Charles University in Prague
Faculty of Mathematics and Physics
Department of Software Engineering

DOCTORAL THESIS

Image Segmentation

Mgr. Stanislav Mikeš

Supervisor: Doc. Ing. Michal Haindl, DrSc.

Prague, January 2010

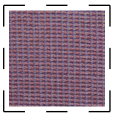
Abstract

Image segmentation is a fundamental part in low level computer vision processing. It has an essential influence on the subsequent higher level visual scene interpretation for a wide range of applications. Unsupervised image segmentation is an ill-defined problem and thus cannot be optimally solved in general.

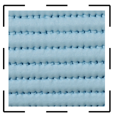
Several novel unsupervised multispectral image segmentation methods based on the underlaying random field texture models (GMRF, 2D/3D CAR) were developed. These segmenters use efficient data representations that allow an analytical solutions and thus the segmentation algorithm is much faster in comparison to methods based on MCMC. All segmenters were extensively compared with the alternative state-of-the-art segmenters with very good results. The MW3AR segmenter scored as one of the best available. The cluster validation problem was solved by a modified EM algorithm. Two multiple resolution segmenters were designed as a combination of a set of single segmenters. To tackle a realistic variable lighting in images, the illumination invariant features were derived and the illumination invariant segmenter was developed.

For the proper evaluation of segmentation results and ranking of algorithms, a unique web-based texture segmentation benchmark was proposed and implemented. It was used for comprehensive comparisons of results of developed algorithms with ten different state-of-the-art segmentation methods. Finally, the proposed methods were validated through use in various applications from a range of different fields.

In the medical imaging field, they were used for automatic segmentation of mammograms into regions of interest. Proposed solutions based on the random field model could also be used in automated inspection systems. Developed segmenters work on aerial images up to a size of 8000×8000 pixels, which are standard in the remote sensing field. The algorithm can also be used in areas related to digital cultural heritage. At last, an advantage of our methods is the need to tune just a few application dependent parameters.



Acknowledgements



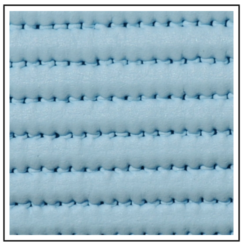
My deepest gratitude goes to my advisor Doc. Ing. Michal Haindl, DrSc. Without his guidance and support, this work would not have been possible. I would also like to thank fellows in our department and those who cooperated on the thesis results. Thanks to all the people that use the benchmark so as to make it meaningful.

I would sincerely like to thank my parents and my sister for being there at all times and for their support and encouragement with my research. Special thanks go to my friend Přemek who has endured my work on the thesis.

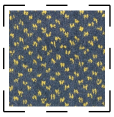
The research was supported by the Czech Science Foundation (GAČR) grant no. 102/08/0593, by the European Union Network of Excellence MUSCLE project (FP6–507752), and by the Ministry of Education, Youth and Sports of the Czech Republic (MŠMT ČR) grants no. 1M0572 DAR, 2C06019. The images of paint slices are provided by courtesy of ALMA laboratory (`http://www.alma-lab.cz`).

I hereby declare that I have worked out the thesis independently and that I have listed all the literature and publications used. I consent to the publication of the thesis.

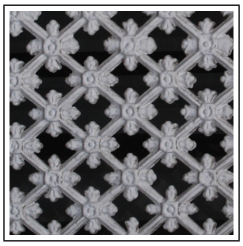
In Prague, January 12, 2010 Stanislav Mikeš



Contents

List of Figures



List of Tables


List of Acronyms

I. ... type I error (*benchmark criterion*)

II. ... type II error (*benchmark criterion*)

2D/3D ... two/three dimensional

a-w ... area weighted

BIB ... bibliography entry for BibTeX (reference management software)

BNSC ... British National Space Centre

BSDS ... Berkeley Segmentation Dataset and Benchmark

BTF ... bidirectional texture function

C ... commission error (*benchmark criterion*)

CA ... the weighted average class accuracy (*benchmark criterion*)

CAR ... causal autoregressive model

CBIR ... content based image retrieval

CC ... craniocaudal

CC ... precision, object accuracy, overall accuracy (*benchmark criterion*)

CI ... comparison index (*benchmark criterion*)

CO ... recall, the weighted average correct assignment (*benchmark criterion*)

CS ... correct detection (*benchmark criterion*)

dD ... Van Dongen metric (*benchmark criterion*)

DDSM ... Digital Database for Screening Mammography

dM ... Mirkin metric (*benchmark criterion*)

dVI ... variation of information (*benchmark criterion*)

EA ... mean class accuracy estimate (*benchmark criterion*)

EDISON ... Edge Detection and Image SegmentatiON

EGBIS ... Efficient Graph-Based Image Segmentation

EM ... expectation maximization algorithm

FIR ... finite impulse response

GCE ... global consistency error (*benchmark criterion*)

GM ... Gaussian mixture

GMRF ... Gaussian Markov random field

GPS ... Global Positioning System

GSRM ... General Statistical Region Merging

GT ... ground truth

K–L ... Karhunen–Loève expansion

KL, KLD ... Kullback Leibler divergence

kNN ... k-nearest neighbours algorithm

LBP ... local binary pattern

LCE ... local consistency error (*benchmark criterion*)

MAP ... maximum a posteriori probability

MC ... multiple segmenter

MCMC ... Markov chain Monte Carlo

ME ... missed error (*benchmark criterion*)

ML ... maximum likelihood

MLO ... mediolateral oblique

MRF ... Markov random field

MS ... mapping score (*benchmark criterion*)

MUSCLE ... Multimedia Understanding through Semantics, Computation and LEarning

NE ... noise error (*benchmark criterion*)

O ... omission error (*benchmark criterion*)

OS ... over-segmentation (*benchmark criterion*)

PCA ... principal component analysis

PNG ... Portable Network Graphics

RGB ... additive colour model with red, green, and blue as primary colours

RI-Spline ... real-imaginary spline

RM ... root mean square proportion estimation error (*benchmark criterion*)

ROI ... region of interest

SC ... single segmenter

SEM ... scanning electron microscopy

SNR ... signal to noise ratio

SWA ... Segmentation by Weighted Aggregation

TFR ... Texture Fragmentation and Reconstruction

US ... under-segmentation (*benchmark criterion*)

UV ... ultraviolet spectrum

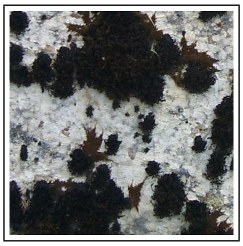
VisTex ... Vision Texture

VR ... virtual reality

VRML ... Virtual Reality Modelling Language

VS ... visible spectrum

WWW ... World Wide Web



List of Notations

$\boldsymbol{\theta}$ ... original centred data space

$\boldsymbol{I}$ ... finite lattice $N_1 \times N_2$

$\boldsymbol{r}$ ... multiindex (row, column index) $r = [r_1, r_2]$

$\boldsymbol{Y_r}$ ... random vector for the r index

$\tilde{\boldsymbol{Y}}_{\boldsymbol{r}}$ ... centered random vector

$\bar{\boldsymbol{Y}}_{\boldsymbol{r}}$ ... transformed vector (after K–L expansion)

$\boldsymbol{\Phi}$ ... second-order statistical moments matrix

$\boldsymbol{T}$ ... transformation matrix

$\boldsymbol{u_j}$ ... eigenvectors of the matrix Φ

$\boldsymbol{I_r}$ ... neighbourhood index set

$\boldsymbol{\eta}$ ... neighbourhood cardinality

$\boldsymbol{s}$ ... multiindex neighbourhood shift

$\boldsymbol{a_s}$ ... RF model parameters

$\boldsymbol{J_r}$ ... sublattice centred on the r index

$\boldsymbol{\delta}$ ... multiindex sublattice J_r half-size

$\boldsymbol{e_r}$ ... noise random variable

$\boldsymbol{\gamma_r}$ ... vector of the a_s model parameters

$\boldsymbol{X_r}$ ... corresponding vector of contextual neighbours Y_{r-s}

$\boldsymbol{A_s}$ ... CAR model parameter matrix

$\boldsymbol{I_r^c}$ ... causal neighbourhood index set

$\boldsymbol{\sigma^2}$ ... unknown noise variance

$\boldsymbol{d}$ ... number of spectral planes

$\boldsymbol{\tilde{\gamma}_r}$ ... local estimation of model parameters

$\boldsymbol{\bar{\gamma}_r}$ ... reduced parametric space

$\boldsymbol{\zeta_r}$ ... local spectral values vector

$\boldsymbol{\Theta_r}$ ... feature space vector

$\boldsymbol{h}$ ... length of feature space vector

$\mathbf{tr}\,\boldsymbol{A_s}$ ... trace of the parametric matrix A_s

$\boldsymbol{\xi_{s,j}}$ ... eigenvalue of the parametric matrix A_s

$\boldsymbol{\breve{\gamma}}$ ... illumination invariant parametric space

$\boldsymbol{\psi}$ ... vector of illumination invariants

$\boldsymbol{K}$ ... number of clusters in K-means, *or* components of Gaussian mixture

$\boldsymbol{\vartheta_r}$ ... the cluster index for Θ_r

$\boldsymbol{\mathcal{I}_i}$ ... cluster set i

$\boldsymbol{\nu_i}$ ... the i-th cluster center, *or* estimated mean value of the i-th component

$\boldsymbol{\Sigma_i}$ ... estimated covariance matrix of the i-th component

$\boldsymbol{\omega_i}$ ... the i-th component, or cluster

$\boldsymbol{p_i}$ ... the i-th component weight

$\boldsymbol{K_{ini}}$ ... initial number of components

$\boldsymbol{K_{opt}}$ ... optimal number of components

$\boldsymbol{\phi}$ ... component weight threshold

$\boldsymbol{\mathcal{L}_t}$ ... EM algorithm likelihood function

$\boldsymbol{w_s}$ ... fixed distance-based weights

$\boldsymbol{\pi_{r,i}}$ ... local component probability

$\boldsymbol{\pi_{thre}}$ ... component probability threshold

$\boldsymbol{M}$ ... number of resolution levels (number of segmenters in multi segmenters approach)

$\boldsymbol{\iota_i}$ ... the i-th sampling factor

$\downarrow^{\iota_m}$ ... downsampling by ι_m sampling factor

$\uparrow^{\iota_m}$ ... upsampling by ι_m sampling factor

$\Xi_i^{(m)}$ ... thematic map set for the m-th resolution and the i-th component

$\boldsymbol{\mathcal{M}}$ ... number of machine segmented regions

$\boldsymbol{\mathcal{N}}$ ... number of ground truth regions

$\boldsymbol{R_i}$ ... machine segmented regions

$\boldsymbol{\bar{R}_j}$ ... ground truth regions

$|\boldsymbol{R}|$ ... corresponding set cardinality

$\setminus$ ... set difference

$\cap$ ... intersection of sets

$\boldsymbol{k}$ ... regions overlap acceptance threshold

$\boldsymbol{n}$ ... number of pixels in the test set

$\boldsymbol{n_{i,j}}$ ... number of pixels interpreted as the i-th class but belonging to the j-th class

$\boldsymbol{n_{i,\bullet}}$ ... number of pixels interpreted as the i-th class

$\boldsymbol{n_{\bullet,i}}$ ... number of pixels belonging into the i-th class

$\hat{\boldsymbol{\imath}}$ ... either i for supervised test, or mapping of the i-th class ground truth into an interpretation segment for unsupervised test

$\boldsymbol{\varphi}$ ... F-measure curve parameter

$\dot{\boldsymbol{S}}, \ddot{\boldsymbol{S}}$ ... segmentations

$\boldsymbol{\varepsilon_r}$ ... non-symmetric local error measure at pixel r

$\boldsymbol{H}(\dot{\boldsymbol{S}})$ ... entropy

$\boldsymbol{I}(\dot{\boldsymbol{S}}, \ddot{\boldsymbol{S}})$ ... mutual information

$\boldsymbol{l}$ ... process history length of the adaptive threshold

$\boldsymbol{\epsilon_r}$ ... prediction error

$\mathtt{r}$ ... recall criterion

$\mathtt{p}$ ... precision criterion

$\mathtt{II}$ ... type II error criterion

$\boldsymbol{n_d}$ ... number of defect pixels

$\boldsymbol{n_i}$ ... number of pixels interpreted as defect pixels

$\boldsymbol{n_c}$ ... number of correctly interpreted defect pixels


Introduction

Chapter 1

1.1 What Is Image Segmentation?

Segmentation is a fundamental process which partitions a data space into meaningful salient regions. It is often used to partition an image into separate regions, which ideally correspond to different real-world objects. It is a critical step [146] towards content analysis and image understanding. Image segmentation essentially affects the overall performance of any automated image analysis system, thus its quality is of the utmost importance. Image regions, homogeneous with respect to some usually textural or colour measure, which result from a segmentation algorithm are analysed in subsequent interpretation steps. Colour and texture are the most important visual cues for segmentation. With the progress of digital image sensors, texture can be more precisely captured and thus texture becomes even more important. Consequently texture-based image segmentation has been an area of intense research activity in the past thirty years and many algorithms were published in consequence of all this effort, starting from simple thresholding methods up to the most sophisticated random field type methods. However many papers are published every year, image segmentation is still far from being solved. Segmentation can be categorized into supervised and much more difficult unsupervised categories.

1.1.1 Supervised Segmentation

Supervised segmentation benefits from prior knowledge of all classes of trainee sets in the classification task to be solved. It consists of two steps – learning and classification. In the learning stage it learns a classifier or set of classifiers on the local characteristics (features) of the training set (eg. texture patches). The training set is divided into class related subsets. During the classification phase the learned classifier is utilized to assign class labels to image pixels.

1.1.2 Unsupervised Segmentation

On the other hand, unsupervised methods which do not assume any prior knowledge of class related trainee sets, which can be learned to help the segmentation process, are obviously more challenging than the supervised ones due to the unknown division of the training set into classes. Additionally, it can be even harder without knowledge of the number of classes in the image to be segmented.

Cluster Validation

The determination of the number of classes actually present in an image is a serious problem. This problem, called the cluster validation problem, remains essentially unsolved. The difficulty of this problem lies, in part, in the inability to provide accurate sampling distributions for various classes and the lack of sufficient regularity conditions [33]. We proposed [61] the hierarchical cluster validation approach where clusters are merged or split as the segmentation algorithm progresses in attempting to solve the segmentation and validation problems simultaneously.

Segmentation Verification

Unsupervised image segmentation is therefore an ill-defined problem, and, without any semantic information from the upper level of computer vision, cannot be satisfactorily solved in its full generality as the human vision system can be. Although many methods are published every year, the problem is still far from being solved. This is, among other reasons, due to missing reliable performance comparisons between different techniques. Rather than advancing the most promising approaches, novel algorithms are often satisfied just being sufficiently different and tested only on a few selected examples. Therefore, a system for proper testing and robust learning of performance characteristics is needed. But it requires large test sets and objective ground truths.

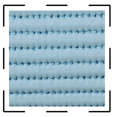
1.2 Thesis Contribution

Several unsupervised multispectral image segmentation methods based on the underlaying random field texture models (GMRF, 2D/3D CAR) are designed and developed. Descriptive models are used instead of usually applied discriminative models. Because descriptive models allow us to reconstruct segmented data space we can rightfully expect to obtain a better quality of the segmentation results than using the standard discriminative approach. The segmenters do not assume any prior knowledge of the number of regions in the image and try to solve cluster validation problem by a modified EM algorithm. Several multiple resolution segmenters are proposed and illumination invariant features are employed to tackle realistic lighting.

A web-based texture segmentation benchmark service is proposed and implemented. Its main goals are test data generating, segmentation results evaluation and algorithms ranking, comparison and development. It provides several benchmark data sets – monospectral, multispectral, or BTF data, and sets testing scale, rotation, and illumination invariance. It is intended for either unsupervised or supervised segmentation methods validation. All developed methods were extensively tested and verified on this benchmark which is rarely done for published alternative segmenters. Further, these methods were also validated on various applications. In the medical imaging field, an unsupervised segmentation method is used for automatic mammogram segmentation into regions of interest. The random field model is utilized for fast defect detection that could be used in automated inspection systems. In the remote sensing field it has to work on images reaching a size of 8000×8000 pixels. And finally, our methods can be used in areas related to digital cultural heritage as well.

1.3 Thesis Overview

The thesis outline is as follows: the next chapter overviews the state-of-the-art methods, followed by chapter 3, which covers a description of texture representation models, clustering methods, and multiple resolution approaches. Chapter 4 concerns segmentation validation and its main content is a developed web-based benchmark framework. The subsequent chapter contains experimental results and comparison of algorithms. In chapter 6 can be found various examples of applications of image segmentation (mammography, defect detection, remote sensing, and cultural heritage). The thesis is concluded in chapter 7 with conclusions and further development.



State of the Art

Chapter 2

Segmentation methods are based on some pixel or region similarity measures in relation to their local neighbourhood. They are usually categorized [117] as region-based, boundary-based, or as a hybrid of the two [13, 34]. Boundary-based methods search for the most dissimilar pixels which represent discontinuities in the image, while region-based methods on the contrary search for the most similar areas.

The similarity measures in texture segmentation methods use some textural spatial-spectral-temporal features such as Markov random field statistics (MRF) [54–56], Gabor filter features [73, 143], local binary pattern (LBP) [106, 113] and many other features, for example [134]: autocorrelation features (ACF), co-occurrence matrix (CM), edge frequency (EF), Law's masks (LM), run length (RL), binary stack method (BSM), texture operators (TO), and texture spectrum (TS).

2.1 Colour Models

Colour is perceived by humans as a combination of tristimuli R (red), G (green), and B (blue) which are usually called primary colours. The RGB colour model is not optimal for all processing tasks and therefore other colour models were developed. Other types of colour representations can be derived from the RGB representation. New colour spaces YIQ, YUV, or $I_1I_2I_3$ can obtained from the RGB representation by linear transformation.

YIQ resp. YUV are used to encode colour information in TV signals for American resp. European systems. The Y component is a measure of the luminance of the colour, and the IQ, resp. UV components jointly describe the hue and the saturation of the colour. These spaces can partly get rid of the correlation between components in RGB representation. Moreover the linear transformation needs less computation time than nonlinear ones, which makes these systems more preferable to nonlinear systems.

Ohta et al. [104] performed experiments of region segmentation to derive a set of effective colour features. They used recursive region splitting and calculated new features by Karhunen-Loeve transformation and they found a set of colour features as $I_1 = (R+G+B)/3$, $I_2 = (R-B)/2$, and $I_3 = (2G-R-B)/4$. They compared $I_1I_2I_3$ with seven other standard colour spaces, and claimed that $I_1I_2I_3$ is more effective in terms of the quality of segmentation and the computational complexity.

For colour image segmentation we need to make colours independent of lightning intensity. The normalized RGB space is defined as $r = R/(R+G+B)$, $g = G/(R+G+B)$, and $b = B/(R+G+B)$. Since $r+g+b = 1$, we can use only two components, determined by the percentage of the RGB components. As the third component may be used the luminance $Y = c_1R + c_2G + c_3R$. Normalization reduces the sensitivity of the illumination changes, but it is noisy under low intensities.

The HSI (hue-saturation-intensity) system is a commonly used system in image processing, which is more convenient to human perception. Colour information is separated into hue and saturation, while intensity describes the brightness of the image. Hue represents dominant basic colour, and saturation the purity of the colour. We can use grey-level segmentation algorithms on the hue component. It is efficient when the image contains non-uniform lighting such as highlights, shading and shadows because hue is invariant to illumination. But hue has singularity near low saturation values. Also, if the intensity is close to white or black, hue and saturation are not so important in distinguishing colours. The $Munsell$ colour system is one of the early methods to describe colours. It is similar to the HSI system, and it use hue, value and chroma components.

The CIE colour system was developed to represent perceptual uniformity. The ability to express human perceptions of colour difference by Euclidean distances is important to colour segmentation. CIE colour spaces expressed colour and intensity information more independently than RGB primary colours. It has three primaries denoted X, Y, and Z, from which can be created several CIE spaces. Typical examples are CIE $(L^*a^*b^*)$ and CIE $(L^*u^*v^*)$.

2.2 Texture Segmentation Methods

Texture segmentation methods can be categorized using various criteria [117], e.g. region / boundary based, MAP / clustering methods, graph theoretic methods, etc.

The clustering approach resulted in agglomerative and divisive algorithms which were modified for image segmentation as region-based merge and split algorithms. KMG and KMC [36] use a common approach of clustering on low-level features using the K-means algorithm [8], forming connected regions, and merging regions until a minimum region size is obtained. KMG uses grey-level features, while KMC uses intensity and colour features in the opponent colour space.

The segmenter [80] combines a bag-of-words recognition component with spatial regularization based on a random field and a Dirichlet process mixture. Bag-of-words models predict the presence of an object within an image; random fields take into account the spatial layout of images and provide local spatial regularization. Larger scale structures are combined with a Dirichlet process mixture.

Different published methods are difficult to compare because of lack of a comprehensive analysis together with accessible experimental data. However, available results indicate that the ill-defined texture segmentation problem is still far from being satisfactorily solved.

2.3 Stochastic Model Based Approach

In stochastic model-based image segmentations, image classes are modelled as random fields and the segmentation problem is posed as a statistical optimization problem [28, 74, 98, 133, 138]. Spatial interaction models and especially Markov random fields-based models are increasingly popular for texture representation [47, 72, 88, 117], etc. These segmenters often employ a doubly-stochastic model to describe both the distribution of regions and the intensity field within a region. Several researchers dealt with the difficult problem of unsupervised segmentation using these models. See for example [3, 48, 87, 109] or [54, 56, 58].

2.4 Region Growing

The basic approach of a region growing algorithm [26, 108] is to start from seed region (mostly one or few pixels) that is assumed to be inside the object to be segmented. The neighbouring pixels to every seed region are evaluated to decide if they should be considered part of the object or not. If they are recognized as similar, they are added to the region and the process continues as long as any undecided pixels remain. Region growing algorithms vary depending on the similarity criteria, seed region selection, the type connectivity used to determine neighbours, and the strategy used to visit neighbouring pixels.

A region merging segmentation technique [112] starts from an oversegmented image using various segmentations of the same image using the Seeded Region Growing algorithm [90] and the merging process is based on the repulsion force between neighbouring pixels criterion.

2.4.1 Blobworld

The Blobworld [12] scheme aims to automatically segment images into a small set of regions (*blobs*) which are coherent in colour and texture [5]. This is achieved by clustering pixels in a joint colour-texture-position eight-dimensional feature space using the EM algorithm. The feature vector is represented by a Gaussian mixture model.

2.4.2 JSEG

The JSEG method [27] is for unsupervised segmentation of colour-texture regions in images and video. This method consists of two independent steps: colour quantization and spatial segmentation. In the first step, colours in the image are quantized to several representative classes that can be used to differentiate regions in the image. The image pixels are then replaced by their corresponding colour class labels, thus forming a class-map of the image. The focus is on spatial segmentation, where a criterion for *good* segmentation using the class-map is proposed. Applying the criterion to local windows in the class–map results in the J–image, in which high and low values correspond to possible boundaries and interiors of colour-texture regions. A region growing method is then used to segment the image based on the multiscale J–images.

2.4.3 TFR

TFR (Texture Fragmentation and Reconstruction) method [124] is an unsupervised colour texture segmentation algorithm which processes independently the spectral and spatial information. The algorithm is composed of two parts. The former provides an over-segmentation of the image, such that basic components for each of the textures which are present are extracted. The latter is a region growing algorithm which reduces drastically the number of regions, and provides a region-hierarchical texture clustering. The over-segmentation is achieved by means of a colour-based clustering (CBC) followed by a spatial-based clustering (SBC). The SBC, as well as the subsequent growing algorithm, make use of a characterization of the regions based on shape and context.

2.4.4 TFR/KLD

TFR/KLD method [125] is an improved version of the TFR algorithm where the region gain has been changed by introducing a Kullback-Leibler Divergence (KLD) term modelling the region similarity in terms of spatial location. The image to be segmented is first discretized and then a hierarchical finite-state region-based model is automatically coupled with the data by means of a sequential optimization scheme, namely TFR algorithm. Both intra- and inter-texture interactions are modelled, by means of an underlying hierarchical finite-state model, and eventually the segmentation task is addressed in a completely unsupervised manner. The output is then a nested segmentation, so that the user may decide the scale at which the segmentation has to be provided. TFR is composed of two steps: the former focuses on the estimation of the states at the finest level of the hierarchy, and is associated with an image fragmentation, or over-segmentation; the latter deals with the reconstruction of the hierarchy representing the textural interaction at different scales.

2.5 Split and Merge

Split and merge techniques [103, 108] start with recursive splitting image into smaller regions until they do not satisfy some homogeneity criterion. The second merging step merges adjacent regions with similar attributes.

2.5.1 GSRM

GSRM (General Statistical Region Merging) method [11] is a statistical approach to region merging where regions are modelled as arbitrary discrete distributions, directly estimated from the pixel values. Under this framework, two region merging criteria are obtained from two different perspectives, leading to information theory statistical measures: the Kullback-Leibler divergence and the Bhattacharyya coefficient. The methods are size-dependent, which assures the size consistency of the partitions but reduces their size resolution. Thus, a size-independent extension of the previous methods, combined with a modified merging order, is also proposed. Additionally, an automatic criterion to select the most statistically significant partitions from the whole merging sequence is available.

2.6 Watershed

Watershed segmentation [82, 100, 111, 127] classifies pixels into regions using gradient descent on image features and analysis of weak points along region boundaries. The image feature space is treated, using a suitable mapping, as a topological surface where higher values indicate the presence of boundaries in the original image data. It uses an analogy with water gradually filling low lying landscape basins. The size of the basins grow with increasing amounts of water until they spill into one another. Small basins (regions) gradually merge together into larger basins. Regions are formed by using local geometric structure to associate the image domain features with local extremes measurement. Watershed techniques produce a hierarchy of segmentations, thus the resulting segmentation has to be selected using either some prior knowledge or manually. These methods are well suited for different measurements fusion and they are less sensitive to user defined thresholds.

2.7 Level Sets

The paradigm of the level set [10, 119] is that it is a numerical method for tracking the evolution of contours and surfaces. Instead of manipulating the contour directly, the contour is embedded as the zero level set of a higher dimensional function called the level-set function. The level-set function is evolved under the control of a differential equation using some image-based features. At any time, the evolving contour can be obtained by extracting the zero level-set from the output. Level sets allow to model arbitrarily complex shapes and topological changes (merging and splitting) are handled implicitly.

2.8 Active Contours

Similar to the level set segmentation are active contour based methods [2, 34, 136], which belong to the variational image segmentation group. This method (active contour model (ACM) or snakes) is based on minimization of the energy functional. This energy is usually defined as a linear combination of the internal and the external energy terms. The internal term represents the internal energy of the contour caused by stretching and bending. The external term is based on local image properties such as edge strength or some region-based statistics.

2.9 Mean Shift Based

A robust clustering technique [22, 23, 46] which does not require prior knowledge of the number of clusters, and does not constrain the shape of the clusters, is mean shift based clustering. This is an iterative technique, as K-Means is, but instead of the means, it estimates the modes of the multivariate distribution underlying the feature space. The number of clusters is obtained automatically by finding the centers of the densest regions in the space (the modes).

2.9.1 EDISON

EDISON (Edge Detection and Image SegmentatiON) segmenter [21] is a mean shift based image segmentation with embedded edge information. To improve the trade-off between the sensitivity of homogeneous region delineation and the oversegmentation of the image, it incorporates an edge magnitude/confidence map into a colour image segmenter based on the mean shift procedure. The method can recover regions with weak but sharp boundaries and thus can provide a more accurate input for high level interpretation modules. Its first filtering step uses the mean shift segmenter in the combined colour $L^*u^*v^*$ and coordinate feature space. The mean shift weights are derived from the edge confidence measure. The second fusion step recursively fuses the basins of attraction of the modes.

2.10 Graph-Theoretic Segmentation

These methods [4, 9, 35, 38, 43, 45, 132] use graph representation for image pixels or regions where usually small neighbourhood elements are mutually connected with weighted graph edges. These weights indicate pairwise element similarities. The segmentation is based on finding groups of nodes that are strongly connected to each other but weakly with the remaining nodes in the graph.

2.10.1 EGBIS

The EGBIS (Efficient Graph-Based Image Segmentation) segmenter [39] is based on a predicate for measuring the evidence for a boundary between two regions. It formulates segmentation as a graph cutting problem and uses dynamic programming to form regions which are guaranteed to be neither too coarse nor fine with respect to a colour edge strength measure within and between regions, then merges regions to a minimum region size. The segmentation algorithm makes greedy decisions but it produces segmentations that satisfy global properties. The algorithm runs in time nearly linear in the number of graph edges and is also fast in practice. An important characteristic of the method is its ability to preserve detail in low-variability image regions while ignoring detail in high-variability regions.

2.10.2 SWA

SWA (Segmentation by Weighted Aggregation) method [45, 130] is a bottom-up weighted aggregation framework that combines structural characteristics of texture elements with filter responses. Each pixel in an image corresponds to a node in a graph, coupled to each of its four neighbours according to their similarity in luminance level. The goal is to cut this graph into pieces. A salient segment in the image is one for which the similarity across its border is small, whereas the similarity within the segment is large. Texture elements are identified at multiple scales. It constructs a graph in which every pixel is a node and neighbouring pixels are connected by an edge. A weight $w_{i,j} > 0$ is associated with each edge reflecting the contrast in the corresponding location in the image. A multiscale procedure is used to find optimal partitions of the graph. The SWA method requires a manual selection of the best segmentation result from the hierarchy of possible segmentation results.

2.11 Fusion of Sub-Segmentations

The underlying idea of these methods [30–32, 112] is to merge a set of different segmentations of the input image obtained previously by standard techniques. The segmentations can be produced using different methods or even the same method with different initial conditions.

2.11.1 TEX-ROI-SEG

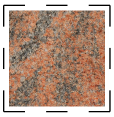
The TEX-ROI-SEG segmenter [31] is a texture extension of colour segmentation algorithm ROI-SEG. It uses covariance matrices of low level features for texture description. These features are efficiently calculated using integral images. Furthermore, a multi-scale extension allows to provide accurate texture segmentation results. ROI-SEG [30] is a unsupervised colour segmentation scheme, which is based on the main idea of combining a set of different sub-segmentation results. It uses an efficient algorithm to compute subsegmentations by an integral image approach for calculating Bhattacharyya distances and a modified version of the Maximally Stable Extremal Region (MSER) detector. The sub-segmentation algorithm gets a region-of-interest (ROI) as input and detects connected regions having similar colour appearance as the ROI. Further it introduces a method to identify ROIs representing the predominant colour and texture regions of an image. Passing each of the identified ROIs to the sub-segmentation algorithm provides a set of different segmentations, which are then combined by analyzing a local quality criterion. The entire approach is fully unsupervised and does not need a priori information about the image scene.

2.12 Feature Based Approach

As is mentioned in the beginning of this chapter, feature-based texture segmentation methods use some textural spatial-spectral-temporal features. One of frequently used features for texture segmentation are Gabor filters.

2.12.1 HGS

The HGS unsupervised segmenter [68] is based on the integration of the Gabor filters with the measurement of colour. Single versions of the method differ in their photometric invariance power (HGS-E no invariance, HGS-W low, HGS-C full invariance). The spatial frequency is measured by sampling the incoming image with a shifted Gaussian in the spatial frequency domain, and the colour is measured by sampling the signal with Gaussian in wavelength domain. The method implies that the colour–texture is measured in the wavelength-Fourier domain. The measurement filter in this domain boils down to a 3D Gaussian, representing a Gabor–Gaussian in the spatial-colour domain.


Texture Segmentation

Chapter 3

Developed unsupervised texture segmentation methods are described in this chapter. They are based on the underlaying random field models for texture representation and subsequent cluster analysis in the parametric feature space. The first section is concerned with the GMRF, 2D and 3D CAR models, respectively, and with the illumination invariants derived from the 3D CAR model as well. The next section deals with cluster analysis and provides details on the K-Means and the EM algorithms. The rest of the chapter is dedicated to the explanation of hierarchical approaches.

3.1 Texture Representation

The adequate representation of general static Lambertian multispectral textures requires three dimensional models. Although full 3D models allow unrestricted spatial-spectral correlation description its main drawback is a large amount of parameters to be estimated and in the case of Markov random field based models (MRF) also the necessity to estimate all these parameters simultaneously. Alternatively, it is possible to factorize the 3D static texture space into several (equal to the number d of spectral bands) 2D subspaces. A combination of several simpler 2D data models with less parameters per model allows more compact texture representation and faster estimation algorithms.

Natural measured texture data space can be decorrelated only approximately thus the independent spectral component representation suffers with some loss of image information. However, because the segmentation is a less demanding application than the texture synthesis, it is sufficient if such a representation maintains the discriminative power of the full model even if its visual modelling strength is slightly compromised.

The Gaussian Markov random field model (GMRF) is described in section 3.1.2, while spectral factorization is explained in section 3.1.1. The alternative to a non-causal GMRF model can be the simultaneous causal autoregressive random field model (CAR). It can be used in the form of the 2D model (see 3.1.3), using spectral decorrelation similarly as the GMRF model, or in the full 3D form (see 3.1.4) employing unrestricted information between spectral planes.

Realistic remote sensing, outdoor, security, and many others applications of these segmenters often have to deal with variable illumination of the segmented scene. Therefore it is important to have illumination invariant texture representation (see 3.1.5).

3.1.1 Spectral Factorization

Spectral factorization using the Karhunen-Loeve expansion transforms the original centred data space θ defined on the rectangular $N_1 \times N_2$ finite lattice I into a new data space with K-L coordinate axes $\bar{Y}$. These new basis vectors are the eigenvectors of the second-order statistical moments matrix

$$\Phi = E\{\tilde{Y}_r \tilde{Y}_r^T\} \tag{3.1}$$

where the multiindex r has two components $r = [r_1, r_2]$, the first component is row and the second one column index, respectively. The projection of the centred random vector $\tilde{Y}_r$ onto the K-L coordinate system uses the transformation matrix

$$T = [u_1^T, u_2^T, \dots u_d^T]^T \tag{3.2}$$

which has single rows u_j that are eigenvectors of the matrix Φ.

$$\bar{Y}_r = T\tilde{Y}_r \tag{3.3}$$

Components of the transformed vector $\bar{Y}_r$ (3.3) are mutually uncorrelated. If we assume further on Gaussian vectors $\bar{Y}_r$ then they are also independent, i.e.,

$$p(\bar{Y}_r) = \prod_{i=1}^{d} p(\bar{Y}_{r,i}) \tag{3.4}$$

and single monospectral random fields can be modelled independently.

3.1.2 GMRF Model

We assume that single monospectral texture factors $(Y_r = \bar{Y}_{r,i})$ can be modelled using a Gaussian Markov random field model (GMRF). This model is obtained if the local conditional density of the MRF model (3.5) is Gaussian:

$$p(Y_r \,|\, Y_{r-s}\ \forall s \in I_r) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{2}\,\sigma^{-2}\,(Y_r - \tilde{\mu}_r)^2\}\ , \tag{3.5}$$

where the mean value is

$$\begin{aligned} E\{Y_r \,|\, Y_{r-s}\ \forall s \in I_r\} &= \tilde{\mu}_r \\ &= \mu_r + \sum_{s \in I_r} a_s (Y_{r-s} - \mu_{r-s}) \end{aligned} \tag{3.6}$$

and $\sigma, a_s\ \forall s \in I_r$ are unknown parameters.

The 2D GMRF model can be expressed as a stationary non-causal correlated noise-driven 2D autoregressive process:

$$Y_r = \sum_{s \in I_r} a_s Y_{r-s} + e_r \tag{3.7}$$

where the noise e_r is random variable with zero mean $E\{e_r\} = 0$.

The e_r noise variables are mutually correlated

$$R_e = E\{e_r e_{r-s}\} = \begin{cases} \sigma^2 & \text{if } s = (0,0), \\ -\sigma^2 a_s & \text{if } s \in I_r, \\ 0 & \text{otherwise.} \end{cases} \tag{3.8}$$

Correlation functions have the symmetry property

$$E\{e_r e_{r+s}\} = E\{e_r e_{r-s}\} \tag{3.9}$$

hence the neighbourhood support set I_r and its associated coefficients have to be symmetric, i.e.,

$$s \in I_r \;\Rightarrow\; -s \in I_r \;\&\; a_s = a_{-s} \;. \tag{3.10}$$

The selection of an appropriate GMRF model support is important to obtain good results in modelling of a given random field. If the contextual neighbourhood is too small it can not capture all details of the random field. Inclusion of the unnecessary neighbours on the other hand adds to the computational burden and can potentially degrade the performance of the model as an additional source of noise. We use the hierarchical neighbourhood system I_r, e.g., the first-order neighbourhood is

$$I_r = \{(0,-1),(0,+1),(-1,0),(+1,0)\}\,,\ \text{etc.} \tag{3.11}$$

An optimal neighbourhood is detected using the correlation method [51] favouring neighbours locations corresponding to large correlations over those with small correlations.

The parameter estimation of a MRF model is complicated by the difficulty associated with computing the normalization constant. Fortunately, the GMRF model is an exception where the normalization constant is easily obtainable. However, either the Bayesian or ML estimate requires the iterative minimization of a nonlinear function. Therefore we use the pseudo-likelihood estimator which is computationally simple although not efficient.

The pseudo-likelihood estimate for a_s parameters evaluated for a sublattice

$$J_r \subset I \quad \text{and} \quad J_r = \{t : |r_1 - t_1| \leq \delta_1 \;\wedge\; |r_2 - t_2| \leq \delta_2\} \tag{3.12}$$

centred on the r index.

The pseudo-likelihood estimate for a_s parameters has the form

$$\gamma_r = [\, a_s : \forall s \in I_r \,] = \left[\sum_{\forall t \in J_r} X_t^T X_t\right]^{-1} \sum_{\forall t \in J_r} X_t^T Y_t \;, \tag{3.13}$$

where

$$X_t = [\, Y_{t-s} : \forall s \in I_t \,] \tag{3.14}$$

and

$$\sigma_r^2 = \frac{1}{|J_r|} \sum_{\forall t \in J_r} \left(Y_{t,i} - \gamma_{r,i}\, X_{t,i}^T \right)^2 \ . \tag{3.15}$$

3.1.3 CAR2D Model

We assume that single monospectral texture factors $(Y_r = \bar{Y}_{r,i})$ can be locally modelled using a 2D simultaneous causal autoregressive random field model (CAR). This model can be expressed as a stationary causal uncorrelated noise driven 2D autoregressive process [64]:

$$Y_r = \gamma X_r + e_r \ \ , \tag{3.16}$$

where $\gamma = [A_1, \ldots, A_\eta]$ is the $1 \times \eta$ parameter matrix, e_r is a white Gaussian noise with zero mean and a constant but unknown variance σ^2, X_r is a corresponding vector of the contextual neighbours Y_{r-s} and $r, r-1, \ldots$ is a chosen direction of movement on the image index lattice I. Further $\eta = card(I_r^c)$ where I_r^c is a causal neighbourhood index set (e.g. $I_r^c = \{(0,+1),(+1,0)\}$).

This texture model uses the 2D simultaneous causal autoregressive random field model and thus it requires the spectral decorrelation (sect. 3.1.1). If we stack single decorrelated mono spectral pixel components into $d \times 1$ vectors Y_r, the model can be formalized using the same equations as the CAR3D model, i.e. (3.19)–(3.29). The CAR2D models differ in having diagonal parameter matrices A_s (3.18) and a diagonal white noise covariance matrix.

3.1.4 CAR3D Model

If we do not want to lose spectral information due to the spectral decorrelation step, we have to use three dimensional models for adequate representation. One of few 3D models which does not require any approximation and can be treated analytically is the 3D simultaneous causal autoregressive random field model (CAR) used in the AR3D+EM segmenter [56].

Static smooth multispectral (or pseudo-colour mammogram) textures require three dimensional models for adequate representation. We assume that single multispectral textures can be locally modelled using a 3D simultaneous causal autoregressive random field model. This model can be expressed as a stationary causal uncorrelated noise driven 3D autoregressive process [64]:

$$Y_r = \sum_{s \in I_r^c} A_s Y_{r-s} + e_r \, , \tag{3.17}$$

where e_r is a white Gaussian noise vector with zero mean and a constant but unknown covariance matrix Σ and r, s are multiindices. The noise vector is uncorrelated with data from a causal neighbourhood index set I_r^c,

$$A_{s_1,s_2} = \begin{pmatrix} a_{1,1}^{s_1,s_2} & \dots & a_{1,d}^{s_1,s_2} \\ \vdots & \ddots & \vdots \\ a_{d,1}^{s_1,s_2} & \dots & a_{d,d}^{s_1,s_2} \end{pmatrix} \tag{3.18}$$

are $d \times d$ parameter matrices where d is the number of spectral bands. $r, r-1, \dots$ is a chosen direction of movement on the image lattice I (e.g. scanning lines rightward and top to down). This model can be analytically estimated using numerically robust recursive statistics hence it is exceptionally well suited for possible real-time texture defect detection applications. The model adaptation is introduced using the standard exponential forgetting factor technique.

Parameter Learning

The model can be written in the matrix form

$$Y_r = \gamma X_r + e_r \, , \tag{3.19}$$

where $\gamma = [\, A_1, \dots, A_\eta]$ is a $d \times d\eta$ parameter matrix, $\eta = card(I_r^c)$ and X_r is a corresponding vector of contextual neighbours Y_{r-s}. To evaluate the conditional mean values $E\{Y_r | Y^{(r-1)}\}$, the one-step-ahead prediction posterior density $p(Y_r \,|\, Y^{(r-1)})$

is needed. If we assume the normal-gamma parameter prior for parameters in (3.17) (alternatively we can assume the Jeffrey's parameter prior) this posterior density has the form of Student's probability density

$$p(Y_r|Y^{(r-1)}) = \frac{\Gamma(\frac{\beta(r)-d\eta+3}{2})\ \pi^{-\frac{1}{2}}\ \lambda_{(r-1)}^{-\frac{1}{2}}}{\Gamma(\frac{\beta(r)-d\eta+2}{2})\ \ (1+X_r^T V_{xx(r-1)}^{-1} X_r)^{\frac{1}{2}}}$$

$$\left(1+\frac{(Y_r-\hat{\gamma}_{r-1}X_r)^T\lambda_{(r-1)}^{-1}(Y_r-\hat{\gamma}_{r-1}X_r)}{1+X_r^T V_{xx(r-1)}^{-1}X_r}\right)^{-\frac{\beta(r)-d\eta+3}{2}}, \tag{3.20}$$

with $\beta(r)-d\eta+2$ degrees of freedom, where the following notation is used:

$$\beta(r) = \beta(0)+r-1\,, \tag{3.21}$$

$$\hat{\gamma}_{r-1}^T = V_{xx(r-1)}^{-1}\,V_{xy(r-1)}\,, \tag{3.22}$$

$$V_{r-1} = \begin{pmatrix} \tilde{V}_{yy(r-1)} & \tilde{V}_{xy(r-1)}^T \\ \tilde{V}_{xy(r-1)} & \tilde{V}_{xx(r-1)} \end{pmatrix} + I\,, \tag{3.23}$$

$$\tilde{V}_{uw(r-1)} = \sum_{k=1}^{r-1} U_k W_k^T\,, \tag{3.24}$$

$$\lambda_{(r)} = V_{y(r)} - V_{xy(r)}^T\,V_{x(r)}^{-1}\,V_{xy(r)}\,. \tag{3.25}$$

where $\beta(0) > 1$ and U, W denote either Y or X vector, respectively. If $\beta(r-1) > \eta$ then the conditional mean value is

$$E\{Y_r|Y^{(r-1)}\} = \hat{\gamma}_{r-1}X_r \tag{3.26}$$

and it can be efficiently computed using the following recursion

$$\hat{\gamma}_r^T = \hat{\gamma}_{r-1}^T + \frac{V_{x(r-1)}^{-1}\,X_r(Y_r-\hat{\gamma}_{r-1}X_r)^T}{1+X_r^T V_{x(r-1)}^{-1}X_r}\,, \tag{3.27}$$

where $V_{x(r-1)} = \sum_{k=1}^{r-1} X_k X_k^T + V_{x(0)}$.

The selection of the appropriate model support (I_r^c) is important to obtain good texture representation but less important for segmentation. The optimal Bayesian decision rule for minimizing the average probability of decision error chooses the maximum posterior probability model, i.e., a model M_i corresponding to $\max_j \{p(M_j|Y^{(r-1)})\}$. If we assume uniform prior for all tested support sets (models) the solution for the optimal model support (I_r^c) can be found analytically. The most probable model given past data is the model M_i $(I_{r,i}^c)$ for which $i = \arg\max_j\{D_j\}$.

$$\begin{aligned} D_j &= -\frac{1}{2}\ln|V_{x(r-1)}| - \frac{\alpha(r)}{2}\ln|\lambda_{(r-1)}| + \\ &+ \frac{d\eta}{2}\ln\pi\left[\ln\Gamma(\frac{\alpha(r)}{2}) - \ln\Gamma(\frac{\beta(0)-d\eta+2}{2})\right], \end{aligned} \tag{3.28}$$

where $\alpha(r) = \beta(r) - d\eta + 2$.

Final Parametric Space

Local texture for each pixel is represented by four parametric vectors. Each vector contains local estimations of the CAR model parameters. These models have identical contextual neighbourhood I_r^c but they differ in their major movement direction (top-down, bottom-up, rightward, leftward), i.e.,

$$\tilde{\gamma}_r^T = \{\hat{\gamma}_r^t, \hat{\gamma}_r^b, \hat{\gamma}_r^r, \hat{\gamma}_r^l\}^T. \tag{3.29}$$

The parametric space $\tilde{\gamma}$ is subsequently smoothed out, rearranged into a vector and its dimensionality is reduced using the Karhunen-Loeve feature extraction $(\bar{\gamma})$. Finally we add the average local spectral values ζ_r to the resulting feature vector

$$\Theta_r = [\bar{\gamma}_r, \zeta_r]^T. \tag{3.30}$$

3.1.5 CAR3D Illumination Invariants

In previous models we assume constant illumination and viewing angles for all scene textures, or alternatively that the Lambert law holds for all scene surfaces. If this assumption cannot be assumed, then all textures have to be treated either as Bidirectional Texture Functions (BTFs) or some illumination invariant features [65, 141] have to be used.

We assume that the two images $\dot{Y}, \ddot{Y}$ acquired with different illumination can be linearly transformed to each other: $\dot{Y}_r = B\,\ddot{Y}_r$, where $\dot{Y}_r, \ddot{Y}_r$ are multispectral pixel values at position r and B is a $d \times d$ transformation matrix. This linear relation holds for changes in brightness and illumination spectrum with Lambertian surface reflectance, or for a model which includes specular reflectance component. Using the above assumption we can derive [141] the illumination invariance of

1. trace: $\operatorname{tr} A_s, \quad s = 1, \ldots, \eta$,
2. eigenvalues: $\xi_{s,j}$ of $A_s, \quad s = 1, \ldots, \eta, \quad j = 1, \ldots, d$,

where A_s are the parameter matrices (3.18).

The parametric space $\tilde{\gamma}$ (3.29) is subsequently transformed into the illumination invariant parametric space $\breve{\gamma}$:

$$\begin{aligned} \breve{\gamma}_r^T &= \left[{}^t\psi,\, {}^b\psi,\, {}^r\psi,\, {}^l\psi\right]^T , \\ {}^\alpha\psi &= \left[{}^\alpha\xi_{1,1}, \ldots, {}^\alpha\xi_{\eta,d},\, \operatorname{tr}\, {}^\alpha A_1, \ldots, \operatorname{tr}\, {}^\alpha A_\eta\right] , \qquad \alpha \in \{t, b, r, l\} . \end{aligned} \tag{3.31}$$

Finally we add the local a_r, b_r components of the Lab colour coordinates to the resulting feature vector (Θ_r).

3.2 Cluster Analysis

Clustering or cluster analysis is an important method in unsupervised image segmentation. Its objective is the grouping of image pixels into the subsets in order to be similar in some sense. Data clustering algorithms can be divided into several types. Hierarchical clustering methods create a hierarchy tree which can be done either bottom–up (*agglomerative*) or top–down (*divisive*). Another type of clustering algorithms are graph-theoretic methods including spectral clustering which deals with finding a cut in a graph [40, 86, 92, 132]. The K–means and the EM algorithm which are described in this section belong to a partitional clustering type. The EM algorithm and its modifications are quite popular in clustering problems [69, 110, 118, 139, 144].

One of the major issues of data clustering is the question of cluster validation, i.e. how many clusters should be found by cluster analysis. Some methods require this number to be given explicitly while others try to find it automatically [18, 24, 99, 137].

3.2.1 K-Means

The input of the algorithm is a set of feature vectors $\Theta_1, \Theta_2, \dots \Theta_{N_1 \times N_2}$ and the number K is the desired number of clusters into which input vectors are divided. The centers of the clusters $\nu_j,\ i = 1, \dots K$ are initialized at the beginning. It can be done either by random choice of the input vectors or using some heuristic, for instance a priori information about distribution of input vectors. After initialization, two steps are repeated – assigning vectors to clusters and the recomputing of the clusters' centers. The algorithm is finished when no more vectors are reassigned to another cluster or the maximum iteration number threshold is reached.

The vectors are assigned to the clusters according to the L_2 distance between the vector and the center of the cluster. The vector is assigned to the cluster whose center is the nearest, i.e.

$$\vartheta_r = \underset{i=1,\dots K}{argmin} \|\Theta_r - \nu_i\|, \quad r = 1, \dots N_1 \times N_2 \,. \tag{3.32}$$

The centers of the clusters are computed as the arithmetic mean of the vectors belonging to the cluster, i.e.

$$\nu_i = \frac{1}{|\mathcal{I}_i|} \sum_{r \in \mathcal{I}_i} \Theta_r, \quad \mathcal{I}_i = \{r : \vartheta_r = i\}, \quad i = 1, \dots K \,. \tag{3.33}$$

3.2.2 Mixture Model Based – EM algorithm

Multispectral or monospectral texture segmentation is done by clustering in the parameter space $\Theta\ (\in \mathcal{R}^h)$ defined on the lattice I where

$$\Theta_r = [\bar{\gamma}_r, \zeta_r]^T \tag{3.34}$$

is the CAR3D parameter vector (3.30) computed for the lattice location r, or

$$\Theta_r = [\gamma_{r,1}, \zeta_{r,1}, \gamma_{r,2}, \zeta_{r,2}, \dots \gamma_{r,d}, \zeta_{r,d}]^T \,, \tag{3.35}$$

where $\gamma_{r,i}$ is the GMRF parameter vector (3.13) computed for the i-th transformed spectral band for the lattice location r and $\zeta_{r,i}$ is the average local spectral value.

We assume that this parametric space can be represented using the Gaussian mixture model (GM) with diagonal covariance matrices due to the previous CAR (GMRF) parametric space decorrelation (using the Karhunen-Loeve transformation). The Gaussian mixture model for CAR (GMRF) parametric representation is as follows:

$$p(\Theta_r) = \sum_{i=1}^{K} p_i \, p(\Theta_r \,|\, \nu_i, \Sigma_i) \,, \tag{3.36}$$

$$p(\Theta_r \,|\, \nu_i, \Sigma_i) = \frac{|\Sigma_i|^{-\frac{1}{2}}}{(2\pi)^{\frac{h}{2}}} \, e^{-\frac{(\Theta_r - \nu_i)^T \Sigma_i^{-1} (\Theta_r - \nu_i)}{2}} \,. \tag{3.37}$$

The mixture model equations (3.36), (3.37) are solved using a modified EM algorithm. The algorithm is initialized using ν_i, Σ_i statistics estimated from the corresponding rectangular subimages obtained by regular division of the input image. An alternative initialization can be a random choice of these statistics. For each possible couple of regions the Kullback Leibler divergence

$$D\left(p(\Theta_r \,|\, \nu_i, \Sigma_i) \,||\, p(\Theta_r \,|\, \nu_j, \Sigma_j)\right) =$$

$$= \int_\Omega p(\Theta_r \,|\, \nu_i, \Sigma_i) \, \log\left(\frac{p(\Theta_r \,|\, \nu_i, \Sigma_i)}{p(\Theta_r \,|\, \nu_j, \Sigma_j)}\right) \, d\Theta_r \tag{3.38}$$

is evaluated and the most similar regions, i.e.,

$$\{i, j\} = \underset{k,l}{argmin} \, D\left(p(\Theta_r \,|\, \nu_l, \Sigma_l) \,||\, p(\Theta_r \,|\, \nu_k, \Sigma_k)\right)$$

are merged together in each step. This initialization results in K_{ini} regions and recomputed statistics ν_i, Σ_i. $K_{ini} > K_{opt}$ where K_{opt} is the optimal number of textured segments to be found by the algorithm. Two steps of the EM algorithm are repeated after initialization:

$$\mathbf{E}: \quad p^{(t)}(\omega_i \mid \Theta_r) = \frac{p_i\, p(\Theta_r \mid \nu_i, \Sigma_i)}{\sum_{j=1}^{K} p_j\, p(\Theta_r \mid \nu_j, \Sigma_j)}$$

$$\mathbf{M}: \quad j = 1, \ldots, K$$

$$p_j^{(t+1)} = \frac{1}{|I|} \sum_{\forall \Theta_r} p^{(t)}(\omega_j \mid \Theta_r)$$

$$\nu_j^{(t+1)} = \frac{\sum_{\forall \Theta_r} \Theta_r\, p^{(t)}(\omega_j \mid \Theta_r)}{\sum_{\forall \Theta_r} p^{(t)}(\omega_j \mid \Theta_r)}$$

$$\Sigma_j^{(t+1)} = \frac{\sum_{\forall \Theta_r} p(\omega_j \mid \Theta_r) \left(\Theta_r - \nu_j^{(t+1)}\right) \left(\Theta_r - \nu_j^{(t+1)}\right)^T}{\sum_{\forall \Theta_r} \; p^{(t)}(\omega_j \mid \Theta_r)} \,. \tag{3.39}$$

Cluster Validation

The optimal number of clusters is determined by the elimination and merging of Gaussian mixture components during the EM algorithm computation. The components with smaller weights than a fixed threshold $p_j < \phi$ (e.g. $\phi = \frac{0.01}{K_{ini}}$) are eliminated. For every pair of components we estimate their Kullback Leibler divergence (3.38). From the most similar pair, the component with the weight smaller than the threshold is merged with its stronger partner and all statistics are actualized using the EM algorithm. The algorithm stops when either the likelihood function has a negligible increase $(\mathcal{L}_t - \mathcal{L}_{t-1} < 0.05)$ or the maximum iteration number threshold is reached.

Postprocessing

The parametric vectors representing image (texture mosaic) pixels are assigned to the clusters according to the highest component probabilities, i.e., Y_r is assigned to the cluster ω_{j^*} if

$$\pi_{r,j^*} = max_j \sum_{s \in I_r} w_s\, p(\Theta_{r-s} \mid \nu_j, \Sigma_j) \,, \tag{3.40}$$

where w_s are fixed distance-based weights, I_r is a rectangular neighbourhood and $\pi_{r,j^*} > \pi_{thre}$ (otherwise the pixel is unclassified). The area of single cluster blobs is evaluated in the post-processing thematic map filtration step. Regions with similar statistics are merged. Thematic map blobs with an area smaller than a given threshold are attached to their neighbour with the highest similarity value.

3.3 Combination of Multiple Segmenters

The concept of decision fusion [76] for high-performance pattern recognition is well known and widely accepted in the area of supervised classification where (often very diverse) classification technologies, each providing complementary sources of information about class membership, can be integrated to provide more accurate, robust and reliable classification decisions than single classifier applications. It should also be noted [116] that a single classifier with a single feature set and a single generalized classification strategy often does not comprehensively capture the large degree of variability and complexity encountered in many application domains while multiple decision combinations can help to alleviate many of these problems encountered from large data variability by acquiring multiple-source information through multiple features extracted from multiple processes.

Similar advantages can also be expected for the unsupervised segmentation applications [57] as is demonstrated further in section 6.1. However, a direct unsupervised application of the supervised classifiers fusion idea is complicated by an unknown number of data hidden classes and consequently a different number of segmented regions in segmentation results to be fused. However, a direct unsupervised application of the supervised classifiers fusion idea is complicated with an unknown number of hidden data classes and consequently a different number of segmented regions in segmentation results to be fused. This method exploits above advantages by combining several unsupervised segmenters of the same type but with different feature sets.

This method combines segmentation results from different resolutions. We assume to down-sample input image Y into M different resolutions $Y^{(m)} = \downarrow^{\iota_m} Y$ with sampling factors ι_m $m = 1, \ldots, M$ identical for both directions and $Y^{(1)} = Y$. The local texture for each pixel $Y_r^{(m)}$ is represented by the 2/3D simultaneous causal autoregressive random field model (CAR) parameter space $\Theta_r^{(m)}$ (3.34) and modelled by the Gaussian mixture model (3.36), (3.37). The detailed description can be found in sections 3.1.4 and 3.2.2.

The resulting mixture model probabilities are mapped to the original fine resolution image space for all $m = 1, \ldots, M$ mixture submodels, i.e.,

$$p(\Theta_r^{(m)}) = \sum_{i=1}^{K^{(m)}} p_i^{(m)} \, p(\Theta_r^{(m)} \,|\, \nu_i^{(m)}, \Sigma_i^{(m)}) \ , \tag{3.41}$$

$$p(\Theta_r^{(m)} \,|\, \nu_i^{(m)}, \Sigma_i^{(m)}) = \frac{|\Sigma_i^{(m)}|^{-\frac{1}{2}}}{(2\pi)^{\frac{h}{2}}} \, e^{-\frac{(\Theta_r^{(m)} - \nu_i^{(m)})^T (\Sigma_i^{(m)})^{-1} (\Theta_r^{(m)} - \nu_i^{(m)})}{2}} \ . \tag{3.42}$$

The M cooperating segmenters deliver their class response in the form of conditional probabilities. Each segmenter produces a preference list based on the mixture component probabilities of a particular pixel belonging a particular class, together with a set of confidence measurement values generated in the original decision-making process.

