk System. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition, ICDAR03*, pages 980–984, 2003.
- [TÓD90] T. Taxt, J.B. Ólafsdóttir, and M. Dæhlenshort. Recognition of Handwritten Symbols. *Pattern Recognition*, 23(11):1155–1166, 1990. doi: 10.1016/0031-3203(90)90113-Y.
- [Tea80] M.R. Teague. Image Analysis Via the General Theory of Moments. *Journal of the Optical Society of America*, 70(8):920–930, 1980. doi: 10.1364/JOSA.70.000920.
- [TC88] C.H. Teh and R.T. Chin. On Image Analysis by the Methods of Moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(4):496–513, 1988. doi: 10.1109/34.3913.
- [TASD00] K. Tombre, C. Ah-Soon, P. Dosch, G. Masini, and S. Tabbone. Stable and Robust Vectorization: How to Make the Right Choices. In *Graphics Recognition Recent Advances*, volume 1941 of *Lecture Notes on Computer Science*, pages 3–18. 2000. doi: 10.1007/3-540-40953-X\_1.
- [TL03] K. Tombre and B. Lamiroy. Graphics Recognition - from Re-Engineering to Retrieval. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition, ICDAR03*, pages 148–155, 2003. doi: 10.1109/ICDAR.2003.1227650.
- [TL08] K. Tombre and B. Lamiroy. Pattern Recognition Methods for Querying and Browsing Technical Documentation. In *Progress in Pattern Recognition, Image Analysis and Applications*, volume 5197 of *Lecture Notes on Computer Science*, pages 504–518. 2008. doi: 10.1007/978-3-540-85920-8\_62.
- [TT95] Ø.D. Trier and T. Taxt. Improvement of “Integrated Function Algorithm” for Binarization of Document Images. *Pattern Recognition Letters*, 16(3):277–283, 1995. doi: 10.1016/0167-8655(94)00101-8.

- [TT89] Y.T. Tsay and W.H. Tsai. Model-guided Attributed String Matching by Split-and-merge for Shape Recognition. *International Journal on Pattern Recognition and Artificial Intelligence*, 3(2):159–179, 1989. doi: 10.1142/S0218001489000140.
- [VD04] E. Valveny and P. Dosch. Symbol Recognition Contest: A Synthesis. In *Graphics Recognition, Recent Advances and Perspectives*, volume 3088 of *Lecture Notes on Computer Science*, pages 368–385. 2004. doi: 10.1007/b99011.
- [VD06] E. Valveny and P. Dosch. Report on the Second Symbol Recognition Contest. In *Graphics Recognition. Ten Years Review and Future Perspectives*, volume 3926 of *Lecture Notes on Computer Science*, pages 381–397. 2006. doi: 10.1007/11767978\_35.
- [VDF08] E. Valveny, P. Dosch, A. Fornés, and S. Escalera. Report on the Third Contest on Symbol Recognition. In *Graphics Recognition. Recent Advances and New Opportunities*, volume 5046 of *Lecture Notes on Computer Science*, pages 321–328. 2008. doi: 10.1007/978-3-540-88188-9\_30.
- [VDW07] E. Valveny, P. Dosch, A. Winstanley, Y. Zhou, S. Yang, L. Yan, L. Wenyin, D. Elliman, M. Delalandre, E. Trupin, S. Adam, and J.M. Ogier. A General Framework for the Evaluation of Symbol Recognition Methods. *International Journal on Document Analysis and Recognition*, 9(1):59–74, 2007. doi: 10.1007/s10032-006-0033-x.
- [vR79] C.J. van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann Newton, MA, USA, 1979.
- [VS94] A. Della Ventura and R. Schettini. Graphic Symbol Recognition Using a Signature Technique. In *Proceedings of the Twelfth International Conference on Pattern Recognition, ICPR94*, pages 533–535, 1994. doi: 10.1109/ICPR.1994.577011.
- [VRL04] P. Viola, J. Rinker, and M. Law. Automatic Fax Routing. In *Document Analysis Systems VI*, volume 3163 of *Lecture Notes on Computer Science*, pages 484–495. 2004. doi: 10.1007/b100557.
- [WF74] R.A. Wagner and M.J. Fischer. The String-to-string Correction Problem. *Journal of the Association for Computing Machinery*, 21(1):168–173, 1974. doi: 10.1145/321796.321811.
- [WSB98] R. Weber, H.J. Schek, and S Blott. A Quantitative Analysis and Performance Study for Similarity-search Methods in High-dimensional Spaces. In *Proceedings of the Twentyfourth International Conference on Very Large Data Bases, VLDB98*, pages 194–205, 1998.
- [WLC09] C.H. Wei, Y. Li, W.Y. Chau, and C.T. Li. Trademark Image Retrieval Using Synthetic Features for Describing Global Shape and Interior Structure. *Pattern Recognition*, 42(3):386–394, 2009. doi: 10.1016/j.patcog.2008.08.019.