3.3.1 Single Segmenters Correspondence

Single-resolution segmentation results cannot be combined without knowledge of the mutual correspondence between regions in all M different-resolution segmentation probabilistic mixture component maps ($K^1 \times \sum_{m=2}^{M} K^m$ combinations). Mutual assignments of two probabilistic maps are solved by using the Munkre's assignment algorithm [102] which finds the minimal cost assignment

$$g : A \mapsto B, \qquad \sum_{\alpha \in A} f(\alpha, g(\alpha))$$

between sets A, B, $|A| = |B| = \kappa$ given the cost function $f(\alpha, \beta)$, $\alpha \in A, \beta \in B$, see Fig. 3.1. α corresponds to the fine resolution probabilistic maps, β corresponds to downsampled probabilistic maps and $f(\alpha, \beta)$ is the Kullback Leibler divergence between probabilistic maps. The algorithm has polynomial complexity instead of exponential for the exhaustive search. Rectangular modification is also known as the Hungarian Algorithm.

$$A = \{a_1, a_2, a_3, a_4\}$$

$$B = \{b_1, b_2, b_3, b_4\}$$

$$g = \{(a_1, b_4), (a_2, b_1), (a_3, b_3), (a_4, b_2)\}$$

total cost: $\sum_{\alpha \in A} f(\alpha, g(\alpha)) = 23$

$$f(\alpha, \beta): \quad \begin{array}{c|cccc} b_4 & \boxed{4} & 8 & 12 & 16 \\ b_3 & 3 & 6 & \boxed{9} & 12 \\ b_2 & 2 & 4 & 6 & \boxed{8} \\ b_1 & 1 & \boxed{2} & 3 & 4 \\ \hline & a_1 & a_2 & a_3 & a_4 \end{array}$$

Figure 3.1: Munkres's assignment algorithm.

3.3.2 Final Parametric Space

The parametric vectors representing image pixels are assigned to the clusters based on our modification of the sum rule according to the highest component probabilities, i.e., Y_r is assigned to the cluster ω_{j^*} if

$$\pi_{r,j^*} = max_j \sum_{s \in I_r} w_s \left(\sum_{m=1}^{M} p(\Theta_{r-s}^{(m)} \,|\, \nu_j^{(m)}, \Sigma_j^{(m)}) \right) , \tag{3.43}$$

where w_s are fixed distance-based weights, I_r is a rectangular neighbourhood and $\pi_{r,j^*} > \pi_{thre}$ (otherwise the pixel is unclassified). The postprocessing step is performed similarly as in section 3.2.2.

3.4 Combination of Multiple Texture Models

The proposed method circumvents the problem of combining multiple unsupervised segmenters [57] by fusing multiple-processed measurements into a single segmenter feature vector. We assume to down-sample input image Y into M different resolutions $Y^{(m)} = \downarrow^{\iota_m} Y$ with sampling factors ι_m $m = 1, \ldots, M$ identical for both directions and $Y^{(1)} = Y$. Local texture for each pixel $Y_r^{(m)}$ is represented using the 3D CAR model parameter space.

For each resolution $Y^{(m)}$ there are four CAR models with the identical contextual neighbourhood I_r^c but they differ in their major movement direction (top-down, bottom-up, rightward, leftward). The local texture for each pixel and M resolutions $\alpha_1, \ldots, \alpha_M$ is represented by four parametric matrices t, b, r, l , e.g. $\hat{\gamma}_r^{i,\alpha_j}$ for $i \in \{t, b, r, l\}, \quad j = 1, \ldots, M$ which are subsequently compressed by using the local PCA (for computational efficiency) into $\tilde{\gamma}_r^{i,\alpha_j}$. Single resolution compressed parameters are composed into M parametric matrices:

$$\tilde{\gamma}_r^{\alpha_j\,T} = \{\tilde{\gamma}_r^{t,\alpha_j}, \tilde{\gamma}_r^{b,\alpha_j}, \tilde{\gamma}_r^{r,\alpha_j}, \tilde{\gamma}_r^{l,\alpha_j}\}^T \qquad j = 1, \ldots, M\,.$$

The parametric space $\tilde{\gamma}^{\alpha_j}$ is subsequently smoothed out, rearranged into a vector and its dimensionality is reduced using the PCA feature extraction ($\bar{\gamma}^{\alpha_j}$). Finally we add the average local spectral values $\zeta_r^{\alpha_j}$ to the resulting feature vector:

$$\Theta_r = [\bar{\gamma}_r^{\alpha_1}, \zeta_r^{\alpha_1}, \ldots, \bar{\gamma}_r^{\alpha_M}, \zeta_r^{\alpha_M}]^T\,. \tag{3.44}$$

Rough scale pixel parameters are simply mapped to the corresponding fine scale locations.

Multispectral, multi-resolution texture segmentation is done by clustering in the combined CAR models parameter space Θ defined on the lattice I where Θ_r is the modified parameter vector (3.44) computed for the lattice location r. The clustering is performed by the EM algorithm (sect. 3.2.2).

3.5 Hierarchy Segmentation

This method (MW3AR) [61] is an extension of the combination of the multiple segmenters method from section 3.3. It also combines segmentation results from different resolutions. We assume to down-sample input image Y into M different resolutions $Y^{(m)} =\downarrow^{\iota_m} Y$ with sampling factors $\iota_m \quad m = 1, \ldots, M$ identical in both horizontal and vertical directions and $Y^{(1)} = Y$. Local texture for each pixel $Y_r^{(m)}$ is represented by the 3D simultaneous causal autoregressive random field model (CAR) parameter space Θ_r (3.30) and modelled by the Gaussian mixture model (3.41), (3.42).

3.5.1 Initialization

The algorithm is initialized using $\nu_i^{(m)}, \Sigma_i^{(m)}$ statistics for each resolution m estimated from the corresponding thematic maps in two subsequent steps:

1. refining direction
$\nu_i^{(m-1)}\left(\forall \Theta_r^{(m-1)} : r \in \uparrow \Xi_i^{(m)}\right), \qquad \Sigma_i^{(m-1)}\left(\forall \Theta_r^{(m-1)} : r \in \uparrow \Xi_i^{(m)}\right),$
$m = M+1, M, \dots, 2\,, \qquad i = 1, \dots, K^{(m)}\,,$

2. coarsening direction
$\nu_i^{(m)}\left(\forall \Theta_r^{(m)} : r \in \downarrow \Xi_i^{(m-1)}\right), \qquad \Sigma_i^{(m)}\left(\forall \Theta_r^{(m)} : r \in \downarrow \Xi_i^{(m-1)}\right),$
$m = 2, 3, \dots, M\,, \qquad i = 1, \dots, K^{(m)}\,,$

where $\Xi_i^{(m)} \subset I \quad \forall m, i\,,$ and the first initialization thematic map $\Xi_i^{(M+1)}$ is approximated by the rectangular subimages obtained by the regular division of the input texture mosaic. All the subsequent refining steps are initialized from the preceding coarser resolution upsampled thematic maps. The final initialization results are from the second coarsening direction where the gradually coarsening segmentations are initialized using the preceding downsampled thematic maps.

3.5.2 Resulting Mixture Probabilities

The Gaussian mixture models are solved by the EM algorithm (section 3.2.2). Resulting mixture model probabilities are then mapped to the original fine resolution image space for all $m = 1, \dots, M$ mixture submodels (3.41), (3.42). The M cooperating segmenters deliver their class response in the form of conditional probabilities. The mutual assignments of two mixture components of different segmenters are solved by Munkre's algorithm (section 3.3.1).

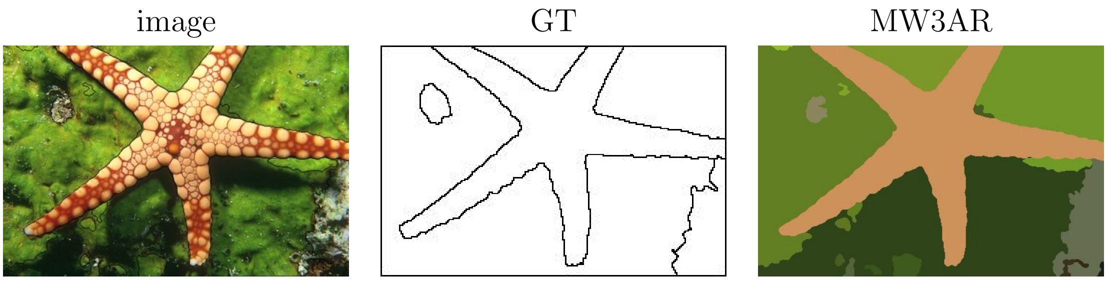


Figure 3.2: Natural image from the BSDS [89], ground truth, and the segmentation result by MW3AR method [61].

3.5.3 Final Parametric Space

The parametric vectors representing texture mosaic pixels are assigned to the clusters based on our modification of the sum rule according to the highest component probabilities, i.e., Y_r is assigned to the cluster ω_{j^*} if

$$\pi_{r,j^*} = max_j \sum_{s \in I_r} w_s \left(\sum_{m=1}^{M} \frac{p^2(\Theta_{r-s}^{(m)} \,|\, \nu_j^{(m)}, \Sigma_j^{(m)})}{\sum_{i=1}^{M} p(\Theta_{r-s}^{(i)} \,|\, \nu_j^{(i)}, \Sigma_j^{(i)})} \right) , \tag{3.45}$$

where w_s are fixed distance-based weights, I_r is a rectangular neighbourhood and $\pi_{r,j^*} > \pi_{thre}$ (otherwise the pixel is unclassified). The postprocessing step is performed similarly as in section 3.2.2.

Figure 3.2 shows the result of the segmentation on natural image (481×321) selected from the Berkeley Segmentation Dataset and Benchmark [89].


Segmentation Evaluation

Chapter 4

The Prague texture segmentation data-generator and benchmark is a web based service (`http://mosaic.utia.cas.cz`) designed to mutually compare and rank different texture segmenters, and to support the development of new segmentation and classification method. The benchmark verifies their performance characteristics on potentially unlimited monospectral, multispectral, bidirectional texture function (BTF) data using an extensive set of prevalent criteria and enables their noise robustness, scale, and rotation or illumination invariance to be tested. It can easily be used for other applications such as feature selection, image compression, and query by pictorial example, etc. The benchmark functionality is demonstrated in the next chapter by evaluation on fourteen previously published image segmentation algorithms.

4.1 Introduction

Unsupervised or supervised texture segmentation is a prerequisite for successful content-based image retrieval, scene analysis, automatic acquisition of virtual models, quality control, security, medical applications and many others. Although more than 1000 different methods were already published [16, 41, 42, 45, 84, 108, 132, 148], this ill-defined problem is still far from being solved and even cannot be solved fully in its generality. In addition to that, very little is known about properties and behaviour of already published segmentation methods and their potential user is left to randomly select one due to the absence of any guidance. This is, among other reasons, due to the lack of a reliable performance comparisons between different techniques because very limited effort was spent to develop suitable quantitative measures of segmentation quality that can be used to evaluate and compare segmentation algorithms. Rather than advance the most promising image segmentation approaches, novel algorithms are often satisfied just being sufficiently different from the previously published ones and tested on only a few carefully selected positive examples. The optimal alternative, which is to check several variants of a developed method and to carefully compare results with the state-of-the-art in this area, is practically impossible because most methods are too complicated and insufficiently described to be implemented in an acceptable period of time.

Although no theoretical property of a method can be proven using any experimental test set, such a set can suggest its performance and ranking in comparison with alternative algorithms. Because there is no available benchmark to fully support segmentation method development, we implemented a solution in the form of a web based data generator and benchmark software. Proper testing and robust learning of performance characteristics require large test sets and objective ground truths which are unfeasible for natural images. Thus, inevitably all such image sets such as the Berkeley benchmark [89] share the same drawbacks – subjectively generated ground truth regions and a limited extent which is very difficult and expensive to enlarge. These problems motivated our preference for random mosaics with randomly filled textures even if they only approximate natural image scenes. The profitable feature of this compromise is the unlimited number of different test images with corresponding objective and free ground truth maps available for each of them.

The segmentation results can be judged [148] either by using manually segmented images as a reference [81], or visually by comparing them to the original images [108], or just by applying quality measures corresponding to human intuition [81, 108, 121]. However, it is difficult to avoid subjective ranking conclusions by using either of the above approaches on limited test databases.

A prior work on the segmentation benchmark is the Berkeley benchmark presented by Martin et al. [89]. This benchmark contains more than 1000 various natural images

(300 in its public version) from the Corel image database, each of which is manually processed by a group of people to get the ground-truth segmentation in the form of partitioning of the image into a set of disjoint segments. Without any special guidance, such manual segmentations reflect subjective human perceptions and therefore, different people usually construct different ground truths on the same image. The Berkeley benchmark suffers from several drawbacks. Besides subjective ground truth, its performance criteria global consistency error (GCE) and local consistency error (LCE) also tolerate unreasonable refinement of the ground truth. Over-segmented machine segmentations have always zero consistency error, i.e., they wrongly suggest an ideal segmentation. The benchmark comparison is based on region borders hence different border localization from the human based drawing can handicap otherwise correct scene segmentation.

Another segmentation benchmark Minerva [129] contains 448 colour and grey scale images of natural scenes which are segmented using four different segmenters, segmented regions are manually labelled and different textural features can be learned from these regions and subsequently used by the kNN supervised classifier. This approach suffers from erroneous ground truth resulting from an imperfect segmenter, manual labelling and inadequate textural feature learning from small regions.

The Outex Texture Database [105] provides a public repository for three types of empirical texture evaluation test suites. It contains 14 classification test suites, one unsupervised segmentation test suite which is formed by 100 texture mosaics and finally one texture retrieval test suite. All mosaics are using the same simple regular ground truth template. The test suites are publicly available on the website (`http://www.outex.oulu.fi`), which allows searching, browsing and downloading of the test image databases. Outex currently provides a limited test repository but does not allow the evaluation of results or ranking of single algorithms.

A psycho-visual evaluation of segmentation algorithms using human observers was proposed in [128]. The test was designed to visually compare two segmentations in each step and to answer if any consensus of the best segmentation exists. While such human judgement certainly allows meaningful evaluation, this approach is too demanding to be applicable in image segmentation research.

The next section (4.2) describes the basic functionality of our benchmark, the exploited data and the benchmark generation algorithm. The following sections present benchmark performance criteria (4.3) and conclusions (4.4). The evaluation of segmentation methods examples can be found in the next chapter (5.2).

4.2 Prague Texture Segmentation Benchmark

The Prague texture segmentation data-generator and benchmark [59, 96] is a web-based service (`http://mosaic.utia.cas.cz`) developed as a part of the EU NoE no. 507752 MUSCLE project. The goal of the benchmark is to produce score, performance, and quality measures for an algorithm's performance for two main reasons: So that different algorithms can be compared to each other, and so that progress toward human-level segmentation performance can be tracked and measured over time. A good experimental evaluation should allow comparison of the current algorithm to several leading alternative algorithms, using as many test images as possible and employing several evaluation measures for comparison (in the absence of one clearly optimal measure). Our benchmark possesses all these features.

Single textures as well as the mosaics generation approach were chosen on purpose to produce unusually difficult tests to allow space for improvement and the creation of better segmentation algorithms in the future.

The benchmark allows one an evaluation of many performance characteristics of a segmenter on a virtually unlimited extent of data. However, the number of tested features requires careful consideration to include only the most important ones. Otherwise the evaluation tables would split to many specialized sub-tables with few comparative results and the benchmark would lose its chief value.

All test regions are created from natural measured textures (stochastic, regular, near-regular) hence they obey the basic texture property – homogeneity at least to certain degree. This may limit evaluation results validity on completely different (textureless) visual data types, for example segmentation of drawings, cartoons, cartographic maps, documents, range maps, characters or 3D scenes with significant geometric distortion.

Luckily, most existing images such as outdoor or indoor photographs, aerial or satellite images [126], material samples [50] or medical images [57] are well approximated by these mosaics and the benchmark ascertainments are informative also for them.

The benchmark operates either in full mode for registered users (unrestricted mode – U) or in a restricted mode. The major differences between both working modes are that the restricted operational mode does not permanently store visitor's data (results, algorithm details, etc.) into its online database and does not allow the creation of custom mosaics. To be able to use fully unrestricted benchmark functionalities the user is required to be registered (registration page).

The benchmark allows you:

- To obtain customized experimental texture mosaics and their corresponding ground truth (U);
- To obtain the benchmark texture mosaic sets with their corresponding ground truth;
- To evaluate visitor's working segmentation results and compare them with state-of-the-art algorithms;
- To update the benchmark database (U) with an algorithm (reference, abstract, benchmark results, code) and use it for subsequent other algorithms benchmarking;
- To grade noise, scale, rotation or illumination endurance of an algorithm;
- To check single mosaics evaluation details (criteria values and resulted thematic maps);
- To compare evaluation details of selected results (graphs and resulted thematic maps);
- To rank segmentation algorithms according to the most common benchmark criteria;
- To obtain LaTeX or MATLAB coded resulting criteria tables (U).

4.2.1 Image Database

Generated texture mosaics as well as the benchmarks are composed of the following texture types: (1) Monospectral textures (derived from the corresponding multispectral textures), (2) Multispectral textures, (3) BTF (bidirectional texture function) textures, (4) rotation invariant texture set, (5) scale invariant texture set, (6) illumination invariant texture set and several invariant combinations (rotation & scale, rotation & illumination, scale & illumination, rotation & scale & illumination) and (7) geometry distorted invariant texture set. Thus, it is possible to evaluate, how the performance of a segmenter changes with texture scale, illumination or rotation.

The benchmark uses colour textures from our large (more than 1000 high resolution colour textures categorized into 10 thematic classes) Prague colour texture database. All these textures are natural textures or man-made material textures which are only approximately homogeneous (i.e. local statistics for single textures are similar but not identical). Hard natural textures were chosen on purpose rather than homogeneous synthesized (for example using Markov random field models) ones because they are more difficult to be correctly segmented for segmentation methods.

The benchmark uses cut-outs from the original textures (1/6 approximately) either in the original resolution or a sub-sampled version. The remaining texture parts are used for the separate test/training sets in the benchmark-supervised mode. The benchmarks use 114 colour / greyscale textures from 10 classes. These textures were selected deliberately to be difficult for the segmenters. We believe that only under difficult conditions we can obtain useful knowledge for segmentation algorithms improvement. The BTF measurements [123] are provided by courtesy of Prof. Reinhard Klein from the Bonn University.

4.2.2 Benchmark Generation

Benchmark datasets are computer generated 512×512 random mosaics filled with randomly selected textures. The random mosaics are generated by using the Voronoi

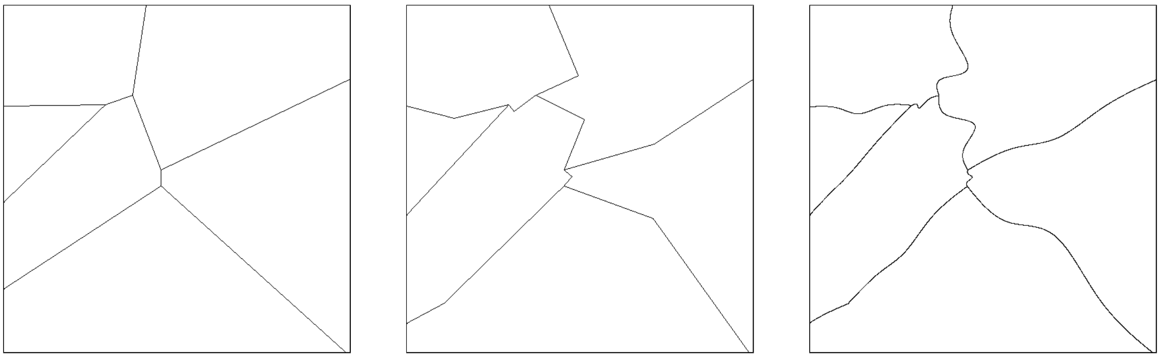


Figure 4.1: Voronoi (left), modified piecewise linear (middle) and spline (right) mosaic borders.

polygon random generator [131]. It firstly creates a Delaunay triangulation, secondly determines the circumcircle centers of its triangles, and thirdly connects these points according to the neighbourhood relations between the triangles. Resulting Voronoi polygons can further be modified (see Fig. 4.1), if required by inserting additional border points into each polygon line. Alternatively to piece-wise linear borders it is possible to generate spline defined borders or suppressed borders using a border area morphing. We exploit the fact that segmenting smaller and irregular objects is more difficult than segmenting bigger and regular objects such as squares or circles.

Colour, greyscale or BTF benchmarks are generated upon request in three quantities (normal = 20, large = 80, huge = 180 test mosaics). But if required, it is easy to automatically generate any number of such mosaics (e.g. hundreds or even thousands). The benchmark archive either in the compressed tar or in zip formats contains images in the PNG format and the data.xml file containing detailed description of all mosaics (number of regions, source component textures, size, etc.). For each texture mosaic there are also the corresponding ground truth and mask images included. The test mosaic layouts and each cell texture membership are randomly generated but with identical initialization of the corresponding random generators, so the requested benchmark sets (for the same size and type) are identical for every visitor.

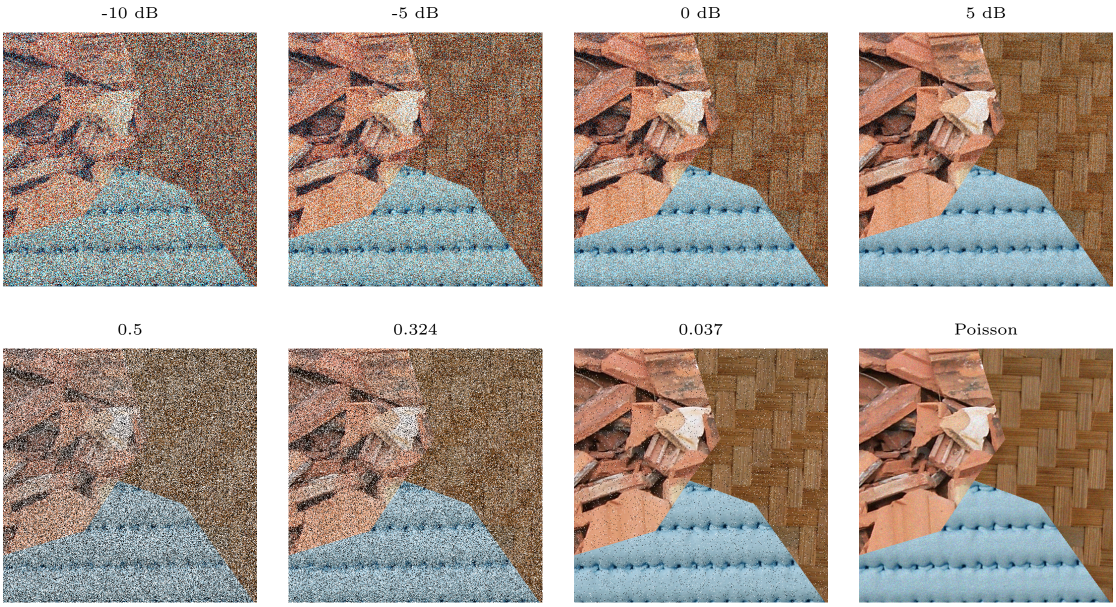


Figure 4.2: Noisy mosaics with different SNR for Gaussian noise (upper row) or different noise probabilities for salt & pepper noise (bottom row).

Noise Corruption

Noise is the important attribute that affects the performance of learning or segmenting algorithms. In real-world applications noise is an integral part of measurements and usually the noise level is unknown. The benchmark enables to test the noise robustness of single segmenters. The benchmark mosaics can be corrupted during their generation with additive Gaussian noise in several signal to noise ratio (SNR) steps (see Fig. 4.2), Poisson or salt & pepper noise. The user can choose between ten SNR steps for the additive Gaussian ($\langle -10; 35 \rangle$ dB) noise or ten steps for the salt & pepper noise (noise probabilities $\langle 0.5; 0.01 \rangle$).

Custom Mosaics

Registered users can benefit from all functions of the underlying benchmark engine. They can design their custom mosaics by specifying the image size, number of cells, number and type of textures to be used as well as the type of cell borders illustrated on Fig. 4.1 (straight lines, piecewise linear, splines or attenuated borders).

Comparative Methods

For each compared algorithm there is a concise description available. Each method contains hyperlinks to further information (author, algorithms details, BIB entry, and WWW external page). Working versions of the segmenter can be compared in the restricted mode. Uploaded temporal results and data in this mode are stored in the database for one hour only and they are deleted after its expiry.

4.3 Performance Criteria

The submitted benchmark results are evaluated and stored (U) in the server database and used for the algorithm ranking according to a chosen criterion. We have implemented the twenty-seven most frequented evaluation criteria categorized into four groups: region-based (5 criteria with the standard threshold + 5 performance curves Fig. 4.3 and performance integrals (eq. 4.1) over all threshold settings), pixel-wise (11+1), consistency measures (2) and clustering comparison criteria (3). The performance criteria mutually compare ground truth image regions with the corresponding machine segmented regions. Symbols ▲ / ▼ further denote the trend of the corresponding criterion value for the better segmenter, i.e. ▲ higher or ▼ lower values than those achieved by some inferior method. All criteria are available on two levels:

- averaged over the corresponding benchmark,
- computed for every individual test mosaics set.

The basic region-based criteria (sect. 4.3.1) available are correct, over-segmentation, under-segmentation, missed and noise. All these criteria are available either for a single threshold parameter setting or as the performance curves and their integrals. Our pixel-wise criteria group (sect. 4.3.2) contains the most frequented classification criteria such as the omission and commission errors, class accuracy, recall, precision, mapping score, etc. The consistency criteria group (sect. 4.3.3) incorporates the global and local consistency errors. Finally, the last criterion set (sect. 4.3.4) contains three clustering comparison measures. By clicking on a required criterion the evaluation table is reordered, according to this chosen criterion.

4.3.1 Region-Based Criteria

The region-based criteria [70] mutually compare the machine segmented regions R_i, $i = 1, \ldots, \mathcal{M}$ with the correct ground truth regions $\bar{R}_j$, $j = 1, \ldots, \mathcal{N}$ where $|R|$ is the corresponding set cardinality. The acceptance of the regions' overlap is controlled by the threshold $k = 0.75$. Single region-based criteria are defined as follows:

▲ **CS (correct detection)** – $\{R_m; \bar{R}_n\}$ iff

1. $|R_m \cap \bar{R}_n| \geq k\ |R_m|$,
2. $|R_m \cap \bar{R}_n| \geq k\ |\bar{R}_n|$.

The ideal segmentation has the same number of correctly detected (CS) regions with very similar shapes and locations (for the required 75 % overlap) as the ground truth map. Neither ground truth region should be ideally over-segmented as well as no machine segmented region should contain more than one corresponding ground truth region (under-segmentation).

▼ **OS (over-segmentation)** – $\{R_{m1}, \ldots, R_{mx}; \bar{R}_n\}$, $2 \leq x \leq \mathcal{M}$ iff

1. $\forall i \in \langle 1; x \rangle, \quad |R_{mi} \cap \bar{R}_n| \geq k\ |R_{mi}|$,
2. $\sum_{i=1}^{x}\ |R_{mi} \cap \bar{R}_n| \geq k\ |\bar{R}_n|$.

▼ **US (under-segmentation)** – $\{R_m; \bar{R}_{n1}, \ldots, \bar{R}_{nx}\}$, $2 \leq x \leq \mathcal{N}$ iff

1. $\sum_{i=1}^{x}\ |R_m \cap \bar{R}_{ni}| \geq k\ |R_m|$,
2. $\forall i \in \langle 1; x \rangle, \quad |R_m \cap \bar{R}_{ni}| \geq k\ |\bar{R}_{ni}|$.

▼ **ME (missed error)** – $\{\bar{R}_n\}$ iff

$\bar{R}_n \notin$ correct detection, $\bar{R}_n \notin$ over-segmentation, $\bar{R}_n \notin$ under-segmentation.

Missed regions are the ground truth regions which were not detected in neither of above categories (CS, OS, US).

▼ **NE (noise error)** – $\{R_m\}$ iff

$R_m \notin$ correct detection, $R_m \notin$ over-segmentation, $R_m \notin$ under-segmentation.

Similarly noise regions are machine segmented regions which do not belong to any of CS, OS or US categories.

Performance Curves

Single region-based criteria are also available as the corresponding performance curves (Fig. 4.3) $CS(k)$, $OS(k)$, $US(k)$, $ME(k)$, $NE(k)$. The curves allow us to compare sensitivity of different segmenters to the changing threshold value $k \in \langle 0.5; 1 \rangle$. Finally the last five region criteria are approximations of the performance curves integrals

$$\bar{f} = \int_{0.5}^{1} f(k)\, dk \,, \tag{4.1}$$

where $f(k)$ is some curve from $\{\, CS(k), OS(k), US(k), ME(k), NE(k)\,\}$. These integral criteria can be found in the brackets (Fig. 4.3) next to the criterion name, but not in the results comparison tables' page.

method	$\uparrow CS$	$\uparrow \overline{CS}$	$\downarrow OS$	$\downarrow \overline{OS}$	$\downarrow US$	$\downarrow \overline{US}$	$\downarrow ME$	$\downarrow \overline{ME}$	$\downarrow NE$	$\downarrow \overline{NE}$
TEX-ROI-SEG	56.37	52.98	11.93	11.54	19.79	17.94	11.55	19.19	10.29	18.68
MW3AR	53.04	49.60	59.53	51.08	3.20	2.93	5.63	16.05	6.96	16.87

Table 4.1: Performance curve integrals and appropriate criteria values comparison for MW3AR and TEX-ROI-SEG methods.

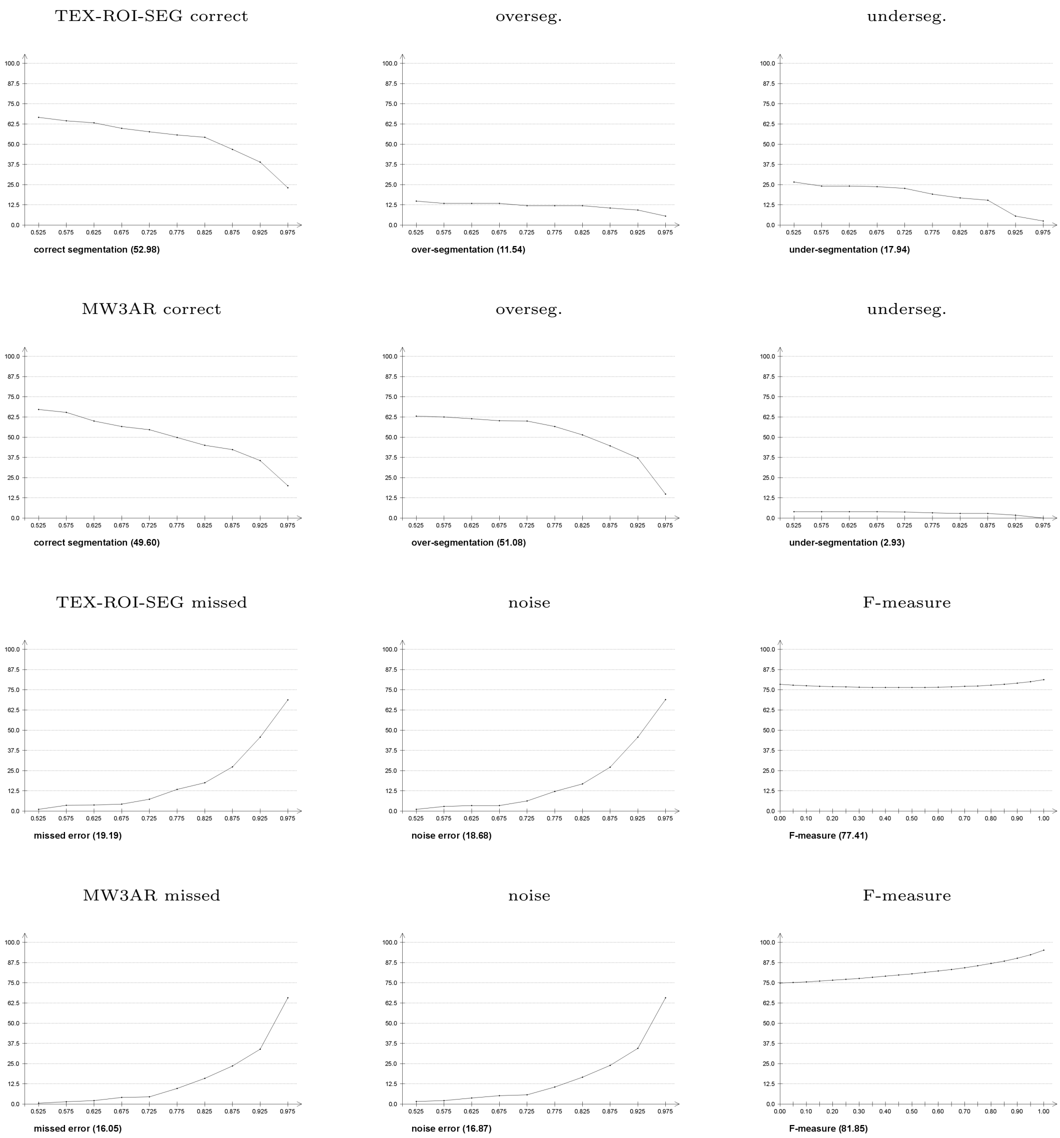


Figure 4.3: Performance curves and the corresponding performance integrals for MW3AR (even rows) and TEX-ROI-SEG methods.

The performance curves, which are shown for the TEX-ROI-SEG and MW3AR methods in Fig. 4.3, and similarly their integrals in Tab. 4.1 both confirm that this single method's behaviour is not too sensitive about the region-based criteria threshold.

There can be observed very similar ranking between performance curves integrals and their appropriate criteria in the Tabs. 5.4–5.5. As expected this threshold mainly effects the inter-region border localization. The localization error difference between the best and the worst method has only slightly diminished over the whole threshold range.

4.3.2 Pixel-Wise Weighted Average Criteria

The pixel-wise criteria were originally developed for the evaluation of supervised classifiers. We generalized them also for unsupervised applications, where their direct application is prevented due to unknown mutual correspondence between segmented and ground truth regions as well as different cardinality of both these region sets. The mutual assignment of machine segmented and ground truth regions for the pixel-wise criteria evaluation is solved by using the Munkre's assignment algorithm [102] which finds the minimal cost assignment $g : A \mapsto B,\ \sum_{\alpha \in A} f(\alpha, g(\alpha))$ between sets A, B, $|A| = |B| = \kappa$ given by the cost function $f(\alpha, \beta)$, $\alpha \in A, \beta \in B$. The algorithm has polynomial complexity instead of exponential for the exhaustive search.

Let us denote $n_{i,\bullet} = \sum_{j=1}^{\mathcal{N}} n_{i,j}$, and $n_{\bullet,i} = \sum_{j=1}^{\mathcal{M}} n_{j,i}$, where $\mathcal{N}, \mathcal{M}$ are the correct number of classes and the interpreted number of classes (or regions), respectively. $\mathcal{K} = \max\{\mathcal{M}, \mathcal{N}\}$, n is the number of pixels in the test set, $n_{i,j}$ is the number of pixels interpreted as the i-th class but belonging into the j-th class. The error matrix $(\{n_{i,j}\})$ extended into $\mathcal{K} \times \mathcal{K}$ is obtained by padding missing entries with zeros. $\hat{\imath}$ is either i for supervised tests or mapping of the i-th class ground truth into an interpretation segment based on the Munkres algorithm (see section 3.3.1) for unsupervised test. The following pixel-wise criteria were implemented:

- ▼ **O (omission error)** – the overall ratio of wrongly interpreted pixels

$$O = \text{median} \left\{ \frac{O_i}{n_{\bullet,i}} \right\}_{i=1}^{\mathcal{N}} = \text{median} \left\{ 1 - \frac{n_{\hat{\imath},i}}{n_{\bullet,i}} \right\}_{i=1}^{\mathcal{N}} \qquad \langle 0; 1 \rangle \,,$$

  where O_i is the i-th class omission error.

▼ **C (commission error)** – the overall ratio of wrongly assigned pixels

$$C = \text{median}\left\{\frac{C_i}{n_{\hat{\imath},\bullet}}\right\}_{\hat{\imath}=1}^{\mathcal{M}} = \text{median}\left\{1 - \frac{n_{\hat{\imath},i}}{n_{\hat{\imath},\bullet}}\right\}_{\hat{\imath}=1}^{\mathcal{M}} \qquad \langle 0;1\rangle\,,$$

where C_i is the i-th class commission error.

▲ **CA (the weighted average class accuracy)**

$$CA = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} \frac{n_{\hat{\imath},i}\, n_{\bullet,i}}{n_{\bullet,i} + n_{\hat{\imath},\bullet} - n_{\hat{\imath},i}} \qquad \langle 0;1\rangle\,,$$

▲ **CO (recall, the weighted average correct assignment)**

$$CO = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} n_{\bullet,i}\, CO_i = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} n_{\hat{\imath},i} \qquad \langle 0;1\rangle\,,$$

▲ **CC (precision, object accuracy, overall accuracy)**

$$CC = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} n_{\bullet,i}\, CC_i = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} \frac{n_{\hat{\imath},i}\, n_{\bullet,i}}{n_{\hat{\imath},\bullet}} \qquad \langle 0;1\rangle\,,$$

▼ **I. (type I error, the weighted probability of wrong assignment of classes pixels)**

$$I = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} (n_{\bullet,i} - n_{\hat{\imath},i}) = 1 - CO \qquad \langle 0;1\rangle\,,$$

▼ **II. (type II error, the weighted probability of commission error)**

$$II = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} \frac{n_{\hat{\imath},\bullet}\, n_{\bullet,i} - n_{\hat{\imath},i}\, n_{\bullet,i}}{n - n_{\bullet,i}} \qquad \langle 0;1\rangle\,,$$

▲ **EA (mean class accuracy estimate)**

$$EA = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} \frac{2n_{\hat{\imath},i}\, n_{\bullet,i}}{n_{\bullet,i} + n_{\hat{\imath},\bullet}} \qquad \langle 0;1\rangle \,.$$

▲ **MS (mapping score)** – emphasizes the error of not recognizing the test data

$$MS = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} (1.5\, n_{\hat{\imath},i} - 0.5 n_{\hat{\imath},\bullet}) \qquad \langle -0.5;1\rangle \,.$$

▼ **RM (root mean square proportion estimation error)**

$$RM = \sqrt{\frac{1}{\mathcal{K}}\sum_{i=1}^{\mathcal{K}} \left(\frac{n_{\hat{\imath},\bullet} - n_{\bullet,i}}{n}\right)^2} \qquad \geq 0$$

indicates unbalance between the omission O_i and commission C_i errors, respectively.

▲ **CI (comparison index)** – includes both these types of errors

$$CI = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} n_{\hat{\imath},i}\sqrt{\frac{n_{\bullet,i}}{n_{\hat{\imath},\bullet}}} = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} n_{\bullet,i}\sqrt{CC_i CO_i} \qquad \langle 0;1\rangle \,,$$

where CC_i, CO_i are the object precision and recall.

CI reaches its maximum either for the ideal segmentation or for equal commission and omission errors for every region (class).

▲ **F – measure (curve)** – see *Performance Curves* in section 4.3.1

$$F = \frac{1}{n}\sum_{i=1}^{\mathcal{K}} n_{\bullet,i}\, \frac{CC_i\, CO_i}{\varphi CO_i + (1-\varphi) CC_i} \qquad \langle 0;1\rangle \,,$$

where $\varphi \in \langle 0;1\rangle$.

For $\varphi = 0.5: \ F = EA$, $\varphi = 0: \ F = CO$ and $\varphi = 1: \ F = CC$.

4.3.3 Consistency Error Criteria

Let $\dot{S}, \ddot{S}$ be two segmentations, $\dot{R}_r$ is the set of pixels corresponding to a region in the $\dot{S}$ segmentation and containing the pixel r, $|R|$ is the set cardinality and $\setminus$ is the set difference. A refinement tolerant measure error was defined [89] at each pixel r:

$$\varepsilon_r(\dot{S}, \ddot{S}) = \frac{|\dot{R}_r \setminus \ddot{R}_r|}{|\dot{R}_r|}\,.$$

This non-symmetric local error measure encodes a measure of refinement in one direction only. Two error measures for entire image are defined:

▼ **GCE (Global Consistency Error)**

$$GCE(\dot{S}, \ddot{S}) = \frac{1}{n} \min \left\{ \sum_r \varepsilon_r(\dot{S}, \ddot{S}), \sum_r \varepsilon_r(\ddot{S}, \dot{S}) \right\}$$

forces all local refinements to be in the same direction while

▼ **LCE (Local Consistency Error)**

$$LCE(\dot{S}, \ddot{S}) = \frac{1}{n} \sum_r \min \left\{ \varepsilon_r(\dot{S}, \ddot{S}), \varepsilon_r(\ddot{S}, \dot{S}) \right\}$$

allows refinement in both directions.

$$LCE,\, GCE \in \langle 0; 1 \rangle\,, \qquad LCE \leq GCE\,.$$

The major problem with these consistency measures is their tolerance for incorrect oversegmentation of the ground truth. If the segmentation is an oversegmented version of the ground truth, or vice versa, the segmentation error is always zero. Thus the trivial segmentations with either all regions containing just one pixel or the whole image being single region are the ideal segmentations $LCE = GCE = 0$ according to both consistency criteria.

4.3.4 Clustering Comparison Criteria

Finally, three clustering comparison criteria [91] are implemented:

▼ **$\mathbf{d_{VI}}$ (variation of information)**

$$d_{VI}(\dot{S}, \ddot{S}) = H(\dot{S}) + H(\ddot{S}) - 2I(\dot{S}, \ddot{S}) \, ,$$

where the entropy is

$$H(\dot{S}) = -\sum_{i=1}^{\dot{\mathcal{K}}} \frac{\dot{n}_i}{n} \log \frac{\dot{n}_i}{n} \, ,$$

$\dot{\mathcal{K}}$, $\ddot{\mathcal{K}}$ are the numbers of sets in segmentations $\dot{S}$, $\ddot{S}$, respectively, $n_{i,j}$ is the number of points in the intersection $n_{i,j} = |\dot{R}_i \cap \ddot{R}_j|$, $\dot{n}_i = |\dot{R}_i|$, $\ddot{n}_j = |\ddot{R}_j|$ and the mutual information is

$$I(\dot{S}, \ddot{S}) = \sum_{i=1}^{\dot{\mathcal{K}}} \sum_{j=1}^{\ddot{\mathcal{K}}} \frac{n_{i,j}}{n} \log \frac{n_{i,j}}{n} \frac{\dot{n}_i}{n} \frac{\ddot{n}_j}{n} \, .$$

It is possible to show that the variation of information complies with symmetry, additivity w.r.t. refinement, additivity w.r.t. join, convex additivity and scale properties (see details in [91]).

▼ **$\mathbf{d_M}$ (Mirkin metric)**

$$d_M(\dot{S}, \ddot{S}) = \frac{\bar{d}_M(\dot{S}, \ddot{S})}{n^2} \, ,$$

where

$$\bar{d}_M(\dot{S}, \ddot{S}) = \sum_{i=1}^{\dot{\mathcal{K}}} \dot{n}_i^2 + \sum_{j=1}^{\ddot{\mathcal{K}}} \ddot{n}_j^2 - 2 \sum_{i=1}^{\dot{\mathcal{K}}} \sum_{j=1}^{\ddot{\mathcal{K}}} n_{i,j}^2 \, ,$$

▼ $\mathbf{d_D}$ **(Van Dongen metric)**

$$d_D(\dot{S},\ddot{S}) = \frac{\bar{d}_D(\dot{S},\ddot{S})}{2n} \,,$$

with

$$\bar{d}_D(\dot{S},\ddot{S}) = 2n - \sum_{i=1}^{\dot{\mathcal{K}}} \max_j n_{i,j} - \sum_{j=1}^{\ddot{\mathcal{K}}} \max_i n_{i,j} \,.$$

4.3.5 Criteria Relationship

The obvious question with the use of so many evaluation criteria by different researchers is if all are really needed. An optimal criterion depends on the intended application which is the reason for so many criteria being used. Tables 5.4 and 5.5 in the next chapter illustrate this observation, there is no segmenter scoring best for all evaluated criteria. Applications which cannot tolerate over-segmentation cannot use consistency measures or under-segmentation. Security applications and defect detectors on the other hand should guarantee low under-segmentation thus the commission error or Van Dongen metric are not the best criteria to consult. Region-based criteria are robust and appropriate for the majority of applications where precise border location is not of the primary interest. For this reason, the benchmark does not prefer any criterion. A user can click on any criterion to reorder the evaluation table according to an intended application or a tested performance characteristic.

Figure 4.4 presents colour coded correlation analysis for twenty one segmentation criteria computed for fourteen segmentation algorithms which were evaluated using our benchmark in the next chapter. While strong correlation between $I.$, CO and EA, CA can be expected, high correlation between ME,NE or CI, MS criteria is less obvious. In this experiment four mutually positively correlated groups of criteria EA, CA, CI, MS; GCE, LCE, ME, NE; CO, CA; $I., dD$ and two negatively correlated groups $I.$ with EA, MS, CA, CO and dD with MS, CA, CO emerged. The lowest mutual correlation with others has the variation of information dVI criterion. It is sufficient to use one representative criterion per correlated criterions group for a concise evaluation of the algorithm.

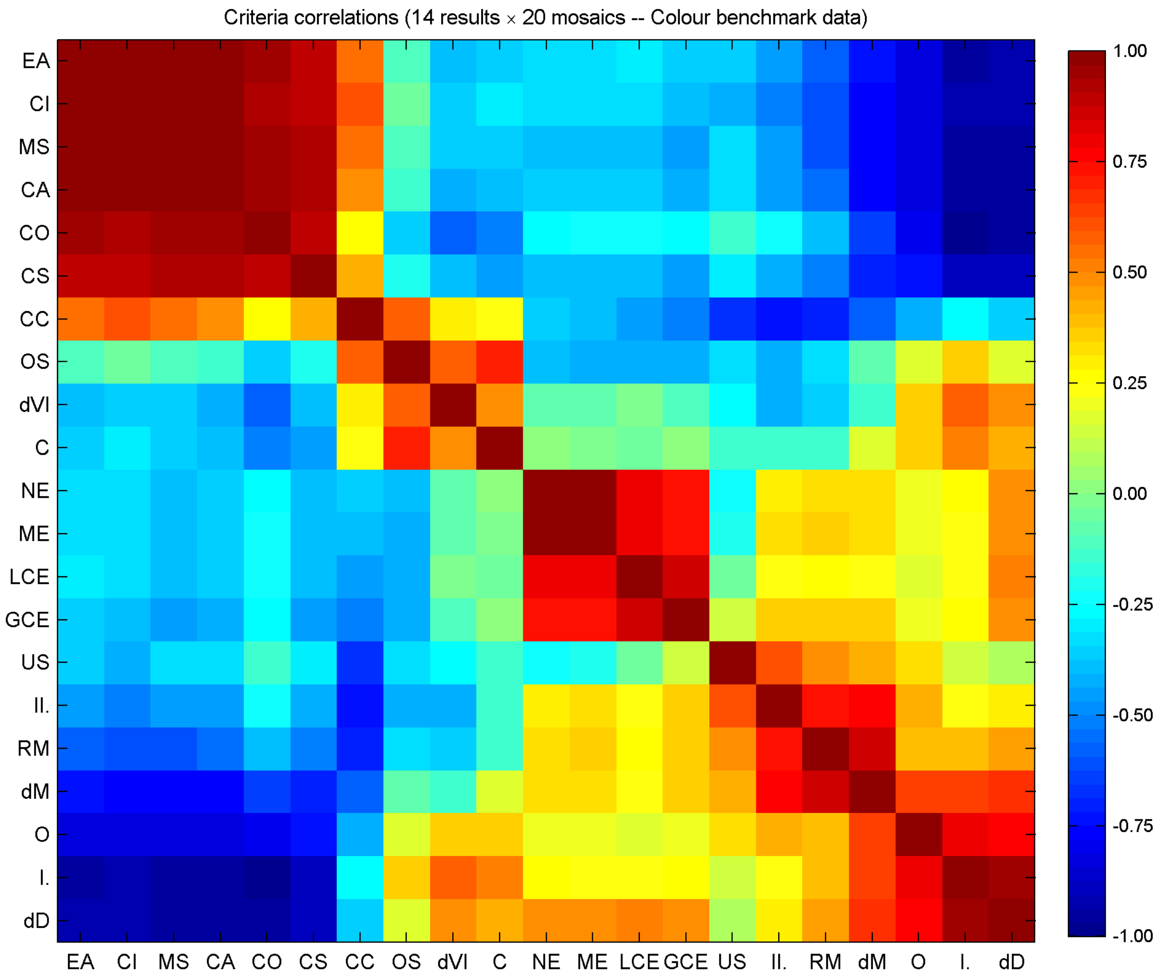
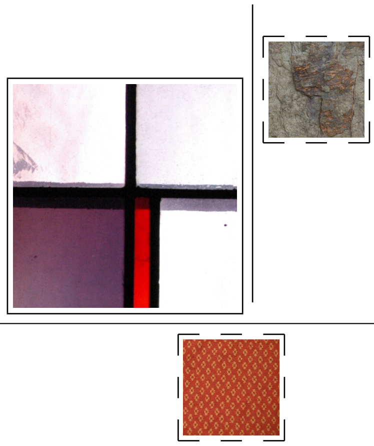


Figure 4.4: Correlation between 21 segmentation criteria computed on 280 segmentation results.

4.4 Conclusions

The implemented supervised / unsupervised segmentation benchmark is a fully automatic web application which enables us to mutually compare image segmentation algorithms and to assist in developing new segmentation methods. The comparison can be done for finalized algorithms with results, descriptions and references stored permanently in the benchmark database and used for subsequent comparison also by other algorithms or for a working version of a segmenter. Segmenters can be ranked based on a chosen criterion from the set of twenty-one regions, pixel, consistency or clustering based criteria. The test mosaics as well as the ground truths are computer generated which guarantees the objectivity of the evaluation and allows for easy generation of extensive test sets which are otherwise infeasible to arrange. The benchmark enables us to test single algorithms on monospectral, multispectral or BTF texture data and to test their noise robustness. Further on, it is possible to test scale, rotation and illumination algorithm invariance or any combination of these properties, so that the researchers can quickly and effectively compare their novel algorithms and verify their performance characteristics. Among important aspects which are not currently tested are mainly the resilience against complex geometric distortions (e.g. foreshortening) and segmentation speed, which cannot be tested because the benchmark only analyses the uploaded segmentation results. Although the benchmark is primarily designed for texture segmenters it gives also good performance insight for any tested image segmenter. The evaluation part of the benchmark is modified to use also user defined ground truth, for example hand segmented natural images. But such results are not stored in the benchmark database and hence they are not available for comparison to other users. Other possible applications such as machine learning methods evaluation, wrapper of filter based feature selection methods comparison, image compression testing, query by pictorial example methods evaluation and some others can easily benefit from the benchmark services as well.



Experimental Results

Chapter 5

Manually created mosaics are used for the evaluation of segmentation results in the first section. This approach was used by many authors for evaluation of segmentation results earlier [14, 15, 75, 147]. However self-made texture mosaics suffer from the impossibility of comparison with segmentation results obtained by different methods. Despite this limitation, many authors still use them in articles published in recent years [83, 85, 93, 114]. This is one of the reasons that leads to the creation of the segmentation benchmark described in the previous chapter (see 4.2) and to using this benchmark for subsequent results evaluation.

5.1 Manually Created Mosaics

Multispectral texture segmentation [54, 95] is done by clustering in the GMRF (see section 3.1.2) parameter space Θ defined on the lattice I where

$$\Theta_r = [\Theta_{r,1}, \Theta_{r,2}, \ldots \Theta_{r,d}]^T$$

$$\Theta_{r,i} = [\gamma_{r,i}, \sigma^2_{r,i}] \, .$$

$\Theta_{r,i}$ is the parameter vector computed for the i-th transformed spectral band for the lattice location r. Clustering is performed by the K-means algorithm, which is described in section 3.2.1.

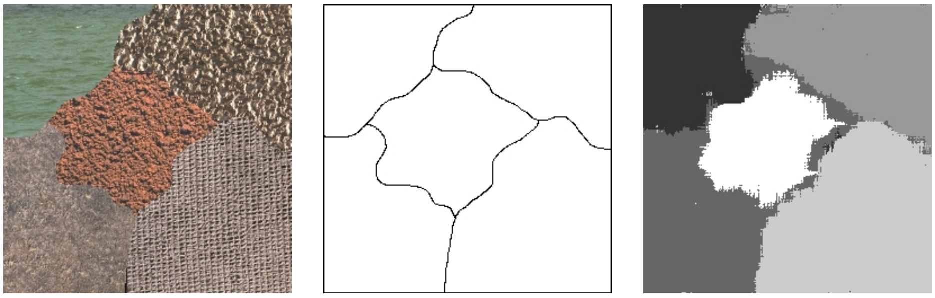


Figure 5.1: Natural texture mosaic (left), optimal texture segmentation (middle), and resulting texture regions (right).

The algorithm was tested on natural colour textures mosaics. The Fig. 5.1–left shows the 256×256 experimental texture mosaic created from five natural colour textures. The texture in the middle of Fig. 5.1–left is a food while the remaining clockwise textures are water, metal, fabric, and stone, respectively. All these textures are from the MIT Media Lab VisTex [1] collection. Natural textures have been chosen rather than synthesized (for example using Markov random field models) because they are expected to be more difficult for the underlying segmentation model. The ideal interclass borders are on the Fig. 5.1–middle and the rough segmentation result, without any postprocessing, is on the Fig. 5.1–right.

The contingency table Tab. 5.1 shows the segmentation performance of the algorithm for single natural textures. The overall probability of correct segmentation for this example is 91.9%. This result can be further improved by an appropriate postprocessing that uses, for example, the minimum area prior information.

true \ *classified*	*water*	*fabric*	*stone*	*metal*	*food*
water	8991	169	398	276	5
fabric	31	17347	458	65	18
stone	22	427	12121	120	68
metal	8	93	968	12497	24
food	700	74	1334	27	9295

Table 5.1: Contingency table of the segmentation result.

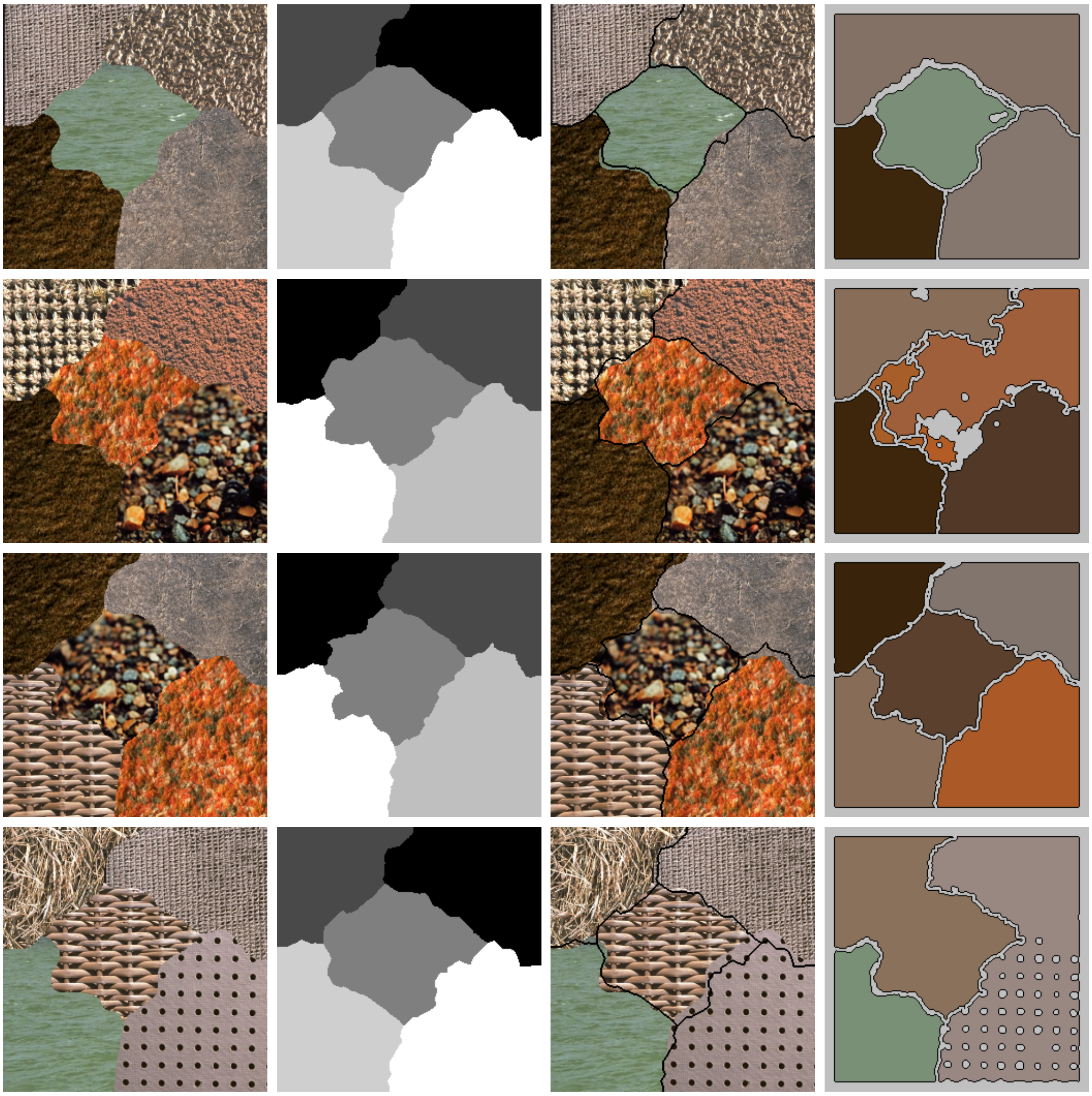


Figure 5.2: Selected experimental texture mosaics (A,B,F,G – downward), GMRF+K-means segmentation results, segmentation maps inserted into original data, and Blobworld segmentation results (rightmost column), respectively.

The Fig. 5.2 shows four 256×256 experimental texture mosaics. The last column demonstrates comparative results from the Blobworld algorithm [12]. The detected interclass borders can be checked on the Fig. 5.2 (third column) where they are in-

	GMRF + K-means				Blobworld [12]			
	pixel-wise [%]	*correct* [%]	*overseg.* [%]	*underseg.* [%]	*pixel-wise* [%]	*correct* [%]	*overseg.* [%]	*underseg.* [%]
A	97.84	97.84	0.00	0.00	79.97	59.95	0.00	30.66
B	97.09	97.09	0.00	0.00	75.35	41.92	0.00	0.00
C	97.64	97.64	0.00	0.00	74.84	0.00	0.00	79.70
D	96.33	96.33	0.00	0.00	63.87	0.00	0.00	0.00
E	96.80	70.47	26.34	0.00	90.96	64.97	0.00	0.00
F	96.29	96.29	0.00	0.00	92.95	90.48	0.00	0.00
G	94.63	94.63	0.00	0.00	76.12	16.55	0.00	31.22
avg.	96.66	92.90	3.76	0.00	79.15	39.12	0.00	20.23

Table 5.2: The segmentation results comparison for GMRF+K-means and Blobworld for mosaics A–G.

serted into the corresponding input mosaics. The second column demonstrates the robust behaviour of our algorithm while the mosaic E on Tab. 5.2 presents the infrequent algorithm failure producing an oversegmented thematic map. Such failures can be corrected by a more elaborated postprocessing step. The Blobworld algorithm [12] on these data performed steadily worse as can be seen in the last column of Fig. 5.2, some areas are undersegmented while other parts of the mosaics are oversegmented. Resulting segmentation results are promising however comparison with other algorithms is difficult because of lack of sound experimental evaluation results in the field of texture segmentation algorithms. The Berkeley segmentation dataset and benchmark proposed in [89] is not appropriate for texture mosaics because it is based on precise region borders localization.

The comparison table Tab. 5.2 shows segmentation performance of the algorithm for single natural textures using the performance metrics described in [70] (*correct* > 70% GT (ground truth) region pixels are correctly assigned, *oversegmentation* > 70% GT pixels are assigned to a union of regions, *undersegmentation* > 70% pixels from a classified region belong to a union of GT regions). The overall probability of correct segmentation for this example is 96.66%. More elaborate postprocessing can improve this result.

5.2 *Colour* Benchmark Data Set

Fourteen different algorithms – ten of the state–of–the–art methods (described in chapter 2): Blobworld (see 2.4.1) [12], JSEG (see 2.4.2) [27], EDISON (see 2.9.1) [21], EGBIS (see 2.10.1) [39], TFR (see 2.4.3) [124], TFR/KLD (see 2.4.4) [125], GSRM supervised KL area-weighted (see 2.5.1) [11], SWA (see 2.10.2) [130], HGS E (see 2.12.1) [68], TEX-ROI-SEG (see 2.11.1) [31], and our four methods (described further in this section): GMRF+EM (see 5.2.1) [54], AR3D+EM (see 5.2.2) [56], AR3D+EM multi (see 5.2.3) [57], MW3AR (see 5.2.4) [61] were tested on natural colour textures mosaics from the benchmark which is described in the previous chapter (see 4.2). The performance comparison is done using the *Colour* benchmark dataset with its normal size, i.e. 20 texture mosaic images.

5.2.1 GMRF+EM

The GMRF+EM method [54] assumes that single decorrelated monospectral texture factors can be represented by a set of local Gaussian Markov random field (GMRF) models evaluated for each pixel centred image window and for each spectral band (see 3.1.2). The segmentation algorithm based on the underlying Gaussian mixture (GM) model operates in the decorrelated GMRF parametric space. The algorithm starts with an oversegmented initial estimation which is adaptively modified until the optimal number of homogeneous texture segments is reached (see 3.2.2).

5.2.2 AR3D+EM

The AR3D+EM method [56] locally represents multispectral texture mosaics by four causal multispectral random field models recursively evaluated for each pixel (see 3.1.4). The segmentation algorithm is based on the underlying Gaussian mixture model as in the GMRF+EM method and it works in the same way.

5.2.3 AR3D+EM multi

The AR3D+EM multi method [57] is based on a combination of several unsupervised segmentation results, each in different resolution, using the sum rule. The representation of multispectral texture mosaics and segmentation part of the algorithm for single-resolution is the same as in the AR3D+EM method. The details about the combination of multiple segmenters can be found in the section 3.3. Three different resolution segmenters are used to obtain results for comparison discussed in this section ($M = 3,\ \iota_1 = 1, \iota_2 = 1.5, \iota_3 = 2$).

5.2.4 MW3AR

The MW3AR method [61] is an unsupervised multispectral, multi-resolution, multiple-segmenter for textured images with unknown number of classes. The segmenter is based on a weighted combination of several unsupervised segmentation results, each in different resolution, using the modified sum rule. This algorithm is an extension of the combination of the multiple segmenters method. It uses the previous hierarchy level EM algorithm result for the initialization of the next level. Further details can be found in section 3.5. Five different resolution levels of hierarchy are used to obtain results for comparison discussed in this section ($M = 5,\ \iota_1 = 1, \iota_2 = 1.333, \iota_3 = 1.6, \iota_4 = 2, \iota_5 = 4$).

5.2.5 Results

Tables 5.4 and 5.5 compare the overall *Colour* benchmark performance. These results demonstrate very good pixel-wise, correct region segmentation, missed error, noise error, and undersegmentation properties of MW3AR method while the oversegmentation results are slightly worse and dVI results are only average. For all the

method	[min]
Blobworld	30
JSEG	$^1/_2$
EDISON	$^1/_5$
EGBIS	$^1/_{50}$
TFR	*n/a*
TFR/KLD	*n/a*
GSRM sup. KL a-w	*n/a*

method	[min]
SWA	1
HGS E	$^2/_3$
TEX-ROI-SEG	*n/a*
GMRF+EM	55
AR3D+EM	7
AR3D+EM multi	14
MW3AR	7

Table 5.3: Approximate time performance of segmentation methods on *Colour* benchmark (run on 2 GHz processor).

pixel-wise criteria or the consistency measures this method is among the best ones. The tables illustrate a significant improvement (e.g. 23 % for the correct segmentation CS criterion) of the newer multi-segmenter method MW3AR over previous multi-segmenter AR3D+EM multi and its single-segmenter version published earlier AR3D+EM in most benchmark criteria. These results can be further improved by sophisticated postprocessing and by the optimisation of the directional models contextual neighbourhoods.

In Tab. 5.3 are shown approximate run times of segmentation 512×512 input image. Time performances for TFR, TFR/KLD, GSRM sup. KL a-w, TEX-ROI-SEG methods are not known however TFR and TFR/KLD are most likely quite time-consuming. The GMRF parameters estimation is slower than the estimation of CAR parameters since the latter uses efficient recursive Bayesian estimation instead of GMRF estimation in a sliding window. Our methods are not optimized for speed therefore the time performances are not so good. Nevertheless CAR3D parameter estimation can be easily parallelized using four CPU cores for single movement directions. And with more effort it could be even further speeded up by employing recent GPUs with hundreds of cores.

Figures 5.3–5.5 show four selected 512×512 experimental benchmark mosaics created from four to eleven natural colour textures. The last four or five rows on

	Benchmark – *Colour*						
label *version* *(RANK)*	Blobworld (11.33)	JSEG (9.67)	EDISON (8.52)	EGBIS (9.67)	GMRF+EM (7.96)	AR3D+EM (6.96)	AR3D+EM multi (6.30)
$\uparrow CS$	*21.01* ^{13}	27.47 ^{11}	*12.68* ^{14}	28.78 ^{10}	31.93 ^{8}	37.42 ^{7}	43.22 ^{6}
$\downarrow OS$	**7.33** ^{3}	38.62 ^{8}	*86.91* ^{14}	19.69 ^{7}	53.27 ^{11}	*59.53* ^{12}	49.27 ^{9}
$\downarrow US$	9.30 ^{8}	5.04 ^{4}	**0.00** ^{1}	*39.15* ^{14}	11.24 ^{9}	8.86 ^{7}	16.55 ^{10}
$\downarrow ME$	*59.55* ^{14}	*35.00* ^{13}	**2.48** ^{1}	20.42 ^{8}	14.97 ^{6}	12.54 ^{5}	**10.30** ^{3}
$\downarrow NE$	*61.68* ^{14}	*35.50* ^{13}	**4.68** ^{1}	21.54 ^{8}	16.91 ^{7}	13.14 ^{5}	12.56 ^{4}
$\downarrow O$	41.45 ^{11}	37.94 ^{10}	*73.17* ^{14}	*44.35* ^{12}	33.61 ^{8}	34.32 ^{9}	21.99 ^{5}
$\downarrow C$	58.94 ^{6}	*92.77* ^{11}	*100.00* ^{12}	82.87 ^{7}	*100.00* ^{12}	*100.00* ^{12}	*87.38* ^{10}
$\uparrow CA$	*46.23* ^{13}	55.29 ^{9}	*31.19* ^{14}	51.10 ^{11}	57.91 ^{8}	59.46 ^{7}	64.51 ^{5}
$\uparrow CO$	*56.04* ^{13}	61.81 ^{11}	*31.55* ^{14}	64.12 ^{8}	63.51 ^{9}	64.81 ^{7}	71.00 ^{6}
$\uparrow CC$	73.62 ^{11}	87.70 ^{8}	**98.09** ^{1}	*72.73* ^{12}	89.26 ^{6}	**91.79** ^{3}	90.14 ^{4}
$\downarrow I.$	*43.96* ^{13}	38.19 ^{11}	*68.45* ^{14}	35.88 ^{8}	36.49 ^{9}	35.19 ^{7}	29.00 ^{6}
$\downarrow II.$	6.72 ^{11}	3.66 ^{7}	**0.24** ^{1}	*7.59* ^{12}	3.14 ^{5}	3.39 ^{6}	3.79 ^{8}
$\uparrow EA$	*58.37* ^{13}	66.74 ^{10}	*41.29* ^{14}	59.88 ^{11}	68.41 ^{8}	69.60 ^{6}	73.90 ^{5}
$\uparrow MS$	*40.36* ^{13}	55.14 ^{9}	*31.13* ^{14}	49.03 ^{11}	57.42 ^{8}	58.89 ^{7}	64.47 ^{5}
$\downarrow RM$	7.96 ^{11}	4.96 ^{7}	**3.21** ^{2}	*8.38* ^{12}	4.86 ^{5}	4.88 ^{6}	4.55 ^{4}
$\uparrow CI$	*61.31* ^{12}	70.27 ^{9}	*50.29* ^{14}	63.11 ^{11}	71.80 ^{7}	73.15 ^{6}	76.51 ^{5}
$\downarrow GCE$	*31.16* ^{14}	*18.45* ^{12}	**3.55** ^{1}	16.64 ^{9}	16.03 ^{8}	12.13 ^{4}	15.31 ^{6}
$\downarrow LCE$	*23.19* ^{14}	11.64 ^{11}	**3.44** ^{1}	8.97 ^{8}	7.31 ^{6}	**6.69** ^{3}	7.97 ^{7}
$\downarrow dM$	*20.03* ^{13}	15.19 ^{7}	16.84 ^{10}	*19.72* ^{12}	15.27 ^{8}	15.43 ^{9}	13.51 ^{5}
$\downarrow dD$	*31.11* ^{13}	23.38 ^{11}	*35.37* ^{14}	21.29 ^{9}	20.63 ^{8}	19.76 ^{7}	16.87 ^{4}
$\downarrow dVI$	15.84 ^{7}	*17.37* ^{13}	*25.65* ^{14}	13.79 ^{4}	*17.32* ^{12}	17.10 ^{10}	16.11 ^{8}
$\uparrow \overline{CS}$	*19.10* ^{13}	29.13 ^{10}	*12.95* ^{14}	30.69 ^{9}	31.04 ^{8}	34.68 ^{7}	40.19 ^{6}
$\downarrow \overline{OS}$	10.81 ^{5}	37.70 ^{8}	*76.33* ^{14}	19.86 ^{7}	49.74 ^{11}	*53.32* ^{13}	44.79 ^{10}
$\downarrow \overline{US}$	8.35 ^{7}	6.38 ^{4}	**0.00** ^{1}	*33.66* ^{14}	11.33 ^{9}	9.24 ^{8}	12.45 ^{10}
$\downarrow \overline{ME}$	*58.54* ^{14}	*34.72* ^{12}	**13.92** ^{1}	28.07 ^{8}	21.92 ^{6}	19.90 ^{4}	22.48 ^{7}
$\downarrow \overline{NE}$	*61.24* ^{14}	*35.38* ^{12}	**15.30** ^{1}	28.74 ^{8}	23.59 ^{6}	20.79 ^{5}	24.13 ^{7}
$\uparrow \overline{F}$	*60.46* ^{13}	69.23 ^{10}	*47.42* ^{14}	62.12 ^{11}	70.79 ^{7}	72.08 ^{6}	75.72 ^{5}

Table 5.4: *Colour* benchmark results (1. part) for the following algorithms: Blobworld, JSEG, EDISON, EGBIS, GMRF+EM, AR3D+EM, AR3D+EM multi; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall − correct assignment; CC = precision − object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dM = Mirkin metric; dD = Van Dongen metric; dVI = variation of information; $\bar{f}$ are the performance curves integrals).

	Benchmark – *Colour*						
label *version* *(RANK)*	TFR (7.70)	TFR/KLD (6.48)	GSRM sup. KL a-w (2.93)	SWA (9.07)	HGS E (10.37)	TEX-ROI- -SEG (4.37)	MW3AR (3.56)
$\uparrow CS$	46.13 ^{5}	51.25 ^{4}	**68.72** ^{1}	*27.06* ^{12}	29.81 ^{9}	**56.37** ^{2}	**53.04** ^{3}
$\downarrow OS$	**2.37** ^{1}	**5.84** ^{2}	9.00 ^{4}	50.21 ^{10}	10.69 ^{5}	11.93 ^{6}	*59.53* ^{13}
$\downarrow US$	*23.99* ^{12}	7.16 ^{6}	6.67 ^{5}	**4.53** ^{3}	*33.76* ^{13}	19.79 ^{11}	**3.20** ^{2}
$\downarrow ME$	26.70 ^{10}	*31.64* ^{12}	15.09 ^{7}	25.76 ^{9}	26.89 ^{11}	11.55 ^{4}	**5.63** ^{2}
$\downarrow NE$	25.23 ^{10}	*31.38* ^{12}	15.16 ^{6}	27.50 ^{11}	25.04 ^{9}	**10.29** ^{3}	**6.96** ^{2}
$\downarrow O$	28.73 ^{6}	19.65 ^{4}	**7.74** ^{1}	33.01 ^{7}	*48.94* ^{13}	**18.21** ^{2}	**19.32** ^{3}
$\downarrow C$	12.50 ^{4}	**9.67** ^{3}	**6.79** ^{1}	85.19 ^{8}	32.39 ^{5}	**9.63** ^{2}	86.19 ^{9}
$\uparrow CA$	61.32 ^{6}	67.45 ^{4}	**78.90** ^{1}	54.84 ^{10}	*49.60* ^{12}	**69.45** ^{3}	**71.89** ^{2}
$\uparrow CO$	73.00 ^{5}	**76.40** ^{3}	**84.74** ^{1}	*60.67* ^{12}	63.37 ^{10}	**78.26** ^{2}	74.66 ^{4}
$\uparrow CC$	*68.91* ^{13}	81.12 ^{10}	89.30 ^{5}	88.17 ^{7}	*66.09* ^{14}	81.24 ^{9}	**95.04** ^{2}
$\downarrow I.$	27.00 ^{5}	**23.60** ^{3}	**15.26** ^{1}	*39.33* ^{12}	36.63 ^{10}	**21.74** ^{2}	25.34 ^{4}
$\downarrow II.$	*8.56* ^{13}	4.09 ^{9}	**2.10** ^{3}	2.11 ^{4}	*13.51* ^{14}	4.16 ^{10}	**0.74** ^{2}
$\uparrow EA$	68.62 ^{7}	75.80 ^{4}	**85.01** ^{1}	66.94 ^{9}	*58.74* ^{12}	**76.31** ^{3}	**80.43** ^{2}
$\uparrow MS$	59.76 ^{6}	65.19 ^{4}	**77.12** ^{1}	53.71 ^{10}	*46.63* ^{12}	**68.88** ^{3}	**71.78** ^{2}
$\downarrow RM$	*8.61* ^{13}	7.21 ^{9}	**4.54** ^{3}	6.11 ^{8}	*13.31* ^{14}	7.37 ^{10}	**3.09** ^{1}
$\uparrow CI$	69.73 ^{10}	77.21 ^{4}	**85.98** ^{1}	70.32 ^{8}	*61.17* ^{13}	**77.86** ^{3}	**82.43** ^{2}
$\downarrow GCE$	15.52 ^{7}	*20.35* ^{13}	13.29 ^{5}	17.27 ^{11}	16.75 ^{10}	**11.98** ^{3}	**8.17** ^{2}
$\downarrow LCE$	*12.03* ^{12}	*14.36* ^{13}	6.93 ^{5}	11.49 ^{10}	10.46 ^{9}	6.71 ^{4}	**5.78** ^{2}
$\downarrow dM$	17.47 ^{11}	12.64 ^{4}	**6.84** ^{1}	13.68 ^{6}	*27.95* ^{14}	**11.74** ^{3}	**8.97** ^{2}
$\downarrow dD$	18.21 ^{6}	18.01 ^{5}	**10.88** ^{1}	*24.20* ^{12}	22.90 ^{10}	**13.66** ^{2}	**14.78** ^{3}
$\downarrow dVI$	**13.04** ^{2}	14.06 ^{5}	14.16 ^{6}	17.16 ^{11}	**12.83** ^{1}	**13.74** ^{3}	16.67 ^{9}
$\uparrow \overline{CS}$	44.21 ^{5}	47.58 ^{4}	**63.96** ^{1}	*26.42* ^{12}	27.82 ^{11}	**52.98** ^{2}	**49.60** ^{3}
$\downarrow \overline{OS}$	**2.32** ^{1}	**5.27** ^{2}	**9.08** ^{3}	44.49 ^{9}	9.70 ^{4}	11.54 ^{6}	*51.08* ^{12}
$\downarrow \overline{US}$	*24.36* ^{12}	7.11 ^{6}	6.58 ^{5}	**5.26** ^{3}	*31.62* ^{13}	17.94 ^{11}	**2.93** ^{2}
$\downarrow \overline{ME}$	29.53 ^{9}	*37.14* ^{13}	19.92 ^{5}	33.36 ^{11}	32.86 ^{10}	**19.19** ^{3}	**16.05** ^{2}
$\downarrow \overline{NE}$	28.91 ^{9}	*37.29* ^{13}	19.86 ^{4}	33.72 ^{11}	32.47 ^{10}	**18.68** ^{3}	**16.87** ^{2}
$\uparrow \overline{F}$	69.42 ^{8}	76.81 ^{4}	**85.71** ^{1}	69.35 ^{9}	*60.51* ^{12}	**77.41** ^{3}	**81.85** ^{2}

Table 5.5: *Colour* benchmark results (2. part) for the following algorithms: TFR, TFR/KLD, GSRM sup., SWA, HGS, TEX-ROI-SEG, MW3AR; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall – correct assignment; CC = precision – object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dM = Mirkin metric; dD = Van Dongen metric; dVI = variation of information; $\bar{f}$ are the performance curves integrals).

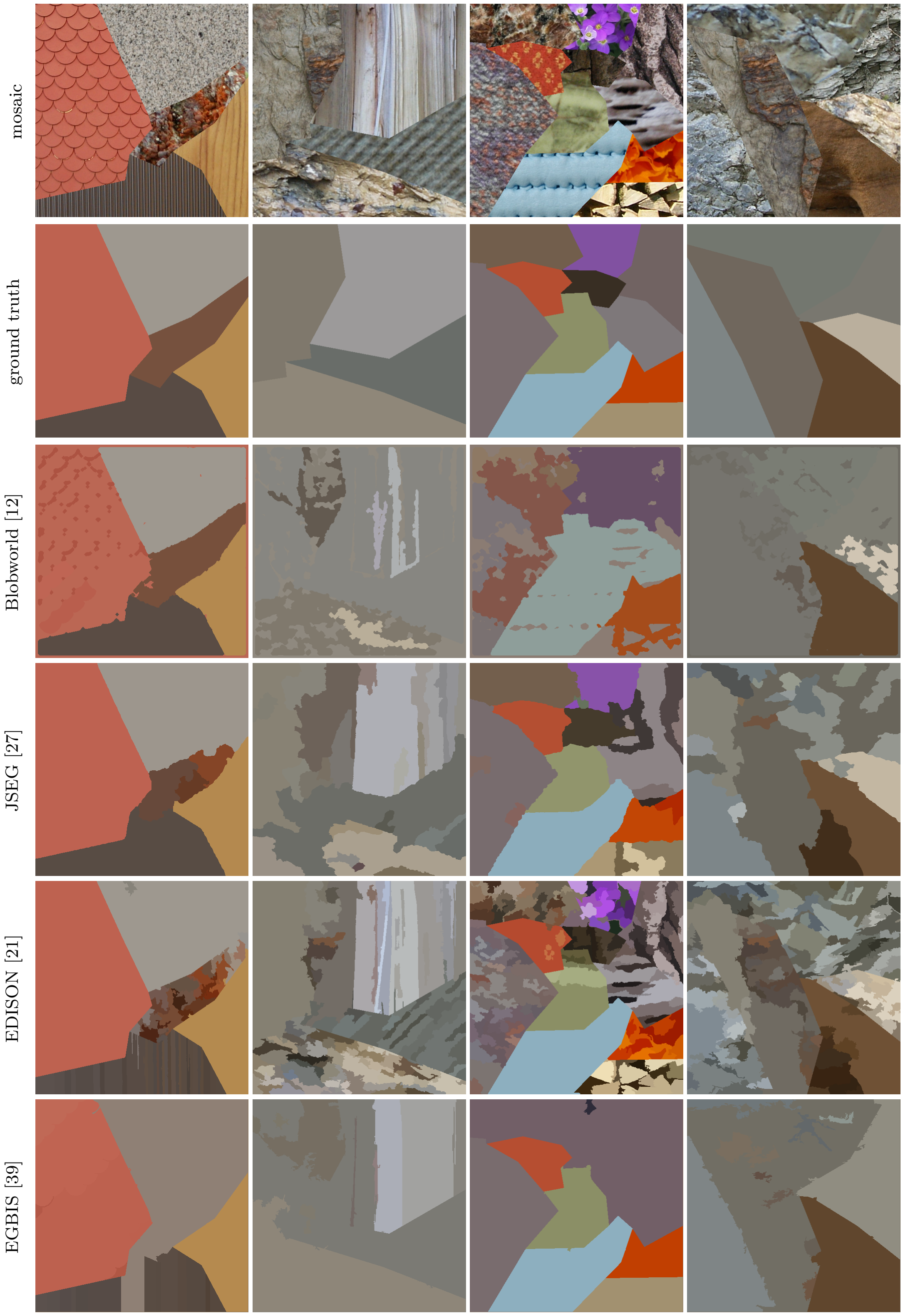


Figure 5.3: Selected experimental texture mosaics, ground truth from the *Colour* benchmark and the corresponding segmentation results (1.part).

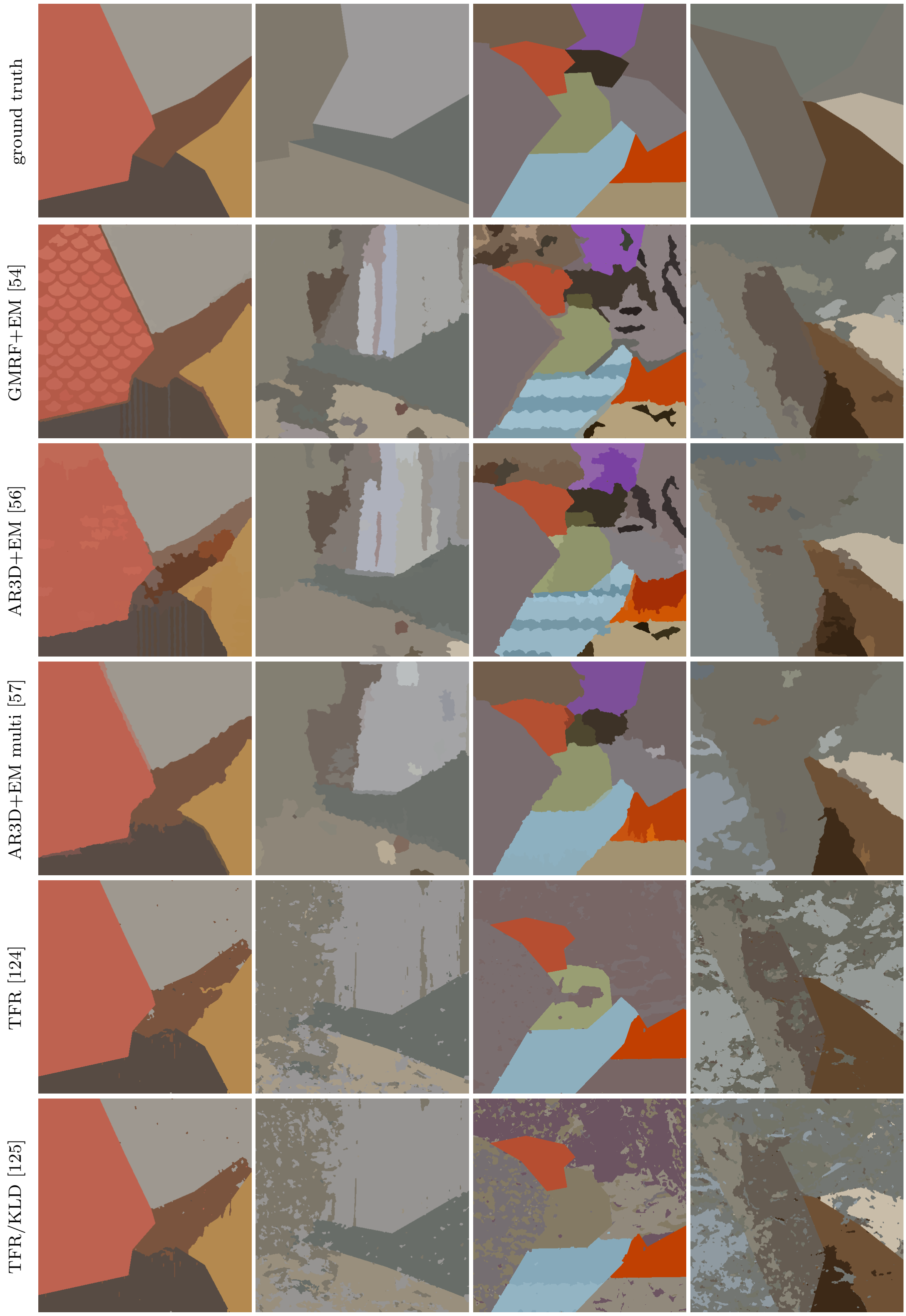


Figure 5.4: Selected ground truth from the *Colour* benchmark and the corresponding segmentation results (2.part).

Figure 5.5: Selected ground truth from the *Colour* benchmark and the corresponding segmentation results (3.part).

these figures demonstrate comparative results from the fourteen above mentioned algorithms. The second, the fifth, and the sixth row on Fig. 5.5 show segmentation results obtained by GSRM sup. KL a-w, TEX-ROI-SEG, and MW3AR methods, respectively. These three algorithms are placed as the best according to the average criteria rank. It is obvious that these methods are among the best ones in the majority of criteria.

The fourth row on Fig. 5.4 demonstrates robust behaviour of AR3D+EM multi algorithm but also infrequent algorithm failures producing the oversegmented thematic map for some textures. The TFR/KLD, AR3D+EM, GMRF+EM, SWA, EGBIS, JSEG, Blobworld, HGS, and EDISON, algorithms on these data performed mostly worse as can be seen in their corresponding rows on Figs. 5.3–5.5 some areas are undersegmented while other parts of the mosaics are oversegmented. The third and the fourth row on fig. 5.4 illustrates also the improvement of the multi-segmenter version of the algorithm at the cost of slight increase in computational complexity.

Visual comparison confirms under-segmentation tendency of EGBIS and HGS E, over-segmentation inclination of Edison, and large missed and noise errors of Blobword. JSEG indicates the second worst both missed and noise errors. The AR3D+EM, GMRF+EM methods produce similar results. The last two rows on fig. 5.4 suggest that TFR and TFR/KLD results would be improved by using an postprocessing (for example minimal region area). The consistency criteria (GCE, LCE) confirm their dubiousness. They prefer the Edison method not because of its good performance but due to its high over-segmentation error.

The overall conclusion supports the superiority of GSRM sup. KL a-w, MW3AR, and TEX-ROI-SEG methods over the tested alternatives when Blobworld, HGS E, JSEG, EGBIS, SWA and EDISON perform consistently worse. The first place of GSRM sup. KL a-w is most probably due to the fact that this method is not an unsupervised one as it uses some prior information.

Complete segmentation results for all fourteen discussed methods can be found in appendix A. All *Colour* benchmark texture mosaics, their ground truths, and appropriate segmentation results are shown in Figs. A.1–A.16 while Figs. A.17–A.30 display graphs with performance curves. The performance of some other methods as well as further details (performance criteria, curves, all test mosaics segmentations, etc.) can be found on the benchmark server.

5.3 Noise Robustness

In Tabs. B.1–B.6 (see appendix B) are shown segmentation results on texture mosaics degraded by added Gaussian noise of several levels from the range of $\langle -10; 35 \rangle$ dB. Resulting segmentations are computed on the *Colour* benchmark by following six algorithms: Blobworld [12], EDISON [21], JSEG [27], EGBIS [39], GMRF+EM [54], and AR2D+EM [55]. Figs. B.1–B.27 show per criterion graphs for all six methods. It can be seen that methods based on random field models (GMRF+EM and AR2D+EM) are more robust to added Gaussian noise (see CS, CA, CO, EA, MS, CI criteria graphs). In most cases the criterion curve starts at a low/high value (depends on the trend of criterion – upward/downward) for the most degraded images and then the value is increasing/decreasing until it reaches a high/low value for a certain noise level. From this level the criterion roughly keeps its value. GMRF+EM and AR2D+EM methods have this level at -5 dB while Blobworld has it around 0 dB. Other algorithms are more sensitive to noise and have this level at approximately 10 dB. The Gaussian noise degradation is visible on images up to the noise level of 25 dB.

5.4 Illumination Robustness

The illumination robustness of segmentation algorithms was tested on the *Colour (Illumination Invariant)* benchmark data set (described in section 4.2). Tab. 5.6 compares the overall benchmark performance of two methods: AR3D+EM ii [63] with its non illumination invariant version AR3D+EM [56] and the HGS method [68] (see 2.12.1) in its both fully illumination invariant version C and the non illumination invariant version E, respectively. The HGS segmenter combines the K-means clus-

label (version) [RANK]	HGS (C) [3.33]		HGS (E) [3.07]		AR3D+EM (ii) [1.70]		AR3D+EM () [1.89]	
	Benchmark – *Colour (Illumination Invariant)*							
↑ *CS*	*9.17*	4	9.55	3	**40.70**	1	34.14	2
↓ *OS*	**12.80**	1	19.30	2	53.02	3	*53.33*	4
↓ *US*	*37.48*	4	30.05	3	16.76	2	**13.29**	1
↓ *ME*	38.41	3	*39.72*	4	**13.96**	1	20.12	2
↓ *NE*	35.36	3	*39.64*	4	**14.85**	1	20.57	2
↓ *O*	*68.87*	4	56.44	3	35.17	2	**31.53**	1
↓ *C*	**51.63**	1	60.20	2	91.72	3	*95.34*	4
↑ *CA*	*35.81*	4	40.20	3	**59.15**	1	57.87	2
↑ *CO*	*50.70*	4	53.61	3	**65.72**	1	64.76	2
↑ *CC*	*60.67*	4	62.45	3	86.36	2	**87.17**	1
↓ *I.*	*49.30*	4	46.39	3	**34.28**	1	35.24	2
↓ *II.*	*16.15*	4	12.11	3	3.83	2	**3.52**	1
↑ *EA*	*46.22*	4	51.44	3	**68.26**	1	68.15	2
↑ *MS*	*28.32*	4	34.80	3	56.91	2	**57.23**	1
↓ *RM*	*16.63*	4	12.93	3	5.89	2	**4.78**	1
↑ *CI*	*50.03*	4	54.22	3	71.32	2	**71.40**	1
↓ *GCE*	21.31	3	*25.36*	4	**14.34**	1	16.99	2
↓ *LCE*	12.23	3	*16.69*	4	**7.62**	1	8.64	2
↓ *dD*	*29.82*	4	29.21	3	**19.82**	1	20.27	2
↓ *dM*	*38.39*	4	29.18	3	16.58	2	**14.64**	1
↓ *dVI*	**12.61**	1	13.98	2	15.80	3	*16.75*	4
↑ $\overline{CS}$	*12.58*	4	14.68	3	**36.68**	1	32.15	2
↓ $\overline{OS}$	**12.38**	1	18.67	2	*49.14*	4	48.30	3
↓ $\overline{US}$	*32.59*	4	26.16	3	14.99	2	**12.29**	1
↓ $\overline{ME}$	42.91	3	*43.20*	4	**21.70**	1	26.61	2
↓ $\overline{NE}$	43.55	3	*43.79*	4	**22.67**	1	27.36	2
↑ $\overline{F}$	*48.94*	4	53.44	3	70.40	2	**70.42**	1

Table 5.6: *Colour (Illumination Invariant)* benchmark results for HGS C, HGS E, AR3D+EM ii, AR3D+EM; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall – correct assignment; CC = precision – object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dD = Van Dongen metric; dM = Mirkin metric; dVI = variation of information; $\bar{f}$ are the performance curves integrals).

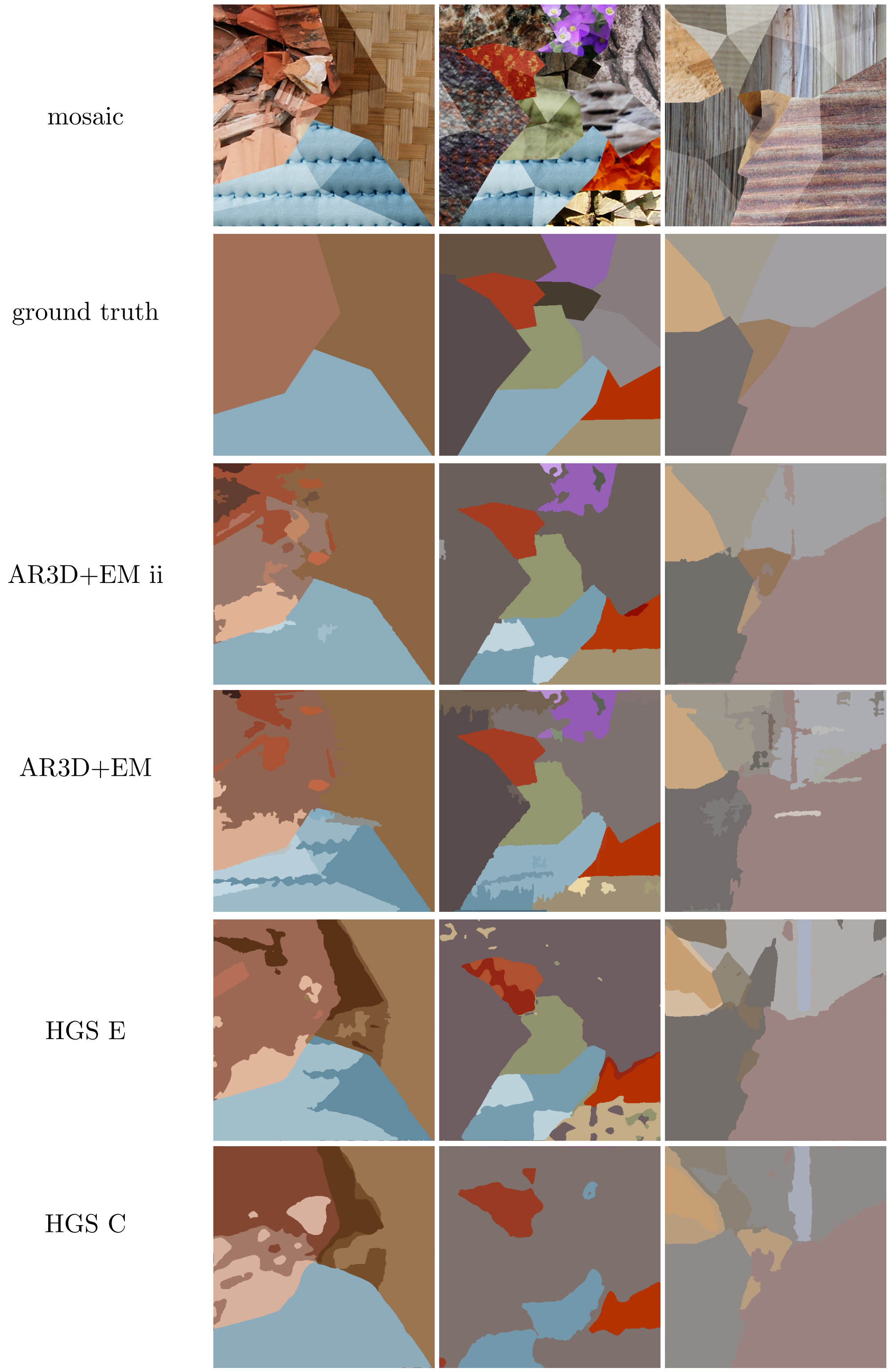


Figure 5.6: *Colour (Illumination Invariant)* benchmark – selected experimental texture mosaics, ground truth from the benchmark and the corresponding segmentation results for AR3D+EM ii, AR3D+EM 1.0, HGS E, and HGS C algorithms.

tering with the region merging step. It uses a Gabor-Gaussian spatial-colour texture representation and its illumination invariant version uses features derived from the Gabor filters applied to log-transformed images.

Results of our methods demonstrate very good performance on all criteria with the exception of oversegmentation tendency and slightly worse variation of information criterion. The important correct region segmentation criterion is four times better than for the HGS method, undersegmentation is low just like missed and noise errors. Our illumination invariant segmenter outperforms its non-invariant counterpart as expected, however the same conclusion cannot be claimed for the HGS method.

Fig. 5.6 shows three selected 512×512 benchmark mosaics created from three to eleven natural colour textures. The last four columns demonstrate comparative results from two alternative methods, both in illumination invariant and non-invariant versions, respectively. The third column demonstrates robust behaviour of our algorithm but also infrequent algorithm failures producing the oversegmented thematic map for some textures. Such failures can be reduced by a more elaborate postprocessing step. The HGS-C, HGS-E algorithms on these data performed steadily worse as can be seen in the last two columns of Fig. 5.6. Some areas are undersegmented while other parts of the mosaics are oversegmented. Resulting segmentation results are promising even if we could compare only one illumination invariant alternative method.

5.5 Benchmark Dataset Size Test

Benchmark data sets (see 4.2.2) are provided in three different quantities: normal size ($\times 1$), large size ($\times 4$), and huge size ($\times 9$) – 20, 80, and 180 images for *Colour* benchmark data set, respectively. Tab. 5.7 shows segmentation results on the *Colour* benchmark for three algorithms: Blobworld (see 2.4.1), JSEG (see 2.4.2), and EGBIS (see 2.10.1). Each method has three result columns – for normal, large, and huge data set size. The results shows that normal size results are reasonable estimations of criteria values while huge size results are almost the same as large size results.

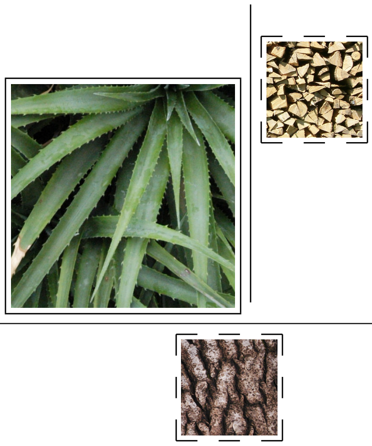
	Benchmark – *Colour*								
	Blobworld			JSEG			EGBIS		
	[normal]	[large]	[huge]	[normal]	[large]	[huge]	[normal]	[large]	[huge]
$\uparrow CS$	*21.01*	**25.83**	24.68	*27.47*	32.73	**33.35**	*28.78*	**30.45**	29.73
$\downarrow OS$	*7.33*	**6.86**	7.08	**38.62**	47.24	*47.26*	*19.69*	17.08	**17.03**
$\downarrow US$	**9.30**	9.55	*10.79*	**5.04**	5.66	*7.50*	*39.15*	**37.25**	37.53
$\downarrow ME$	*59.55*	54.63	**54.12**	*35.00*	24.88	**21.54**	**20.42**	23.83	*23.94*
$\downarrow NE$	*61.68*	57.79	**57.33**	*35.49*	25.38	**22.10**	**21.54**	*23.53*	23.37
$\downarrow O$	*41.45*	**40.88**	40.98	*37.91*	36.44	**34.83**	44.35	**43.62**	*47.07*
$\downarrow C$	*58.94*	55.29	**53.99**	**92.77**	92.98	*93.89*	*82.87*	**77.32**	78.87
$\uparrow CA$	*46.23*	50.29	**50.30**	*55.32*	57.90	**58.32**	**51.10**	49.41	*49.10*
$\uparrow CO$	*56.04*	59.77	**59.93**	*61.85*	63.73	**64.32**	**64.12**	63.08	*62.77*
$\uparrow CC$	*73.62*	**76.13**	74.72	*87.70*	**89.05**	88.71	**72.73**	72.46	*72.12*
$\downarrow I.$	*43.96*	40.23	**40.07**	*38.15*	36.27	**35.68**	**35.88**	36.92	*37.23*
$\downarrow II.$	*6.72*	**6.29**	6.58	*3.66*	**2.90**	2.98	**7.59**	8.13	*8.15*
$\uparrow EA$	*58.37*	**62.08**	61.97	*66.76*	68.96	**69.14**	**59.88**	57.78	*57.54*
$\uparrow MS$	*40.36*	44.71	**44.95**	*55.18*	57.66	**58.05**	**49.03**	47.14	*46.78*
$\downarrow RM$	*7.96*	**7.36**	7.54	*4.95*	**4.67**	4.76	**8.38**	*8.87*	8.86
$\uparrow CI$	*61.31*	**64.73**	64.40	*70.29*	72.24	**72.35**	**63.12**	61.33	*61.09*
$\downarrow GCE$	31.16	*31.58*	**30.82**	*18.46*	16.33	**15.89**	*16.64*	16.45	**16.24**
$\downarrow LCE$	23.19	*23.39*	**22.82**	*11.64*	10.05	**9.63**	**8.97**	*9.30*	9.13
$\downarrow dVI$	*15.84*	15.75	**15.66**	17.36	*17.44*	**17.35**	*13.79*	**13.71**	13.78
$\downarrow dM$	*20.03*	**18.07**	18.16	*15.18*	13.15	**13.06**	**19.72**	20.00	*20.39*
$\downarrow dD$	*31.11*	29.71	**29.18**	*23.36*	21.76	**21.27**	**21.29**	21.85	*21.92*
$\uparrow \overline{CS}$	*19.10*	23.12	**23.35**	*29.19*	31.90	**32.80**	**30.69**	30.23	*29.49*
$\downarrow \overline{OS}$	*10.81*	**8.55**	9.03	**37.70**	*43.91*	43.27	*19.86*	**17.35**	17.54
$\downarrow \overline{US}$	**8.35**	8.92	*10.25*	6.38	**6.19**	*7.10*	*33.66*	**32.35**	32.61
$\downarrow \overline{ME}$	*58.54*	56.85	**55.22**	*34.72*	29.42	**27.82**	**28.07**	30.91	*31.20*
$\downarrow \overline{NE}$	*61.24*	60.24	**58.73**	*35.38*	29.95	**28.29**	**28.74**	30.95	*31.19*
$\uparrow \overline{F}$	*60.46*	**63.96**	63.70	*69.25*	71.28	**71.40**	**62.12**	60.23	*59.99*

Table 5.7: Blobworld, JSEG and EGBIS results for *Colour* benchmark with data set sizes [normal], [large], [huge]; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall – correct assignment; CC = precision – object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dVI = variation of information; dM = Mirkin metric; dD = Van Dongen metric; $\bar{f}$ are the performance curves integrals).

	Benchmark – *Colour*								
	Blobworld			JSEG			EGBIS		
	[normal]	[large]	[huge]	[normal]	[large]	[huge]	[normal]	[large]	[huge]
$\uparrow CS$	*−14.87%*	**+4.66%**	0.00%	*−17.63%*	−1.86%	**0.00%**	*−3.20%*	**+2.42%**	0.00%
$\downarrow OS$	*+3.53%*	**−3.11%**	0.00%	**−18.28%**	−0.04%	*0.00%*	*+15.62%*	+0.29%	**0.00%**
$\downarrow US$	**−13.81%**	−11.49%	*0.00%*	**−32.80%**	−24.53%	*0.00%*	*+4.32%*	**−0.75%**	0.00%
$\downarrow ME$	*+10.03%*	+0.94%	**0.00%**	*+62.49%*	+15.51%	**0.00%**	**−14.70%**	−0.46%	*0.00%*
$\downarrow NE$	*+7.59%*	+0.80%	**0.00%**	*+60.59%*	+14.84%	**0.00%**	**−7.83%**	*+0.68%*	0.00%
$\downarrow O$	*+1.15%*	**−0.24%**	0.00%	*+8.84%*	+4.62%	**0.00%**	−5.78%	**−7.33%**	*0.00%*
$\downarrow C$	*+9.17%*	+2.41%	**0.00%**	**−1.19%**	−0.97%	*0.00%*	*+5.07%*	**−1.97%**	0.00%
$\uparrow CA$	*−8.09%*	−0.02%	**0.00%**	*−5.14%*	−0.72%	**0.00%**	**+4.07%**	+0.63%	*0.00%*
$\uparrow CO$	*−6.49%*	−0.27%	**0.00%**	*−3.84%*	−0.92%	**0.00%**	**+2.15%**	+0.49%	*0.00%*
$\uparrow CC$	*−1.47%*	**+1.89%**	0.00%	*−1.14%*	**+0.38%**	0.00%	**+0.85%**	+0.47%	*0.00%*
$\downarrow I.$	*+9.71%*	+0.40%	**0.00%**	*+6.92%*	+1.65%	**0.00%**	**−3.63%**	−0.83%	*0.00%*
$\downarrow II.$	*+2.13%*	**−4.41%**	0.00%	*+22.82%*	**−2.68%**	0.00%	**−6.87%**	−0.25%	*0.00%*
$\uparrow EA$	*−5.81%*	**+0.18%**	0.00%	*−3.44%*	−0.26%	**0.00%**	**+4.07%**	+0.42%	*0.00%*
$\uparrow MS$	*−10.21%*	−0.53%	**0.00%**	*−4.94%*	−0.67%	**0.00%**	**+4.81%**	+0.77%	*0.00%*
$\downarrow RM$	*+5.57%*	**−2.39%**	0.00%	*+3.99%*	**−1.89%**	0.00%	**−5.42%**	*+0.11%*	0.00%
$\uparrow CI$	*−4.80%*	**+0.51%**	0.00%	*−2.85%*	−0.15%	**0.00%**	**+3.32%**	+0.39%	*0.00%*
$\downarrow GCE$	+1.10%	*+2.47%*	**0.00%**	*+16.17%*	+2.77%	**0.00%**	*+2.46%*	+1.29%	**0.00%**
$\downarrow LCE$	+1.62%	*+2.50%*	**0.00%**	*+20.87%*	+4.36%	**0.00%**	**−1.75%**	*+1.86%*	0.00%
$\downarrow dVI$	*+1.15%*	+0.57%	**0.00%**	+0.06%	*+0.52%*	**0.00%**	*+0.07%*	**−0.51%**	0.00%
$\downarrow dM$	*+10.30%*	**−0.50%**	0.00%	*+16.23%*	+0.69%	**0.00%**	**−3.29%**	−1.91%	*0.00%*
$\downarrow dD$	*+6.61%*	+1.82%	**0.00%**	*+9.83%*	+2.30%	**0.00%**	**−2.87%**	−0.32%	*0.00%*
$\uparrow \overline{CS}$	*−18.20%*	−0.99%	**0.00%**	*−11.01%*	−2.74%	**0.00%**	**+4.07%**	+2.51%	*0.00%*
$\downarrow \overline{OS}$	*+19.71%*	**−5.32%**	0.00%	**−12.87%**	*+1.48%*	0.00%	*+13.23%*	**−1.08%**	0.00%
$\downarrow \overline{US}$	**−18.54%**	−12.98%	*0.00%*	−10.14%	**−12.82%**	*0.00%*	*+3.22%*	**−0.80%**	0.00%
$\downarrow \overline{ME}$	*+6.01%*	+2.95%	**0.00%**	*+24.80%*	+5.75%	**0.00%**	**−10.03%**	−0.93%	*0.00%*
$\downarrow \overline{NE}$	*+4.27%*	+2.57%	**0.00%**	*+25.06%*	+5.87%	**0.00%**	**−7.86%**	−0.77%	*0.00%*
$\uparrow \overline{F}$	*−5.09%*	**+0.41%**	0.00%	*−3.01%*	−0.17%	**0.00%**	**+3.55%**	+0.40%	*0.00%*

Table 5.8: *Colour* benchmark data set sizes relative comparison for Blobworld, JSEG and EGBIS; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall – correct assignment; CC = precision – object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dVI = variation of information; dM = Mirkin metric; dD = Van Dongen metric; $\bar{f}$ are the performance curves integrals).

Hence it is not necessary to compute more than two times of segmentation results for huge size. Normal size results can be used as algorithm performance overview in just quarter of computation time however large size data set should be used for more accurately method comparison. The differences between normal, large, and huge size results for Blobworld method could be caused by the nature of the Blobworld random algorithm producing different results for each run even on a single image. In Tab. 5.8 is the relative comparison of the results w. r. t. huge data size results. EGBIS method has the most consistent results over all data set sizes. The difference between results of large and huge size are negligible. Similar data set size dependency holds for benchmark criteria – some criteria are more robust to data set size (EA, CC, dVI) while others are more variant (ME, NE).



Applications

Chapter 6

6.1 Mammography

Breast cancer is the leading cause of death [115, 140] among all cancers for middle-aged women in most developed countries. Thus a significant effort is currently focused on cancer prevention and early detection which can significantly reduce the mortality rate. X-ray screening mammography is the most frequented method for breast cancer early detection although it is not without problems [115] such as rather large minimum detectable tumor size, higher mammogram sensitivity for older women or radiation exposition.

Automatic mammogram analysis is still a difficult task due to a wide variation of breast anatomy, nevertheless a computer-aided diagnosis system can successfully assist a radiologist, and can be used as a second opinion. The first step in a such system is detection of suspicious potentially cancerous regions of interest. Several approaches to detect these regions of interest (ROI) were published [122, 140] mostly based on supervised learning. One important task for radiologists when interpreting mammograms consists in evaluating the proportion of fatty and fibroglandular tissue with respect to whole breast because the fibroglandular tissue has a higher probability of containing a breast cancer than fatty tissue.

This study proposes an unsupervised segmentation method for fast automatic mammogram segmentation into the regions of interest (ROI) using a statistical random field based texture representation [60, 62]. Presented methods detect the fibroglandular tissue regions from either craniocaudal (CC) or mediolateral oblique (MLO) views and thus can help focus a radiologist to this most important breast region.

The method can be enriched also by a tool to numerically evaluate the cancer risk based on the proportion of fatty and fibroglandular tissue. Spatial interaction models and especially Markov random fields-based models are increasingly popular for texture representation [47, 72, 117], etc. Several researchers dealt with the difficult problem of unsupervised segmentation using these models see for example [3, 48, 54, 87, 109].

6.1.1 Breast Detector

The unsupervised detector starts with automatic breast area detection because it can be cheaply computed and simplifies the subsequent regions of interest detection. This is performed using simple histogram thresholding with the automatically selected threshold. Because all mammograms contain one or several labels, the binarized mammogram contains several white regions. We compute their areas and all but the largest one are discarded and merged with the background. In this stage the algorithm also decides the breast orientation on the mammogram (left or right). Fig. 6.1–breast mask shows resulting detected breast area (in inverted grey levels). The following detection of regions of interest is performed only in the breast region ignoring the background area set in the mask template.

6.1.2 Experimental Results

The algorithm was tested on mammograms from the Digital Database for Screening Mammography (DDSM) from the University of South Florida [67]. This database contains 2620 four view (left and right craniocaudal (CC) and mediolateral oblique (MLO)) mammograms in different resolutions. Single mammograms cases are divided into normal, benign, benign without callback volumes and cancer.

Two methods were tested – the first one is the combination of multiple segmenters. Fig. 6.2 demonstrates benefits of the multiple segmenter approach (MC) over its sin-

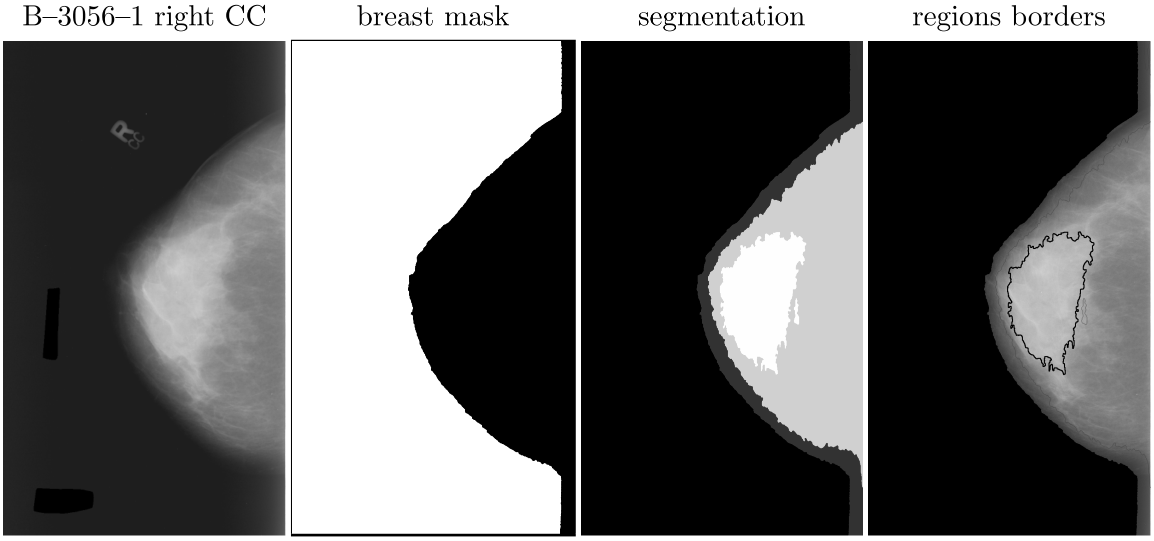


Figure 6.1: Normal right breast mammogram (patient age 58, but with a cancerous lesion in the left breast), the detected breast area, segmentation result and detected regions of interest, respectively.

gle segmenter (SC) counterpart. The MC detector determined more accurately the cancer tissue while the single segmenter found only the corresponding larger region of interest with the cancer lesion. Smooth greyscale mammogram textures require two dimensional models for adequate representation hence 2D CAR models were used in this method that is described in section 3.3.

Our experiments are done with two segmenters ($M = 2$) using sampling factors $\iota_1 = 4, \iota_2 = 8$ and the causal neighbourhood with ten neighbours ($\eta = 10$). Fig. 6.2 show right mammogram of a patient age 65 with detected irregular, ill defined lesion type. Both segmenters (single as well as multiple) correctly found the region of interest with the cancer lesion. The multiple segmenter found also the cancer lesion itself. Similarly, Fig. 6.3 demonstrates region of interest containing an ill defined lobulated cancer lesion found by the pathologist.

The second method is the combination of multiple texture models that is described in section 3.4. Smooth pseudo-colour mammogram (original greyscale mammogram and

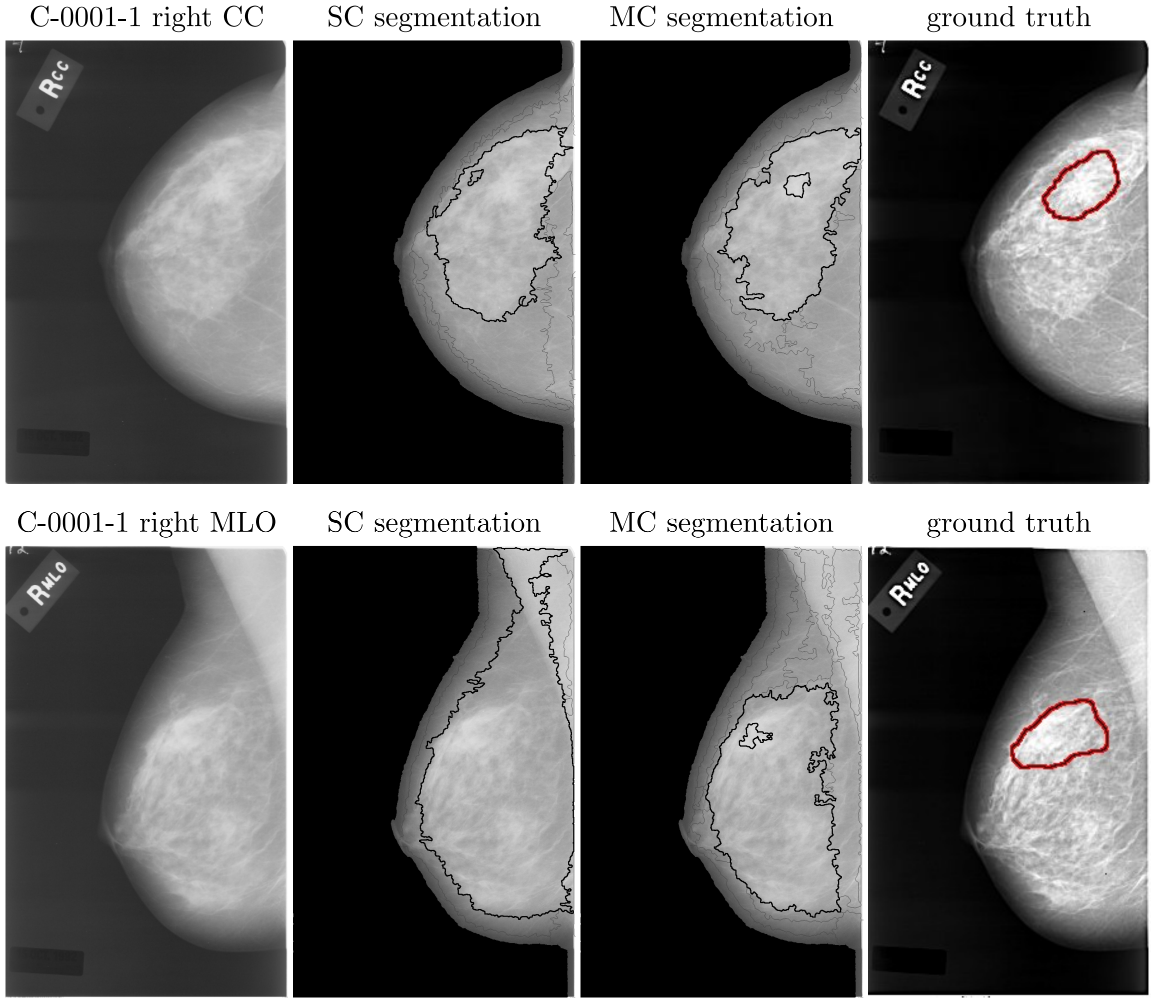


Figure 6.2: Cancer case mammogram (patient age 65), radiologist associated ground truth and detected regions of interest using single segmenter (SC) and multiple segmenter (MC) approach, respectively.

its two nonlinear gamma transformations) textures require three dimensional models for adequate representation hence we used 3D CAR models here. Finally, after the segmentation, regions which have grey level mean value difference from the median mean value (over the same type of digitised mammograms) of cancerous ground truth regions larger than a specified threshold are eliminated.