- [WD97] L. Wenyin and D. Dori. A Protocol for Performance Evaluation of Line Detection Algorithms. *Machine Vision and Applications*, 9(5):240–250, 1997. doi: 10.1007/s001380050045.
- [WD98] L. Wenyin and D. Dori. Incremental Arc Segmentation Algorithm and its Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):424–431, 1998. doi: 10.1109/34.677280.
- [WZY07] L. Wenyin, W. Zhang, and L. Yan. An Interactive Example-driven Approach to Graphics Recognition in Engineering Drawings. *International Journal on Document Analysis and Recognition*, 9(1):13–29, 2007. doi: 10.1007/s10032-006-0025-x.
- [WJ96] D.A. White and R. Jain. Similarity Indexing with the SS-Tree. In *Proceedings of the Twelfth International Conference on Data Engineering, ICDE96*, pages 516–523, 1996. doi: 10.1109/ICDE.1996.492202.
- [WR83] J.M. White and G.D. Roher. Image Thresholding for Optical Character Recognition and Other Applications Requiring Character Image Extraction. *IBM Journal of Research and Development*, 27(4):400–411, 1983.
- [WPW95] E. Wiener, J.O. Pedersen, and A.S. Weigend. A Neural Network Approach to Topic Spotting. In *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval, SDAIR95*, pages 317–332, 1995.
- [WJ06] C. Wolf and J.M. Jolion. Object Count/Area Graphs for the Evaluation of Object Detection and Segmentation Algorithms. *International Journal on Document Analysis and Recognition*, 8(4):280–296, 2006. doi: 10.1007/s10032-006-0014-0.
- [Wol90] H.J. Wolfson. On Curve Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):483–489, 1990. doi: 10.1109/34.55108.
- [Yan05] S. Yang. Symbol Recognition via Statistical Integration of Pixel-Level Constraint Histograms: A New Descriptor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2):278–281, 2005. doi: 10.1109/TPAMI.2005.38.
- [YN07] M. Yoshida and H. Nakagawa. Web Document Parsing: A New Approach to Modeling Layout-language Relations. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 203–207, 2007. doi: 10.1109/ICDAR.2007.4378704.
- [ZR72] C.T. Zahn and R.Z. Roskies. Fourier Descriptors for Plane Closed Curves. *IEEE Transactions On Computer*, 21(3):269–281, 1972.
- [ZL02] D. Zhang and G. Lu. Shape-Based Image Retrieval Using Generic Fourier Descriptor. *Signal Processing*, 17:825–848, 2002. doi: 10.1016/S0923-5965(02)00084-X.

- [ZL04] D. Zhang and G. Lu. Review of Shape Representation and Description Techniques. *Pattern Recognition*, 37:1–19, 2004. doi: 10.1016/j.patcog.2003.07.008.
- [ZW07] W. Zhang and L. Wenyin. A New Vectorial Signature for Quick Symbol Indexing, Filtering and Recognition. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 536–540, 2007. doi: 10.1109/ICDAR.2007.4378767.
- [ZD07] G. Zhu and D. Doerman. Automatic Document Logo Detection. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR07*, pages 864–868, 2007. doi: 10.1109/ICDAR.2007.4377038.
- [Zuw06] D. Zuwala. *Reconnaissance de Symboles Sans Connaissance A Priori*. PhD thesis, Laboratoire Lorrain de Recherche en Informatique et ses Applications, LORIA, 2006.
- [ZT06] D. Zuwala and S. Tabbone. A Method for Symbol Spotting in Graphical Documents. In *Document Analysis Systems VII*, volume 3872 of *Lecture Notes on Computer Science*, pages 518–528. 2006. doi: 10.1007/11669487\_46.

---

**Final Acknowledgment**

This work has been partially supported by the CICYT projects TIN2006-15694-C02-02 and DPI2007-614452 (Ministerio Ciencia y Tecnología) and by the Spanish research programme Consolider Ingenio 2010: MIPRCV (CSD2007-00018).

---