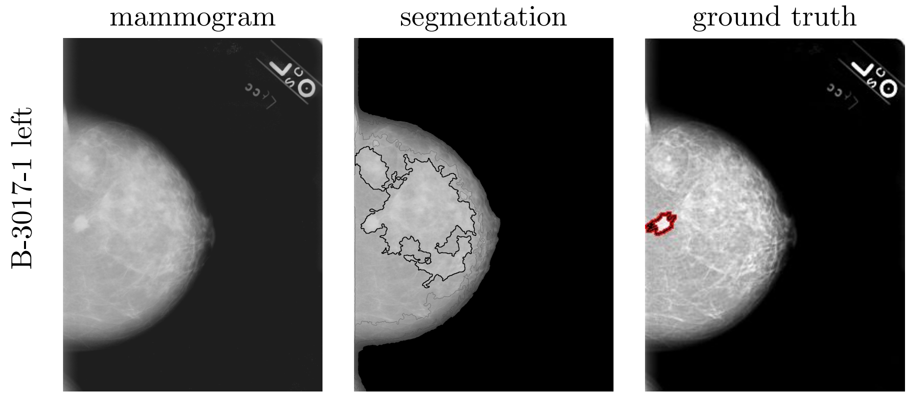


Figure 6.3: Cancer case mammogram (patient age 48) and its detected ROI.

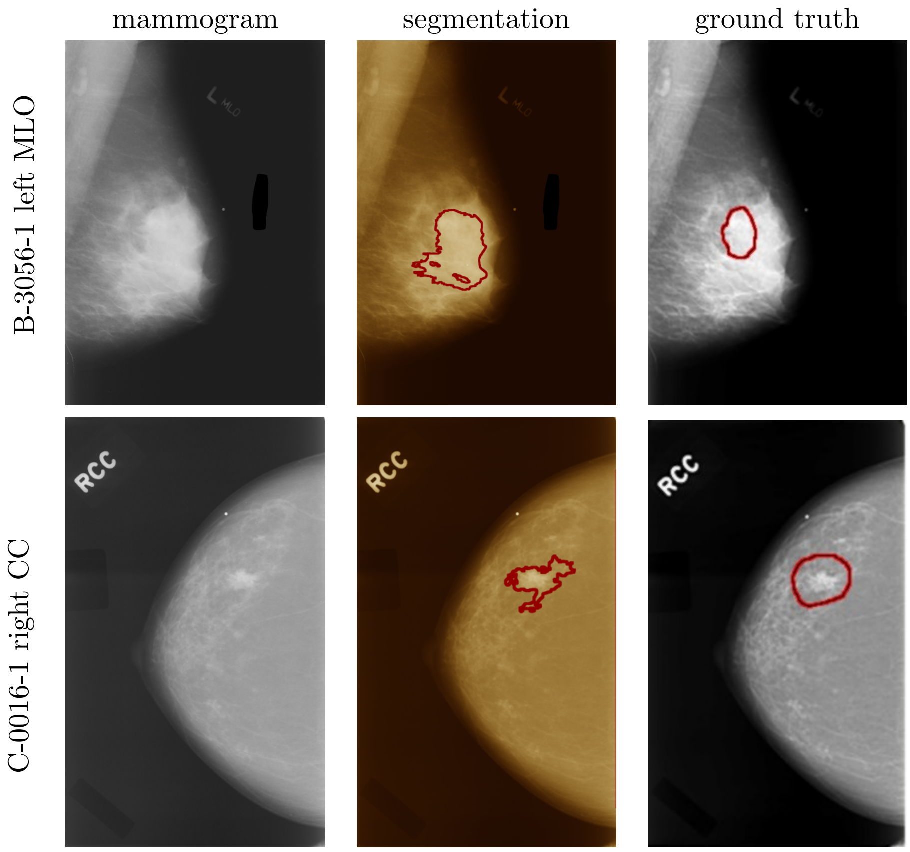


Figure 6.4: Cancerous mammograms (patients age 58 (top) and 80 (bottom)), radiologist associated ground truth and detected regions of interest using the multiple segmenter approach, respectively.

Experiments are done with three resolutions ($M = 3$) using sampling factors $\iota_1 = 2, \iota_2 = 4, \iota_3 = 8$ and the causal neighbourhood with fourteen neighbours ($\eta = 14$). Fig. 6.4–top show left MLO mammogram of a patient age 58 with detected malignant asymmetric lesion and the right CC mammogram (Fig. 6.4–bottom) of a patient age 80 with detected irregular, spiculated malignant lesion type. The segmenter correctly found the region of interest with the cancer lesion on both mammograms.

The detected region of interest results Figs. 6.1–6.4 demonstrate very good region segmentation and low oversegmentation properties of our method. Resulting ROI segmentation results are promising however comparison with other algorithms is difficult because of lack of sound experimental evaluation results in the field of screening mammography segmentation.

6.2 Defect Detection

Traditional manual inspection of material surfaces is labour- and cost-intensive and offers a major bottleneck in the high-speed production lines [78]. Many defects are very difficult to detect manually; it is estimated [135] that a highly trained human operator can detect about 60% to 70% of leather material defects. The advantages of automated visual inspection are well known; repeatability, reliability and accuracy. Unfortunately very few practical automated inspection systems for automated inspection of textile surfaces are available mainly due to their computational costs [44]. Texture imperfections are either non-textured or different textured patches that locally disrupt the homogeneity of a texture image [17]. Quality is a topical issue [94] in manufacturing, designed to ensure that defective products are not allowed to reach the customer. Since in many areas, the quality of a surface is best characterized by its texture, texture analysis plays an important role in automatic visual inspection of surfaces. The major textile texture defects reported by [94] were, missing threads (causing dark lines on the image), gathered knots and oil stains (causing small dark regions on the image), gathered threads (causing dark curves on the image), and tiny holes on the fabrics. Due to the nature of the weaving process, the majority of the defects on the textile web occur along two directions i.e. horizontal and vertical [78].

Defect detection in textured materials can be very subjective task, since defects can be a very subtle and not well localized in space, which may lead to a small modification in the power spectrum.

The conventional approach [17] is to compute texture features in a local subwindow and to compare them with the reference values representing a perfect pattern. The method [94] preprocesses a grey level textile texture with histogram modification and median filtering. The image is subsequently thresholded using the 2D CAR model predictor and finally smoothed with another median filter run. Another approach for detection of grey level textured defects using linear FIR filters with optimised energy separation was proposed in [78]. Similarly the defect detection [135] is based on a set of optimised filters applied to wavelet sub-bands and tuned for a defect type. Method [44] uses translation invariant 2D RI-Spline wavelets for textile surface inspection. The grey level texture is removed using the wavelet shrinkage approach and defects are subsequently detected by simple thresholding. Contrary to above approaches the presented method uses the multispectral information.

6.2.1 Detection Algorithm

We assume that multispectral textured image can be represented by 3D CAR model which is described in section 3.1.4. Single multispectral pixels are classified as belonging to the defective area based on their corresponding prediction errors. If the prediction error is larger than the adaptive threshold

$$|E\{Y_r \,|\, Y^{(r-1),(s-1)}\} - Y_r| > \frac{2.7}{l} \sum_{i=1}^{l} |E\{Y_{r-i} \,|\, Y^{(r-i-1),(s-i-1)}\} - Y_{r-i}| \tag{6.1}$$

then the pixel r is classified as a detected defect pixel. l in (6.1) is a process history length of the adaptive threshold and the constant 2.7 was found experimentally. The one-step-ahead predictor

$$E\{Y_r \,|\, Y^{(r-1),(s-1)}\} = \hat{\gamma}_{s-1} X_r \tag{6.2}$$

differs from the corresponding predictor (3.26) in using parameters $\hat{\gamma}_{s-1}$ which were learned only in the flawless texture area ($s < r$). The whole algorithm is extremely

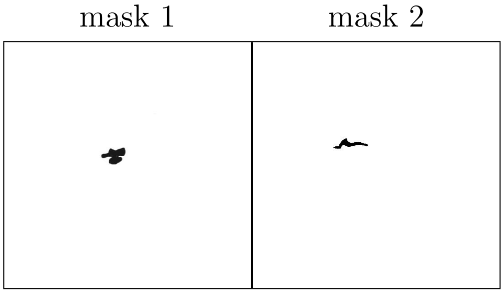


Figure 6.5: Defect masks used in this study for experimental texture mosaics.

fast because the adaptive threshold is updated recursively:

$$|\epsilon_{r+1}| > \frac{2.7}{l} \left[\sum_{i=0}^{l-1} |\epsilon_{r-i}| \right] ,$$

where ϵ_r is the prediction error $\epsilon_r = E\{Y_r \,|\, Y^{(r-1),(s-1)}\} - Y_r$ and $\hat{\gamma}_{s-1}$ is the parametric matrix which is not changing. Hence the algorithm can be easily applied in real time surface quality control.

The presented method was tested on the set of artificially damaged 512×512 colour textile textures, so the ground truth (Fig. 6.5) for every pixel is well known and cannot be influenced by a subjective evaluation. All tested images are colour ($d = 3$) however it is obvious that the method allows any number of spectral bands. The performance of the algorithm is tested using the usual recall ($\mathtt{r}$), precision ($\mathtt{p}$), and the type II ($\mathtt{II}$) error criteria. Let us denote the number of defect pixels n_d, number of pixels interpreted as defect pixels n_i and the number of correctly interpreted defect pixels n_c. The performance criteria are then as follows:

$$\mathtt{r} = \frac{n_c}{n_d} , \quad \mathtt{p} = \frac{n_c}{n_i} , \quad \mathtt{II} = \frac{n_i - n_c}{n - n_d} .$$

6.2.2 Experimental Results

Recall estimates the probability that the reference pixels will be correctly assigned, precision is the defect accuracy estimate relative to the error due to wrong assignment

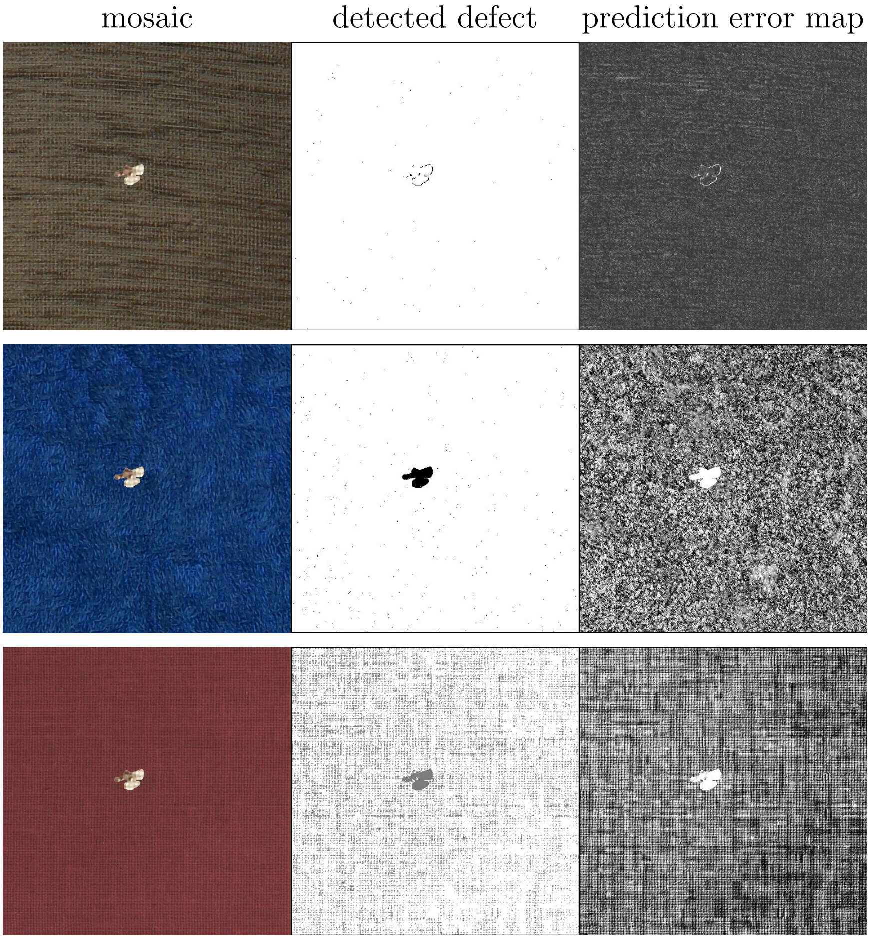


Figure 6.6: Defect detection on texture mosaics (a–c) using the defect mask Fig. 6.5–left

and the type II error estimates the probability of the commission error. All these criteria have range $\langle 0; 1\rangle$. Single defects were simulated by replacing irregular parts of textile textures with different but as similar as possible textile texture. All textures are from our large (more than 1000 high resolution colour textures categorized into 10 thematic classes) colour texture database. All results presented are without any post-processing such as isolated defect pixels filtering to demonstrate basic performance of the presented method. Figs. 6.6, 6.7 exhibit correct defect detection which is also clearly visible on the corresponding prediction maps. Tab. 6.1 presents robust per-

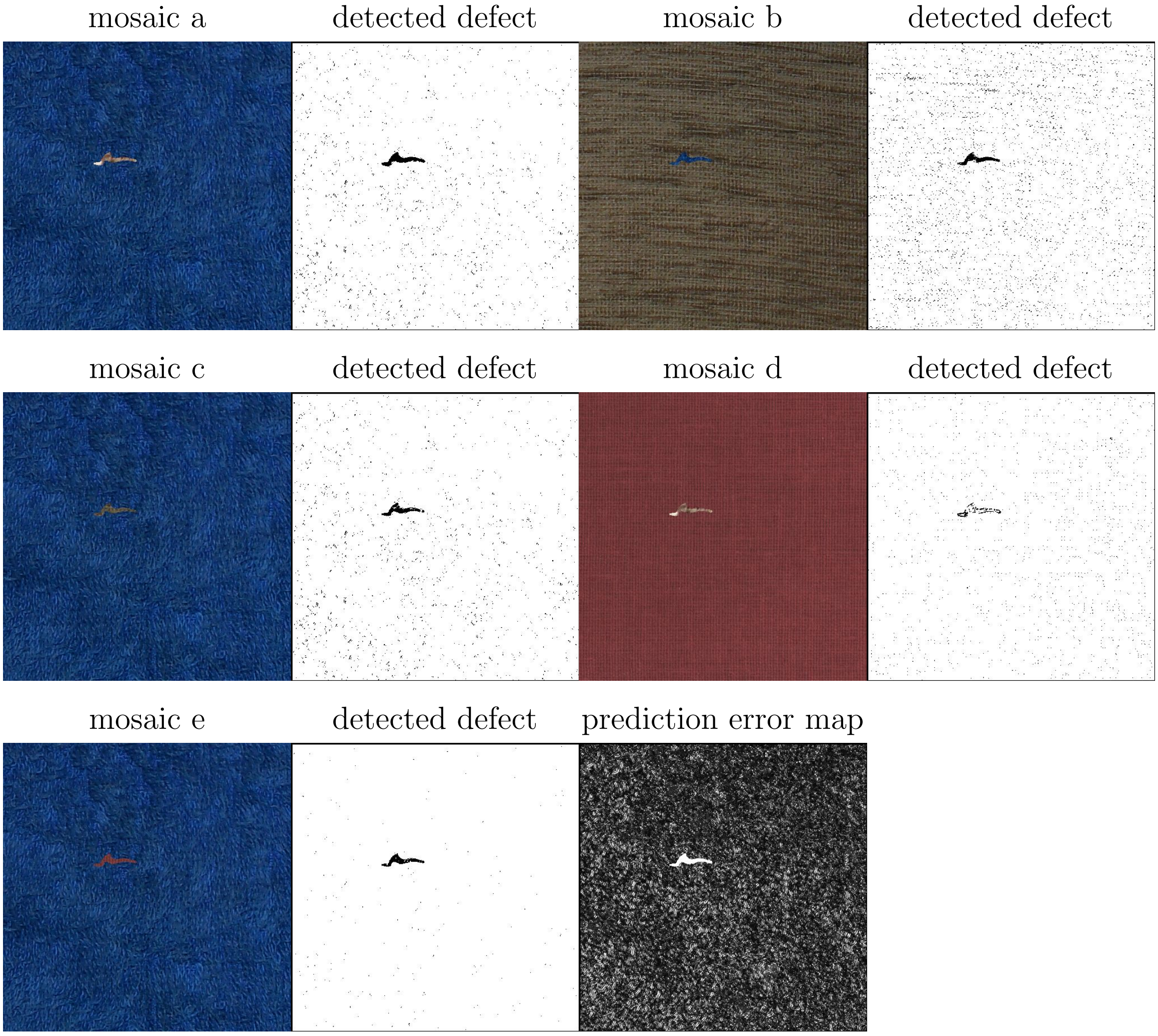


Figure 6.7: Defect detection on texture mosaics (a–e) using the defect mask Fig. 6.5–right

formance with high recall values even for hardly visible defects (Fig. 6.7–b,e). Even for lower recall values (Fig. 6.6–a,c, Fig. 6.7–d) the defect is clearly outlined. Both precision and type II criteria are expectedly low respectively high in failure examples (Fig. 6.8). Finally, the method was successfully evaluated on skin disease treatment progress monitoring application. Fig. 6.9 illustrates a patient with pemphigus vulgaris skin disease and its automatically detected regions which are subsequently compared

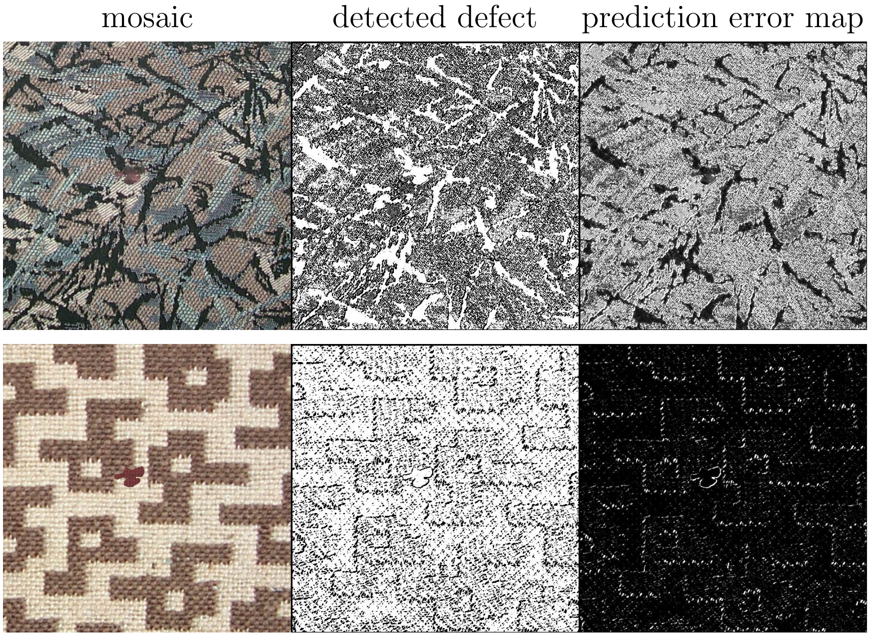


Figure 6.8: Failures on highly structured textures.

mosaic (row)	recall (`r`)	precision (`p`)	type II (`II`) error
Fig. 6.6 – a	0.22	0.01	0.09
Fig. 6.6 – b	1.00	0.70	0.00
Fig. 6.6 – c	0.11	0.62	0.00
Fig. 6.7 – a	1.00	0.70	0.00
Fig. 6.7 – b	0.93	0.10	0.02
Fig. 6.7 – c	0.92	0.19	0.01
Fig. 6.7 – d	0.34	0.11	0.01
Fig. 6.7 – e	0.93	0.71	0.00
Fig. 6.8 – a	0.06	0.00	0.45
Fig. 6.8 – b	0.13	0.00	0.21

Table 6.1: Performance criteria

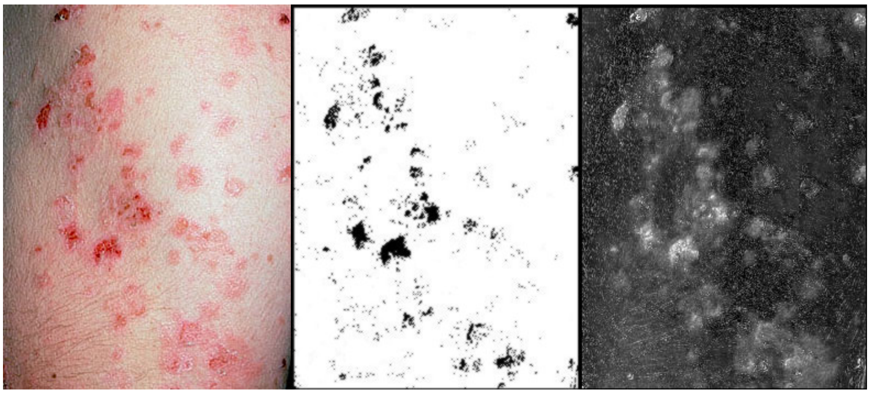


Figure 6.9: Monitoring of the pemphigus vulgaris skin disease progress.

with previous checking to monitor a disease treatment efficiency. Defects were detected using simple models with causal neighbourhoods containing either 3 or 8 sites ($\eta \in \{3, 8\}$) (Tab. 6.2) and adaptive learning on uncorrupted quarter of every texture mosaic. Processing time in Tab. 6.2 is for unoptimized code and can be easily further decreased.

Fig. 6.8 indicates type of highly structured textures which are out of the means of the presented simple probabilistic model. Although even on these examples the defect was correctly detected, the method simultaneously detects also large textile design patterns which cannot be distinguished from the defect. A possible solution is to filter out these design artifacts using prior information such as regularity, size or spectral content.

η	learning $[s]$	detection $[s]$
3	1	7
8	12	15

Table 6.2: Time performance on the HP 9000/785 Unix machine

Most published texture defect detection methods do not use the multispectral information. Our method takes advantage of both multispectral as well as spatial information. The method is simple, extremely fast and robust in comparison with these alternative methods. The presented method results are encouraging, all simulated defects on fine granularity textile textures were correctly localized as well as sick skin patches in real dermatology application. The method will fail on highly structured textures due to limited low frequencies modelling power of the underlying probabilistic model. The presented method can be easily generalized for gradually changing (e.g. illumination, colour, etc.) texture defect detection by exploiting its adaptive learning capabilities.

6.3 Remote Sensing – Aerial Images

Segmentation of remote sensing imagery for various applications (e.g. agriculture, geological survey, military and security, weather forecast, terrain classification, astronomy, the detection of changes and anomalies, etc.) is a challenging task due to huge amounts of data measured by satellite or airborne sensors. Large remote sensing images suffer not only with geometric and radiometric distortions problems but

Figure 6.10: Aerial Lmw 4800 $\times$ 4800 image (left), its detail (middle), and the corresponding unsupervised segmentation (right), respectively.

also with various challenges due to the high heterogeneity both within and across classes. The within class heterogeneity is due to the difference of acquisition process, orientation, and intrinsic appearance [37].

Unsupervised segmentation methods (sections 3.1.2, 3.1.4 and 3.2.2) were modified to be able to handle large aerial images (up to 8000×8000) distributed by the British National Space Centre (BNSC) as a CDROM called "Window On The UK". These aerial images (Fig. 6.10) cover both urban and rural areas of the United Kingdom. The parametric space Θ (3.30) build over large images from this set requires efficient memory handling and distance based region class merging to avoid expensive memory swapping during the segmentation. Segmentation results illustrated on Fig. 6.10–right do not use any prior information except the minimal region area. This parameter can be easily determined from the image resolution and the intended thematic map application.

6.4 Cultural Heritage – Material Analysis

The image processing methods play an important role in very distant application areas such as art restoration [6]. Painting materials research, which helps to choose of the proper materials for restoration, is the field where the system Nephele tries to facilitate the work of restorers. Nephele [7] is a system for processing, description and archiving material analyses used during art restoration.

The aim of the material analyses of painting layers is to identify inorganic and organic compounds using microanalytical methods, and to describe stratigraphy and morphology of layers. Painting materials analysis is described in the form of a report and stored in the database, which could serve as a knowledge base for further restoration cases. For such usage, it is very important to have effective tools to look-up the relevant reports. The visual similarity between images contained in reports can imply that the used technique/materials on the analyzed artwork is the same/similar as in the archived report or that it can point to the same author. Thus, the image-based data retrieval is often used nowadays besides the traditional text-based search.

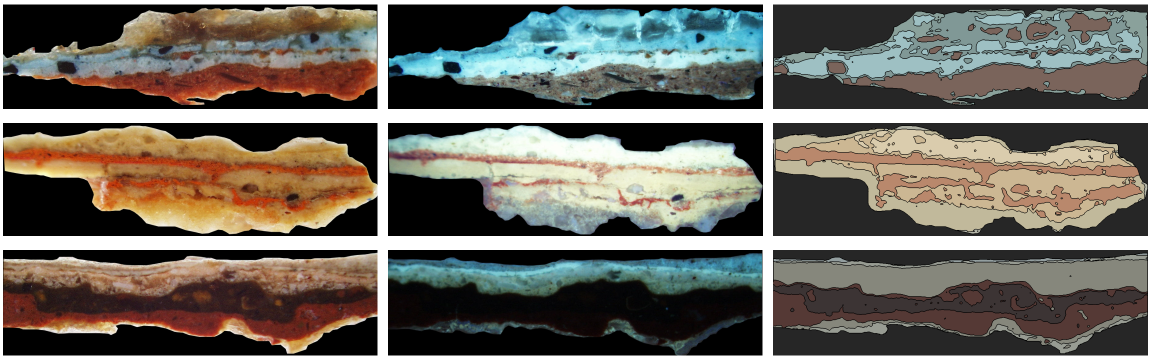


Figure 6.11: Colour layers segmentations of selected paint slices; left column – images in the visible spectrum, middle column – images in the ultraviolet spectrum, and right column – resulting segmentations.

Stratigraphy (learning about layers) is usually studied in the visible spectrum (VS), in the ultraviolet spectrum (UV), and by means of the scanning electron microscopy (SEM). Image segmentation can be used for the colour layer estimation. Input information consists of a set of three RGB channels of VS and three RGB channels of UV specimen images. In figure 6.11 are shown input images in VS, UV spectra and results of colour layers segmentation of selected paint slices. These segmentations are obtained by MW3AR segmentation method (see 3.5) using five different resolutions ($M = 5,\ \iota_1 = 1, \iota_2 = 1.333, \iota_3 = 2, \iota_4 = 2.666, \iota_5 = 4$). It uses a fixed number of components of Gaussian mixture ($K = 5$) in the EM clustering step and the CAR3D neighbourhood with ten neighbours ($\eta = 10$). The postprocessing steps are omitted. More segmentation results can be found in appendix C.

6.5 Virtual Reality Modelling

Image segmentation methods can be used in another cultural heritage application. In virtual reality, which is widely used in the field of cultural heritage nowadays, several issues are solved by image segmentation. For reliable experience from the virtual reality high quality textures are needed. Such realistic textures are very space demanding and in the case of BTF textures the complexity is even more higher. Regardless of increasing internet connection speed and capacity of hard drives texture modelling and compression are essential to solve this problem. And the image segmentation is a significant step in dealing with it. Another issue is obtaining a three-dimensional model. Image segmentation plays an important role in the low-level processing during the construction of the virtual models based on a real world.

This section describes a method for automatic navigation inside a complex virtual scene, demonstrated on a large virtual model of the Department of Modern Art of the National Gallery in Prague. The basic navigation graph structure is constructed semi-automatically and it is subsequently locally changed by the exhibition editor which places new exhibition panels into the building interior and thus locally changes the navigation route structure. The optimal navigation route is automatically generated using graph algorithms and user defined constraints.

6.5.1 Introduction

Virtual or augmented reality systems (VR) are a natural way how to visualize, manipulate and interact with complex digitized information about real world objects in a simple human way. Recent progress in computer technology, range cameras and the corresponding computer graphics and computer vision methods enables to build an ever-growing number of more and more complex VR scenes in various application areas such as 3D games, military or civilian training simulators, architectural models, archeological applications, World Wide Web, digitized cultural heritage sites, etc. These advances in technology allow the shift from text oriented information systems to full 3D graphical ones.

Distributed virtual reality information systems vastly improve the access of citizens, disadvantaged people, or professionals to culture knowledge bases collected in museums or galleries. Many cultural heritage monuments endangered by crowds of visitors or even already closed for public can be accessed through their virtual models. Other monuments stolen, damaged, moved from their natural environment (e.g. Elgin's marbles, Codex Gigas, etc.) can be completed with their original setup. Restoration plans, exhibition planning, manipulating of fragile physical objects, environment changes and many other cultural heritage maintenance problems can be cheaply and safely solved in simulated virtual information systems. Finally some cultural heritage can be preserved only in digital form due to natural disasters or human ignorance. Another obvious application is virtual information and simulation systems for environments too dangerous, hostile, or even inaccessible for humans such as radiation contaminated environment, body interior for microsurgery treatment, etc.

Range and vision sensors are already common and their mutual registration can be done using either standard photogrammetric techniques or an appropriate sensor setup. A range camera is a device which can acquire a 2D raster of depth measurements, as measured from a plane (orthographic) or single point (perspective) on the camera. Such cameras constitute the core part of any virtual model acquisition system together with spectral cameras and the accompanying image processing and modelling software methods.

3D graphical communication and presentation creates a natural environment for users because such a virtual reality information environment simulates the natural surrounding for human beings in which people are accustomed to orient themselves. Single objects are presented in their mutual contextual relations in the simulated realistic time-spatial space and hence offers far richer information than the usual textual, still image or multimedia databases. The VR environment can not only approximate some real world experience but it can provide unique experiences which are either impractical, dangerous or utterly impossible to achieve in real world. A serious problem is the user friendliness of the user interface of a 3D browser. Effective use of this interface requires some experience from the user which is not always the case.

A solution to this problem is automatic path generation that defines a trajectory of the virtual walk through. The parameters of the path (starting point, end point, usage of a staircase etc.) are defined by the user. This solution helps us to solve another critical problem: the performance of a computer to be high enough to generate the virtual walk through in real time. Even experienced users will appreciate a reduction in user's fatigue because in most of the existing VR systems, the user has to input

Figure 6.12: National Gallery in Prague virtual model.

the moving event with a mouse or keyboard continuously if he wants to travel in the virtual environment.

While navigation in real world, i.e. travelling to a specific target location, is often a challenging and not completely understood problem [25], especially in an unknown environment (e.g. city, forest, sea) and many support tools were developed from simple compass to sophisticated GPS (Global Positioning System) based navigators, navigation in VR environment [19, 20, 77, 79, 101, 107, 120] is even more difficult due to many missing real world cues. A major problem for users of virtual environments is maintaining knowledge of their location and orientation while they move through the space because perceptual judgements are biased within a virtual environment. Several tests have shown that users wearing a head mounted display for example underestimate dimensions of space, which might be caused by limited field of view. Several solutions to selected virtual navigation problems were published, e.g. constant navigation velocity [79], collision avoidance, path adjustment, gender factor support [142] or navigation support tools [25] but this problem is still not satisfactorily solved.

The proposed solution for navigation in complex virtual information system is demonstrated on the complex model Fig. 6.12 of the Department of Modern Art of the National Gallery in Prague. In order to test the navigation algorithm on real data we created an accurate and realistic virtual model of this huge gallery building [97]. The gallery has seven exhibition floors and two large exhibition halls in the ground floor. All seven floors of the gallery building interior can be automatically navigated using the method detailed in the following sections and it uses the scaleable model approach proposed in [145].

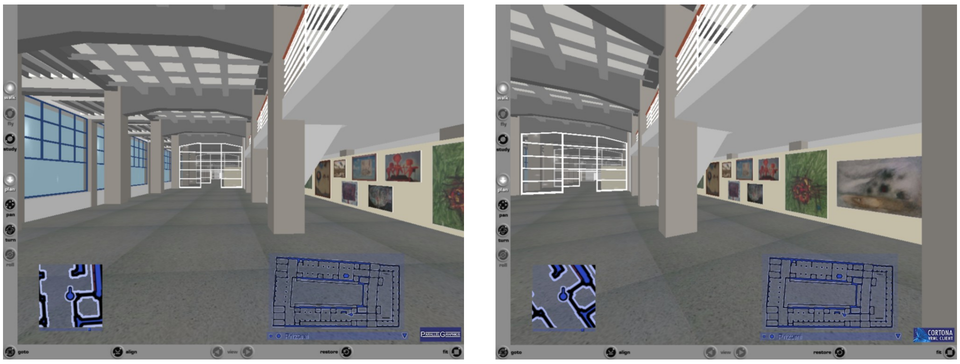


Figure 6.13: Current position visualization.

6.5.2 Actual Position Visualization

Actual position in a complex virtual scene is depicted as the highlighted point in an overlayed transparent map and its local detail (left overlayed map) or building floor plan Fig. 6.13. The map window can be moved by mouse to any appropriate screen location and the detailed map can be scaled. This point is continuously moving as the user or an avatar moves in the virtual scene and the map detail is rotating according to the view angle (compare both maps in Fig. 6.13). If we leave a building floor for another one, the floor plan is switched accordingly to the actual one. Each plan is labelled with the corresponding floor number or label (i.e. ground floor – Přízemí).

6.5.3 Preset Routes

Manual creation of virtual reality models of real world scenes and navigation routes inside them is tedious and error-prone as the scene complexity increases and automation may substantially reduce the laboriousness of the whole process. Possible routes are determined to large extent by the building designed and this information can be exploited for possible navigation network setting if we are prepared to compromise full generality of possible routes. For example we assume that a visitor will never walk closer than half meter from the walls, enter each exit in its center, larger spaces are covered with walking loops with minimal diameter one meter, etc. Each

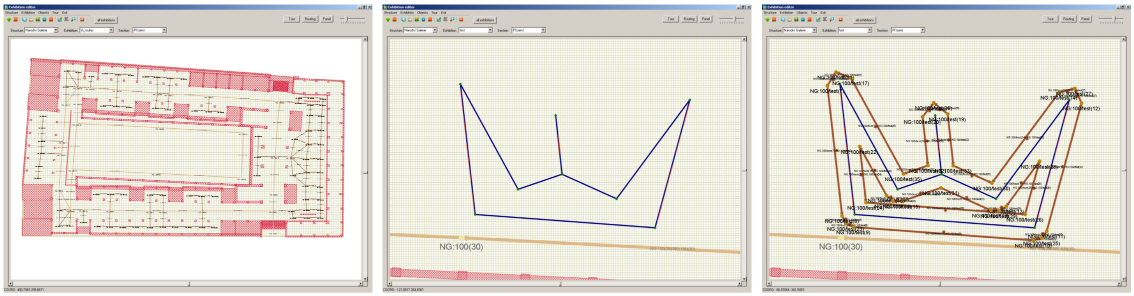


Figure 6.14: Generated gallery ground floor preset routes (left), user generated exhibition panels floor plan (middle) and the corresponding automatically generated navigation subgraph for this exhibition (right).

floor plan is than supplemented with a preset routes graph structure given by the basic building structure. Single corridors, lifts, staircases are represented as graph edges, while doors, branchings or turning points are graph vertices. This prior graph structure which represents initialization of navigation routes can be generated semi-automatically based on the floor plans. Narrow corridors have single graph edges while wider corridors or halls can have even several graph loops (see Fig. 6.14–left loop booked in the middle hall). This automatically proposed graph structure (primary graph) can be interactively edited using the exhibition editor described in the following section. Superfluous edges or vertices can be removed while new edges and vertices can be added. Single edges or vertices can be also shifted to other positions. Vertices can be also supplemented with additional attributes such as emergency exit, lift, staircase, door, etc.

6.5.4 Exhibition Editor

Our Virtual National Gallery allows to interactively build virtual exhibitions using our exhibition editor Fig. 6.15. This editor was devised for the National Gallery exhibition architects to support and speed up their exhibitions proposals. The editor loads requested floor plan and allows to insert single exhibition panels Fig. 6.15–right and to specify their parameters such as single dimensions, colour, covering material, etc.

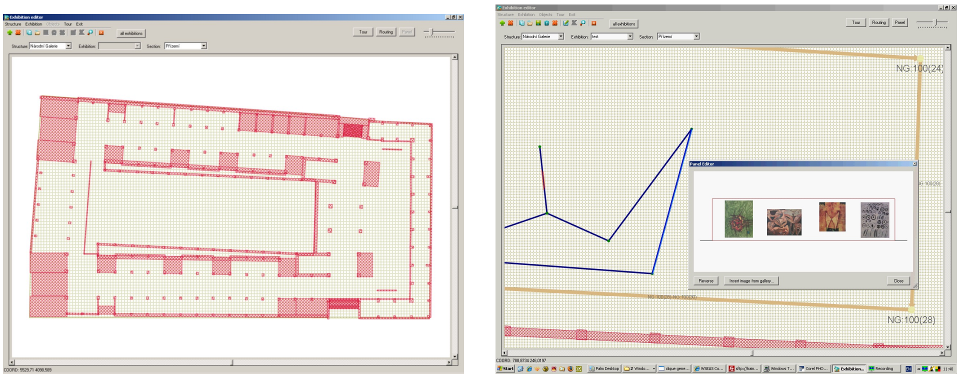


Figure 6.15: Exhibition editor with floor map (left) and single panel editor window (right).

Single paintings from the gallery are subsequently set out on these exhibition panels and other supplementary data can be attached e.g. information about a painter in the corresponding pop-up window. When the exhibition editing is ready, it is exported into the VRML building model and can be immediately checked in the browser. However, these newly inserted panels (Fig. 6.14–middle) change the corresponding part of the underlying navigation graph. The first step is automatic generation of a subgraph around these new panels Fig. 6.14–right. This new subgraph is inserted into the preset primary graph with higher priority than has the corresponding primary subgraph. The primary graph edges and vertices which are overlaid by this new subgraph are temporarily disabled (Fig. 6.16–right dash-and-dot brown edge). If the subgraph does not overlay any part of the primary graph (e.g. a new exhibition in the previously empty hall far from default primary graph hall paths) it is automatically connected to the nearest primary graph vertices. When the exhibition is later changed or removed the primary subgraph can be restored to its original shape (Fig. 6.16–left).

6.5.5 Optimal Path Search

The solution to this problem is to generate a sort of movie that represents the virtual walk-through in the scene. Such a movie can be played forwards and backwards

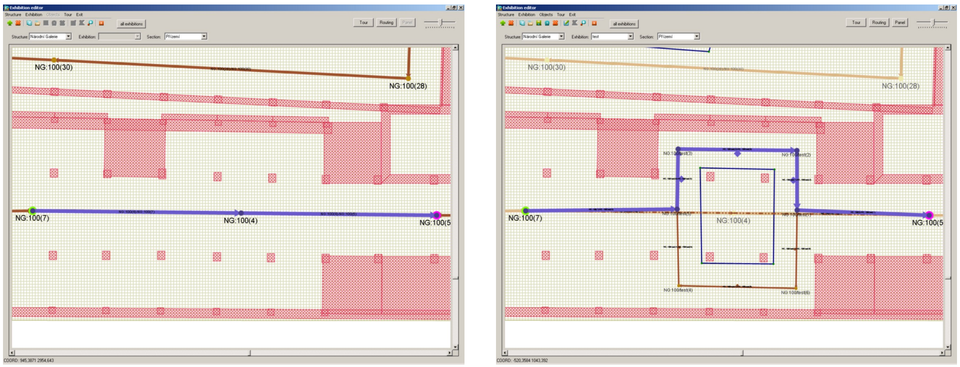


Figure 6.16: Navigation route detail (left – blue edge) and its modification by the editor (right).

thus providing necessary information about every part of the trajectory the user goes virtually through. Generation of a path from parameters given by the user is done automatically in a module that considers the ground plan of a 3D scene as a labelled graph. The labels represent various kinds of information like accessibility of some location from the point of view of handicapped persons etc. A single edge attribute is also its physical length, thus it is possible to estimate real time needed to walk a specified route in the real National Gallery Prague palace as well as time needed for an exhibition sightseeing tour.

The process of the path finding is in principle finding the optimal path in a given graph. Motion planning has been studied for several decades and many motion planning algorithms were published, however VR path planing is simpler than robot motion planning [71] in unknown dynamic environment and simple graph path methods can be used.

The shortest path is found using the Dijkstra algorithm [29] and it represents the required navigation route through the gallery building where we assume only static obstacles and environment changes restricted to the exhibition editor. Such requests can be emergency evacuation assessment, routes for handicapped visitors, educative thematic routes, time limited gallery visit proposals or simply visitor's information support.

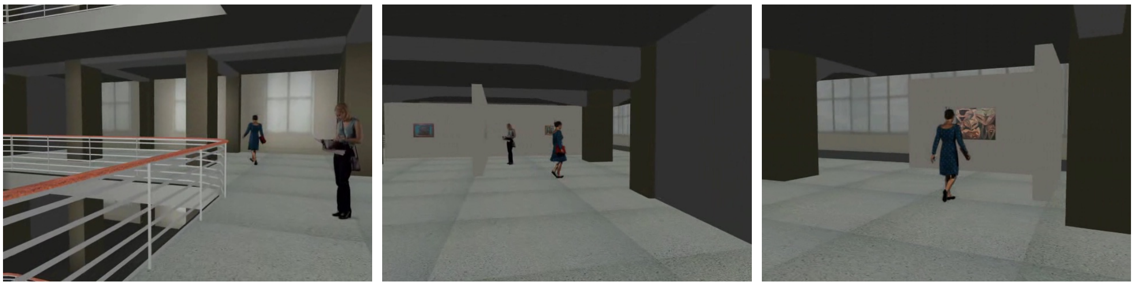


Figure 6.17: Animated navigation route using avatars.

This navigation route is subsequently used for generation of a movie that represents the virtual walkthrough. This walkthrough can be demonstrated using avatars Fig. 6.17 or simulating the eye view of a visitor Fig. 6.18. The user interface has only few features. They do not require some specific knowledge (forward, backward, stop etc.). This fact allows the use of the navigational system even for novice users. A visitor can watch not only animated thematic visits to his or her selected artistic subjects, possible from home over internet, but can also print a map with the proposed personalized route.

6.5.6 Experimental Results

The virtual National Gallery in Prague Department of Modern Art model was partly automatically acquired using our setup for automatic acquisition of virtual reality models using a laser scanner Konica Minolta Vivid 9i and our modelling software. All parts of this system [49, 66] are fully functional and were tested with satisfactory results on gallery small real objects.

Unfortunately, this range camera is not capable to measure large building structures and thus we were not able to acquire range data from the gallery building itself. Single architectural shapes in the model were created instead using building blueprints and acquired photographs. Virtual reality systems require object surfaces covered with realistic nature-like colour textures to enhance realism in virtual scenes. The model surface materials are represented by synthetic textures generated using our multi-

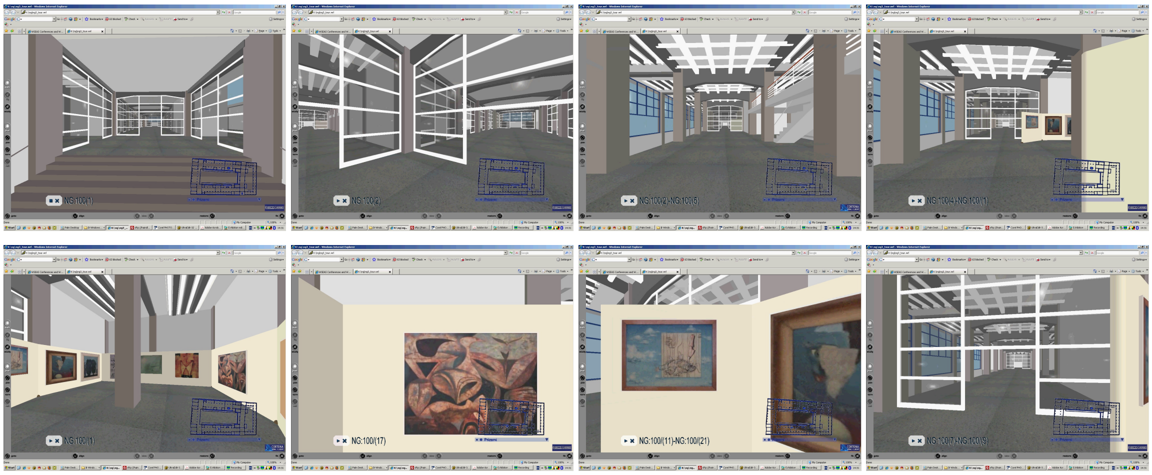


Figure 6.18: Navigation frames (0, 5, 10, 24 top; 26, 32, 39, 43 bottom).

scale Markov random field based methods [52, 53]. The texture Markov random field based model consists of a set of Gaussian Markov random field submodels for single orthogonal mono-spectral single-resolution texture factors. Parameters of the Markov random field submodels are estimated and subsequently used for given factors synthesis. The resulting synthetic colour texture is composed from these mono-spectral single-resolution factors after corresponding inverse transformations. Although these colour texture models are slightly spectrally compromised due to this spectral decorrelation transformation, the appearance of colour synthetic textures in the gallery model is very good and nearly visually indiscernible from their natural counterparts.

Figs. 6.12, 6.13, 6.17, 6.18 show different images from the Virtual National Gallery model. The gallery model is created in the VRML2 language. Single building floors are separated VRML scenes which are automatically loaded whenever avatars or users are moving from one scene into another using, for example, virtual lifts or model staircases.

Fig. 6.18 presents selected eight frames from the automatically generated route to reach a newly installed small exhibition on three panels. The given task was to find the shortest route from the gallery entrance to the selected exhibition part (three panels with eight paintings) for a regular visitor, survey these paintings and return to the exit. The system found automatically the correct route, its length in meters, estimated sightseeing time and animated this tour in the realistic setting of the virtual gallery model. It is also possible to print the floor plan with the highlighted suggested route. If we require disabled person (or wheel chair) constraint, the generated route will avoid staircases (Fig. 6.18 top row leftmost image) in exchange for the lift with a slightly longer route.

6.5.7 Conclusion

The proposed solution for the virtual information system construction is demonstrated on the model of the Department of Modern Art of the National Gallery. This collection of images, drawings and statues from the period of 20th century is located in a functionalist building in Prague. In order to test the navigation algorithm on real data we modelled manually and semi-automatically the interior and exterior of the whole gallery palace. Automatic acquisition of virtual models from registered range and colour real image data and automatic generation of navigation routes in the virtual scene is possible combining novel efficient and robust methods indicated in the article. Very complex scenes with large non-planar faced objects, still require human feedback and corrections. However even in this case the model acquisition procedure significantly simplifies a virtual model building task.

Although recent technological advances already enable automatic or at least semi-automatic construction of complex distributed virtual models, further research is still needed to enhance the physical look and feel of resulting models together with their performance and storage requirements. The VRML2 language has many functional restrictions for example missing support for the most advanced material representation – the bidirectional texture function and some better distributed virtual reality modelling language is clearly required. Current state-of-art of image analysis has its limitations as well in reliable image and range segmentation of complex or inhomogeneously lighted scenes.


Conclusion

Chapter 7

Several novel unsupervised multispectral image segmentation methods based on the underlaying random field texture models (GMRF, 2D/3D CAR) were developed. These segmenters use efficient data representations that allow an analytical solutions and thus the segmentation algorithm is much faster in comparison to methods based on MCMC. All segmenters were extensively compared with the alternative state-of-the-art segmenters with very good results. The MW3AR segmenter scored as one of the best available. The cluster validation problem was solved by a modified EM algorithm. Two multiple resolution segmenters were designed as a combination of a set of single segmenters. To tackle a realistic variable lighting in images, the illumination invariant features were derived and the illumination invariant segmenter was developed.

For the proper evaluation of segmentation results and ranking of algorithms, a unique web-based texture segmentation benchmark was proposed and implemented. It was used for comprehensive comparisons of results of developed algorithms with ten different state-of-the-art segmentation methods. Finally, the proposed methods were validated through use in various applications from a range of different fields.

In the medical imaging field, they were used for automatic segmentation of mammograms into regions of interest. Proposed solutions based on the random field model could also be used in automated inspection systems. Developed segmenters work on aerial images up to a size of 8000×8000 pixels, which are standard in the remote sensing field. The algorithm can also be used in areas related to digital cultural heritage. At last, an advantage of our methods is the need to tune just a few application dependent parameters.

7.1 Further Research

In spite of all the results achieved in the thesis, many areas still offer space for future development. Several suggested areas of further research, plans, and possible applications are listed below.

(a) Improvements in texture representation can be done using more complex MRF. Possible optimization to represent an image by using a set of competing random field models.

(b) Advancement in the area of cluster validation leading toward more reliable estimation of the number of regions.

(c) Developing better combination of several segmenters (multi-segmenter approach) allowing the capture of textures with higher complexity.

(d) Incorporating semi-supervised learning for remote sensing solving incomplete texture training set.

(e) Enhancement of the segmentation benchmark and validation of the coherence between the benchmark and real image scene segmentation results based on statistical comparisons.

(f) Extension of the segmentation method from still image segmentation to video segmentation.

(g) Implementing proposed methods more efficiently using parallelization.

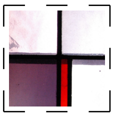
Applications

(A) Content-based image retrieval

(B) Medical image analysis – retina segmentation

(C) Range images segmentation

Image segmentation is a fundamental part in low level computer vision processing. It has a crucial influence on the subsequent higher level visual scene interpretation for a wide range of applications. Unsupervised image segmentation is an ill-defined problem and thus cannot be optimally solved in general. Therefore, a reasonable way is to advance promising existing methods, use concise and better data models and exploit all available information to develop an efficient and robust segmenter.

Colour Benchmark Results

Appendix A

In this appendix can be found *Colour* benchmark (described in section 4.2) texture mosaics, ground truths, segmentation results and performance curves for fourteen different algorithms – Blobworld [12], JSEG [27], EDISON [21], EGBIS [39], GMRF+EM [54], AR3D+EM [56], AR3D+EM multi [57], TFR [124], TFR/KLD [125], GSRM supervised KL area-weighted [11], SWA [130], HGS E [68], TEX-ROI-SEG [31], MW3AR [61]. Further details can be found in section 5.2.

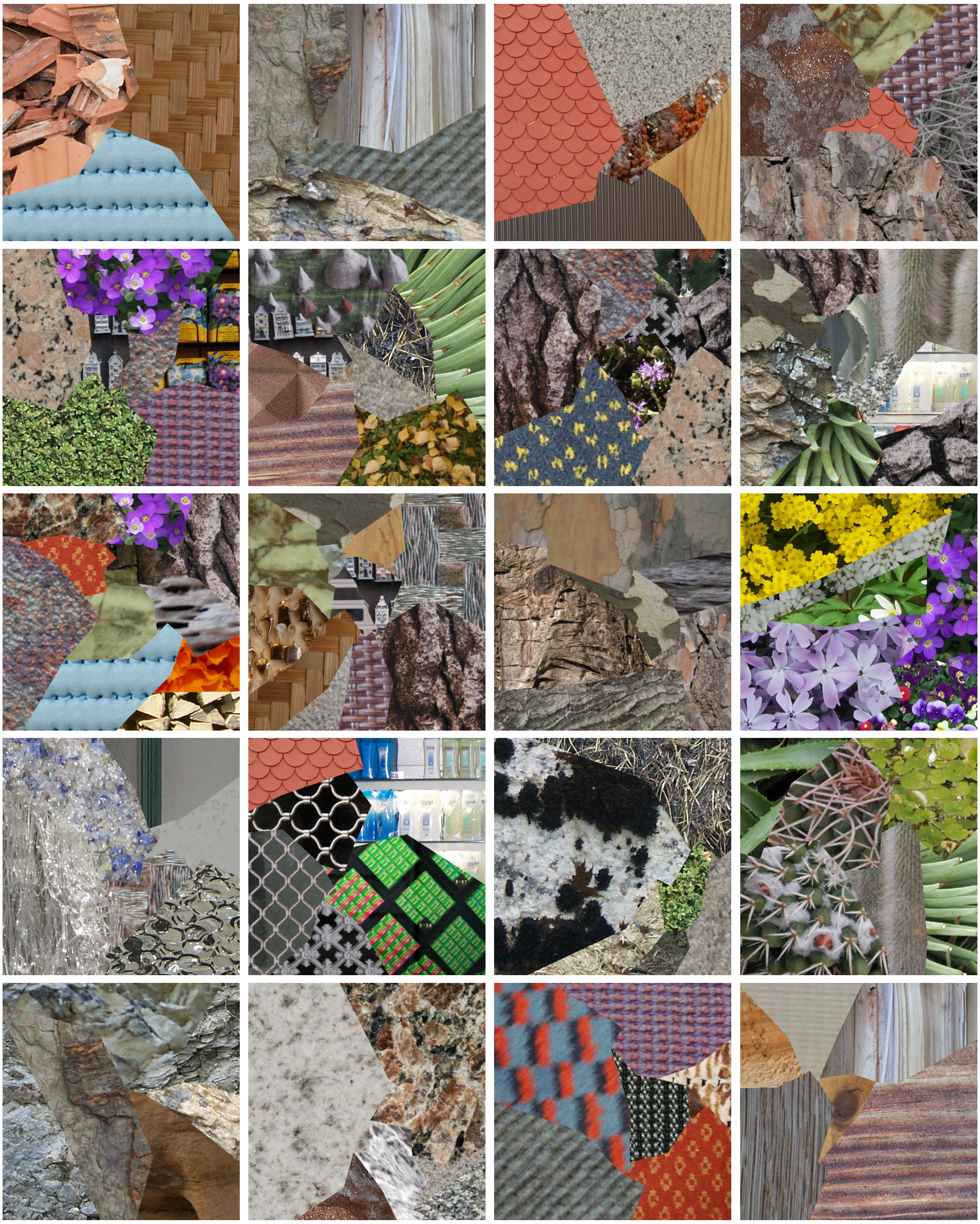


Figure A.1: *Colour* benchmark – texture mosaics.

Figure A.2: *Colour* benchmark – ground truths.

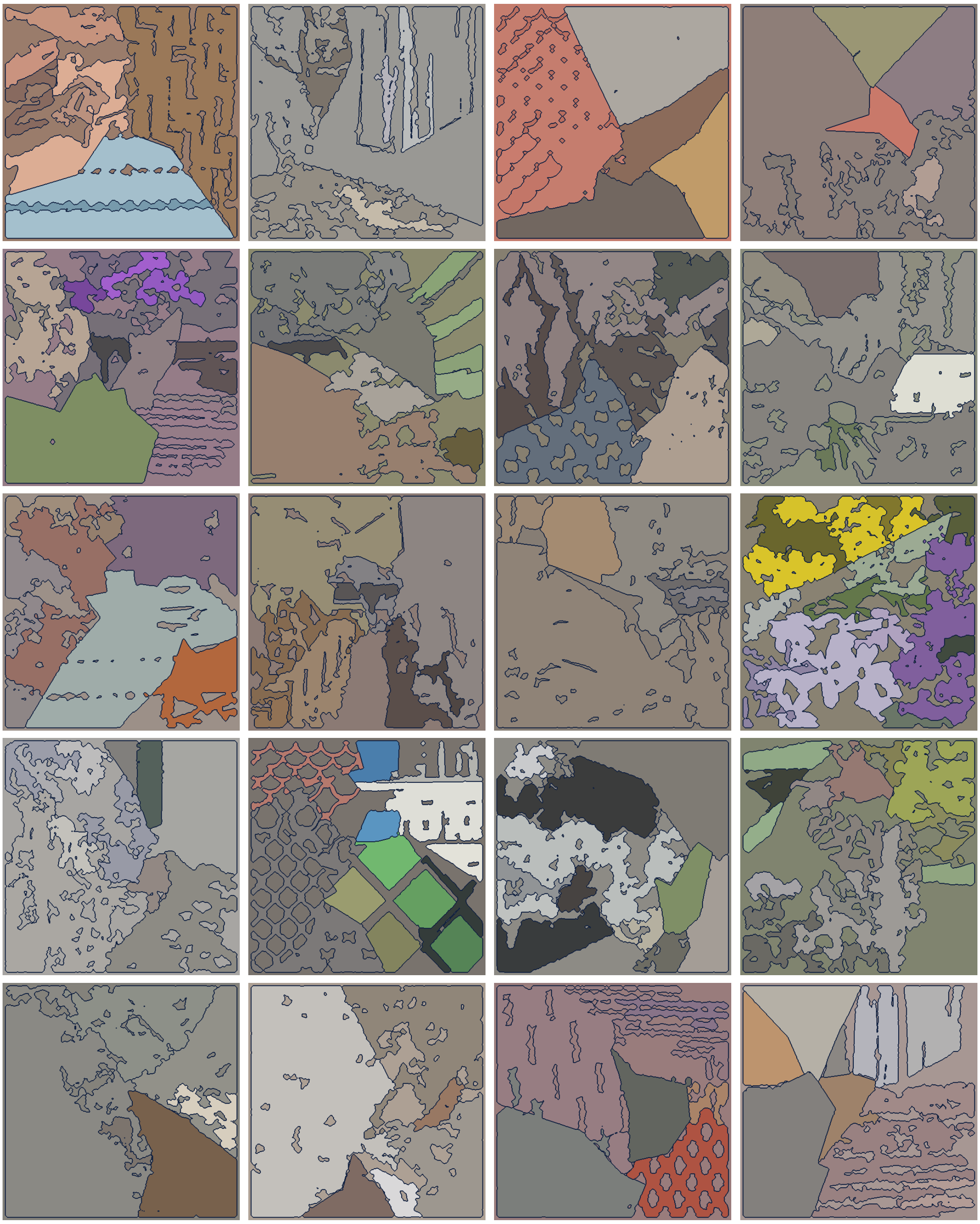


Figure A.3: *Colour* benchmark – segmentation results – Blobworld.

Figure A.4: *Colour* benchmark – segmentation results – JSEG.

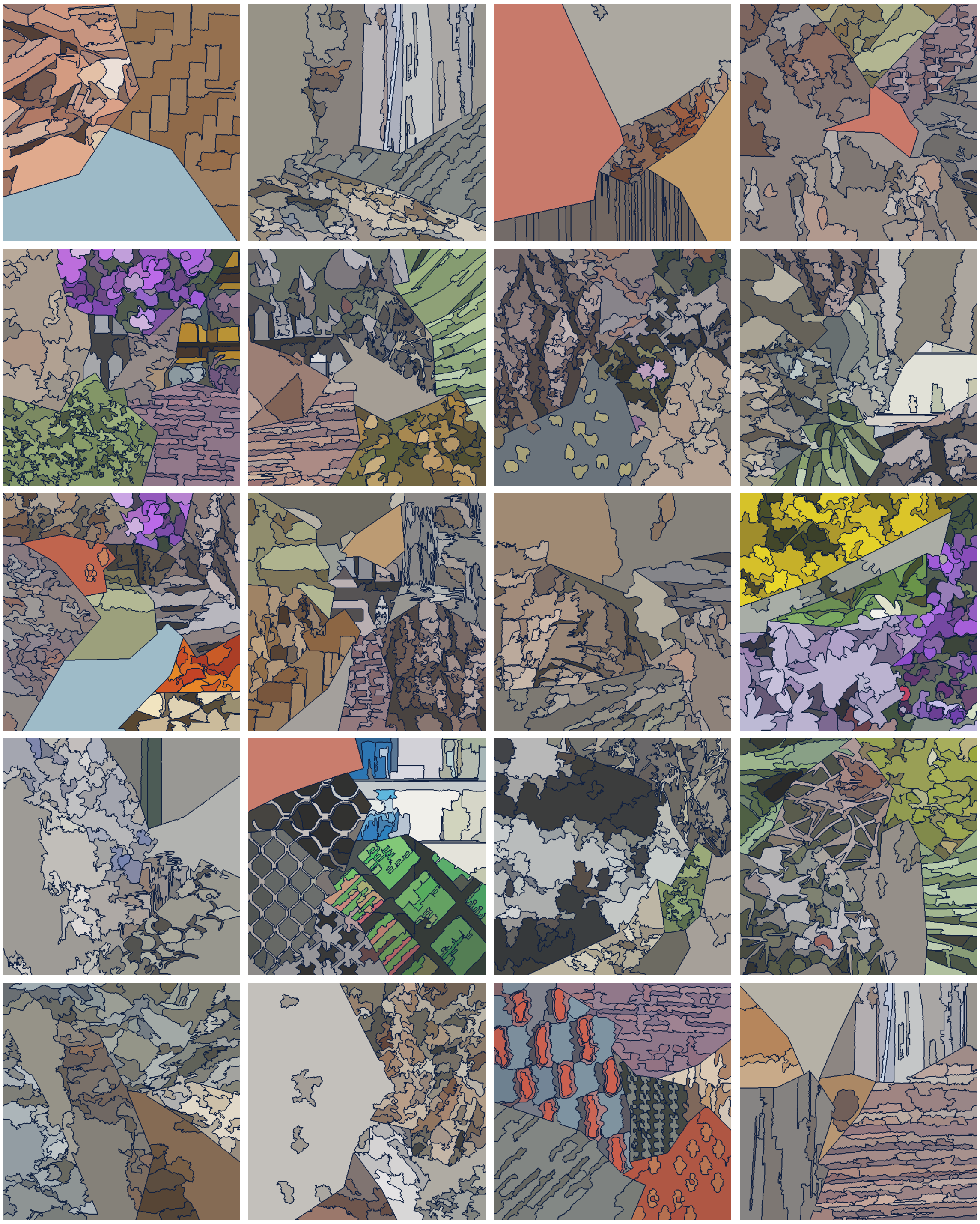


Figure A.5: *Colour* benchmark – segmentation results – EDISON.

Figure A.6: *Colour* benchmark – segmentation results – EGBIS.

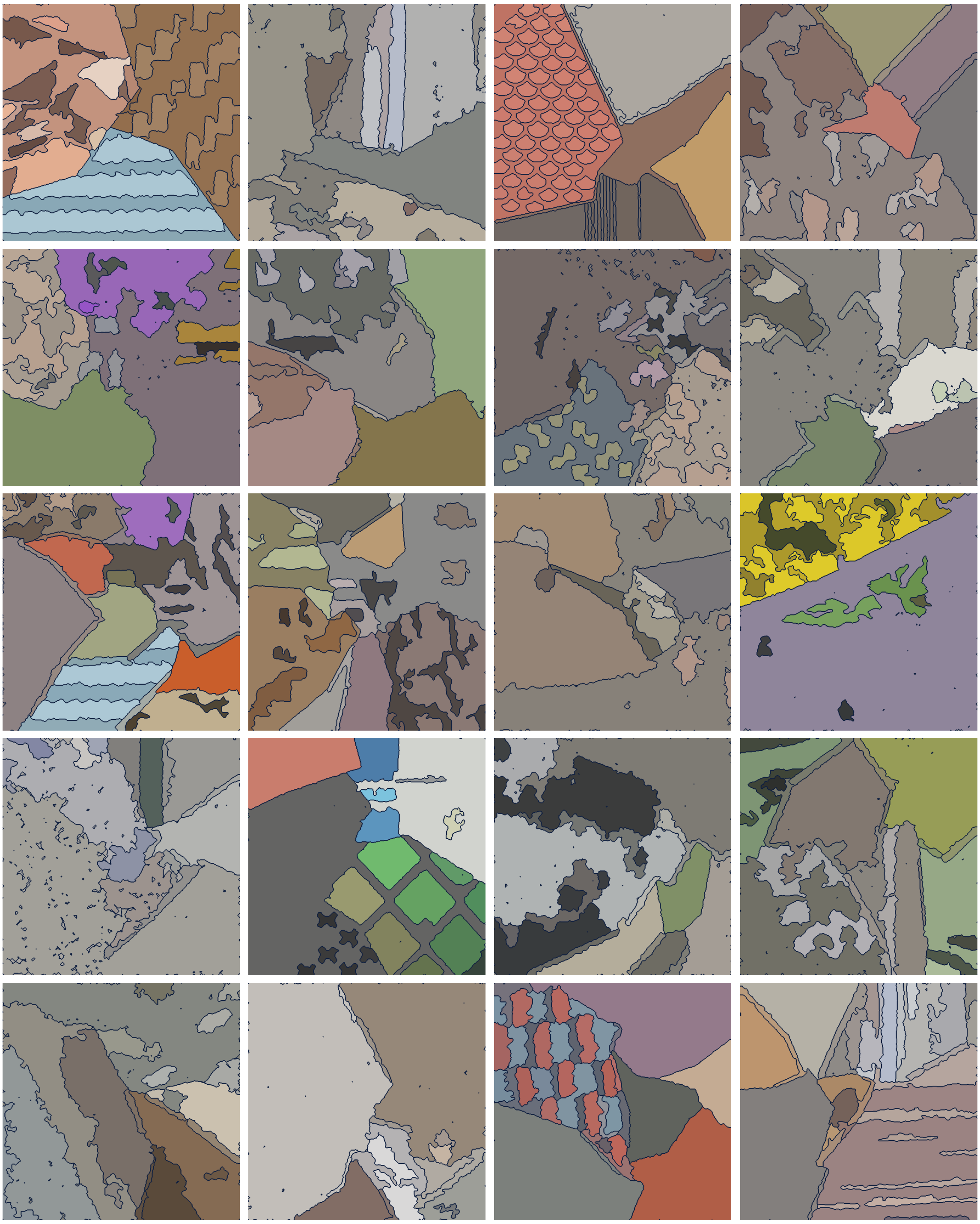


Figure A.7: *Colour* benchmark – segmentation results – GMRF+EM.

Figure A.8: *Colour* benchmark – segmentation results – AR3D+EM.

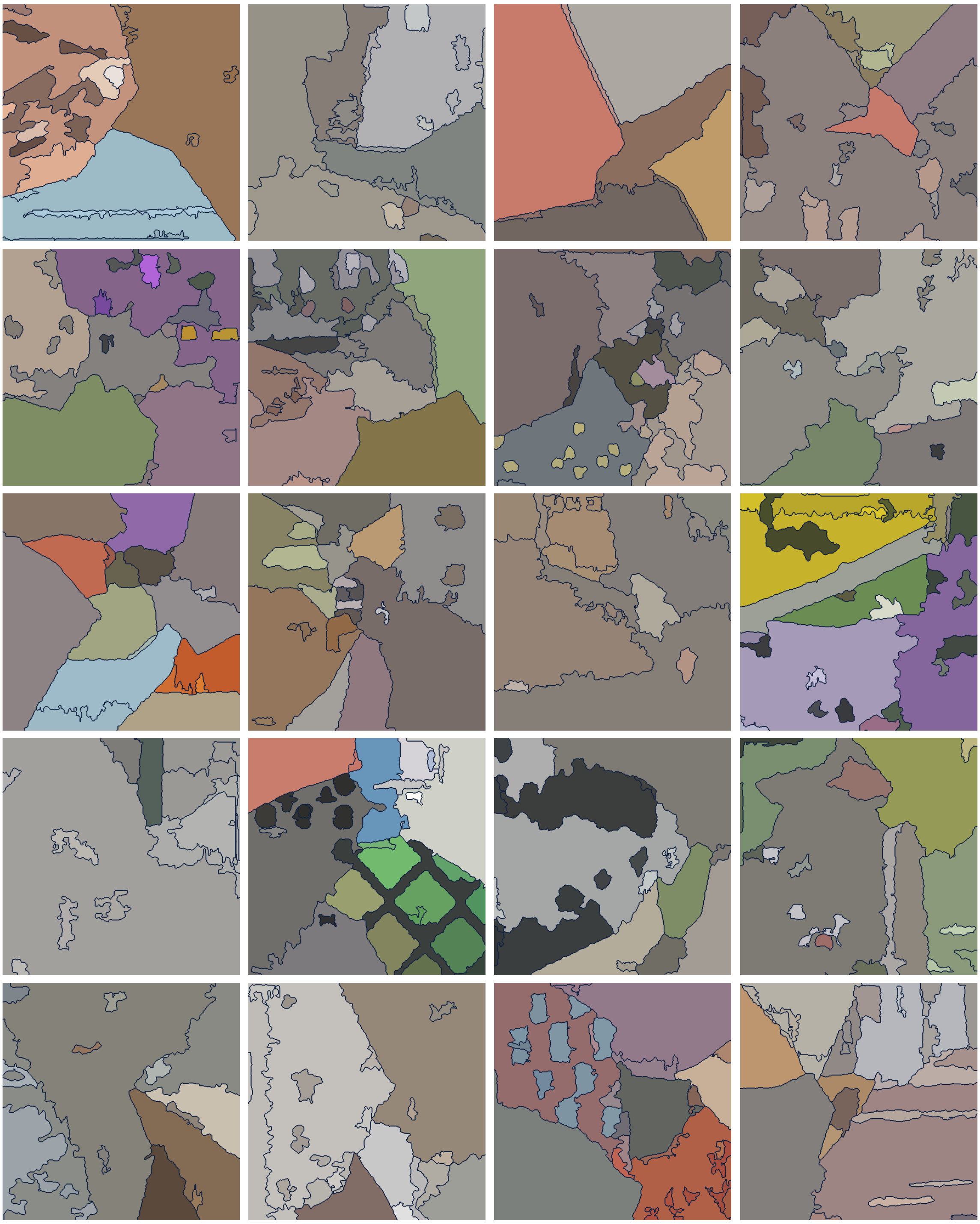


Figure A.9: *Colour* benchmark – segmentation results – AR3D+EM multi.

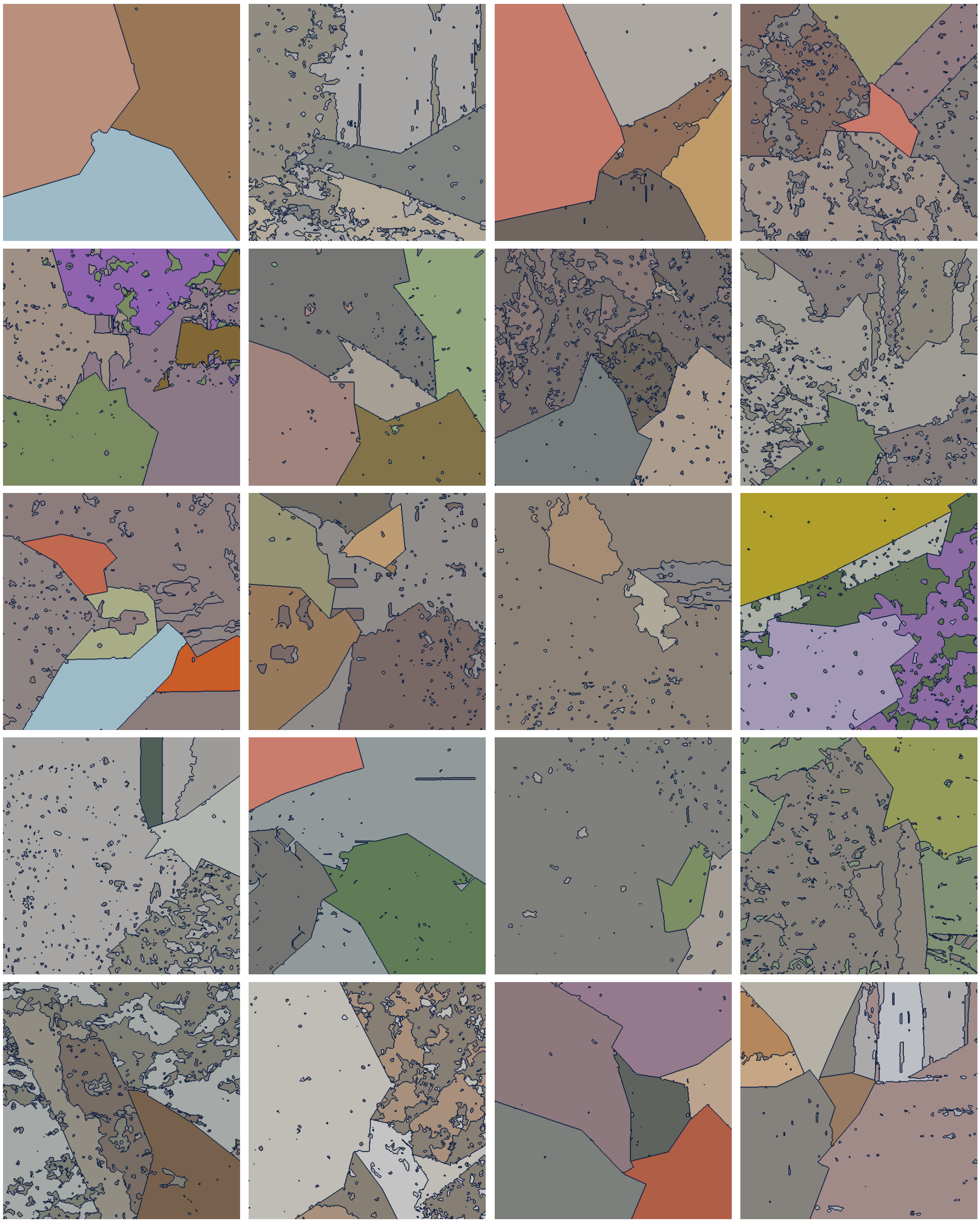


Figure A.10: *Colour* benchmark – segmentation results – TFR.

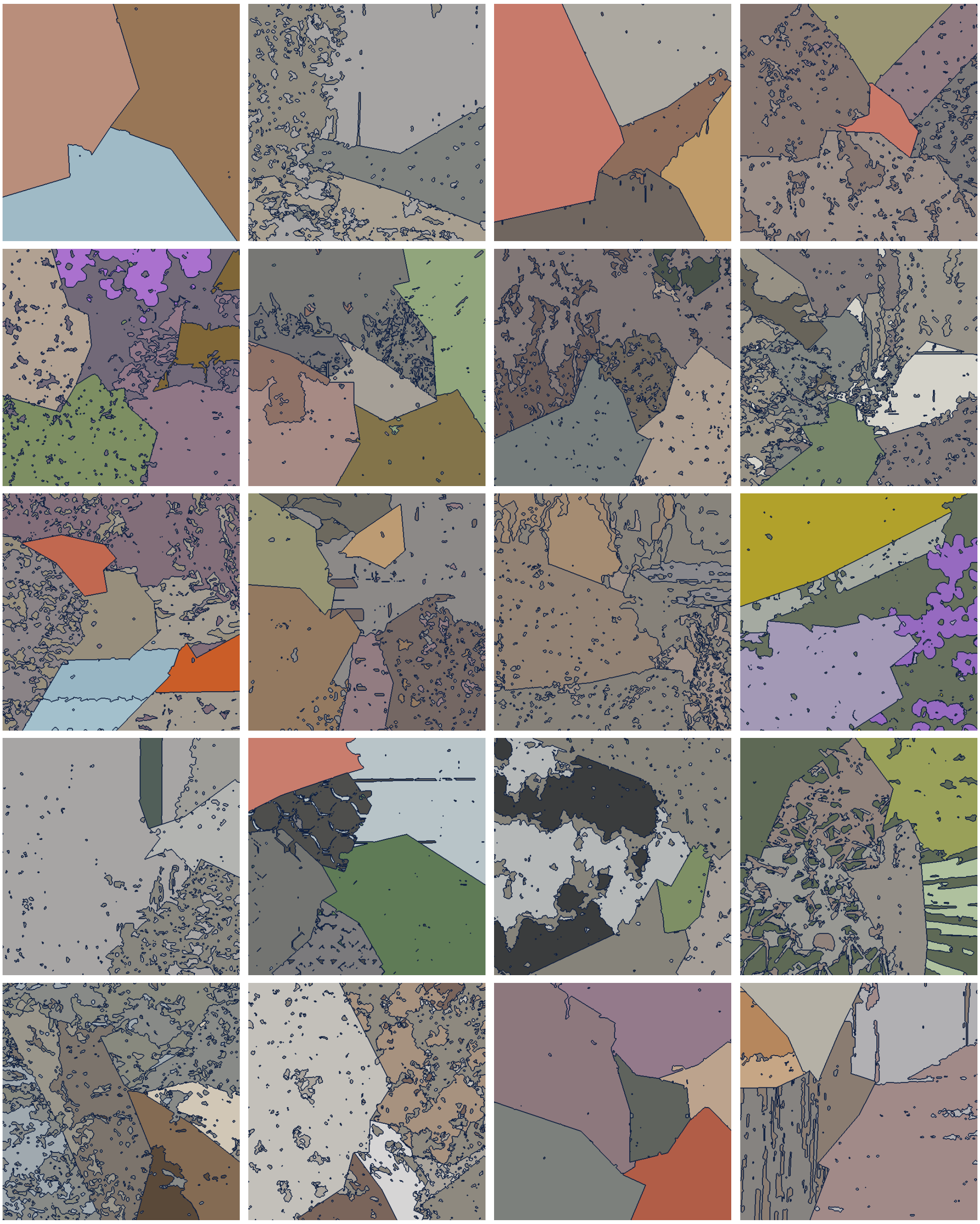


Figure A.11: *Colour* benchmark – segmentation results – TFR/KLD.

Figure A.12: *Colour* benchmark – segmentation results – GSRM sup. (KL a-w).

Figure A.13: *Colour* benchmark – segmentation results – SWA.

Figure A.14: *Colour* benchmark – segmentation results – HGS (E).

Figure A.15: *Colour* benchmark – segmentation results – TEX-ROI-SEG.

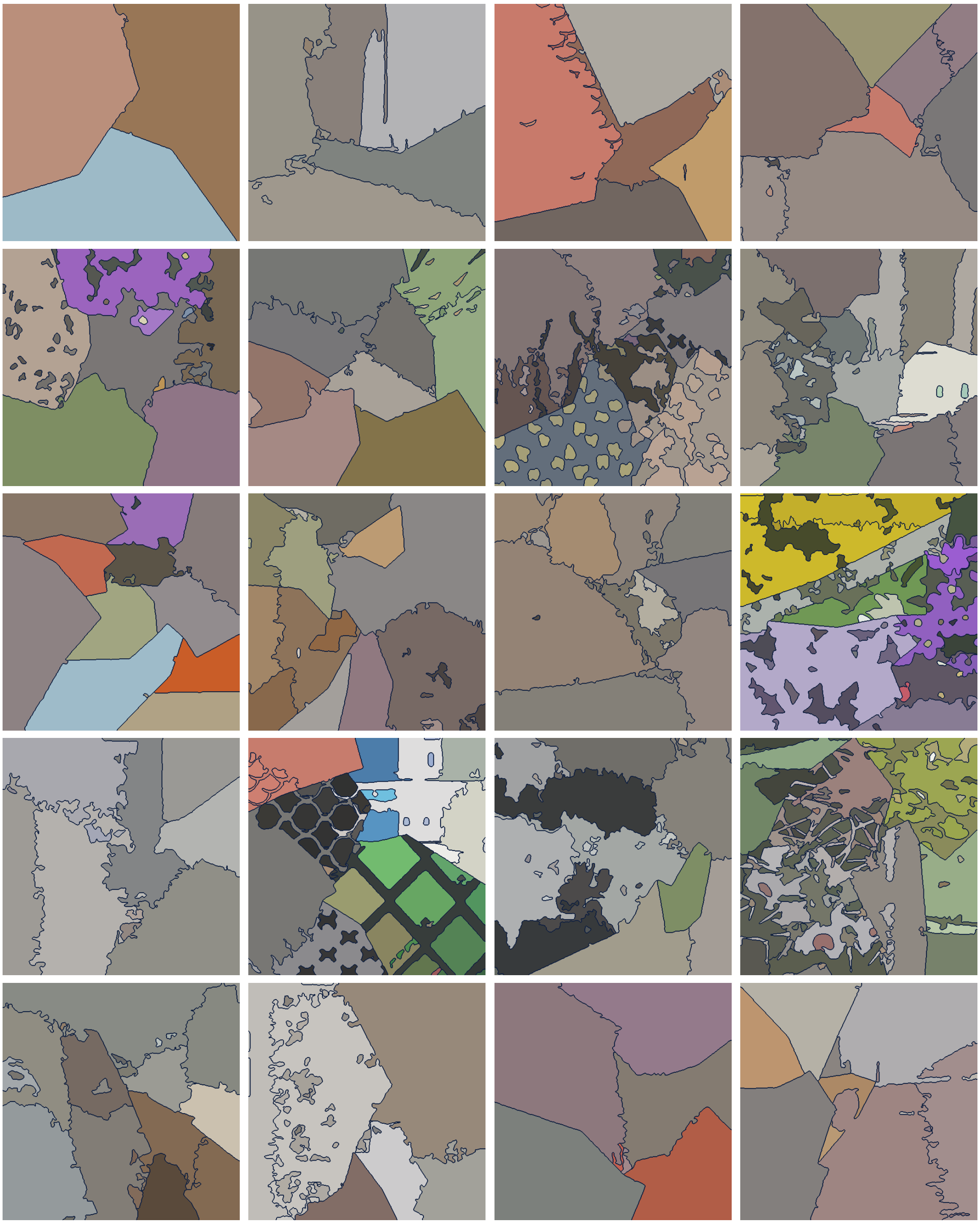


Figure A.16: *Colour* benchmark – segmentation results – MW3AR.

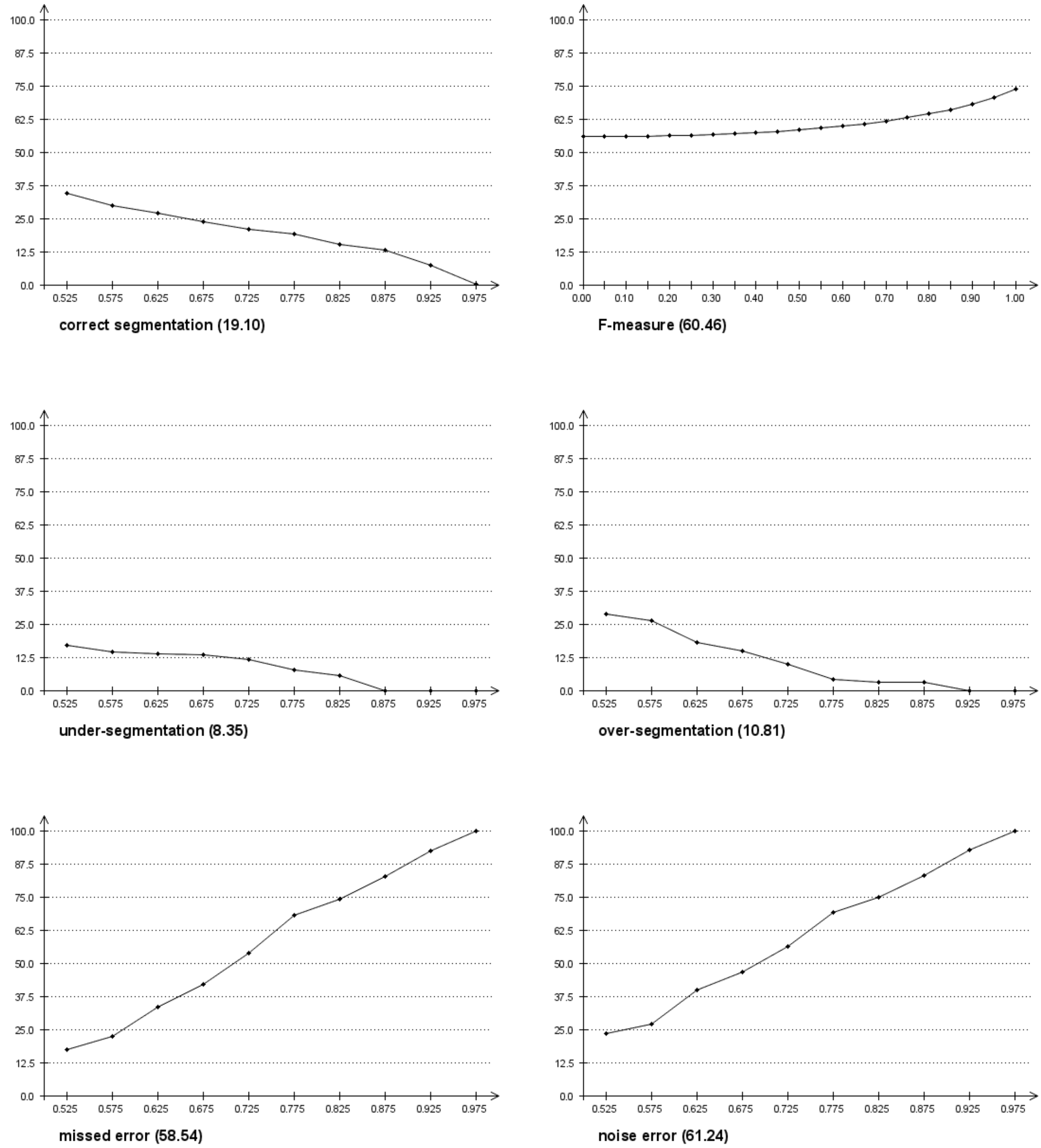


Figure A.17: *Colour* benchmark – performance curves – Blobworld.

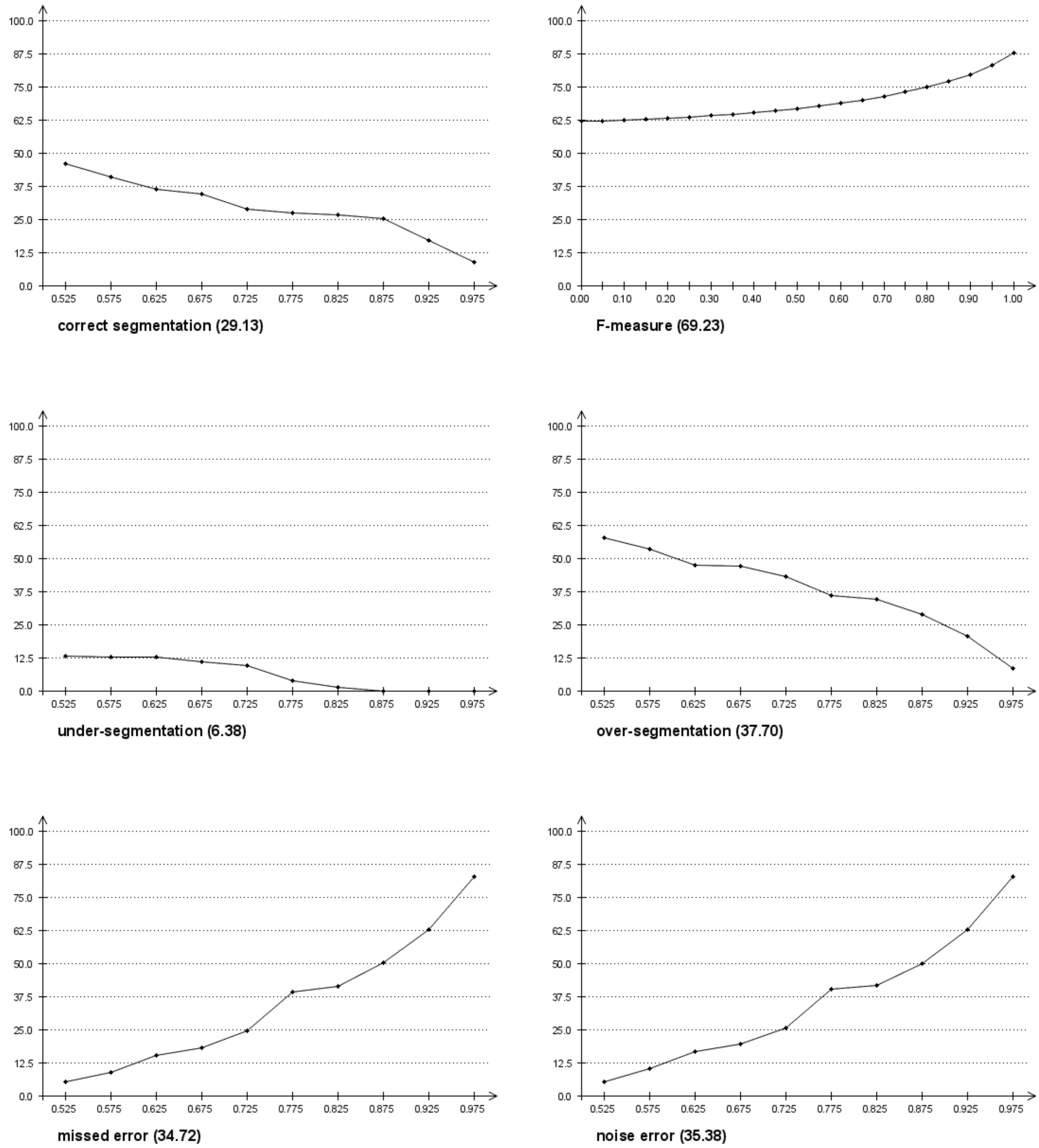


Figure A.18: *Colour* benchmark – performance curves – JSEG.

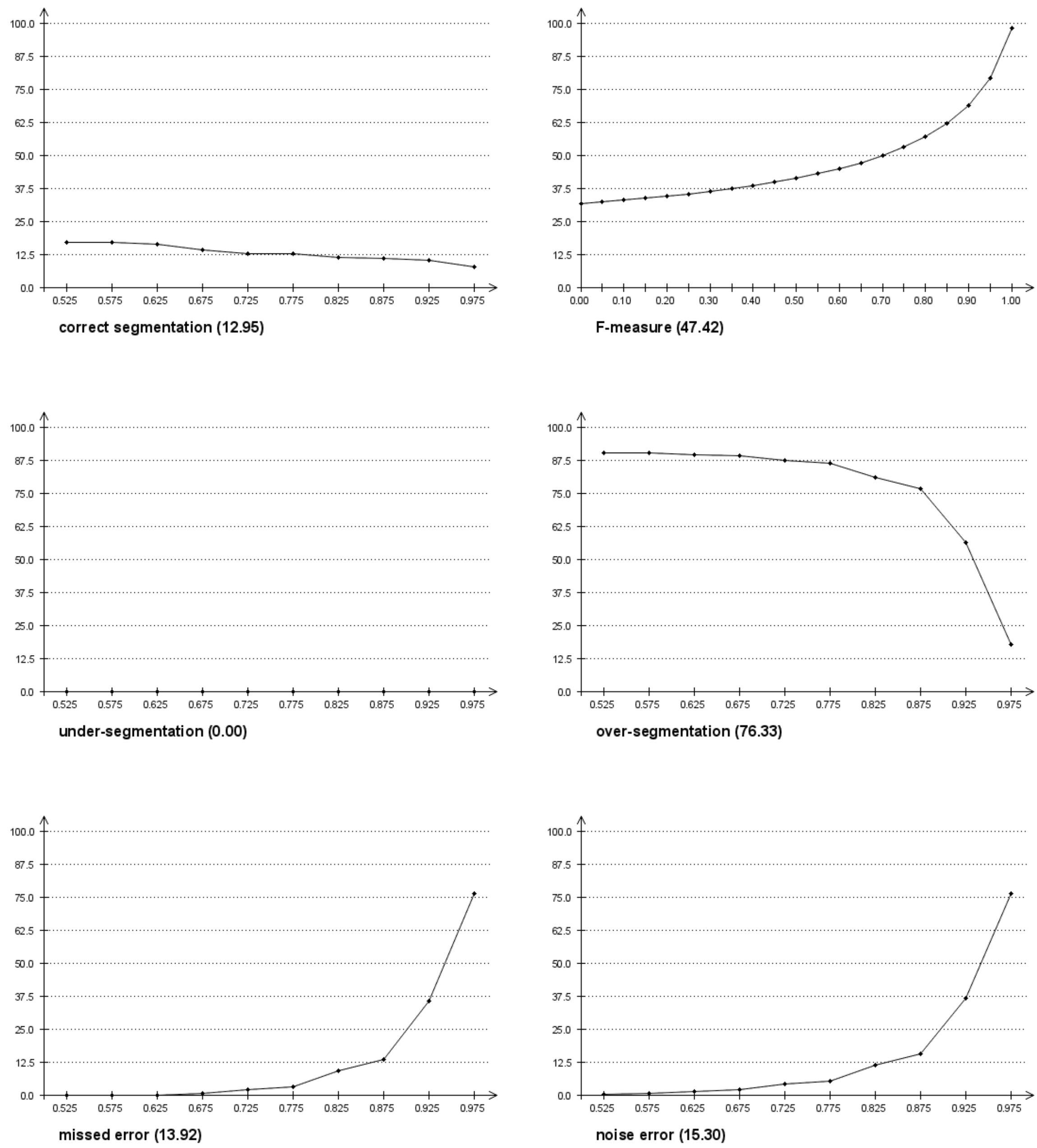


Figure A.19: *Colour* benchmark – performance curves – EDISON.

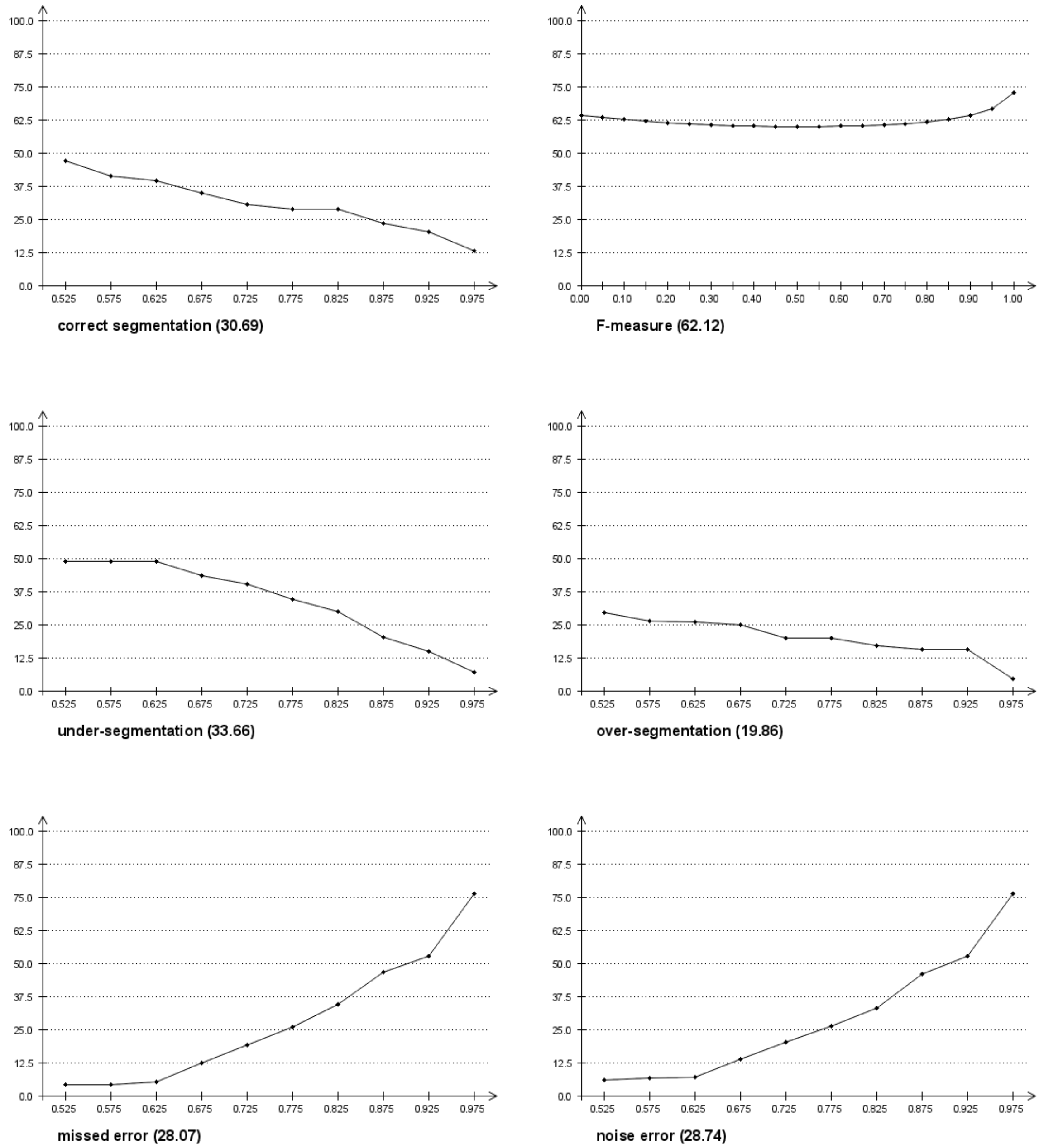


Figure A.20: *Colour* benchmark – performance curves – EGBIS.

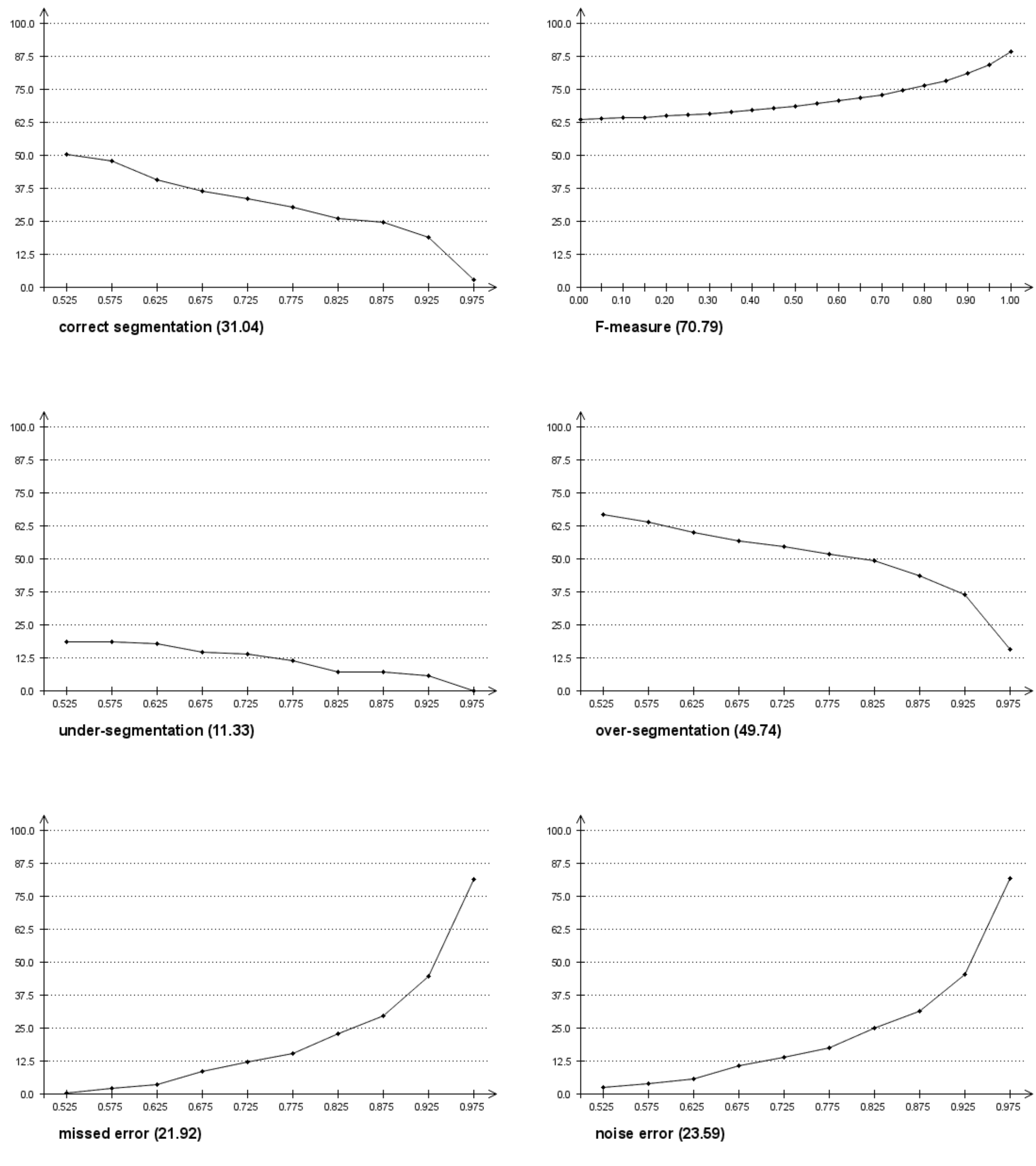


Figure A.21: *Colour* benchmark – performance curves – GMRF+EM.

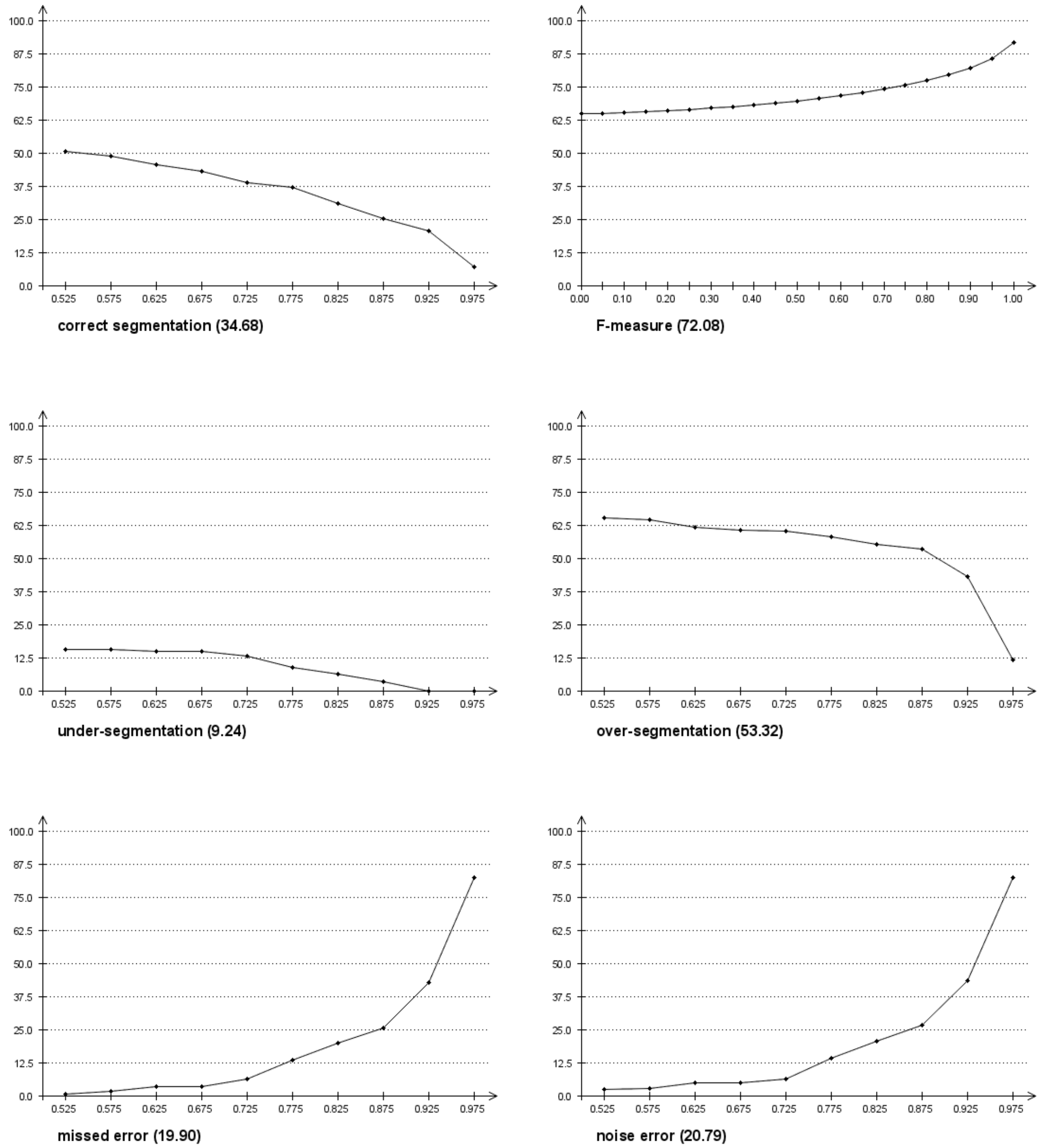


Figure A.22: *Colour* benchmark – performance curves – AR3D+EM.

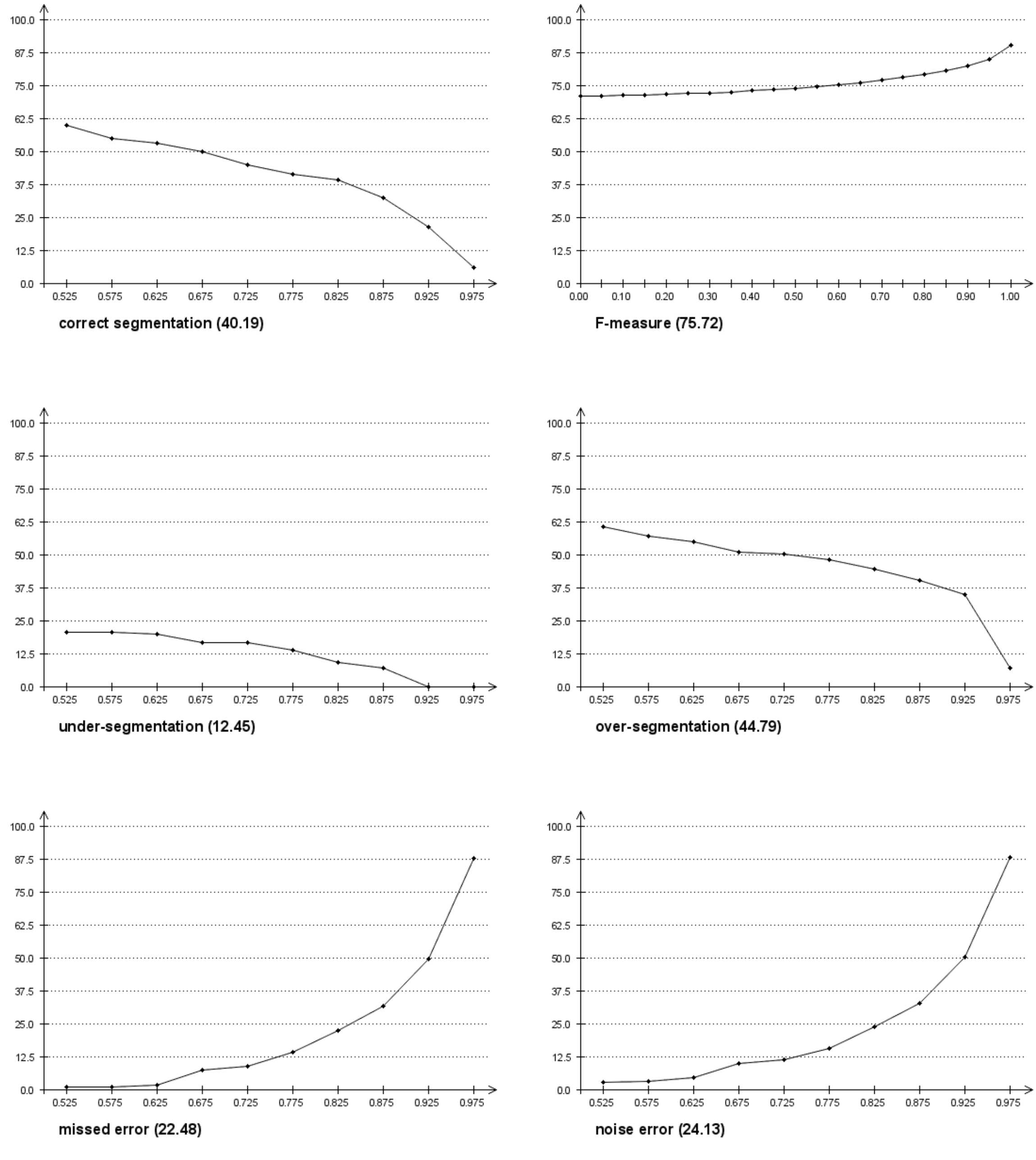


Figure A.23: *Colour* benchmark – performance curves – AR3D+EM multi.

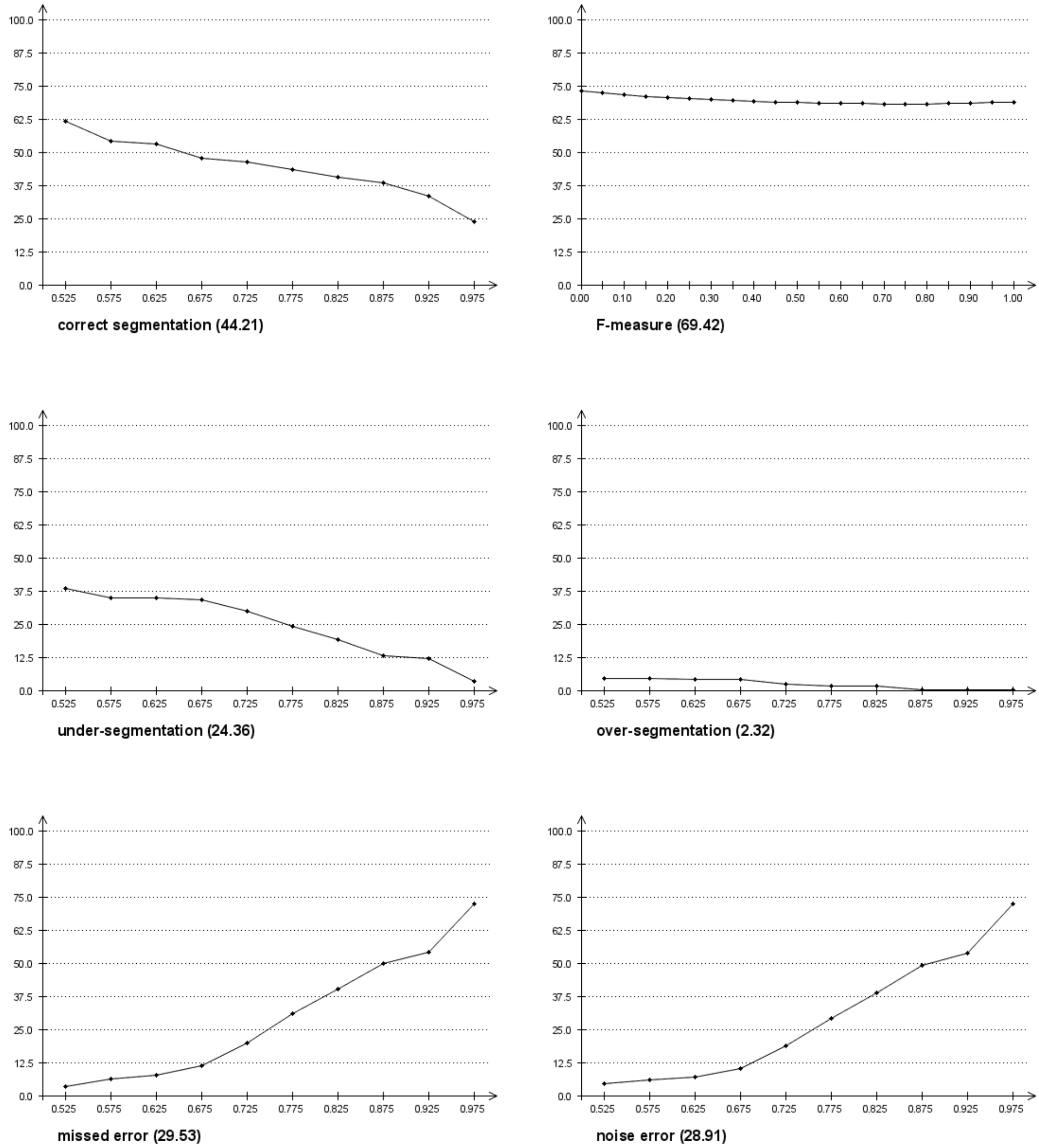


Figure A.24: *Colour* benchmark – performance curves – TFR.

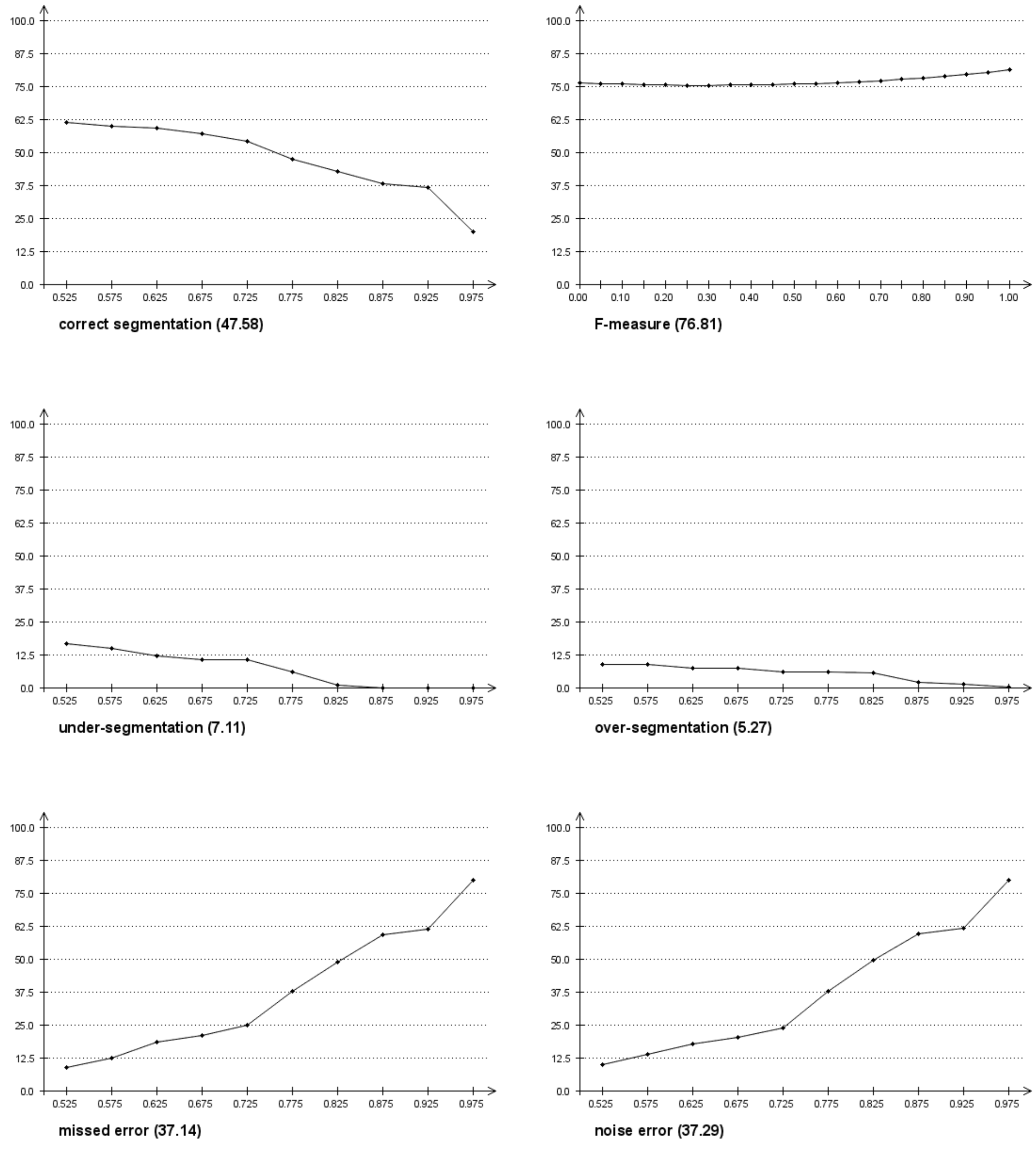


Figure A.25: *Colour* benchmark – performance curves – TFR/KLD.

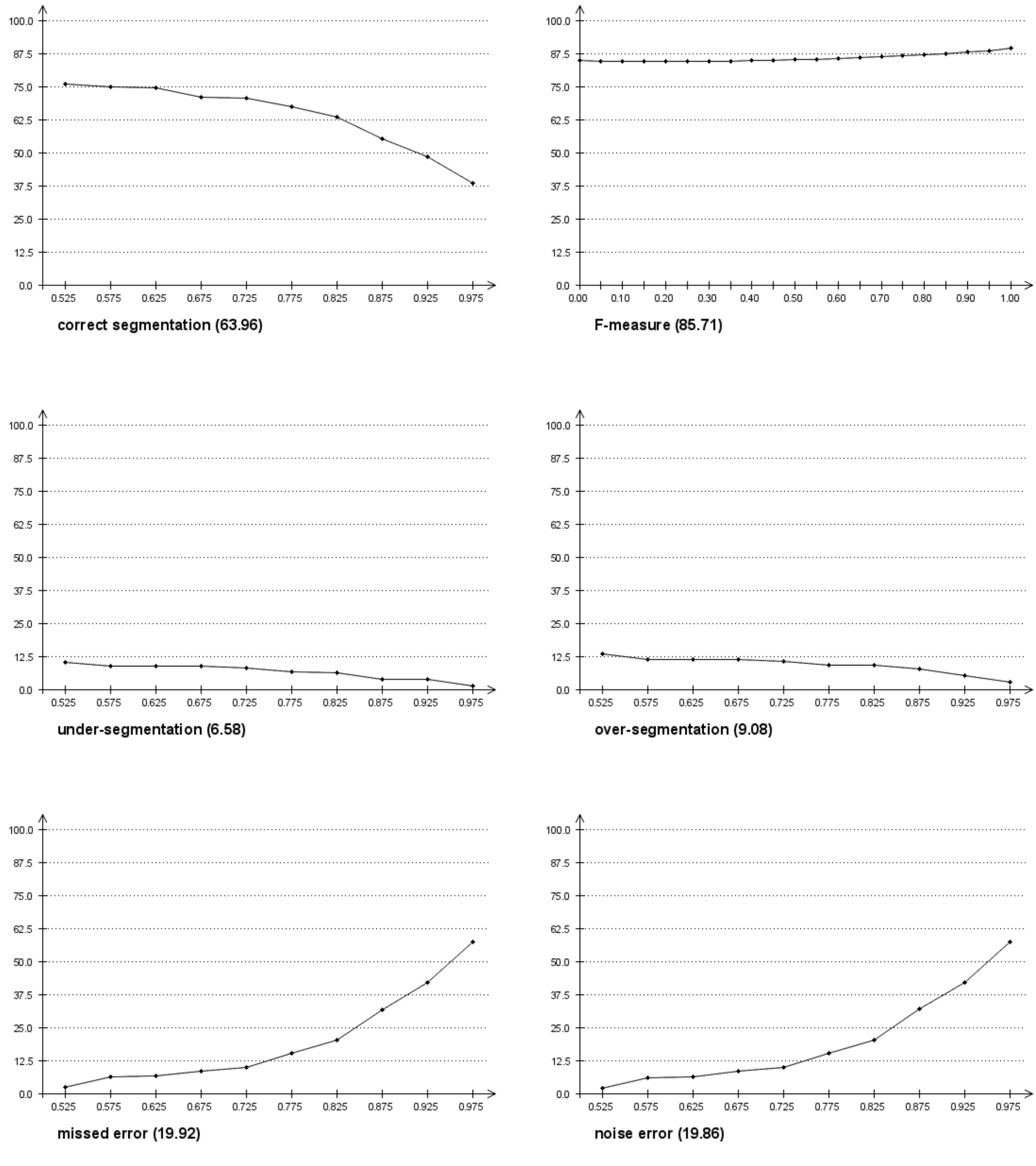


Figure A.26: *Colour* benchmark – performance curves – GSRM sup. (KL a-w).

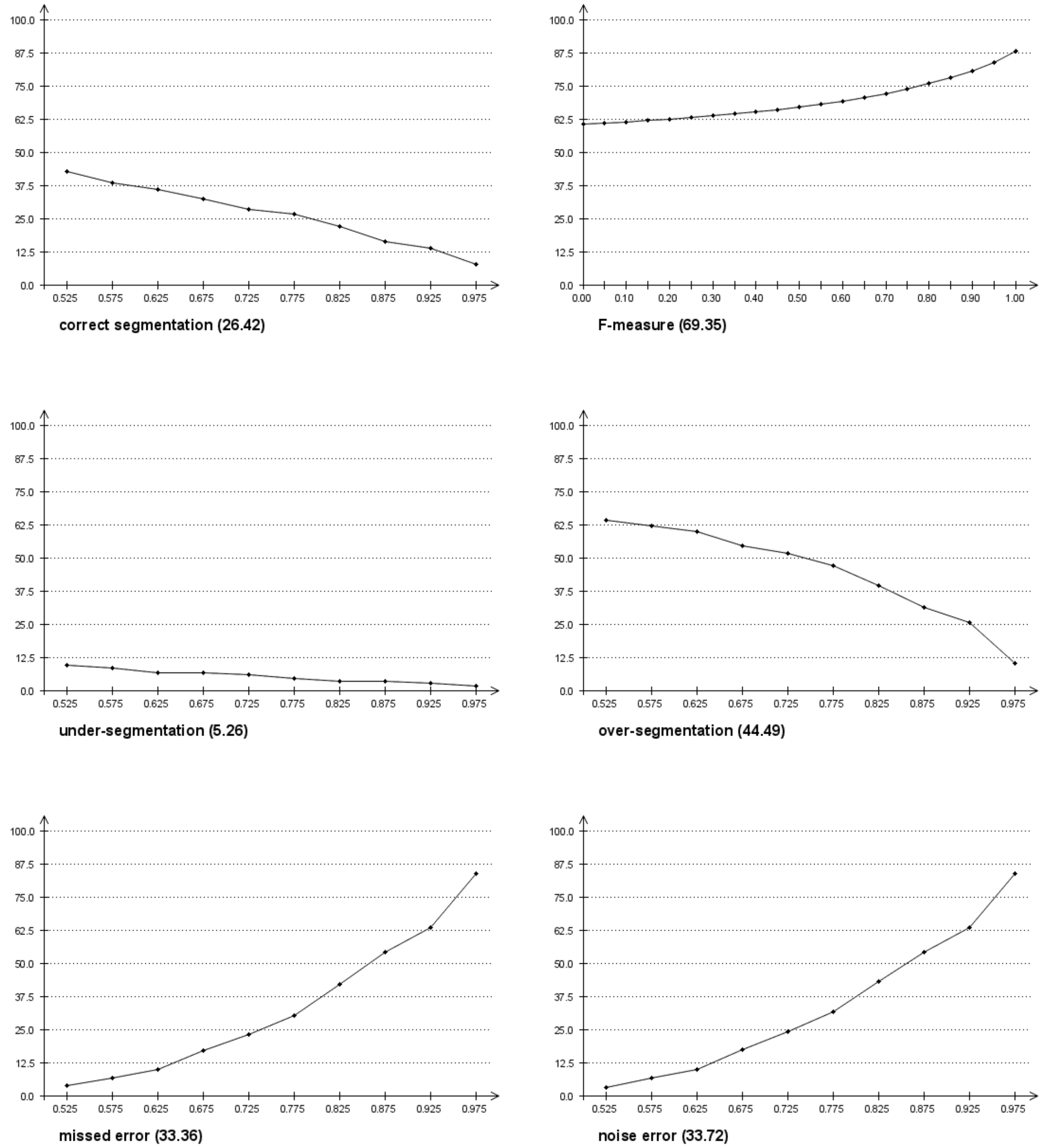


Figure A.27: *Colour* benchmark – performance curves – SWA.

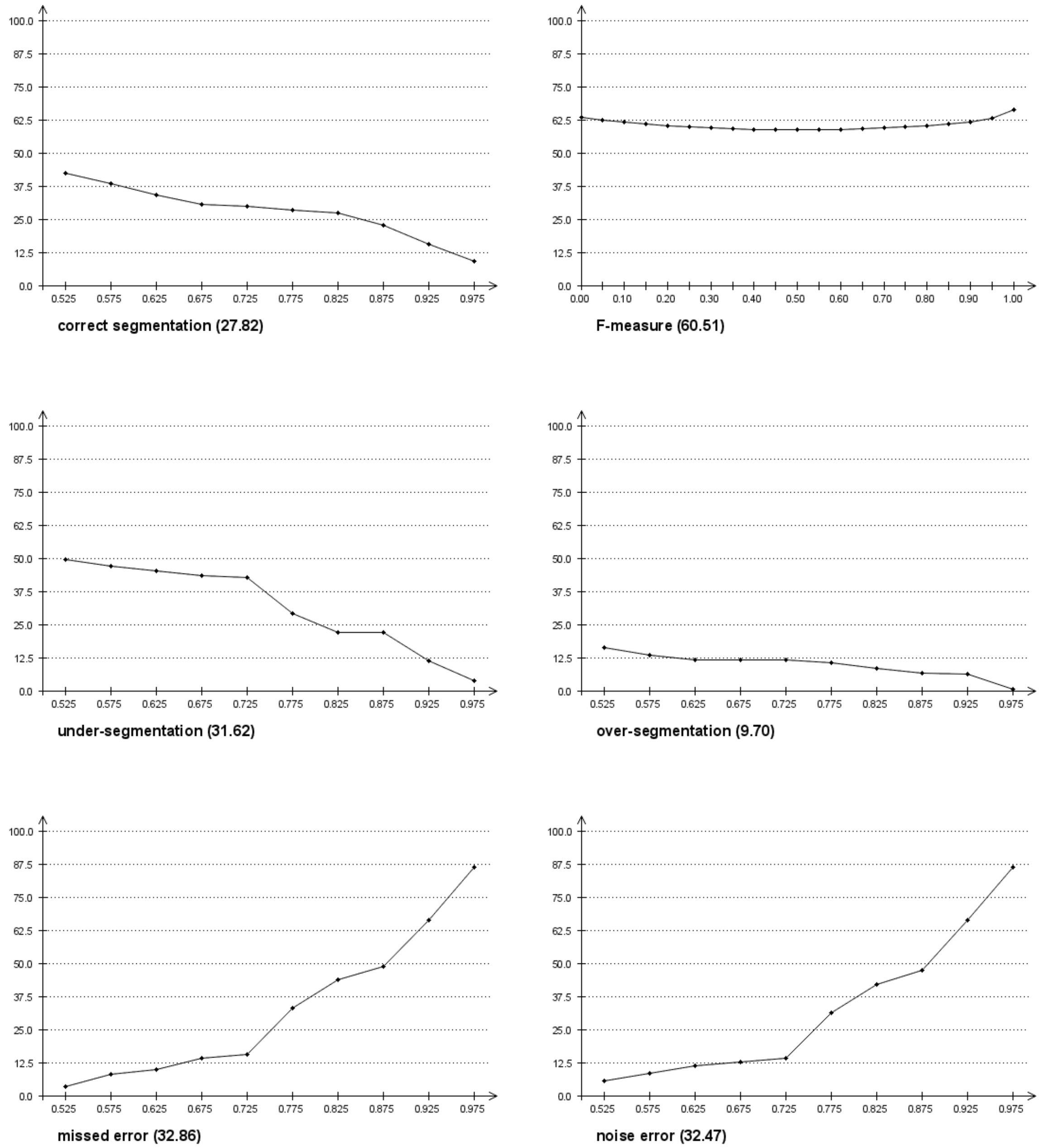


Figure A.28: *Colour* benchmark – performance curves – HGS (E).

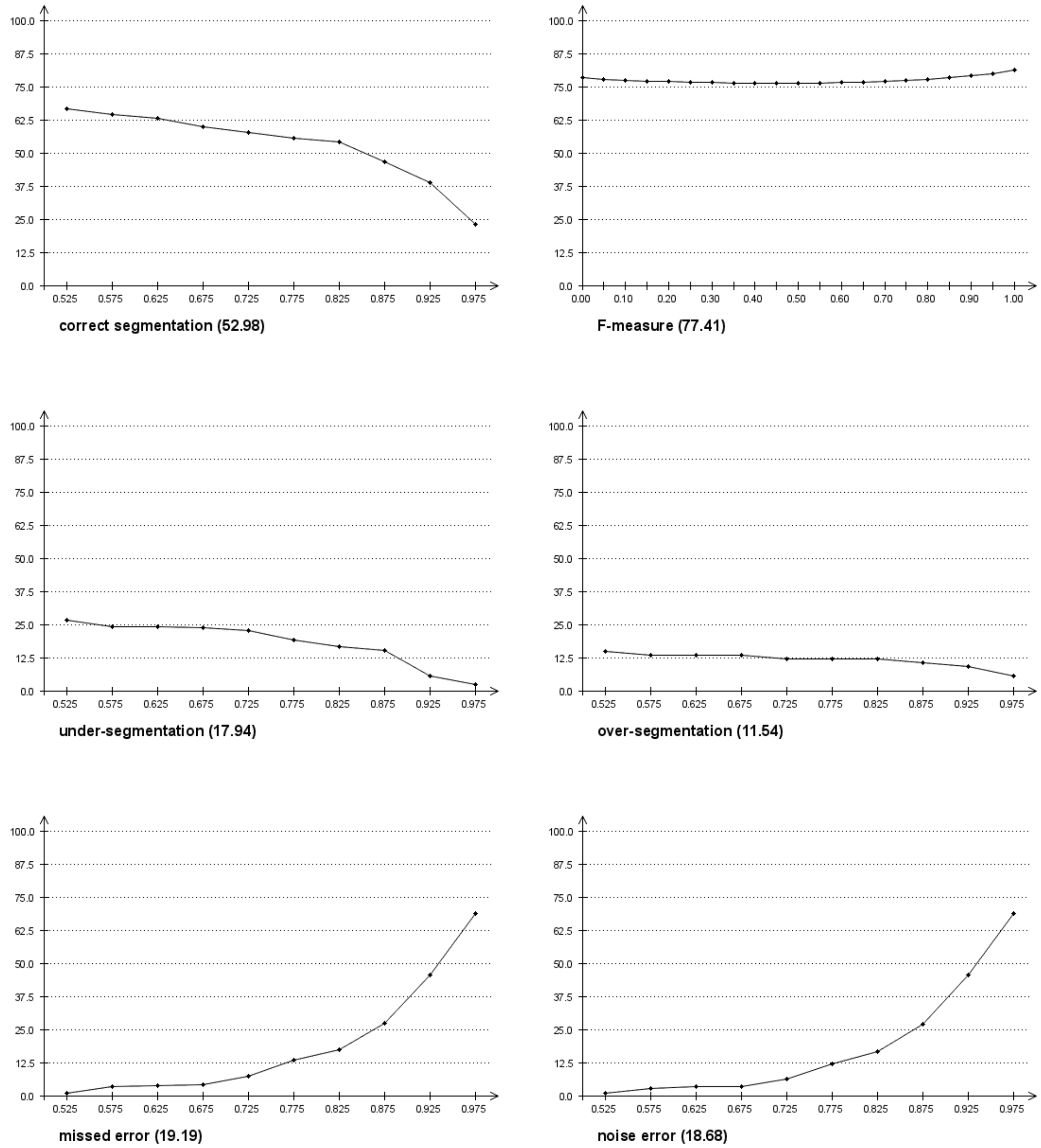


Figure A.29: *Colour* benchmark – performance curves – TEX-ROI-SEG.

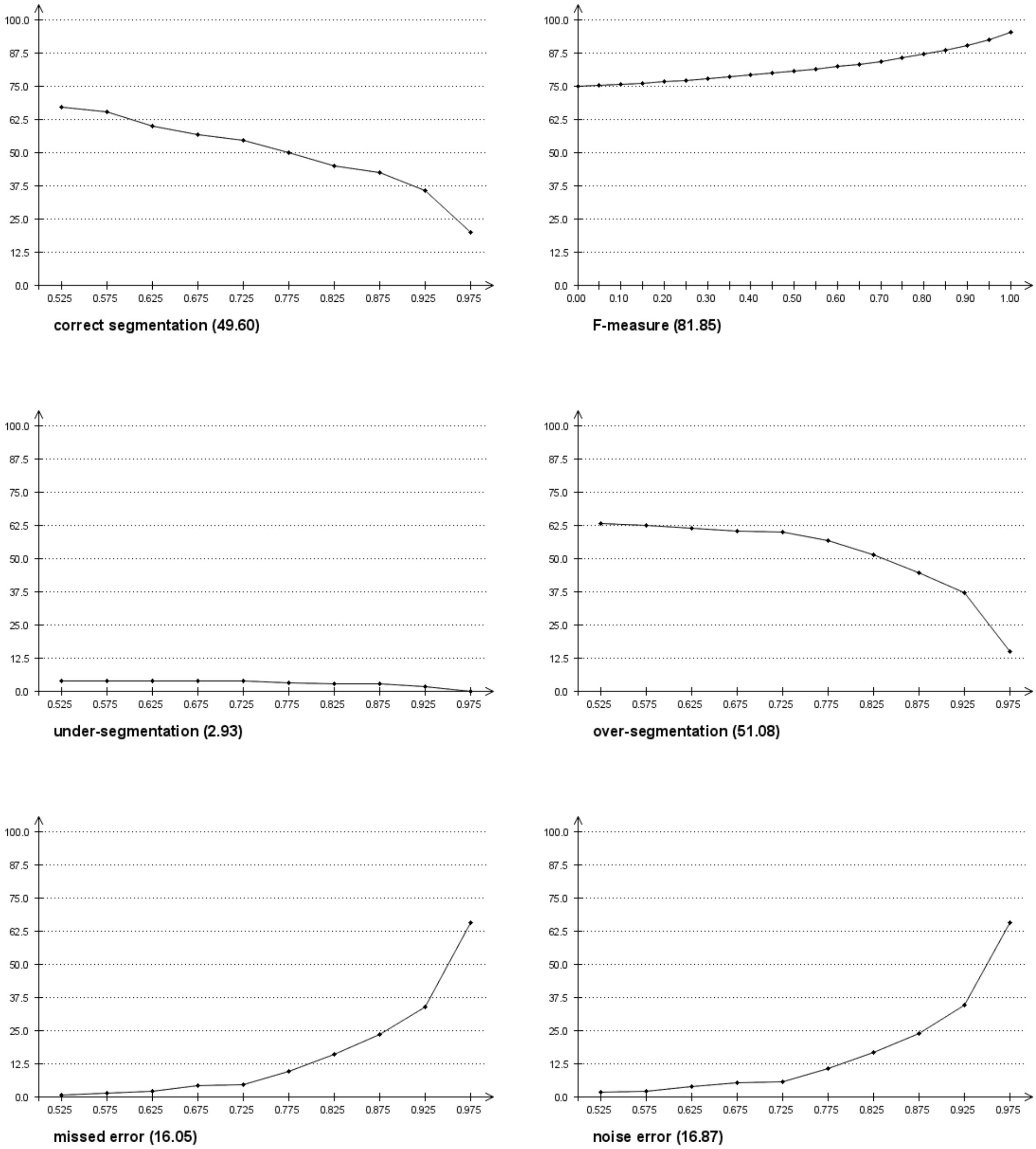
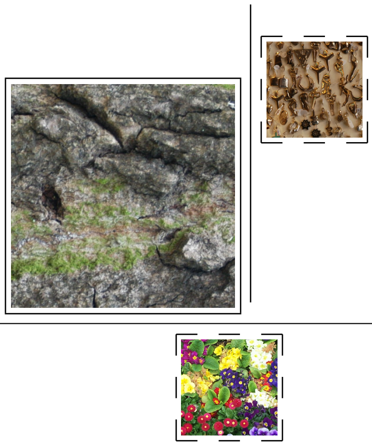


Figure A.30: *Colour* benchmark – performance curves – MW3AR.



Noise Robustness Results

Appendix B

In this appendix can be found *Colour* benchmark (described in section 4.2) segmentation results (tables and graphs) of texture mosaics degraded by Gaussian noise of different levels. Resulting segmentations are computed by six algorithms – Blobworld [12], EDISON [21], JSEG [27], EGBIS [39], GMRF+EM [54], AR2D+EM [55]. Further details can be found in section 5.3.

	Colour Benchmark – Blobworld										
	−10 dB	−5 dB	0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB	no noise
$\uparrow CS$	*9.83*	15.11	25.28	17.66	21.07	24.89	23.55	15.97	17.19	**26.24**	21.01
$\downarrow OS$	**1.17**	2.71	11.79	6.19	7.07	14.19	*14.23*	10.58	7.35	5.27	7.33
$\downarrow US$	*45.39*	25.65	17.38	20.72	11.87	12.89	**4.86**	10.16	11.69	11.44	9.30
$\downarrow ME$	**41.84**	54.62	44.09	52.61	56.72	47.04	54.33	*60.32*	59.02	53.76	59.55
$\downarrow NE$	**43.52**	55.92	45.61	55.12	58.47	49.80	57.53	*63.29*	62.20	55.89	61.68
$\downarrow O$	*74.49*	63.01	43.13	49.92	44.07	41.00	44.59	44.55	44.94	**37.19**	41.45
$\downarrow C$	55.00	60.55	53.22	**48.66**	56.12	56.63	65.81	64.04	*74.80*	65.27	58.94
$\uparrow CA$	*28.03*	36.01	49.06	42.17	47.18	49.48	**49.49**	44.43	45.57	48.92	46.23
$\uparrow CO$	*45.06*	50.09	59.39	54.72	58.10	**59.86**	58.13	54.68	55.13	58.40	56.04
$\uparrow CC$	*43.04*	54.55	70.21	63.52	70.99	72.03	**75.62**	73.69	75.19	74.75	73.62
$\downarrow I.$	*54.94*	49.91	40.61	45.28	41.90	**40.14**	41.87	45.32	44.87	41.60	43.96
$\downarrow II.$	*18.95*	13.45	6.90	10.18	6.86	6.78	**5.53**	6.81	5.66	6.06	6.72
$\uparrow EA$	*37.54*	46.34	60.25	53.32	58.81	61.39	**61.52**	56.92	57.94	60.38	58.37
$\uparrow MS$	*18.33*	27.30	43.46	35.06	41.33	**45.04**	43.96	38.49	39.69	43.74	40.36
$\downarrow RM$	*17.30*	13.29	8.27	10.13	8.14	6.98	**6.98**	8.04	8.35	7.74	7.96
$\uparrow CI$	*40.37*	48.99	62.38	55.90	61.42	63.47	**63.99**	60.15	61.21	63.16	61.31
$\downarrow GCE$	**25.61**	30.61	28.83	30.91	32.25	30.99	30.09	*33.02*	30.93	29.19	31.16
$\downarrow LCE$	**19.93**	22.71	21.78	22.71	22.81	23.17	22.57	*24.52*	23.02	22.82	23.19
$\downarrow dVI$	**12.36**	13.58	15.13	14.54	15.36	15.62	15.92	15.95	*16.08*	15.80	15.84
$\downarrow dM$	*41.81*	33.24	19.84	24.48	18.82	**18.16**	18.62	20.24	20.00	19.17	20.03
$\downarrow dD$	*34.53*	33.49	28.82	30.99	29.58	**28.81**	29.84	32.59	31.97	29.59	31.11
$\uparrow \overline{CS}$	*8.30*	14.62	23.63	18.34	20.48	22.05	22.30	16.89	17.97	**23.88**	19.10
$\downarrow \overline{OS}$	**2.24**	5.03	9.59	6.03	8.38	*13.67*	13.52	12.52	11.62	10.13	10.81
$\downarrow \overline{US}$	*40.37*	23.30	14.48	20.49	14.38	11.11	**7.35**	9.77	10.34	10.37	8.35
$\downarrow \overline{ME}$	**45.83**	54.85	52.09	54.17	54.43	52.69	54.48	*59.16*	57.36	53.63	58.54
$\downarrow \overline{NE}$	**48.35**	56.24	54.00	57.38	57.17	55.12	57.93	*62.51*	60.92	57.01	61.24
$\uparrow \overline{F}$	*39.53*	48.21	61.78	55.14	60.66	62.87	**63.28**	59.21	60.26	62.35	60.46

Table B.1: Blobworld results for *Colour* benchmark with Gaussian noise; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall - correct assignment; CC = precision - object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dVI = variation of information; dM = Mirkin metric; dD = Van Dongen metric; $\bar{f}$ are the performance curves integrals).

	Colour Benchmark – EDISON										
	−10 dB	−5 dB	0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB	no noise
$\uparrow CS$	*0.00*	*0.00*	*0.00*	2.98	10.86	12.91	12.05	**12.92**	12.80	10.78	12.68
$\downarrow OS$	**80.40**	87.91	*93.18*	92.97	83.79	91.97	89.74	88.53	91.01	90.88	86.91
$\downarrow US$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
$\downarrow ME$	*10.47*	4.93	1.73	1.77	9.13	**0.87**	2.26	1.51	1.47	0.91	2.48
$\downarrow NE$	*17.52*	10.05	5.37	4.78	11.06	**2.82**	4.64	3.70	3.38	2.91	4.68
$\downarrow O$	89.86	90.38	*90.67*	82.56	73.86	**72.71**	73.74	73.72	75.24	76.48	73.17
$\downarrow C$	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
$\uparrow CA$	8.83	8.27	*7.98*	17.36	28.05	30.38	30.25	31.01	30.16	30.05	**31.19**
$\uparrow CO$	8.91	8.35	*8.01*	17.70	28.84	30.65	30.64	31.42	30.56	30.39	**31.55**
$\uparrow CC$	96.71	97.14	**98.66**	97.68	*96.37*	98.42	97.89	98.30	97.84	96.74	98.09
$\downarrow I.$	91.09	91.65	*91.99*	82.30	71.16	69.35	69.36	68.58	69.44	69.61	**68.45**
$\downarrow II.$	0.07	0.06	**0.02**	0.26	*0.52*	0.14	0.30	0.22	0.33	0.35	0.24
$\uparrow EA$	15.72	14.84	*14.39*	25.94	37.83	40.27	40.41	41.12	40.10	40.42	**41.29**
$\uparrow MS$	8.65	8.12	*7.92*	17.05	27.77	30.32	30.16	31.03	30.08	29.81	**31.13**
$\downarrow RM$	3.17	3.06	**3.00**	3.05	3.11	3.19	3.23	3.18	3.23	*3.24*	3.21
$\uparrow CI$	28.03	27.19	*26.97*	36.84	46.97	49.38	49.55	50.17	49.22	49.37	**50.29**
$\downarrow GCE$	*9.25*	6.95	4.59	4.11	5.53	**3.39**	3.88	3.83	3.53	3.60	3.55
$\downarrow LCE$	*9.25*	6.95	4.59	4.03	5.44	**3.33**	3.78	3.69	3.43	3.55	3.44
$\downarrow dVI$	30.75	31.15	*31.35*	29.03	26.40	25.98	25.73	25.76	25.79	25.85	**25.65**
$\downarrow dM$	21.12	21.18	*21.18*	19.74	17.68	16.99	17.01	16.93	17.08	17.18	**16.84**
$\downarrow dD$	*48.77*	48.14	47.51	42.48	37.41	35.76	35.84	35.48	35.84	35.95	**35.37**
$\uparrow \overline{CS}$	0.03	0.03	*0.00*	3.28	10.32	12.51	12.01	12.80	12.30	11.11	**12.95**
$\downarrow \overline{OS}$	**61.89**	69.79	77.73	*80.23*	74.51	80.12	77.96	77.19	79.45	78.45	76.33
$\downarrow \overline{US}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
$\downarrow \overline{ME}$	*31.92*	25.32	18.65	15.86	18.67	**13.30**	14.96	13.84	13.84	14.14	13.92
$\downarrow \overline{NE}$	*35.81*	28.20	20.87	17.64	20.29	**14.68**	16.62	15.25	15.13	15.57	15.30
$\uparrow \overline{F}$	23.87	23.00	*22.70*	33.23	44.00	46.46	46.63	47.28	46.30	46.52	**47.42**

Table B.2: EDISON results for *Colour* benchmark with Gaussian noise; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall - correct assignment; CC = precision - object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dVI = variation of information; dM = Mirkin metric; dD = Van Dongen metric; $\bar{f}$ are the performance curves integrals).

	Colour Benchmark – JSEG										
	−10 dB	−5 dB	0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB	no noise
$\uparrow CS$	*0.00*	0.21	1.95	23.29	31.68	30.39	**32.07**	30.44	30.97	29.02	27.47
$\downarrow OS$	**0.00**	**0.00**	0.35	15.33	28.09	37.45	44.98	42.84	*49.73*	48.89	38.62
$\downarrow US$	*100.00*	99.77	89.47	42.71	13.02	8.32	5.63	6.12	**2.62**	8.03	5.04
$\downarrow ME$	**0.00**	**0.00**	7.71	24.55	31.88	33.86	25.17	29.01	27.26	22.34	*35.00*
$\downarrow NE$	**0.00**	**0.00**	6.11	23.96	31.16	33.73	26.36	28.55	27.63	24.06	*35.50*
$\downarrow O$	*100.00*	*100.00*	95.16	55.33	34.13	34.00	**31.58**	36.70	36.15	36.72	37.94
$\downarrow C$	68.92	67.45	**49.23**	69.45	91.00	94.36	92.55	*95.13*	92.61	94.49	92.77
$\uparrow CA$	*10.55*	10.78	13.63	40.70	55.34	55.11	**57.66**	54.02	56.01	54.63	55.29
$\uparrow CO$	*31.08*	31.28	33.19	54.54	**63.93**	62.34	63.75	60.60	61.50	60.79	61.81
$\uparrow CC$	*10.55*	10.78	22.21	63.64	82.99	85.09	88.66	88.10	88.59	**89.02**	87.70
$\downarrow I.$	*68.92*	68.72	66.81	45.46	**36.07**	37.66	36.25	39.40	38.50	39.21	38.19
$\downarrow II.$	*31.08*	30.95	27.15	13.10	5.08	4.70	3.88	3.88	**3.59**	3.64	3.66
$\uparrow EA$	*15.49*	15.72	19.60	48.90	65.80	66.09	**68.89**	65.27	67.20	65.74	66.74
$\uparrow MS$	*-3.38*	-3.07	-0.22	34.65	53.48	53.17	**56.47**	53.22	55.02	53.59	55.14
$\downarrow RM$	*30.44*	30.36	29.36	14.44	6.40	6.16	5.16	5.59	5.17	5.54	**4.96**
$\uparrow CI$	*17.91*	18.13	22.97	52.81	69.08	69.46	**72.11**	69.25	70.67	69.64	70.27
$\downarrow GCE$	**0.00**	0.03	4.56	16.37	*22.07*	21.01	18.19	18.81	15.91	17.87	18.45
$\downarrow LCE$	**0.00**	0.03	2.11	6.54	11.20	11.30	10.54	11.06	9.81	10.50	*11.64*
$\downarrow dVI$	**8.47**	8.50	9.21	12.83	15.82	16.45	16.92	17.24	*17.47*	17.26	17.37
$\downarrow dM$	*78.24*	77.82	72.22	33.96	16.42	16.82	**15.06**	16.01	15.25	15.54	15.19
$\downarrow dD$	*34.46*	34.37	34.37	24.87	22.20	22.89	**21.97**	23.64	22.52	23.12	23.38
$\uparrow \overline{CS}$	*0.58*	0.77	1.72	22.97	30.53	29.36	**30.99**	29.58	29.91	28.93	29.13
$\downarrow \overline{OS}$	**0.00**	**0.00**	0.51	15.14	26.94	36.12	39.93	40.23	*45.17*	44.11	37.70
$\downarrow \overline{US}$	*100.00*	99.77	85.33	40.46	11.34	9.35	7.47	7.16	**5.75**	6.56	6.38
$\downarrow \overline{ME}$	**0.00**	0.02	11.64	27.36	*36.83*	36.21	31.85	32.76	29.50	30.50	34.72
$\downarrow \overline{NE}$	**0.00**	0.02	11.24	27.68	*36.64*	36.22	33.33	32.98	31.14	31.89	35.38
$\uparrow \overline{F}$	*17.20*	17.43	21.97	51.61	68.11	68.47	**71.17**	68.08	69.65	68.48	69.23

Table B.3: JSEG results for *Colour* benchmark with Gaussian noise; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall - correct assignment; CC = precision - object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dVI = variation of information; dM = Mirkin metric; dD = Van Dongen metric; $\bar{f}$ are the performance curves integrals).

	Colour Benchmark – EGBIS										
	−10 dB	−5 dB	0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB	no noise
$\uparrow CS$	*0.11*	0.47	4.29	9.06	14.86	**29.72**	24.43	25.07	27.46	24.75	24.58
$\downarrow OS$	63.37	39.01	**20.12**	20.61	31.44	54.64	61.20	73.13	*77.17*	73.76	75.65
$\downarrow US$	**0.00**	4.15	29.17	*36.55*	22.47	11.49	7.85	1.99	6.13	4.54	8.19
$\downarrow ME$	26.50	*50.78*	44.60	35.99	30.57	13.56	15.25	11.78	**3.19**	8.22	4.12
$\downarrow NE$	34.54	*53.67*	43.01	34.99	31.54	15.21	16.73	14.33	**4.73**	10.03	6.19
$\downarrow O$	*80.63*	68.26	72.22	67.98	52.95	38.67	46.47	42.66	**38.11**	41.96	47.26
$\downarrow C$	*100.00*	*100.00*	94.91	**93.92**	98.29	*100.00*	95.17	*100.00*	*100.00*	*100.00*	*100.00*
$\uparrow CA$	*17.57*	29.43	28.51	31.97	43.80	**52.80**	50.01	50.21	51.03	49.85	48.07
$\uparrow CO$	*18.35*	34.99	42.41	45.89	54.52	**58.74**	54.47	53.39	54.14	53.00	51.29
$\uparrow CC$	92.57	82.67	69.40	*68.06*	80.10	90.40	90.71	93.22	94.37	93.33	**94.99**
$\downarrow I.$	*81.65*	65.01	57.59	54.11	45.48	**41.26**	45.53	46.61	45.86	47.00	48.71
$\downarrow II.$	**0.45**	4.44	*13.69*	12.76	7.49	3.18	2.14	1.34	1.53	1.47	1.54
$\uparrow EA$	*28.94*	43.52	40.33	42.45	55.32	**63.81**	61.31	61.92	62.24	61.45	59.69
$\uparrow MS$	*16.90*	27.69	23.46	25.93	42.12	**52.93**	50.19	50.28	51.22	49.96	48.30
$\downarrow RM$	**3.93**	4.70	9.97	*11.42*	6.58	4.46	4.23	4.03	4.12	4.05	4.21
$\uparrow CI$	*39.41*	50.01	46.03	47.92	59.98	**68.24**	66.03	66.83	67.16	66.43	65.24
$\downarrow GCE$	14.48	24.49	*26.84*	22.89	22.93	15.86	13.01	9.84	**9.08**	9.57	9.27
$\downarrow LCE$	14.38	*19.86*	15.81	11.35	10.54	7.62	7.38	6.92	5.23	6.07	**5.09**
$\downarrow dVI$	*27.08*	21.85	15.54	**14.36**	16.36	18.15	19.32	20.02	19.96	20.18	20.22
$\downarrow dM$	20.55	22.24	*35.84*	35.36	21.58	14.65	14.83	**14.37**	14.58	14.59	14.85
$\downarrow dD$	*45.97*	39.29	34.63	31.13	26.70	**23.33**	25.53	25.91	24.67	25.62	26.01
$\uparrow \overline{CS}$	*0.15*	2.02	4.99	9.55	17.84	**27.81**	24.63	24.22	25.88	23.65	22.52
$\downarrow \overline{OS}$	50.35	35.05	**17.69**	18.08	28.78	45.92	55.82	64.57	66.89	65.25	*68.69*
$\downarrow \overline{US}$	**0.00**	4.28	28.04	*32.07*	20.91	10.52	7.75	2.23	5.62	4.53	6.92
$\downarrow \overline{ME}$	42.47	*54.86*	49.44	42.60	36.77	25.90	22.29	21.21	15.05	17.78	**14.48**
$\downarrow \overline{NE}$	46.80	*56.62*	48.71	43.81	37.57	27.34	23.27	22.45	**15.97**	19.16	16.04
$\uparrow \overline{F}$	*36.10*	48.05	44.29	46.25	58.58	**66.92**	64.61	65.37	65.68	64.93	63.57

Table B.4: EGBIS results for *Colour* benchmark with Gaussian noise; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall - correct assignment; CC = precision - object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dVI = variation of information; dM = Mirkin metric; dD = Van Dongen metric; $\bar{f}$ are the performance curves integrals).

	Colour Benchmark – GMRF+EM										
	−10 dB	−5 dB	0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB	no noise
↑ *CS*	*24.77*	35.25	32.71	**36.82**	36.18	34.47	32.37	33.38	34.32	33.18	31.93
↓ *OS*	**25.07**	37.14	49.76	53.46	48.99	53.62	52.68	49.08	*61.67*	49.96	53.27
↓ *US*	*22.16*	8.74	14.45	8.30	13.51	12.06	12.15	**6.11**	7.18	7.94	11.24
↓ *ME*	*38.87*	25.75	17.12	19.71	19.41	15.84	17.97	21.66	**12.80**	16.99	14.97
↓ *NE*	*38.95*	26.11	17.66	19.75	18.46	15.70	18.62	23.12	**14.32**	17.55	16.91
↓ *O*	*47.34*	34.52	40.89	33.44	35.38	33.26	32.16	31.23	30.53	**30.50**	33.61
↓ *C*	**91.88**	94.03	95.85	97.66	97.18	*100.00*	99.23	*100.00*	*100.00*	*100.00*	*100.00*
↑ *CA*	*45.72*	**59.13**	55.67	58.90	57.37	55.78	56.07	58.84	59.08	56.55	57.91
↑ *CO*	*57.18*	**65.95**	62.16	64.54	64.01	61.59	62.15	63.91	62.97	62.01	63.51
↑ *CC*	*73.36*	87.36	87.67	89.70	86.28	88.53	89.97	90.59	**92.15**	90.02	89.26
↓ *I.*	*42.82*	**34.05**	37.84	35.46	35.99	38.41	37.85	36.09	37.03	37.99	36.49
↓ *II.*	*9.20*	3.59	4.45	3.35	4.80	3.43	3.48	3.08	**1.58**	3.18	3.14
↑ *EA*	*56.43*	69.57	65.83	69.29	67.07	66.05	66.72	69.51	**70.02**	67.17	68.41
↑ *MS*	*41.05*	58.21	54.08	58.57	56.30	54.81	54.99	57.93	**59.24**	56.24	57.42
↓ *RM*	*7.76*	5.10	5.89	4.98	5.43	5.47	5.78	4.89	**3.87**	5.05	4.86
↑ *CI*	*59.91*	72.51	69.50	72.69	70.42	69.79	70.66	72.81	**73.35**	70.89	71.80
↓ *GCE*	*28.51*	19.82	15.08	14.72	15.67	13.63	15.35	14.56	**12.42**	15.36	16.03
↓ *LCE*	*19.21*	11.02	7.71	7.71	7.46	**7.20**	7.52	8.46	8.07	7.38	7.31
↓ *dVI*	**15.05**	16.42	16.73	17.06	16.42	17.28	17.11	17.35	*18.45*	17.62	17.32
↓ *dM*	*24.90*	14.25	17.95	14.61	17.00	16.14	17.59	15.90	**12.82**	15.70	15.27
↓ *dD*	*27.94*	20.69	21.50	**20.20**	20.69	21.43	21.55	20.87	21.04	21.47	20.63
↑ $\overline{CS}$	*22.42*	34.62	30.94	**35.16**	34.27	32.07	30.72	33.08	32.12	31.18	31.04
↓ $\overline{OS}$	**22.19**	35.43	45.04	47.88	45.40	49.60	46.20	45.79	*55.04*	47.41	49.74
↓ $\overline{US}$	*19.87*	9.60	11.61	9.52	11.94	10.88	11.51	7.20	**6.58**	9.21	11.33
↓ $\overline{ME}$	*45.38*	28.84	25.77	24.31	25.95	22.67	25.66	27.30	22.77	22.88	**21.92**
↓ $\overline{NE}$	*46.42*	29.77	26.46	24.95	26.42	**23.05**	26.40	28.41	24.29	24.04	23.59
↑ $\overline{F}$	*58.84*	71.63	68.40	71.69	69.42	68.67	69.48	71.83	**72.37**	69.78	70.79

Table B.5: GMRF+EM results for *Colour* benchmark with Gaussian noise; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall - correct assignment; CC = precision - object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dVI = variation of information; dM = Mirkin metric; dD = Van Dongen metric; $\bar{f}$ are the performance curves integrals).

	Colour Benchmark – AR2D+EM										
	−10 dB	−5 dB	0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB	no noise
$\uparrow CS$	*27.54*	37.67	38.18	35.53	40.77	35.24	36.16	37.39	42.49	39.84	**43.88**
$\downarrow OS$	**24.59**	42.62	41.29	32.61	45.94	45.70	46.16	47.29	48.30	*49.81*	48.55
$\downarrow US$	*41.72*	28.67	23.50	26.48	21.80	25.26	24.19	26.02	23.67	**20.62**	21.89
$\downarrow ME$	17.33	**10.58**	13.33	17.80	16.80	*18.07*	14.95	13.31	10.70	16.33	13.27
$\downarrow NE$	18.10	**11.34**	14.34	17.91	17.90	*18.91*	15.27	14.61	11.92	17.70	14.84
$\downarrow O$	*54.97*	36.65	41.14	40.74	36.39	41.39	36.82	41.54	**32.99**	42.50	34.31
$\downarrow C$	**89.22**	98.10	96.71	91.84	93.76	96.80	95.68	*98.86*	93.69	93.29	93.88
$\uparrow CA$	*41.81*	56.31	56.51	53.69	56.87	52.91	55.36	54.08	59.02	56.60	**60.27**
$\uparrow CO$	*54.49*	65.78	64.85	62.97	64.76	61.29	63.13	62.06	66.38	63.85	**67.36**
$\uparrow CC$	*71.51*	82.93	84.36	81.89	83.40	81.94	82.33	80.14	**85.37**	85.11	83.66
$\downarrow I.$	*45.51*	34.22	35.15	37.03	35.24	38.71	36.87	37.94	33.62	36.15	**32.64**
$\downarrow II.$	*12.09*	6.83	6.30	7.79	**6.17**	8.65	8.42	8.83	6.21	7.96	7.54
$\uparrow EA$	*49.97*	65.04	65.24	61.96	64.91	61.53	64.18	62.50	67.15	65.12	**68.19**
$\uparrow MS$	*35.57*	54.48	54.32	50.71	54.13	49.09	52.44	50.94	56.50	52.88	**57.40**
$\downarrow RM$	*11.00*	7.04	6.68	8.73	6.95	7.09	6.41	6.89	6.82	6.40	**6.20**
$\uparrow CI$	*54.37*	68.48	68.59	65.67	68.34	65.12	67.38	65.73	70.38	68.55	**70.92**
$\downarrow GCE$	*17.94*	16.07	14.83	12.79	13.44	15.98	15.27	15.46	**11.67**	16.83	13.54
$\downarrow LCE$	*9.26*	6.53	5.79	**5.20**	5.46	6.03	6.80	6.13	5.56	6.60	6.02
$\downarrow dVI$	**13.50**	14.67	15.09	14.37	14.98	15.01	*15.35*	15.24	14.98	15.04	14.79
$\downarrow dM$	*32.60*	20.31	**19.32**	24.00	19.60	22.90	20.68	22.71	19.86	20.18	19.35
$\downarrow dD$	*25.45*	19.29	19.44	20.20	19.44	21.07	20.35	20.62	18.44	20.02	**17.91**
$\uparrow \overline{CS}$	*24.42*	34.99	35.17	34.22	36.94	31.59	33.86	33.11	38.35	35.33	**40.30**
$\downarrow \overline{OS}$	**21.55**	39.68	37.40	30.43	42.82	42.75	43.52	43.11	44.18	45.04	*46.40*
$\downarrow \overline{US}$	*34.73*	23.14	22.36	26.56	20.71	23.38	17.57	21.56	19.02	**16.25**	20.08
$\downarrow \overline{ME}$	*28.70*	20.83	**19.07**	21.17	20.06	23.36	26.12	23.36	20.53	26.64	19.50
$\downarrow \overline{NE}$	*30.05*	21.67	**20.22**	21.55	20.81	24.82	27.10	24.58	21.93	28.39	21.56
$\uparrow \overline{F}$	*52.99*	67.42	67.58	64.55	67.29	64.05	66.46	64.74	69.42	67.49	**70.07**

Table B.6: AR2D+EM results for *Colour* benchmark with Gaussian noise; (Benchmark criteria: CS = correct segmentation; OS = over-segmentation; US = under-segmentation; ME = missed error; NE = noise error; O = omission error; C = commission error; CA = class accuracy; CO = recall - correct assignment; CC = precision - object accuracy; I. = type I error; II. = type II error; EA = mean class accuracy estimate; MS = mapping score; RM = root mean square proportion estimation error; CI = comparison index; GCE = Global Consistency Error; LCE = Local Consistency Error; dVI = variation of information; dM = Mirkin metric; dD = Van Dongen metric; $\bar{f}$ are the performance curves integrals).

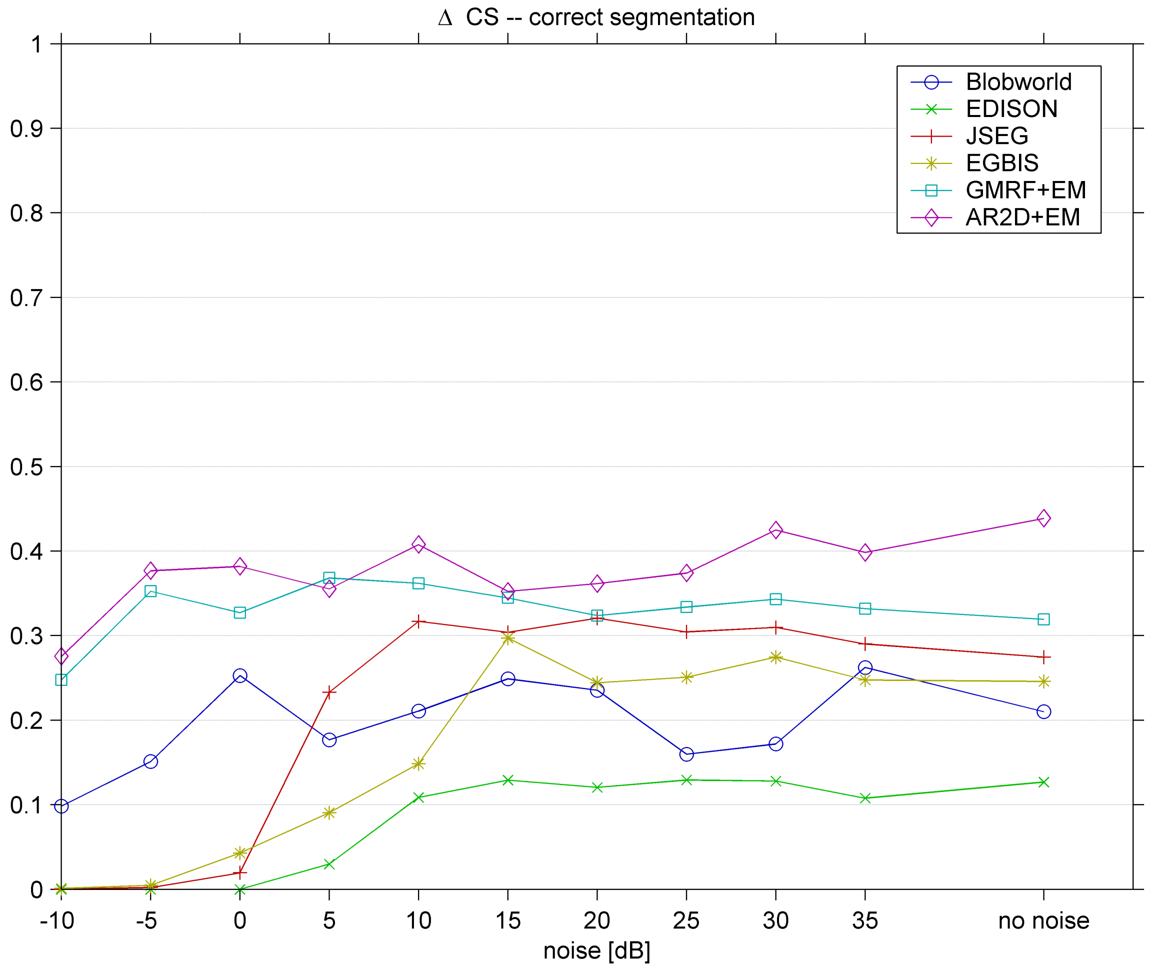


Figure B.1: Noise robustness graph – CS – correct segmentation.

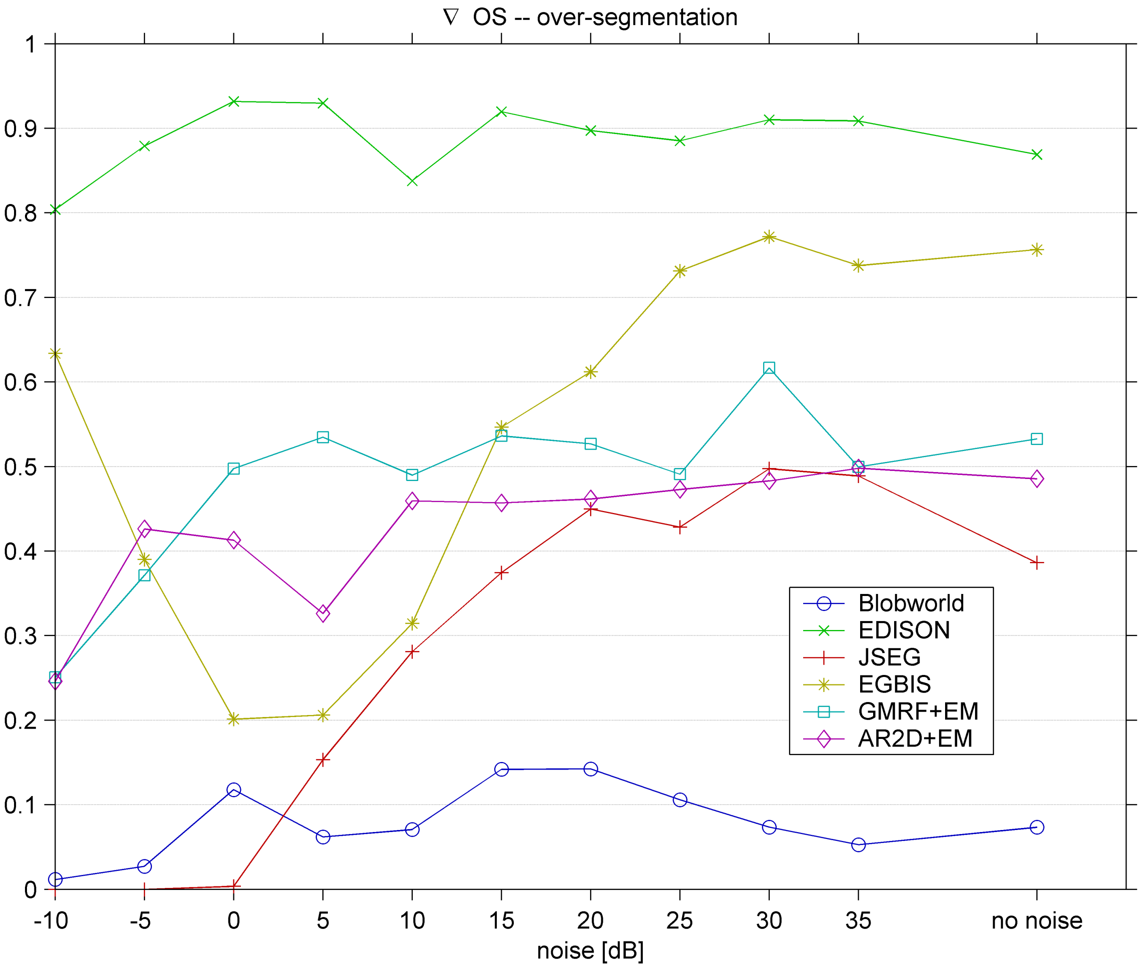


Figure B.2: Noise robustness graph – OS – over-segmentation.

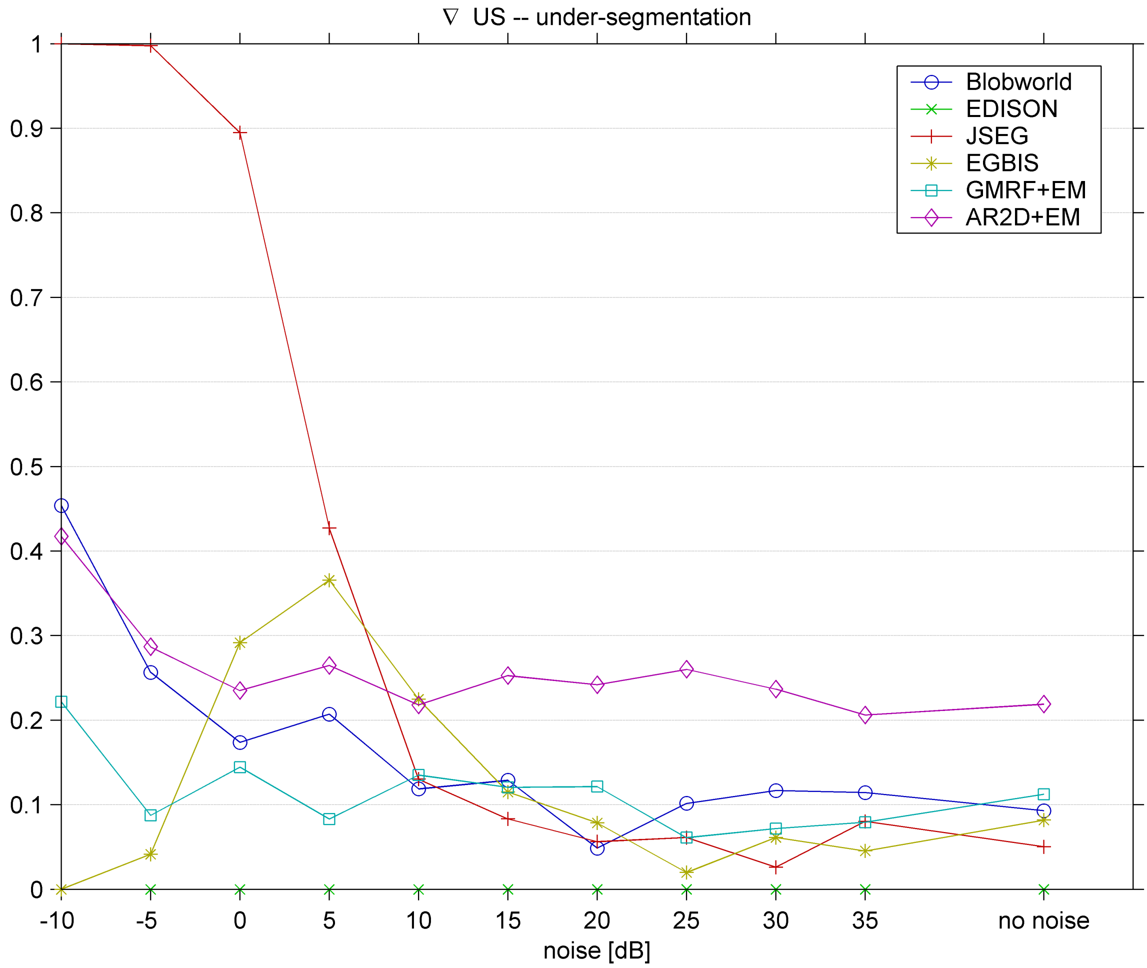


Figure B.3: Noise robustness graph – US – under-segmentation.

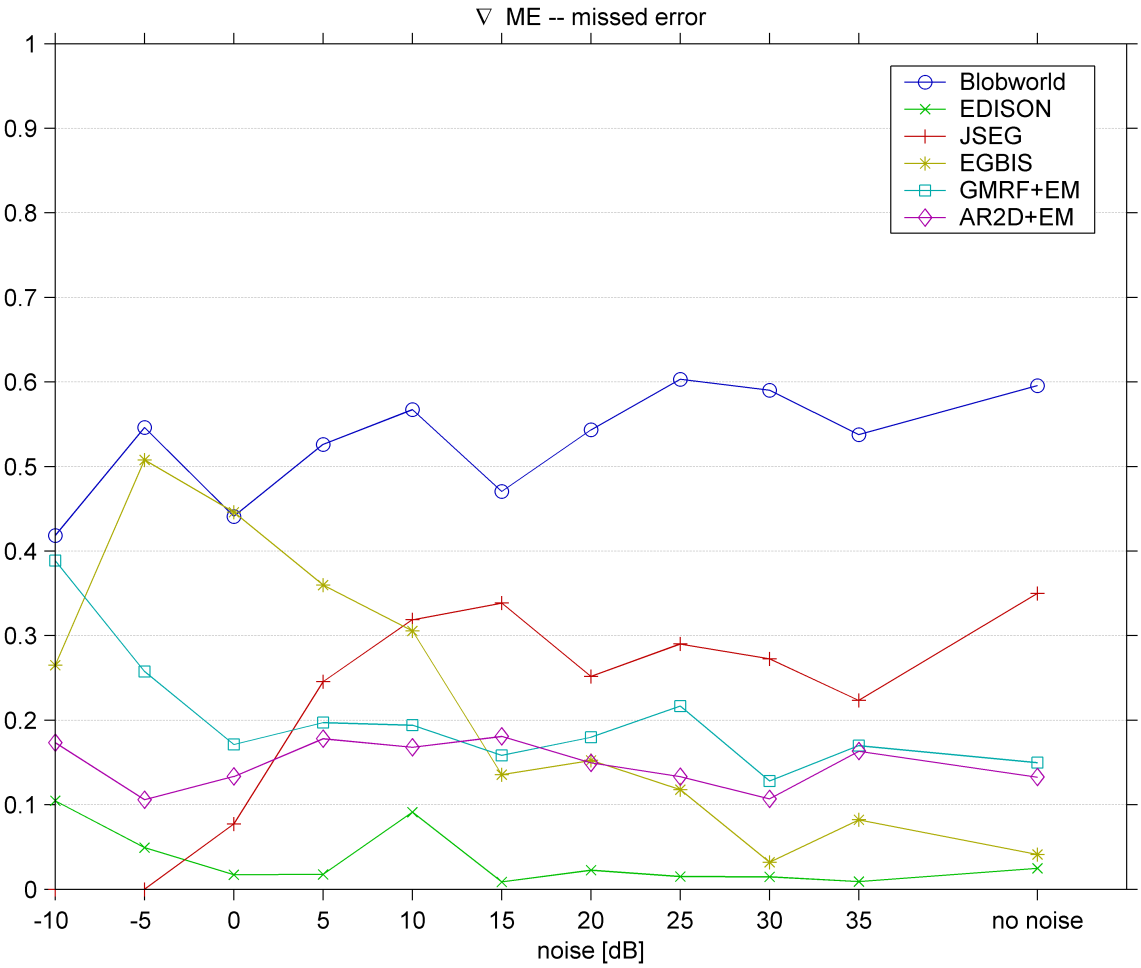


Figure B.4: Noise robustness graph – ME – missed error.

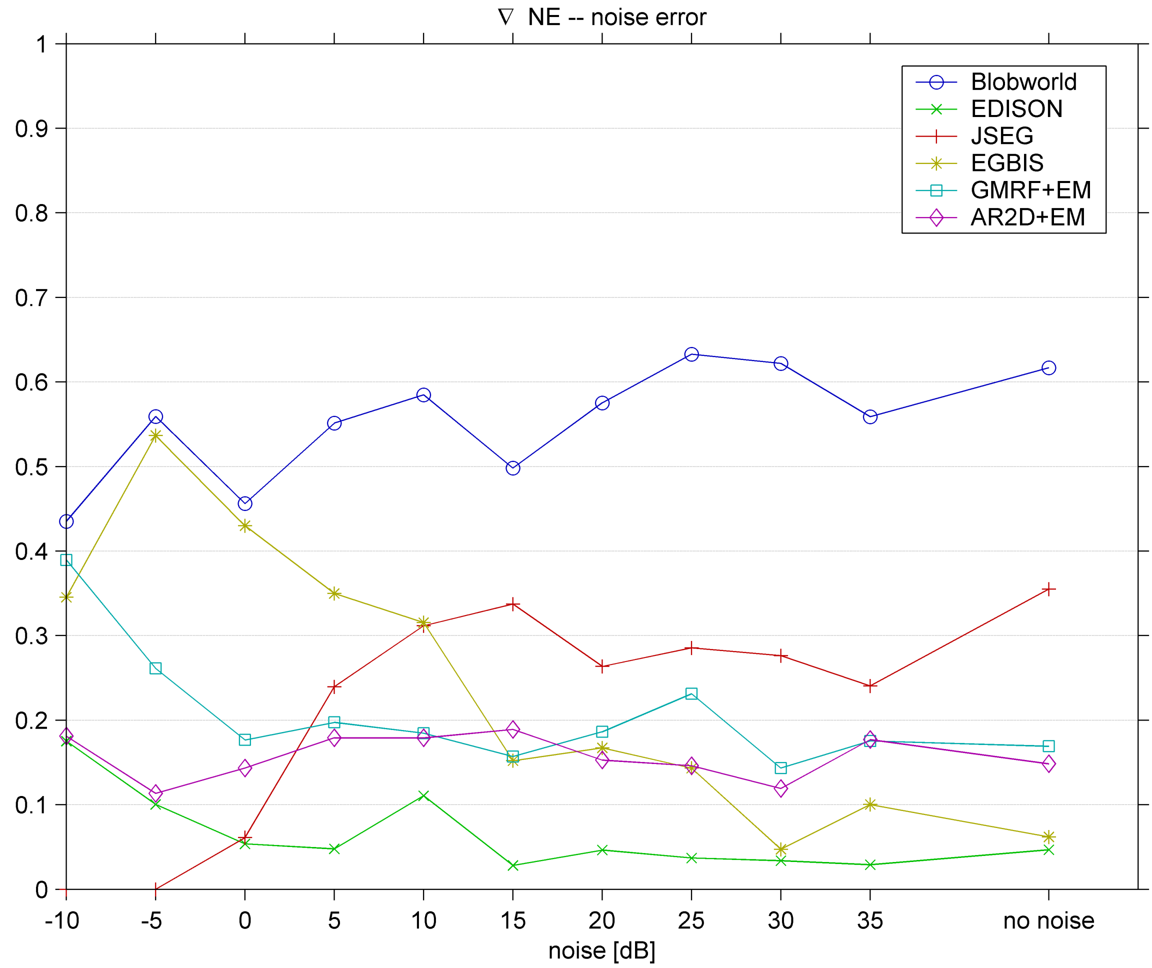


Figure B.5: Noise robustness graph – NE – noise error.

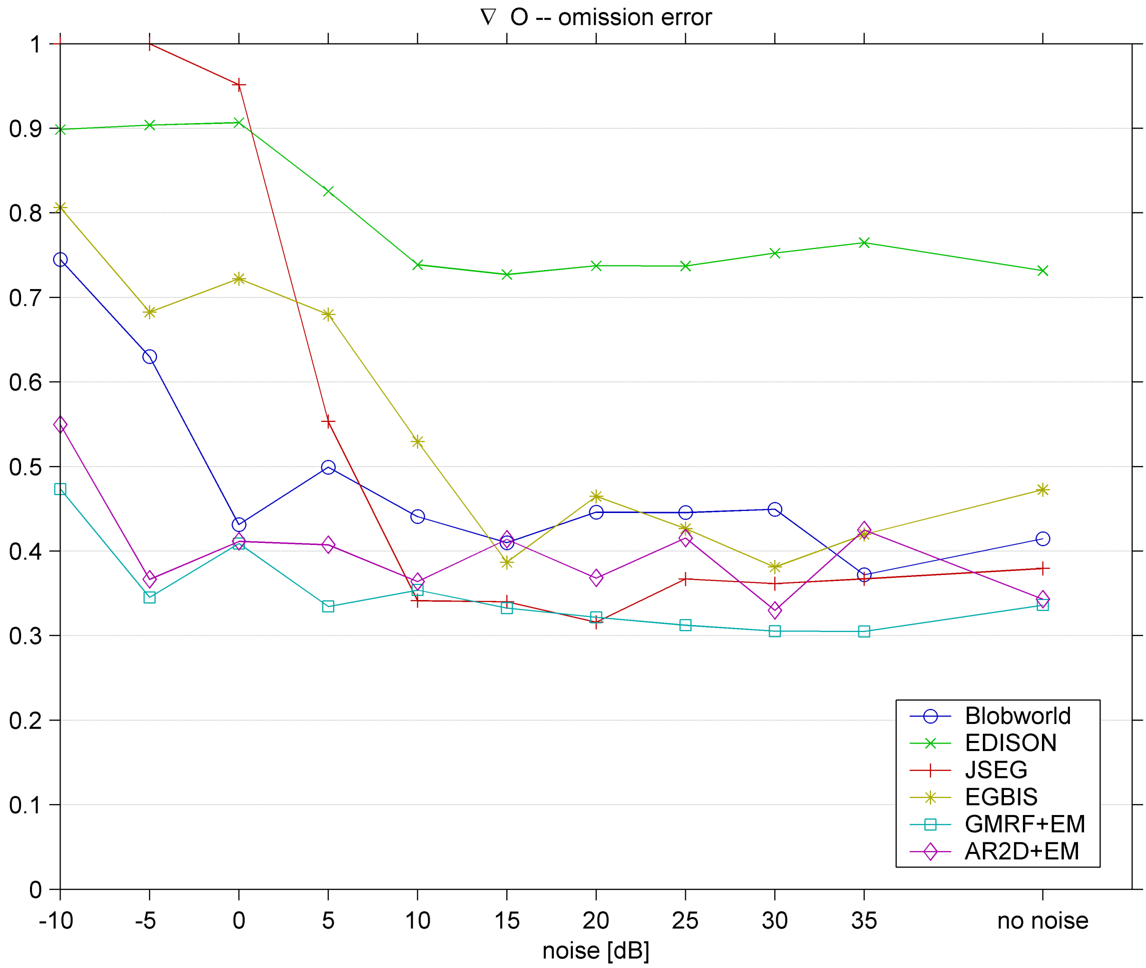


Figure B.6: Noise robustness graph – O – omission error.

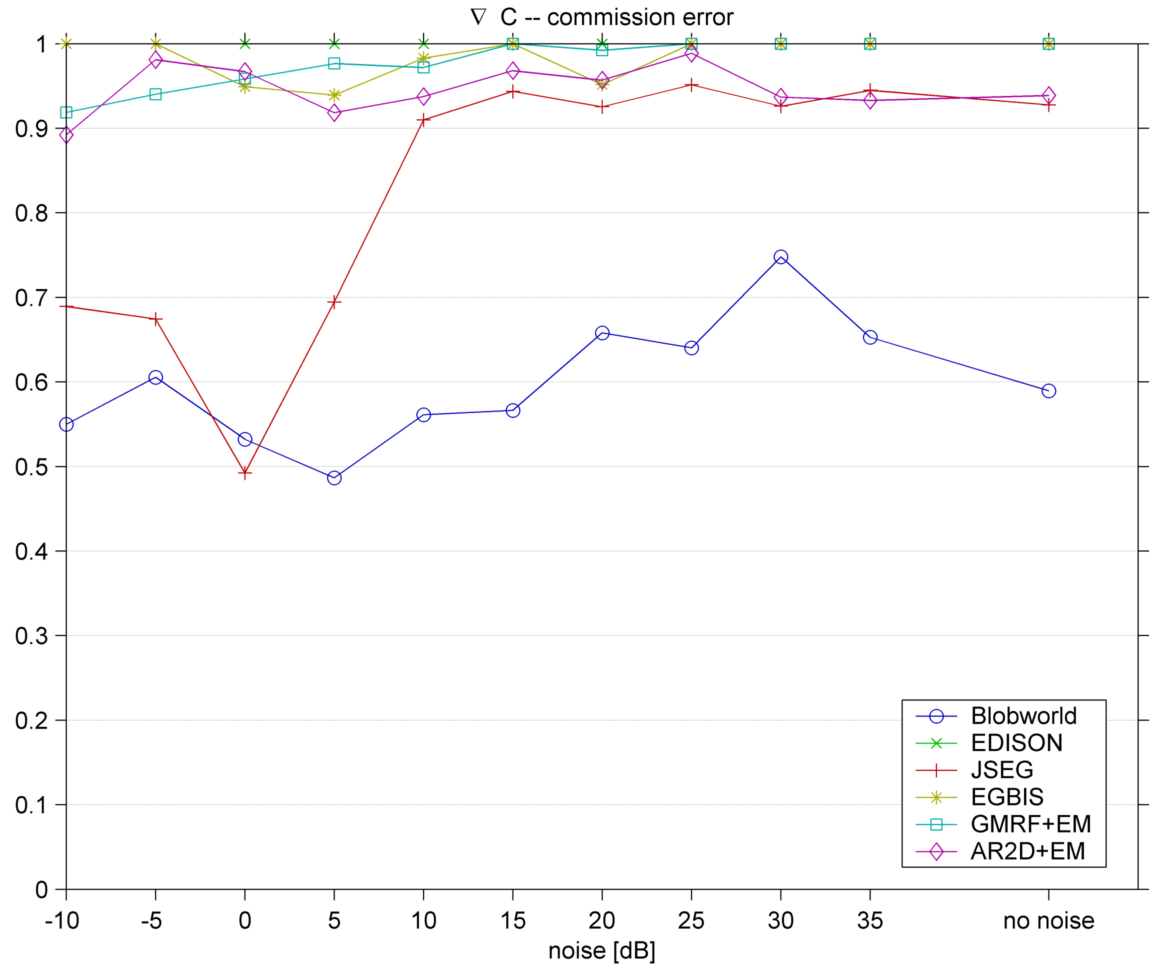


Figure B.7: Noise robustness graph – C – commission error.

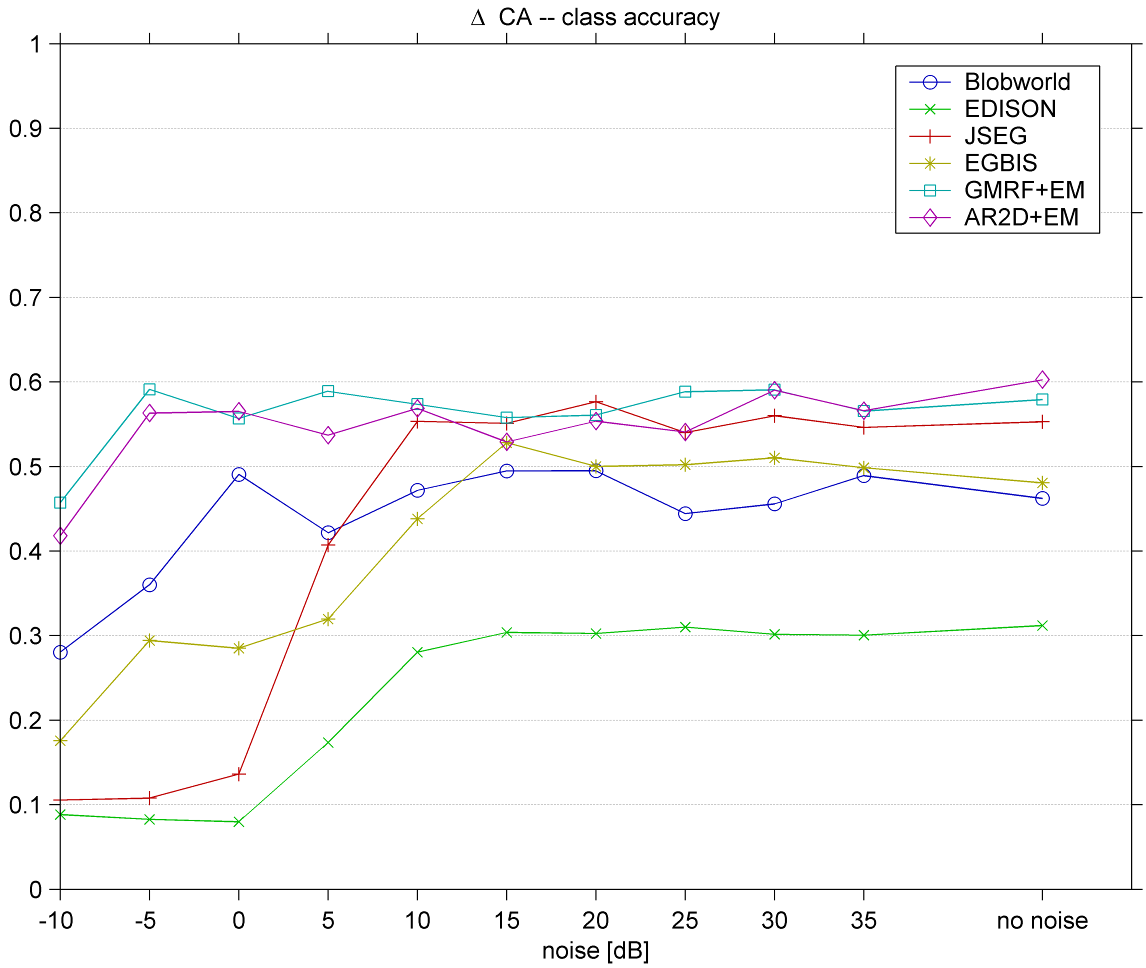


Figure B.8: Noise robustness graph – CA – class accuracy.

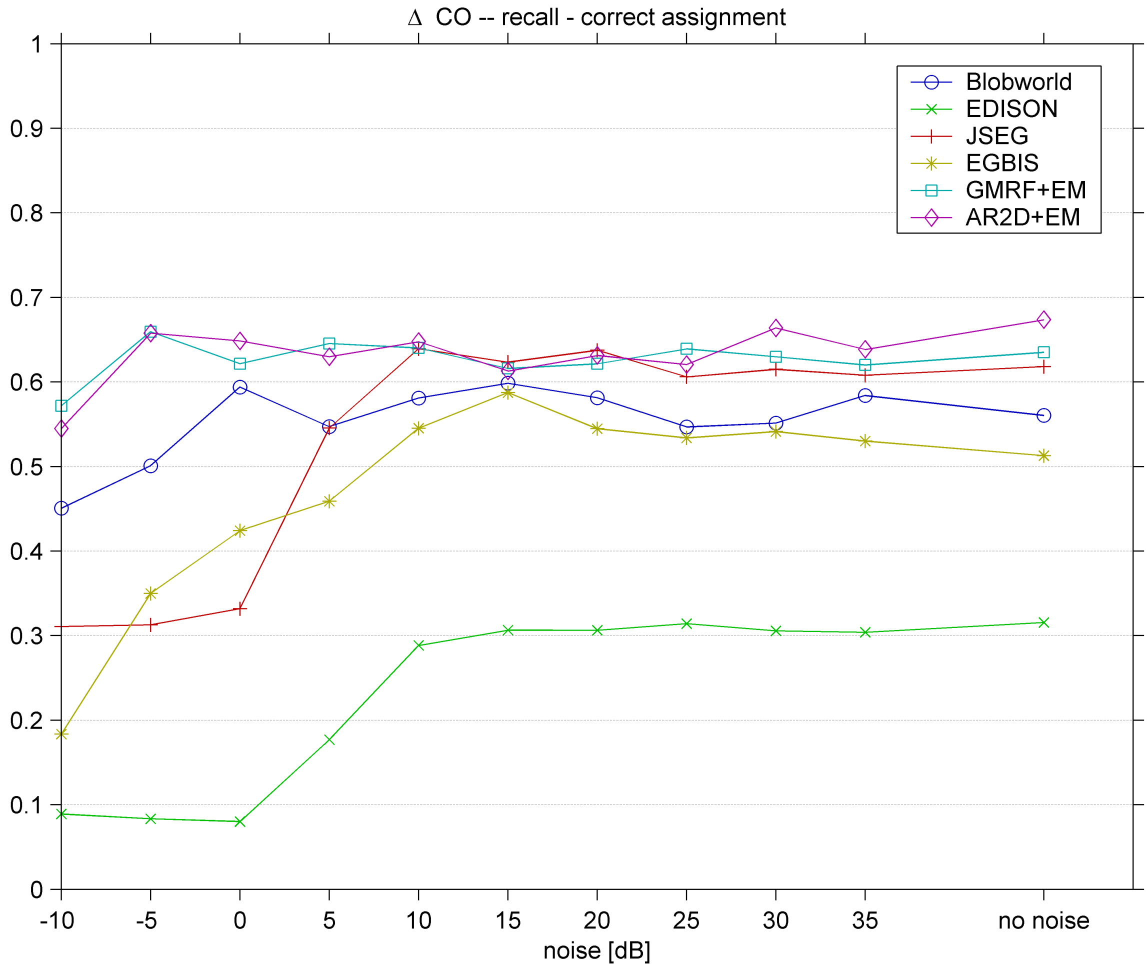


Figure B.9: Noise robustness graph – CO – recall - correct assignment.

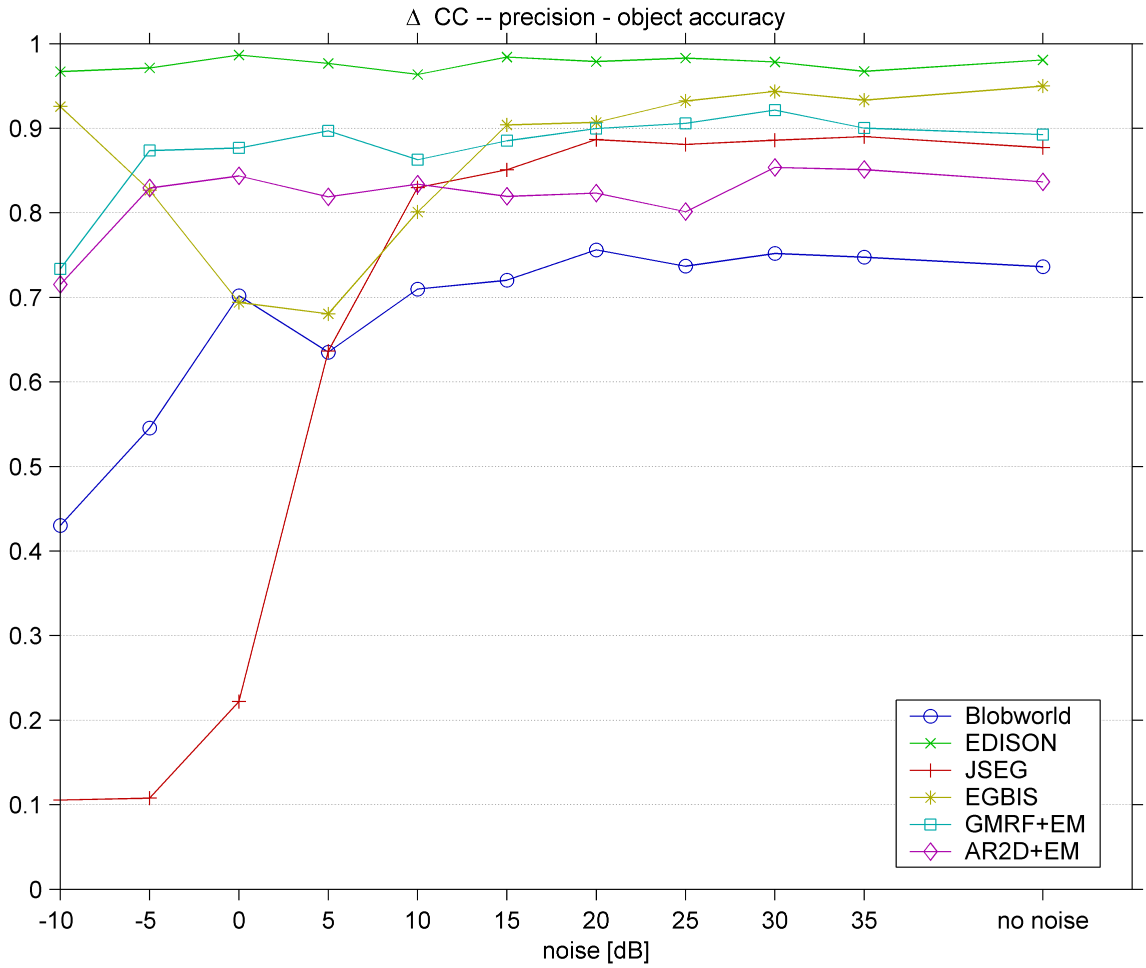


Figure B.10: Noise robustness graph – CC – precision - object accuracy.

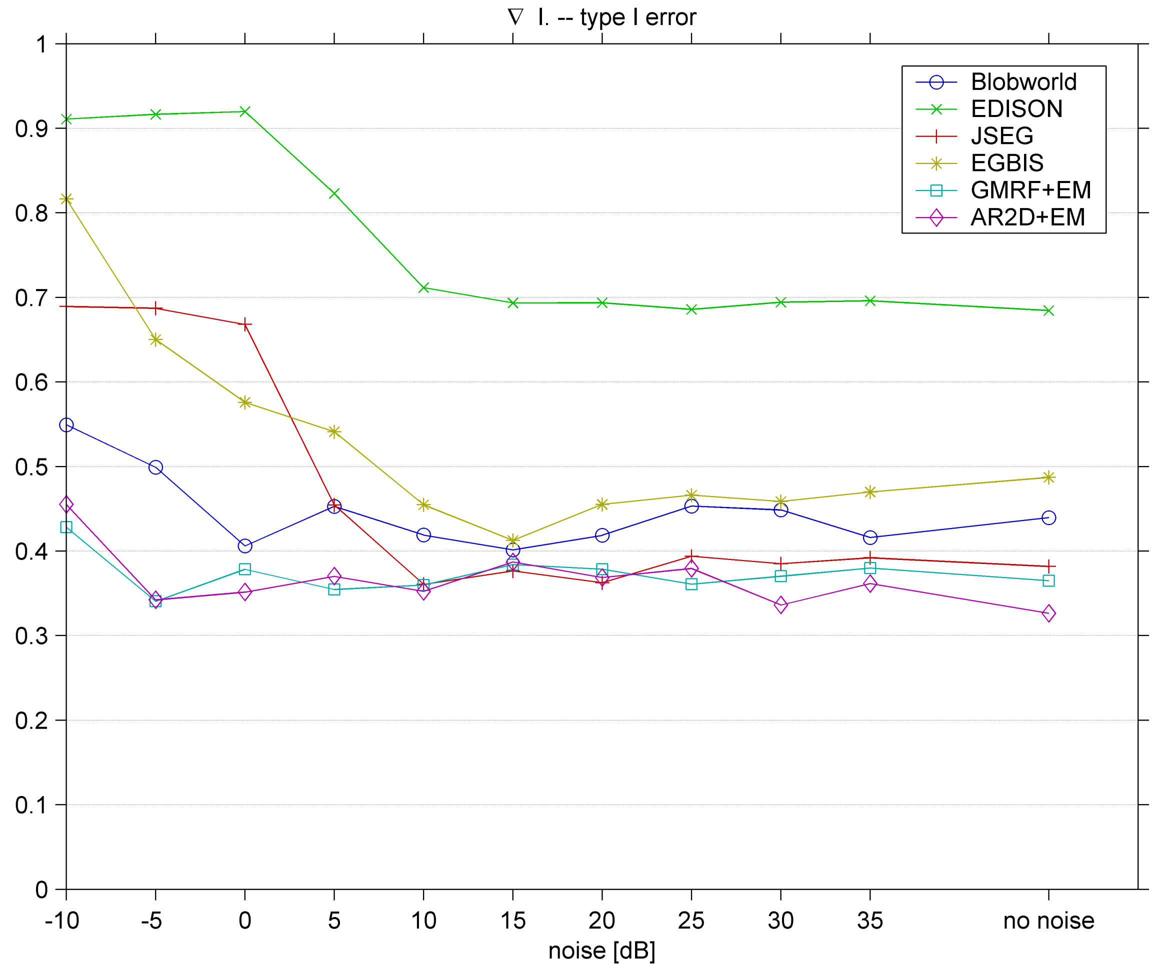


Figure B.11: Noise robustness graph – $I.$ – type I error.

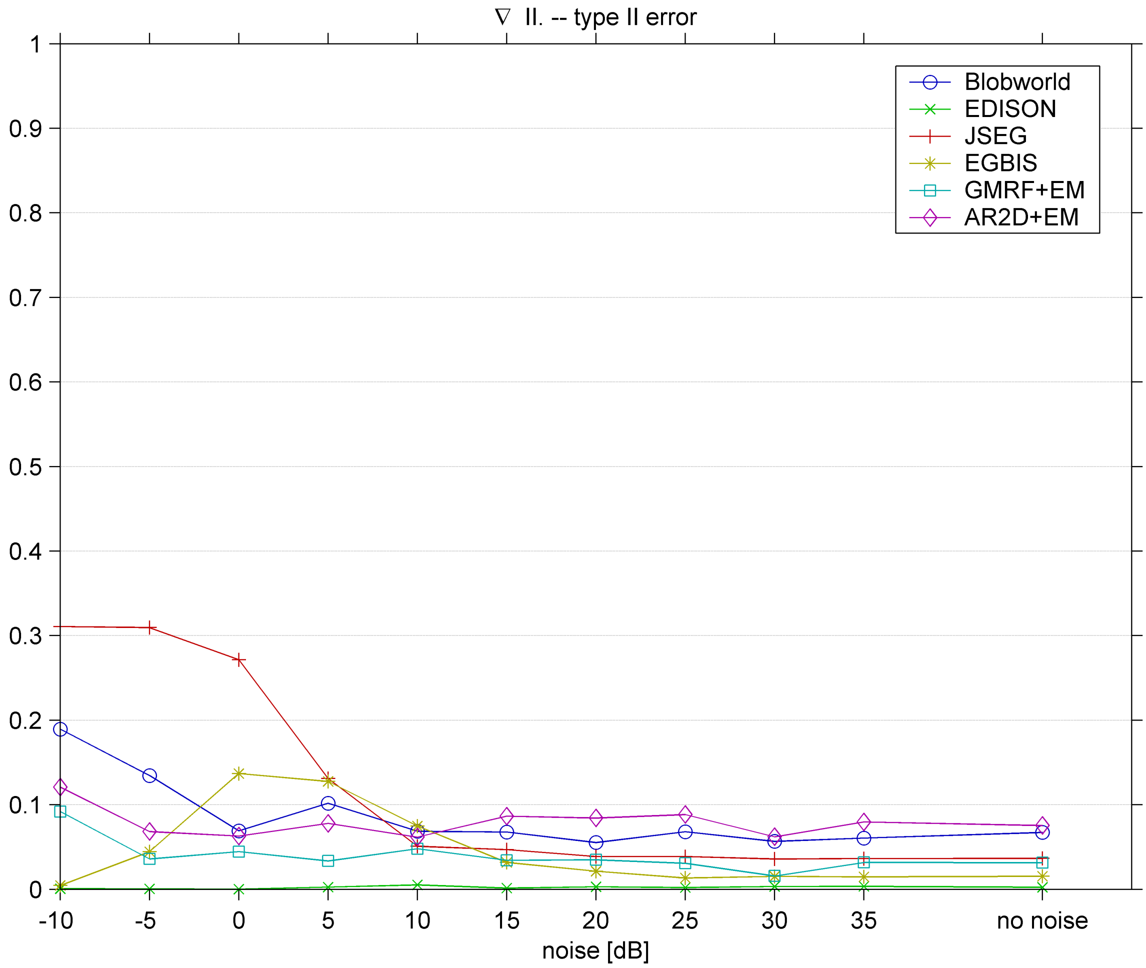


Figure B.12: Noise robustness graph – $II.$ – type II error.

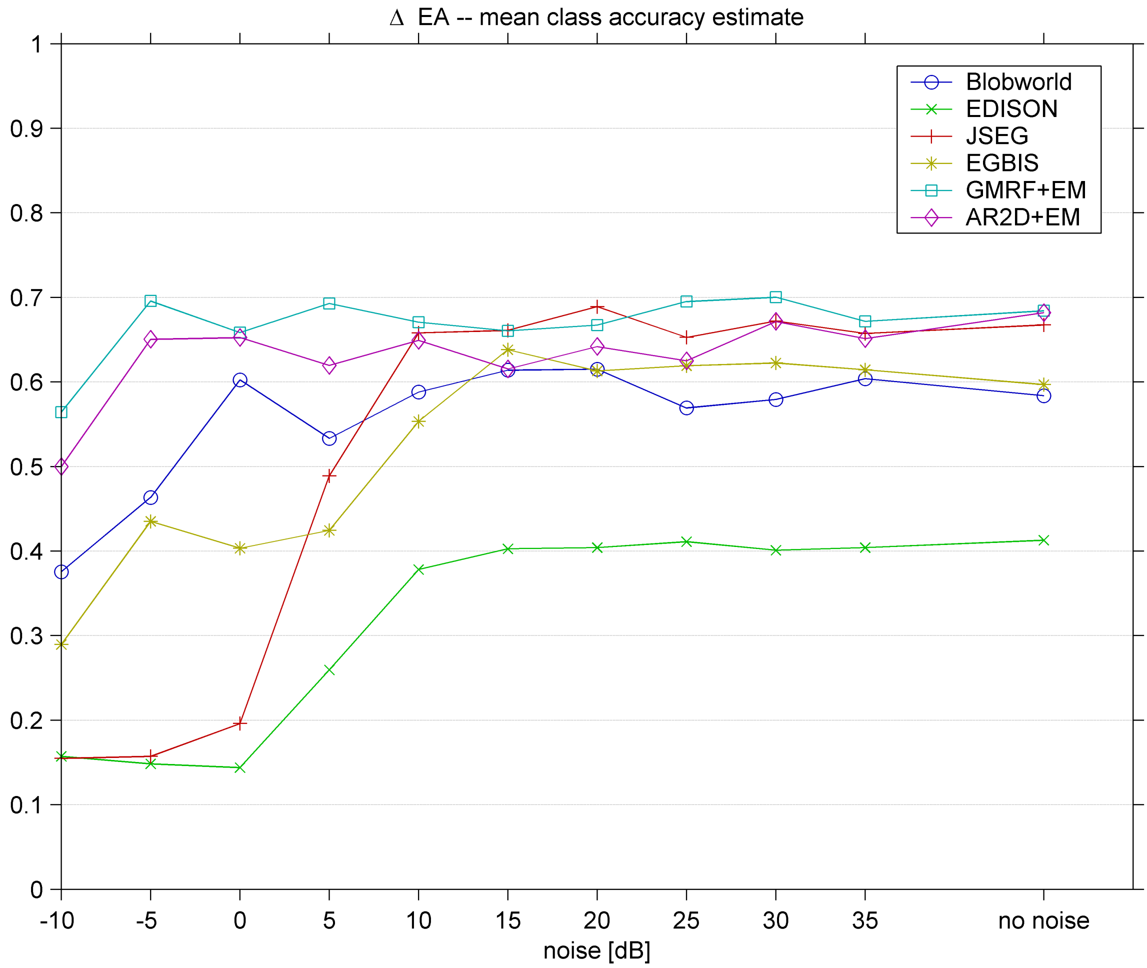


Figure B.13: Noise robustness graph – EA – mean class accuracy estimate.

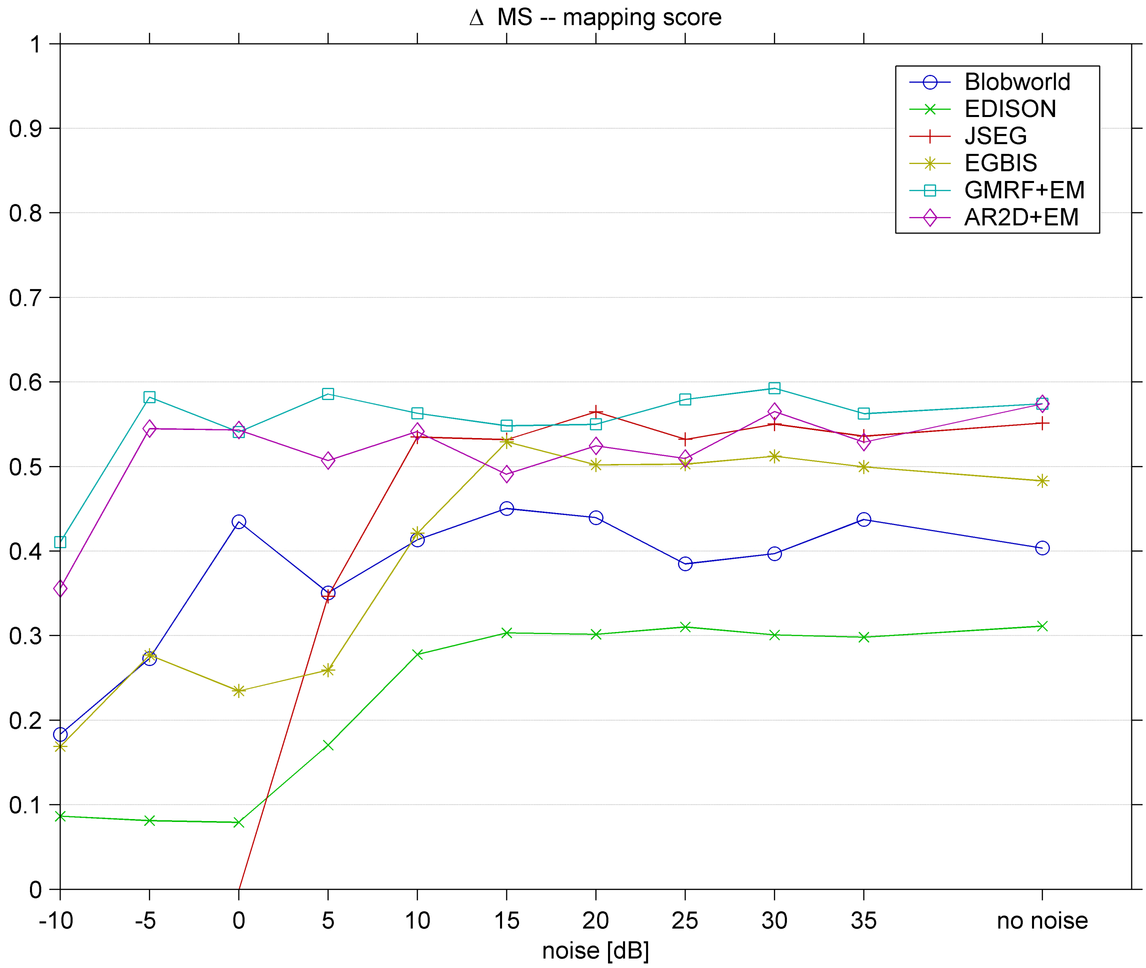


Figure B.14: Noise robustness graph – MS – mapping score.

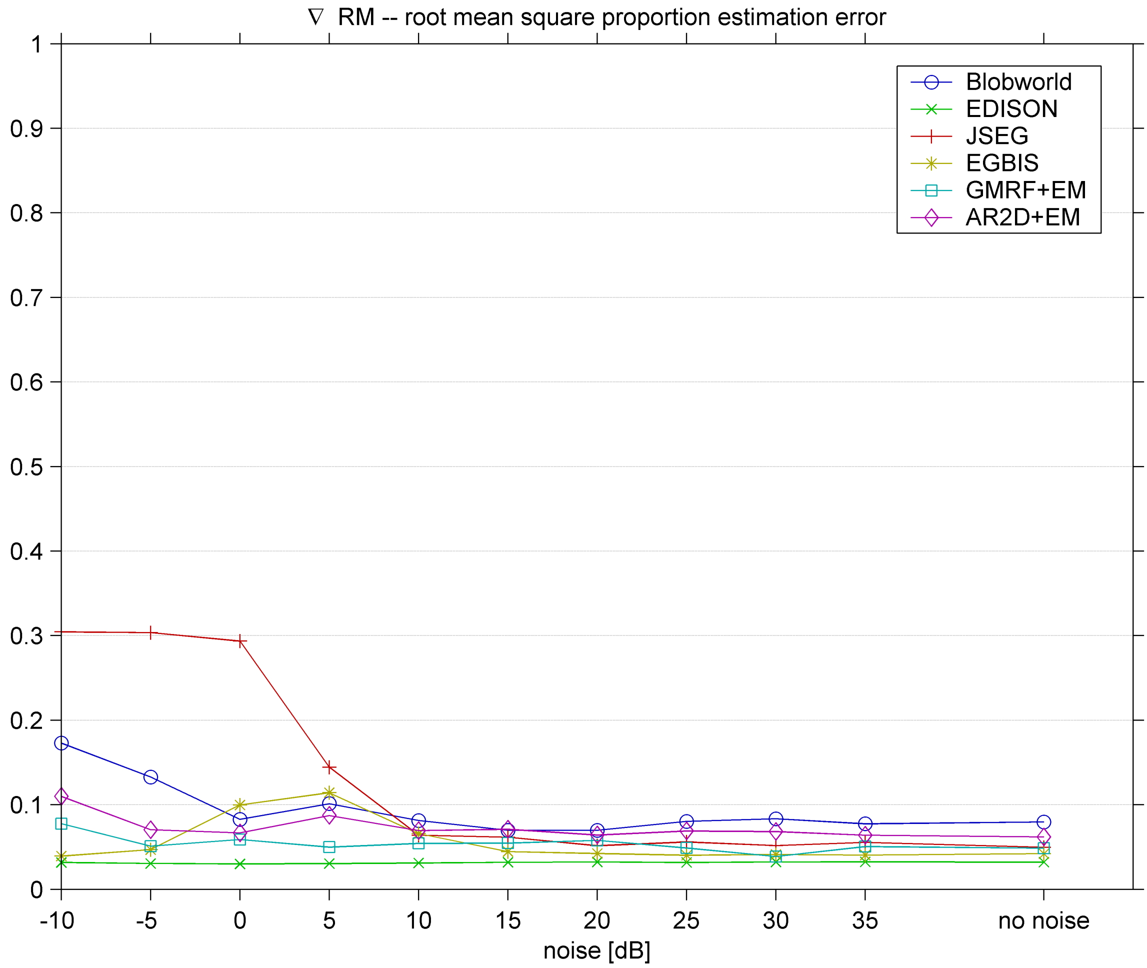


Figure B.15: Noise robustness graph – RM – root mean square proportion estimation error.

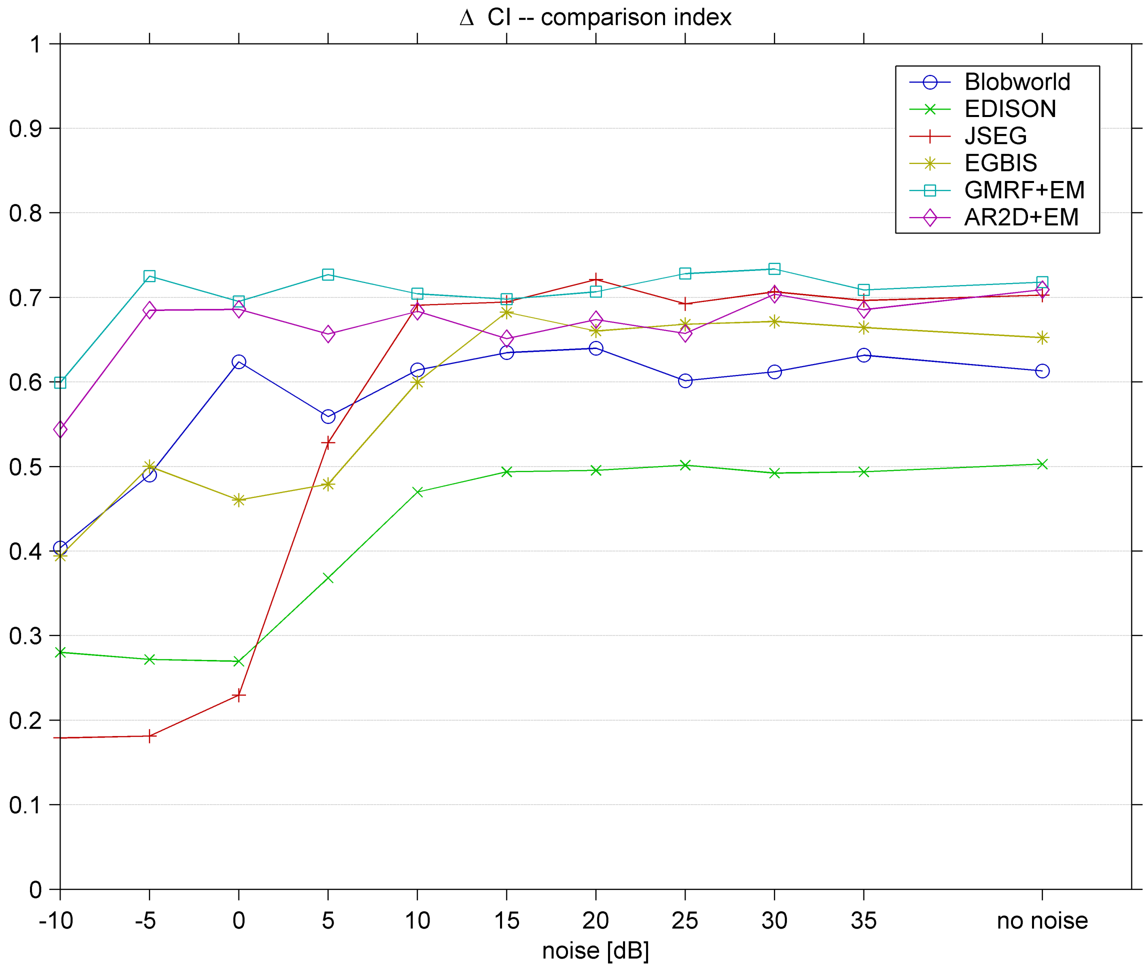


Figure B.16: Noise robustness graph – CI – comparison index.

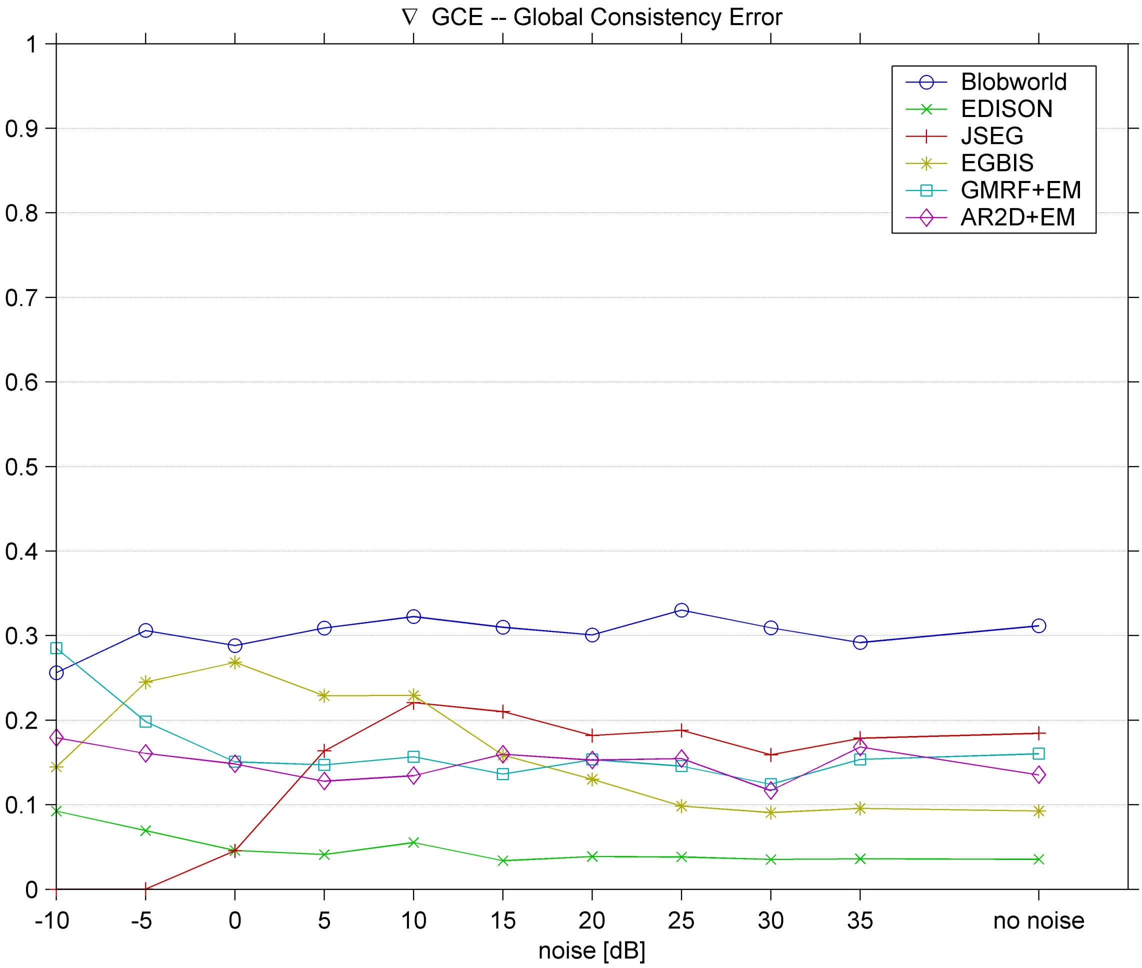


Figure B.17: Noise robustness graph – GCE – Global Consistency Error.

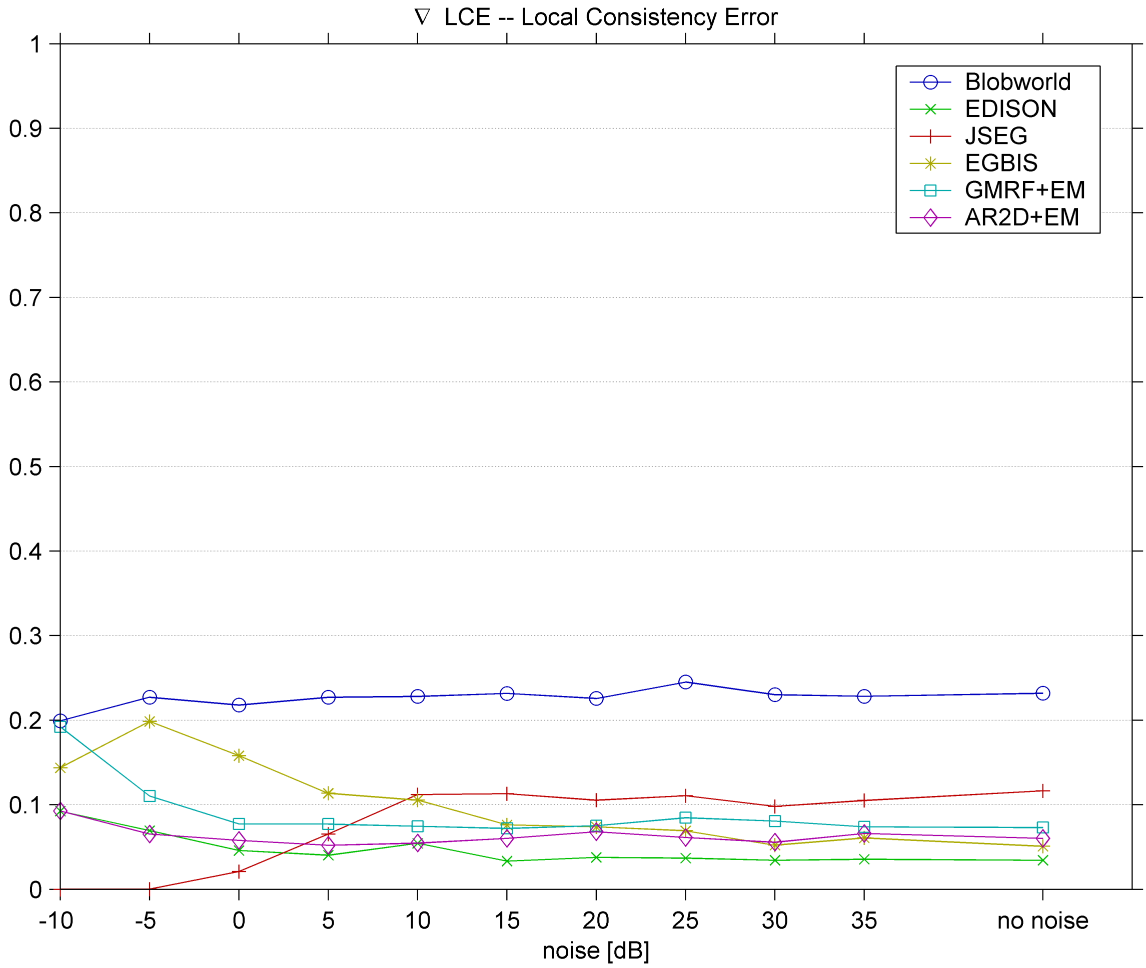


Figure B.18: Noise robustness graph – LCE – Local Consistency Error.

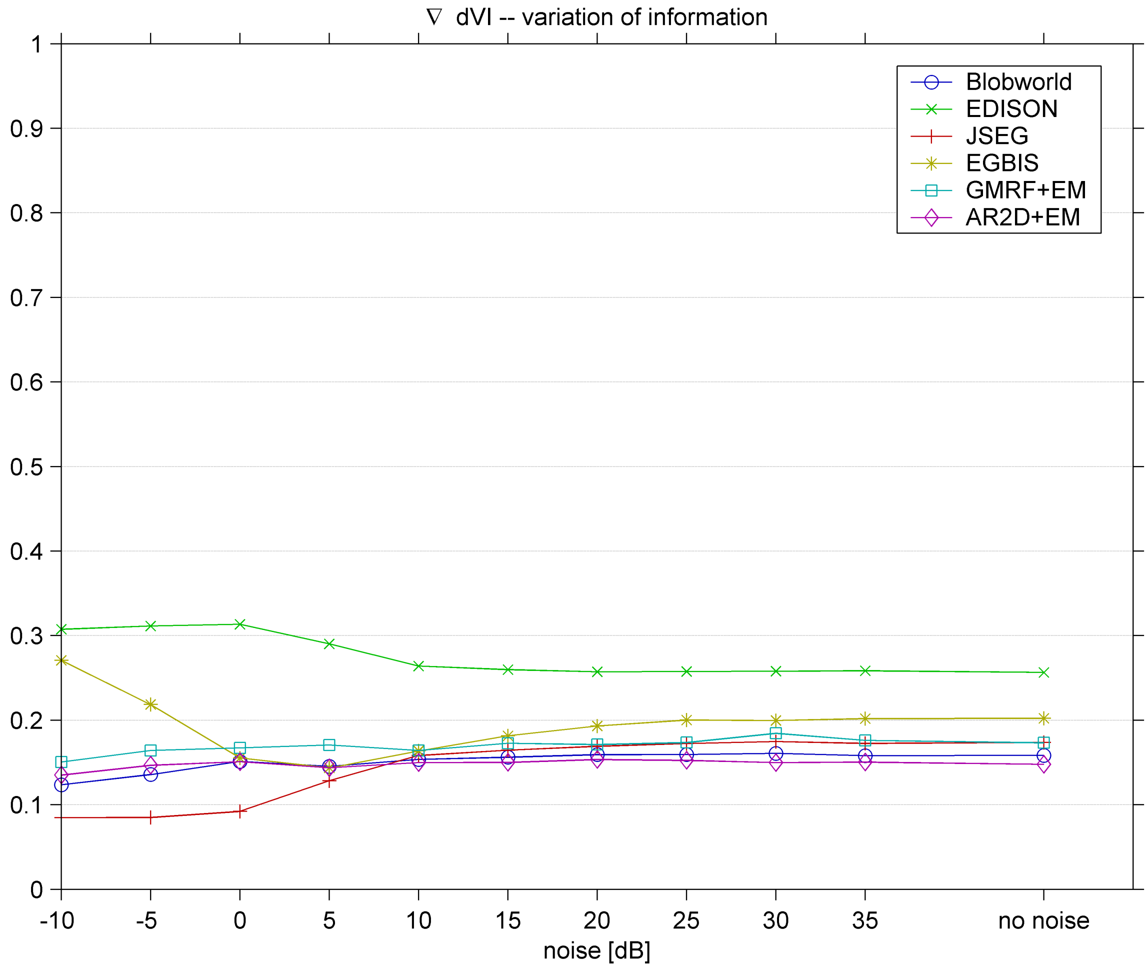


Figure B.19: Noise robustness graph – dVI – variation of information.

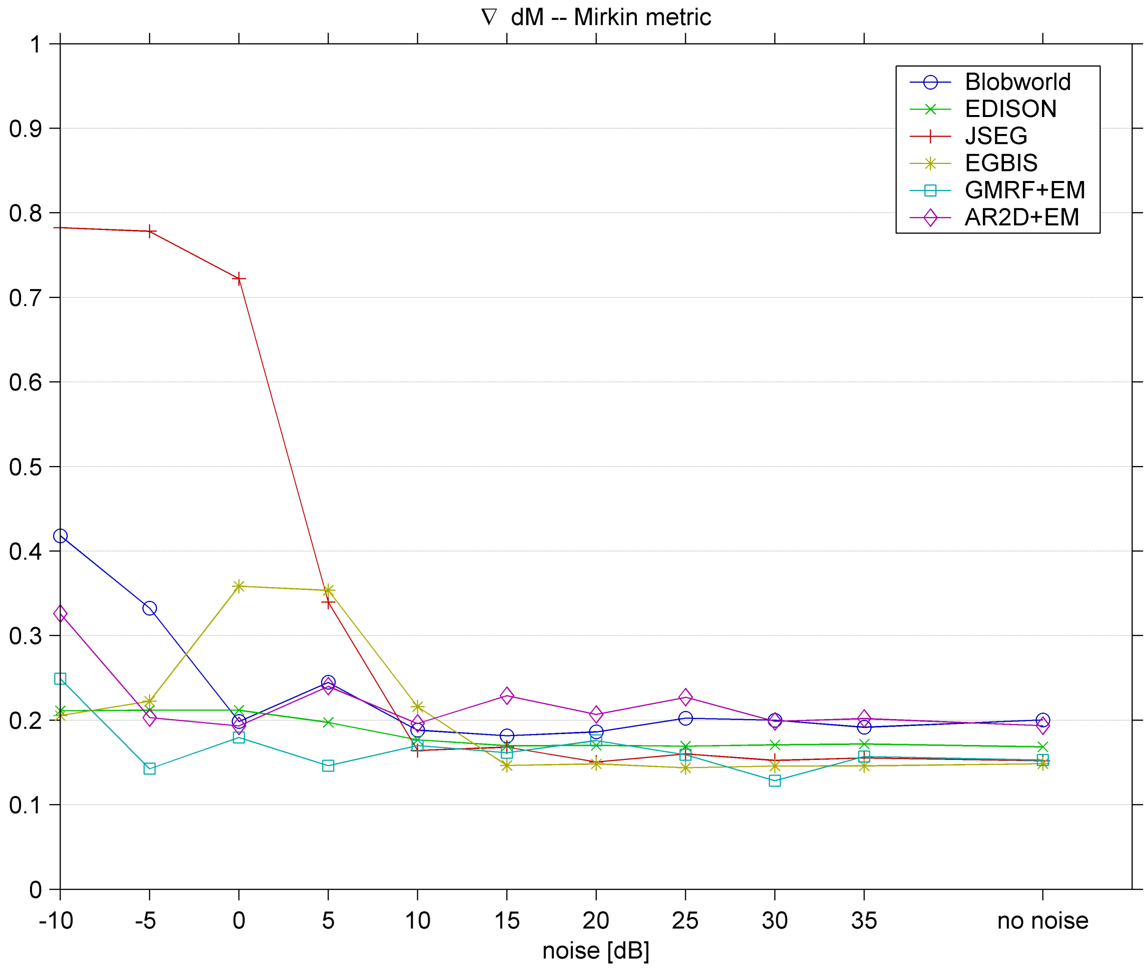


Figure B.20: Noise robustness graph – dM – Mirkin metric.

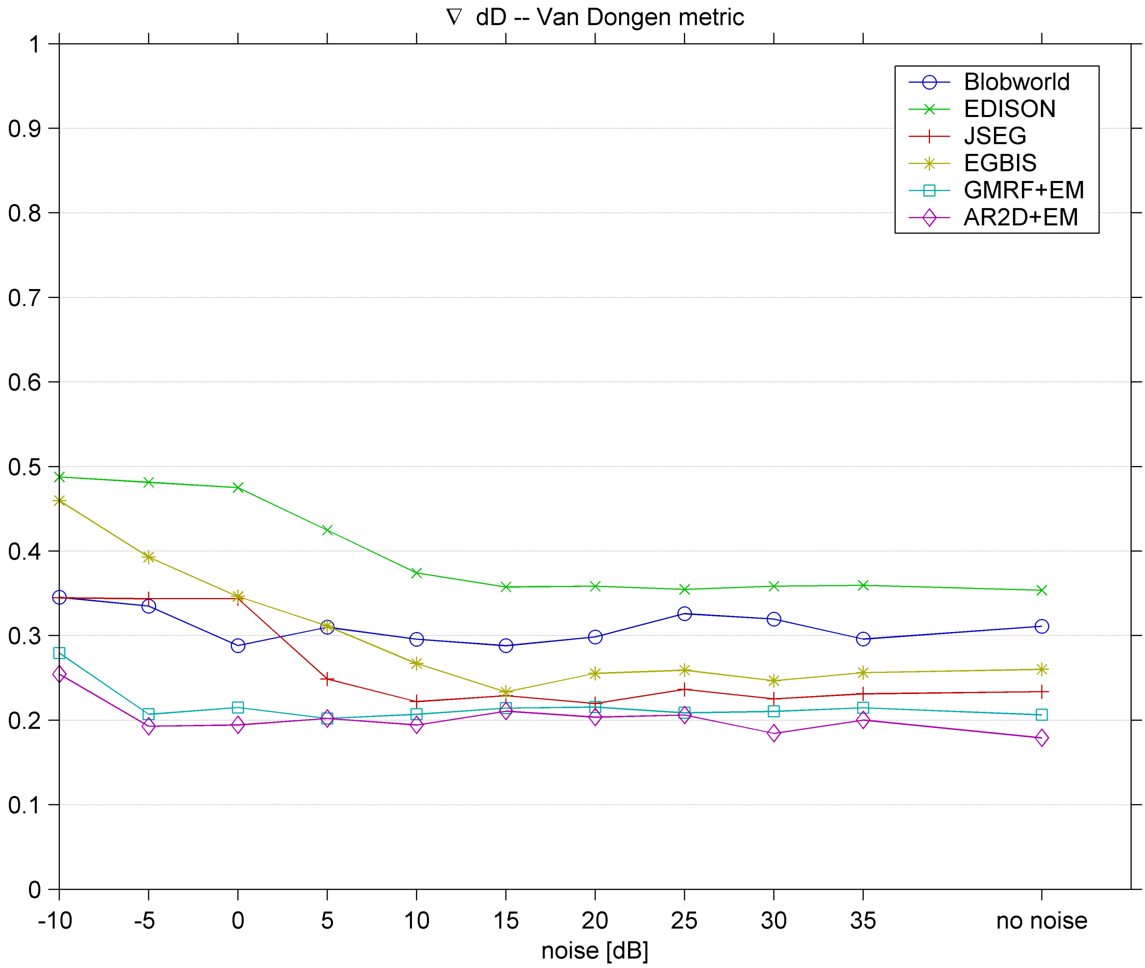


Figure B.21: Noise robustness graph – dD – Van Dongen metric.

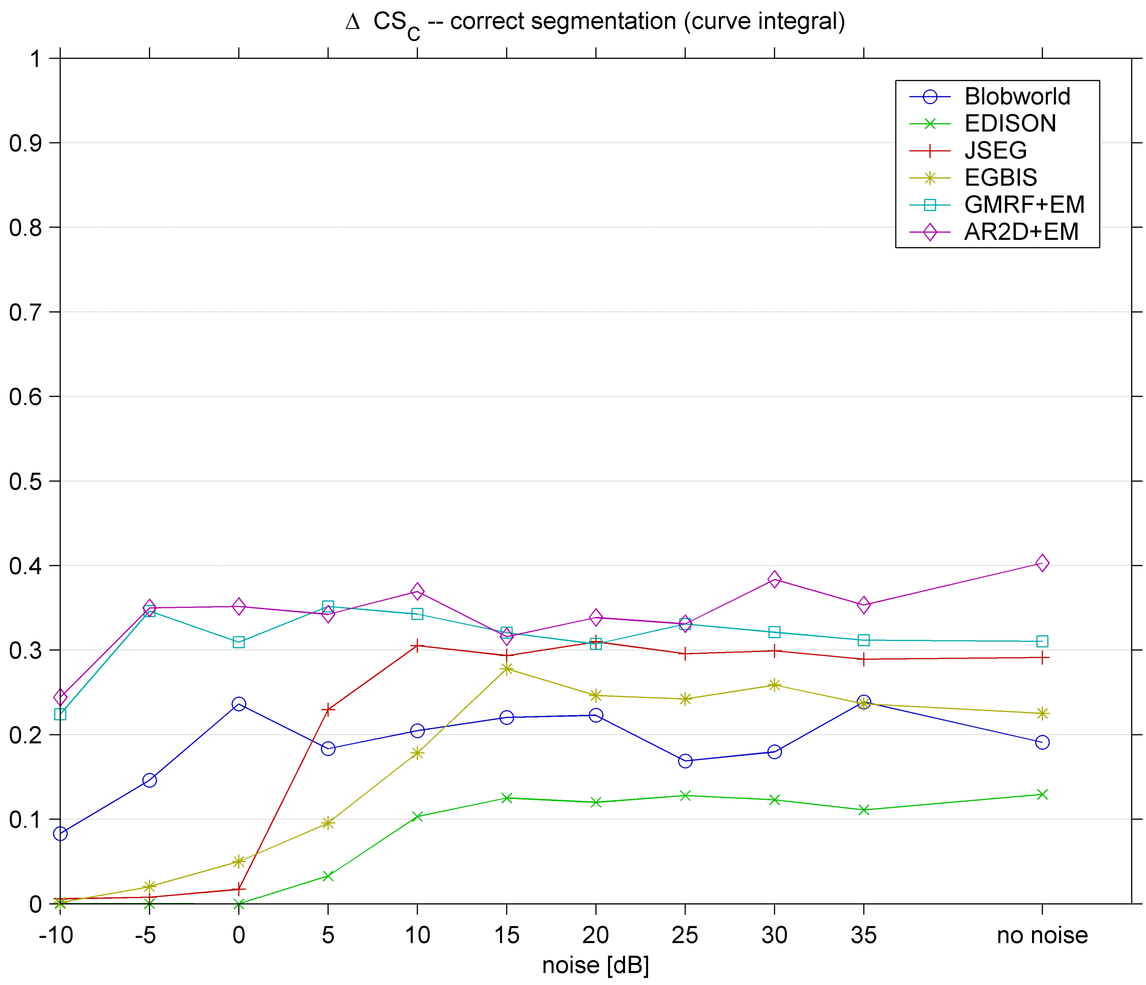


Figure B.22: Noise robustness graph – $\overline{CS}$ – correct segmentation (curve integral).

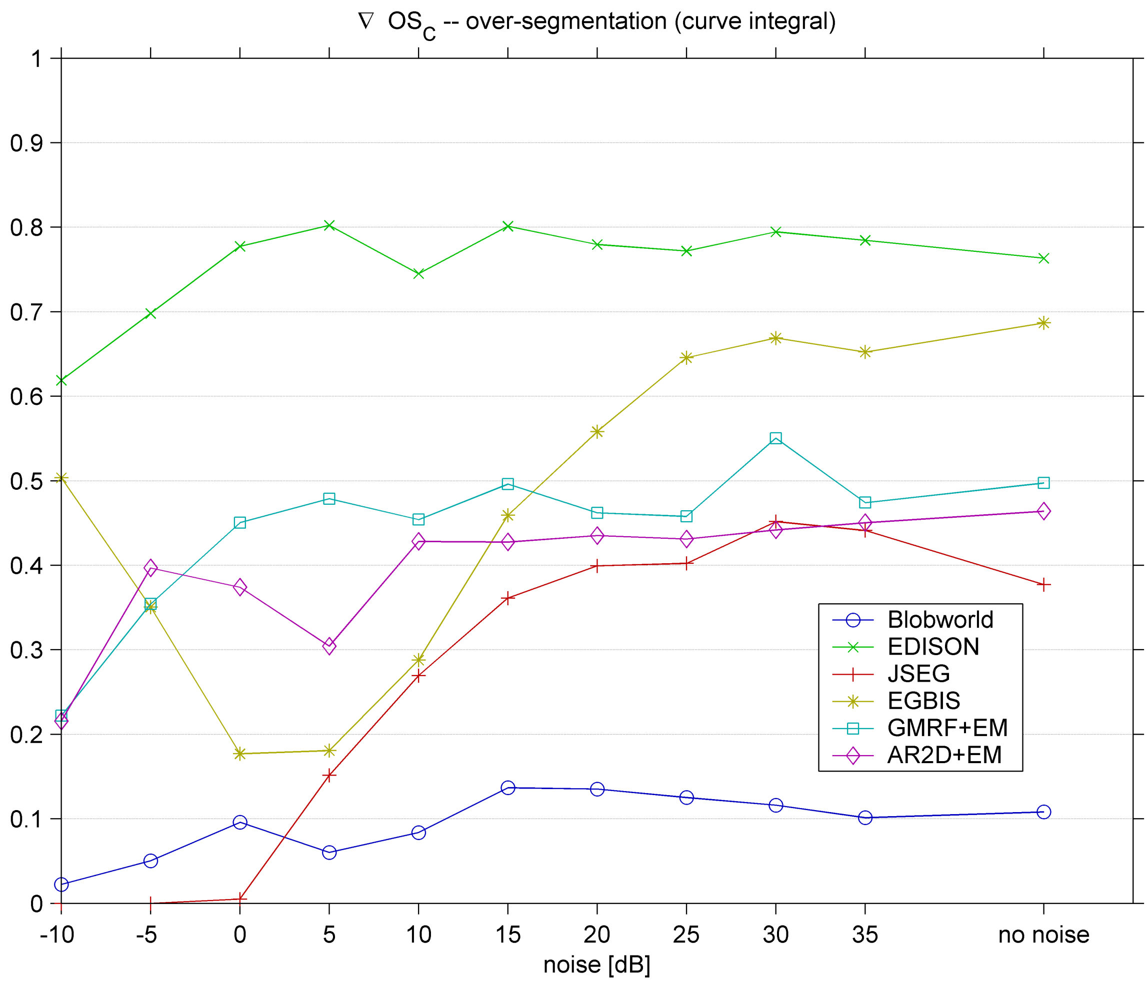


Figure B.23: Noise robustness graph – $\overline{OS}$ – over-segmentation (curve integral).

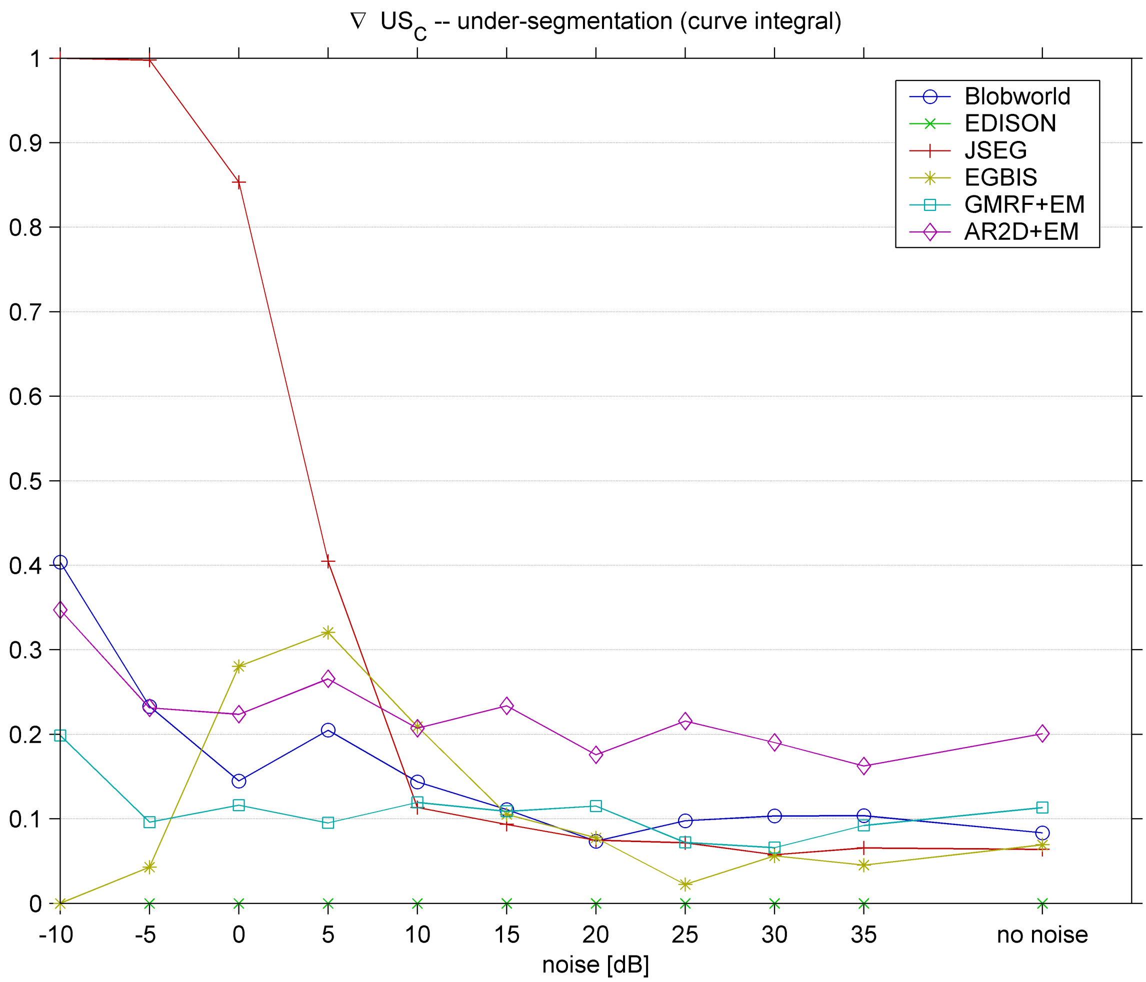


Figure B.24: Noise robustness graph – $\overline{US}$ – under-segmentation (curve integral).

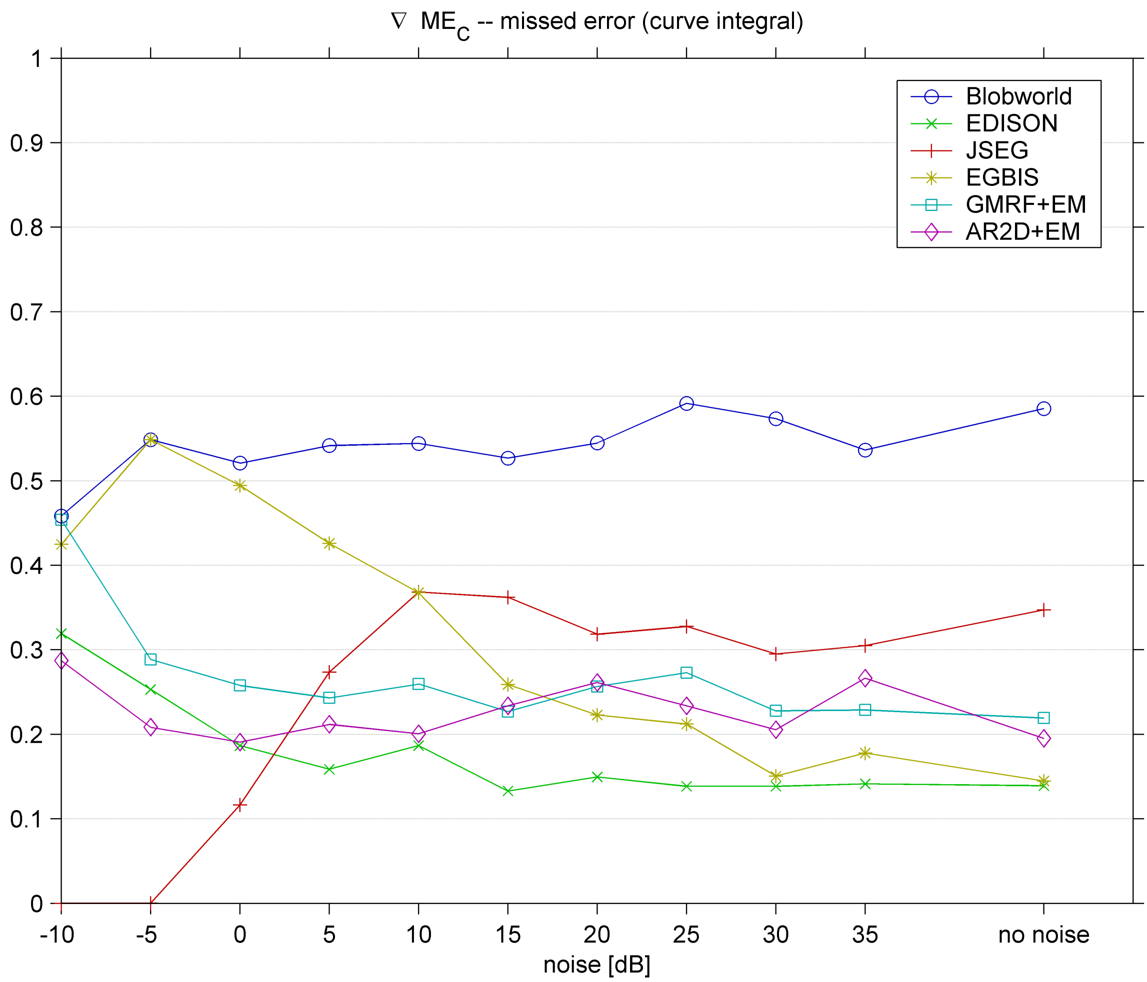


Figure B.25: Noise robustness graph – $\overline{ME}$ – missed error (curve integral).

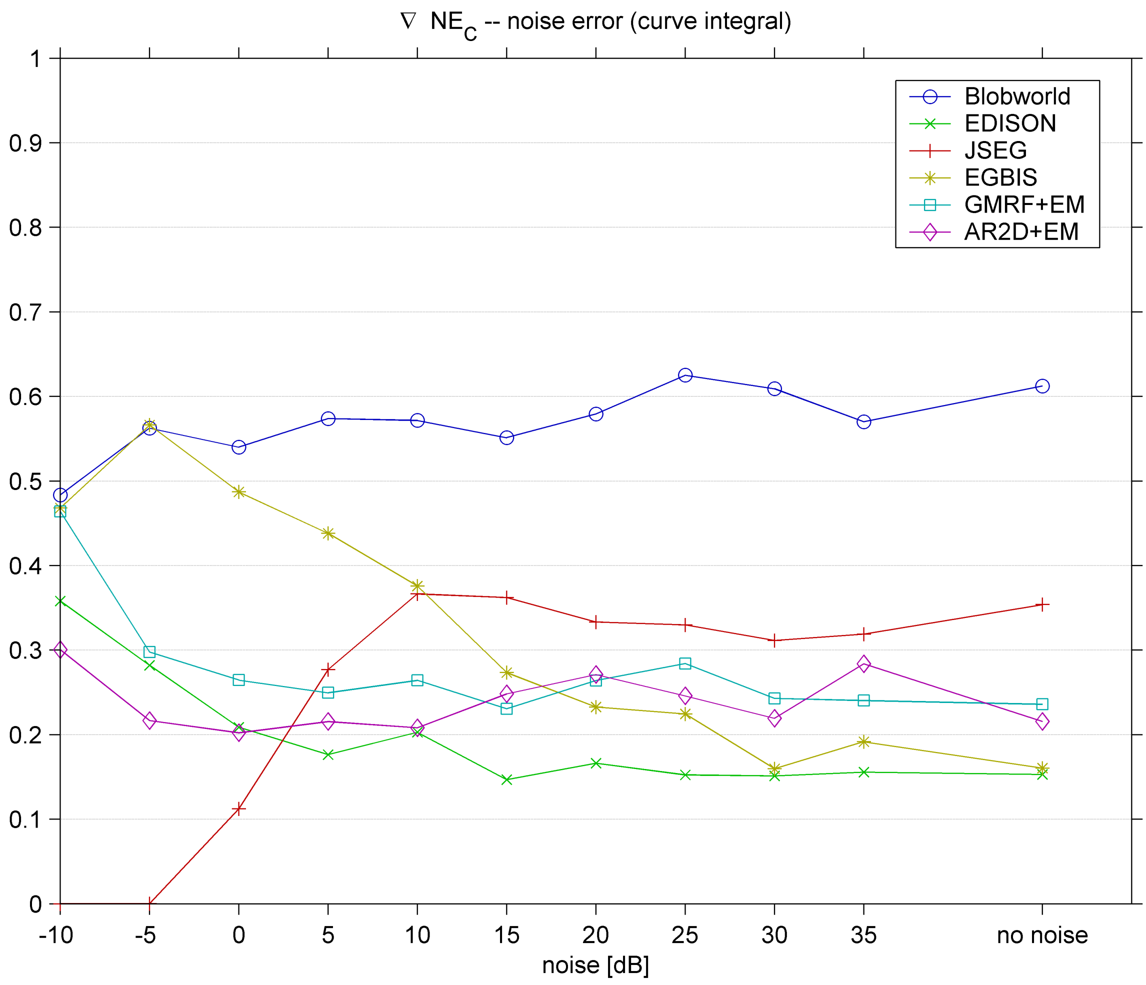


Figure B.26: Noise robustness graph – $\overline{NE}$ – noise error (curve integral).

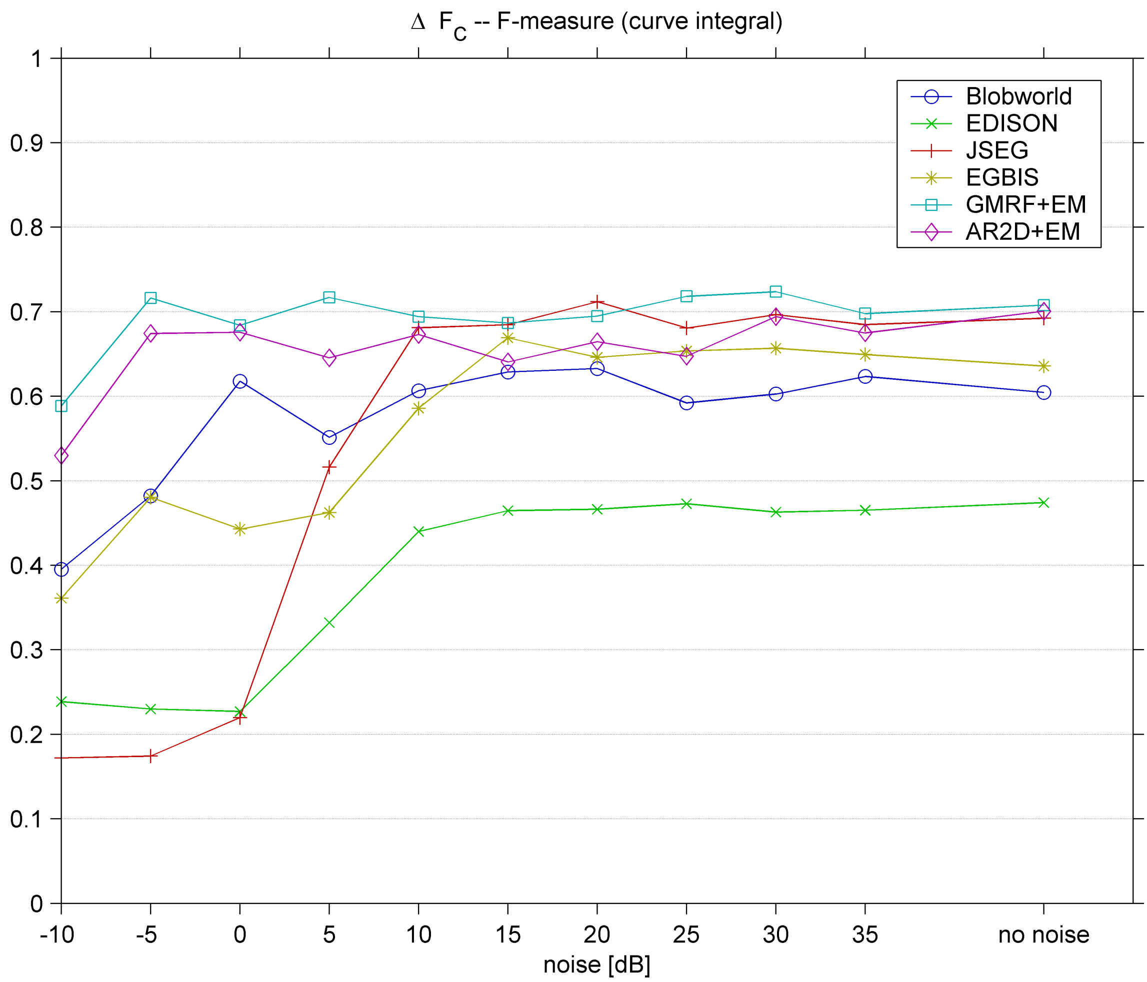
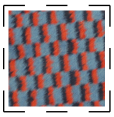


Figure B.27: Noise robustness graph – $\overline{F}$ – F-measure (curve integral).

Colour Layers Segmentation

Appendix C



In this appendix are results of segmentation of paint slices. Further details can be found in section 6.4.

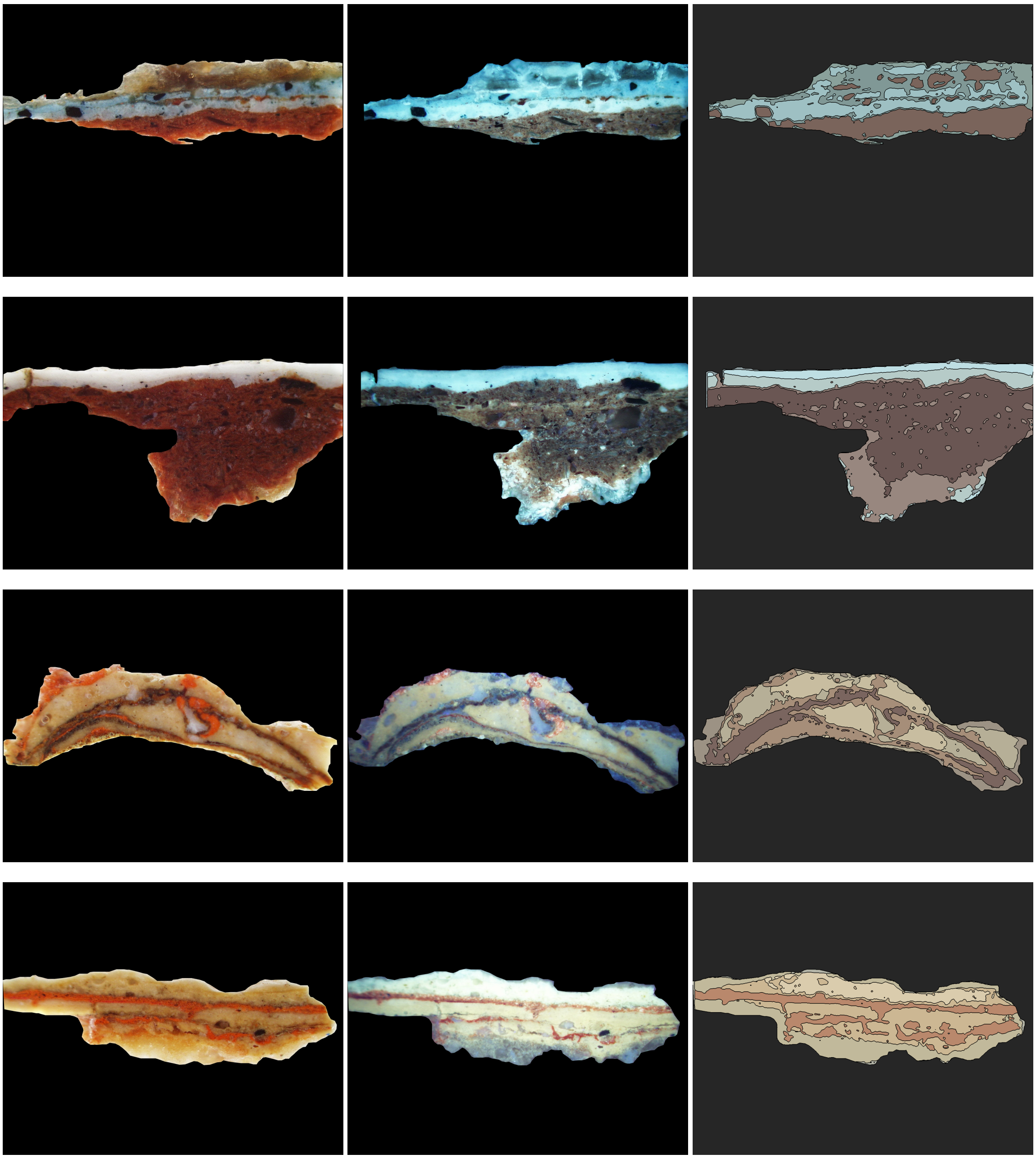


Figure C.1: Colour layers segmentation of paint slices – 019, 024, 025, 028; left column – images in the visible spectrum, middle column – images in the ultraviolet spectrum, and right column – resulting segmentations.

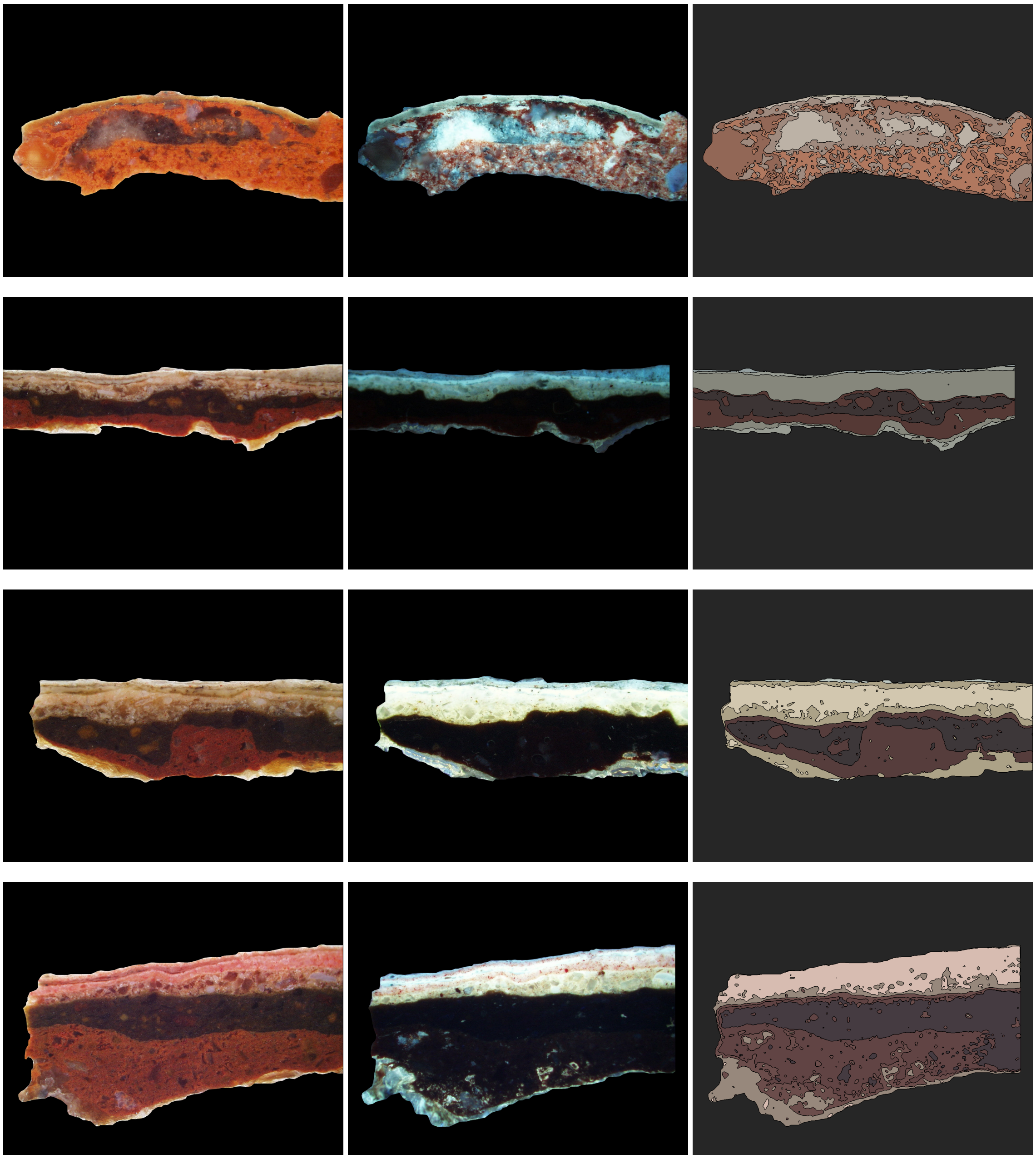


Figure C.2: Colour layers segmentation of paint slices – 030, 033, 034, 035; left column – images in the visible spectrum, middle column – images in the ultraviolet spectrum, and right column – resulting segmentations.

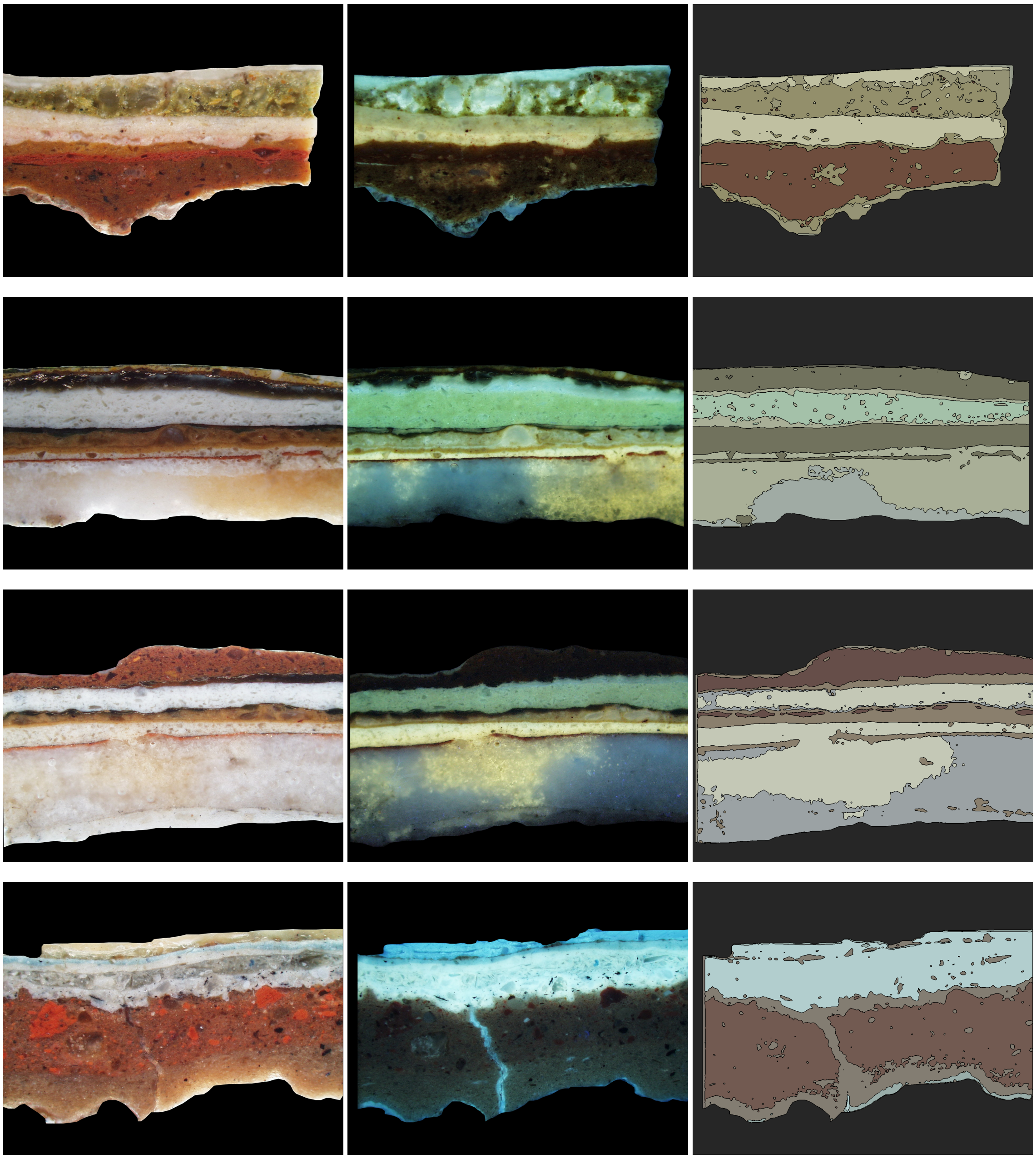
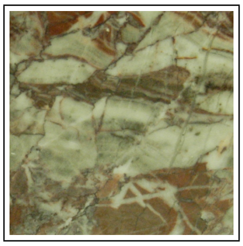
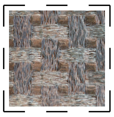


Figure C.3: Colour layers segmentation of paint slices – 038, 042, 043, 045; left column – images in the visible spectrum, middle column – images in the ultraviolet spectrum, and right column – resulting segmentations.





Bibliography

[1] Vision texture (VisTex) database. Technical report, Vision and Modeling Group, MIT Media Laboratory, 1995. *http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html*.

[2] T. Abe and Y. Matsuzawa. Clustering-based control of active contour model. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume II, pages 663–667, Washington, DC, USA, August 2002. IEEE Computer Society.

[3] P. Andrey and P. Tarroux. Unsupervised segmentation of Markov random field modeled textured images using selectionist relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):252–262, March 1998.

[4] A. Barbu and S.-C. Zhu. Multigrid and multi-level Swendsen–Wang cuts for hierarchic graph partition. In *Proceedings of the Computer Vision and Pattern Recognition, CVPR 2004*, volume II, pages 731–738, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[5] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using EM and its application to content-based image retrieval. In *Proceedings of the Sixth International Conference on Computer Vision, ICCV 1998*, pages 675–682. IEEE Computer Society, 1998.

[6] M. Beneš, B. Zitová, J. Hradilová, and D. Hradil. Image processing in material analyses of artworks. In A. Ranchordas and H. Araújo, editors, *Proceedings of the Third International Conference on Computer Vision Theory and Applications, VISAPP 2008*, volume 1, pages 521–524. INSTICC – Institute for Systems and Technologies of Information, Control and Communication, January 2008.

[7] M. Beneš, B. Zitová, F. Jan, J. Hradilová, and D. Hradil. The image processing system for art specimens: Nephele. In *Proceedings of the 14th European Signal Processing Conference, EUSIPCO 2006*, pages 1–5, September 2006.

[8] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press Inc., New York, 1995.

[9] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *Proceedings of the Ninth International Conference on Computer Vision, ICCV 2003*, pages 26–33. IEEE Computer Society, 2003.

[10] T. Brox and J. Weickert. Level set segmentation with multiple regions. *IEEE Transactions on Image Processing*, 15(10):3213–3218, October 2006.

[11] F. Calderero and F. Marqués. General region merging approaches based on information theory statistical measures. In *Proceedings of the International Conference on Image Processing, ICIP 2008*, pages 3016–3019. IEEE, October 2008.

[12] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In *Proceedings of the Third International Conference on Visual Information and Information Systems, VISUAL 1999*, pages 509–516, London, UK, 1999. Springer-Verlag.

[13] A. Chakraborty and J. S. Duncan. Game-theoretic integration for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(1):12–30, 1999.

[14] P. Chanyagorn and K. B. Eom. Texture segmentation using moving average modeling approach. In *Proceedings of the International Conference on Image Processing, ICIP 2000*, volume 2, pages 116–119. IEEE, 2000.

[15] J.-L. Chen and A. Kundu. Unsupervised texture segmentation using multichannel decomposition and hidden Markov models. *IEEE Transactions on Image Processing*, 4(5):603–619, 1995.

[16] H. Cheng, X. Jiang, Y. Sun, and J. Wang. Color image segmentation: Advances and prospects. *Pattern Recognition*, 34:2259–2281, 2001.

[17] D. Chetverikov. Detecting defects in texture. In *Proceedings of the International Conference on Pattern Recognition, ICPR 1998*, pages 61–63, 1988.

[18] Y.-M. Cheung. Maximum weighted likelihood via rival penalized EM for density mixture clustering with automatic model selection. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):750–761, 2005.

[19] L. Chittaro, V. K. Gatla, and S. Venkataraman. The interactive 3D breakaway map: A navigation and examination aid for multi-floor 3D worlds. In *Proceedings of the 2005 International Conference on Cyberworlds, CW 2005*, pages 59–66, Washington, DC, USA, 2005. IEEE Computer Society.

[20] L. Chittaro, R. Ranon, and L. Ieronutti. VU-flow: A visualization tool for analyzing navigation in virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1475–1485, 2006.

[21] C. M. Christoudias, B. Georgescu, and P. Meer. Synergism in low level vision. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume IV, pages 150–155, Washington, DC, USA, August 2002. IEEE Computer Society.

[22] D. Comaniciu. Image segmentation using clustering with saddle point detection. In *Proceedings of the International Conference on Image Processing, ICIP 2002*, volume III, pages 297–300. IEEE, 2002.

[23] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

[24] C. Constantinopoulos and A. C. Likas. Image modeling and segmentation using incremental Bayesian mixture models. In W. G. Kropatsch, M. Kampel, and A. Hanbury, editors, *Proceedings of the 12th International Conference on Computer Analysis of Images and Patterns, CAIP 2007*, volume 4673 of *Lecture Notes in Computer Science*, pages 596–603. Springer, August 2007.

[25] R. P. Darken and J. L. Sibert. A toolset for navigation in virtual environments. In *Proceedings of the 6th Annual Symposium on User Interface Software and Technology*, pages 157–166, New York, NY, USA, November 1993. ACM Press.

[26] H. Deng and D. A. Clausi. Unsupervised image segmentation using a simple MRF model with a new implementation scheme. *Pattern Recognition*, pages 2323–2335, 2004.

[27] Y. Deng and B. Manjunath. Unsupervised segmentation of color-texture regions in images and video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8):800–810, August 2001.

[28] H. Derin and H. Elliot. Modeling and segmentation of noisy and textured images using Gibbs random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):39–55, 1987.

[29] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[30] M. Donoser and H. Bischof. ROI–SEG: Unsupervised color segmentation by combining differently focused sub results. In *Proceedings of the Computer Vision and Pattern Recognition conference, CVPR 2007*, pages 1–8. IEEE Press, June 2007.

[31] M. Donoser and H. Bischof. Using covariance matrices for unsupervised texture segmentation. In B. Lovell, D. Laurendeau, and R. Duin, editors, *Proceedings*

of the 19th International Conference on Pattern Recognition, ICPR 2008, pages 1–4. IEEE Computer Society, December 2008.

[32] M. Donoser, M. Urschler, M. Hirzer, and H. Bischof. Saliency driven total variation segmentation. In *Proceedings of the 12th International Conference on Computer Vision, ICCV 2009*, 2009. To appear.

[33] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd ed.)*. John Wiley and Sons, New York, 2001.

[34] O. Ecabert and J.-P. Thiran. Variational image segmentation by unifying region and boundary information. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume II, pages 885–888, Washington, DC, USA, August 2002. IEEE Computer Society.

[35] F. J. Estrada and A. D. Jepson. Quantitative evaluation of a novel image segmentation algorithm. In *Proceedings of the Computer Vision and Pattern Recognition, CVPR 2005*, volume II, pages 1132–1139. IEEE, 2005.

[36] M. Everingham, H. Muller, and B. T. Thomas. Evaluating image segmentation algorithms using the pareto front. In *Proceedings of the 7th European Conference on Computer Vision, ECCV 2002 – Part IV*, volume 2353 of *Lecture Notes in Computer Science*, pages 34–48, London, UK, 2002. Springer-Verlag.

[37] J. Fauqueur, N. G. Kingsbury, and R. Anderson. Semantic discriminant mapping for classification and browsing of remote sensing textures and objects. In *Proceedings of the International Conference on Image Processing, ICIP 2005*, pages 846–849, 2005.

[38] P. Felzenszwalb and D. Huttenlocher. Image segmentation using local variation. In *Proceedings of the Computer Vision and Pattern Recognition, CVPR 1998*, pages 98–104. IEEE, 1998.

[39] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, September 2004.

[40] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

[41] J. Freixenet, X. Muñoz, D. Raba, J. Martí, and X. Cufí. Yet another survey on image segmentation: Region and boundary information integration. In *Proceedings of the 7th European Conference on Computer Vision, ECCV 2002*, volume III, pages 408–422, London, UK, 2002. Springer-Verlag.

[42] K. S. Fu and J. K. Mui. A survey on image segmentation. *Pattern Recognition*, 13:3–16, 1981.

[43] Z. Fu and A. Robles-Kelly. A fast hierarchical approach to image segmentation. In B. Lovell, D. Laurendeau, and R. Duin, editors, *Proceedings of the 19th International Conference on Pattern Recognition, ICPR 2008*, pages 1–4. IEEE Computer Society, December 2008.

[44] H. Fujiwara, Z. Zhang, H. Toda, and H. Kawabata. Textile surface inspection by using translation invariant wavelet transform. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, volume 3, pages 1427–1432. IEEE, IEEE, 2003.

[45] M. Galun, E. Sharon, R. Basri, and A. Brandt. Texture segmentation by multiscale aggregation of filter responses and shape elements. In *Proceedings of the International Conference on Computer Vision, ICCV 2003*, pages 716–723, 2003.

[46] B. Georgescu, I. Shimshoni, and P. Meer. Mean shift based clustering in high dimensions: A texture classification example. In *Proceedings of the Ninth International Conference on Computer Vision, ICCV 2003*, volume 1, pages 456–463, Los Alamitos, CA, USA, October 2003. IEEE Computer Society.

[47] M. Haindl. Texture synthesis. *CWI Quarterly*, 4(4):305–331, December 1991.

[48] M. Haindl. Texture segmentation using recursive Markov random field parameter estimation. In K. Bjarne and J. Peter, editors, *Proceedings of the 11th Scandinavian Conference on Image Analysis, SCIA 1999*, pages 771–776, Lyngby, Denmark, June 1999. Pattern Recognition Society of Denmark.

[49] M. Haindl and J. Filip. Extreme compression and modeling of bidirectional texture function. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1859–1865, 2007.

[50] M. Haindl, J. Grim, and S. Mikeš. Texture defect detection. In W. G. Kropatsch, M. Kampel, and A. Hanbury, editors, *Proceedings of the 12th International Conference on Computer Analysis of Images and Patterns, CAIP 2007*, volume 4673 of *Lecture Notes in Computer Science*, pages 987–994. Springer, Aug. 2007.

[51] M. Haindl and V. Havlíček. Prototype implementation of the texture analysis objects. Technical Report 1939, ÚTIA AV ČR, Praha, Czech Republic, 1997.

[52] M. Haindl and V. Havlíček. A multiscale colour texture model. In *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, pages 255–258. IEEE Computer Society, August 2002.

[53] M. Haindl and V. Havlíček. A simple multispectral multiresolution Markov texture model. In *The 2nd International Workshop on Texture Analysis and Synthesis, Texture 2002*, pages 63–66, Glasgow, June 2002. Heriot-Watt University.

[54] M. Haindl and S. Mikeš. Model-based texture segmentation. In A. Campilho and M. Kamel, editors, *Image Analysis and Recognition, International Conference on Image Analysis and Recognition, ICIAR 2004*, volume 3212 of *Lecture Notes in Computer Science*, pages 306–313, Berlin, 2004. Springer-Verlag.

[55] M. Haindl and S. Mikeš. Colour texture segmentation using modelling approach. In S. Singh, M. Singh, and C. Apte, editors, *Pattern Recognition and Image Analysis, Third International Conference on Advances in Pattern Recognition,*

ICAPR 2005, volume 3687 of *Lecture Notes in Computer Science*, pages 484–491, Berlin, Aug. 2005. Springer.

[56] M. Haindl and S. Mikeš. Unsupervised texture segmentation using multispectral modelling approach. In Y. Tang, S. Wang, D. Yeung, H. Yan, and G. Lorette, editors, *Proceedings of the 18th International Conference on Pattern Recognition, ICPR 2006*, volume II, pages 203–206, Los Alamitos, Aug. 2006. IEEE Computer Society.

[57] M. Haindl and S. Mikeš. Unsupervised texture segmentation using multiple segmenters strategy. In M. Haindl, J. Kittler, and F. Roli, editors, *Proceedings of the 7th International Workshop on Multiple Classifier Systems, MCS 2007*, volume 4472 of *Lecture Notes in Computer Science*, pages 210–219. Springer, May 2007.

[58] M. Haindl and S. Mikeš. *Pattern Recognition*, chapter Unsupervised Texture Segmentation, pages 306–315. IN–TECH, Vienna, Nov. 2008.

[59] M. Haindl and S. Mikeš. Texture segmentation benchmark. In B. Lovell, D. Laurendeau, and R. Duin, editors, *Proceedings of the 19th International Conference on Pattern Recognition, ICPR 2008*, pages 1–4. IEEE Computer Society, Dec. 2008.

[60] M. Haindl and S. Mikeš. Unsupervised mammograms segmentation. In B. Lovell, D. Laurendeau, and R. Duin, editors, *Proceedings of the 19th International Conference on Pattern Recognition, ICPR 2008*, pages 1–4. IEEE Computer Society, Dec. 2008.

[61] M. Haindl, S. Mikeš, and P. Pudil. Unsupervised hierarchical weighted multi-segmenter. In J. A. Benediktsson, J. Kittler, and F. Roli, editors, *Proceedings of the 8th International Workshop on Multiple Classifier Systems, MCS 2009*, volume 5519 of *Lecture Notes in Computer Science*, pages 272–282. Springer, June 2009.

[62] M. Haindl, S. Mikeš, and G. Scarpa. Unsupervised detection of mammogram regions of interest. In B. A. et al., editor, *Proceedings of the 11th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems, KES 2007*, volume 4694 of *Lecture Notes in Computer Science*, pages 33–40. Springer, 2007.

[63] M. Haindl, S. Mikeš, and P. Vácha. Illumination invariant unsupervised segmenter. In *Proceedings of the International Conference on Image Processing, ICIP 2009*, pages 4025–4028. IEEE, November 2009.

[64] M. Haindl and S. Šimberová. *Theory & Applications of Image Analysis*, chapter A Multispectral Image Line Reconstruction Method, pages 306–315. World Scientific Publishing Co., Singapore, 1992.

[65] M. Haindl and P. Vácha. Illumination invariant texture retrieval. In Y. Tang, S. Wang, D. Yeung, H. Yan, and G. Lorette, editors, *Proceedings of the 18th International Conference on Pattern Recognition, ICPR 2006*, volume III, pages 276–279, Los Alamitos, August 2006. IEEE Computer Society.

[66] M. Haindl and P. Žid. Multimodal range image segmentation. In D. A. Obinata Goro, editor, *Vision Systems, Segmentation & Pattern Recognition*, chapter 2. I-Tech Education and Publishing, Vienna, 2007.

[67] M. Heath, K. Bowyer, D. Kopans, R. Moore, and W. P. Kegelmeyer. The digital database for screening mammography. In M. Yaffe, editor, *Proceedings of the Fifth International Workshop on Digital Mammography*, pages 212–218. Medical Physics Publishing, 2001.

[68] M. A. Hoang, J.-M. Geusebroek, and A. W. M. Smeulders. Color texture measurement and segmentation. *Signal Processing*, 85(2):265–275, 2005.

[69] X. Hong, S. I. McClean, B. W. Scotney, and P. J. Morrow. Model-based segmentation of multimodal images. In W. G. Kropatsch, M. Kampel, and A. Hanbury, editors, *Proceedings of the 12th International Conference on Computer Analysis of Images and Patterns, CAIP 2007*, volume 4673 of *Lecture Notes in Computer Science*, pages 604–611. Springer, August 2007.

[70] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. Bowyer, D. W. Eggert, A. Fitzgibbon, and R. B. Fisher. An experimental comparison of range image segmentation algorithms. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 18(7):673–689, July 1996.

[71] Y. K. Hwang and N. Ahuja. Gross motion planning—A survey. *ACM Computing Surveys*, 24(3):219–291, 1992.

[72] R. Kashyap. Image models. In T. Y. Young and K. S. Fu, editors, *Handbook of Pattern Recognition and Image Processing*. Academic Press, New York, 1986.

[73] Z. Kato, T.-C. Pong, and S. G. Qiang. Multicue MRF image segmentation: Combining texture and color features. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume I, pages 660–663, Washington, DC, USA, August 2002. IEEE Computer Society.

[74] P. A. Kelly, H. Derin, and K. D. Hartt. Adaptive segmentation of speckled images using a hierarchical random field model. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 36(10):1628–1641, October 1988.

[75] C. Kervrann and F. Heitz. A Markov random field model-based approach to unsupervised texture segmentation using local and global spatial statistics. *IEEE Transactions on Image Processing*, 4(6):856–862, 1995.

[76] J. Kittler, A. Hojjatoleslami, and T. Windeatt. Weighting factors in multiple expert fusion. In *Proceedings of the British Machine Vision Conference, BMVC 1997*, pages 41–50. BMVA, BMVA, 1997.

[77] R. Kopper, T. Ni, D. A. Bowman, and M. Pinho. Design and evaluation of navigation techniques for multiscale virtual environments. In *Virtual Reality*, pages 175–182. IEEE, March 2006.

[78] S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. In *ICCV '03: Proceeding of the International Conference on Computer Vision*, pages 1150–1157, 2003.

[79] T. Kwon, T. Kim, and Y.-C. Choy. A new navigation/traveling method in virtual environment. In *TENCON*. IEEE, IEEE, 1999.

[80] D. Larlus, J. Verbeek, and F. Jurie. Category level object segmentation by combining bag-of-words models with Dirichlet processes and random fields. *International Journal of Computer Vision*, 2009. In press.

[81] S. U. Lee, S. Y. Chung, and R. H. Park. A comparitive performance study of several global thresholding techniques for segmentation. *Computer Vision, Graphics, and Image Processing*, 52:171–190, 1990.

[82] S. Lefèvre. Knowledge from markers in watershed segmentation. In W. G. Kropatsch, M. Kampel, and A. Hanbury, editors, *Proceedings of the 12th International Conference on Computer Analysis of Images and Patterns, CAIP 2007*, volume 4673 of *Lecture Notes in Computer Science*, pages 579–586. Springer, August 2007.

[83] K.-H. Liang and T. Tjahjadi. Adaptive scale fixing for multiscale texture segmentation. *IEEE Transactions on Image Processing*, 15(1):249–256, January 2006.

[84] L. Lucchese and S. K. Mitra. Color image segmentation: A state–of–the–art survey. *Proceedings of the Indian National Science Academy (INSA-A)*, 67(2):207–221, 2001.

[85] Q. Luo and T. M. Khoshgoftaar. Unsupervised multiscale color image segmentation based on MDL principle. *IEEE Transactions on Image Processing*, 15(9):2755–2761, August 2006.

[86] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *International Journal of Computer Vision*, 43(1):7–27, 2001.

[87] B. S. Manjunath and R. Chellapa. Unsupervised texture segmentation using Markov random field models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:478–482, 1991.

[88] J. Mao and A. Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern Recognition*, 25:173–188, 1992.

[89] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceeding of the 8th International Conference on Computer Vision, ICCV 2001*, volume 2, pages 416–423, July 2001.

[90] A. Mehnert and P. Jackway. An improved seeded region growing algorithm. *Pattern Recognition Letters*, 18(10):1065–1071, October 1997.

[91] M. Meilă. Comparing clusterings: An axiomatic view. In *Proceedings of the 22nd International Conference on Machine Learning, ICML 2005*, pages 577–584, New York, NY, USA, 2005. ACM.

[92] M. Meilă and W. Pentney. Clustering by weighted cuts in directed graphs. In *Proceedings of the 7th SIAM International Conference on Data Mining, SDM 2007*, Minneapolis, Minnesota, USA, April 2007. SIAM.

[93] J. Melendez, D. Puig, and M. A. García. Comparative evaluation of classical methods, optimized Gabor filters and LBP for texture feature selection and classification. In W. G. Kropatsch, M. Kampel, and A. Hanbury, editors, *Proceedings of the 12th International Conference on Computer Analysis of Images and Patterns, CAIP 2007*, volume 4673 of *Lecture Notes in Computer Science*, pages 912–920. Springer, August 2007.

[94] R. Meylani, A. Ertuzun, and A. Ercil. A comparative study on the adaptive lattice filter structures in the context of texture defect detection. In *ICECS 1996*, volume II, pages 976–979, 1996.

[95] S. Mikeš and M. Haindl. Multispectral texture segmentation. In J. Šafránková, editor, *WDS '03 Proceedings of Contributed Papers: Part I - Mathematics and Computer Sciences*, pages 221–225, Prague, 2003. Matfyzpress.

[96] S. Mikeš and M. Haindl. Prague Texture Segmentation Data Generator and Benchmark. *ERCIM News*, (64):67–68, 2006.

[97] S. Mikeš, M. Haindl, and R. Holub. Navigation in virtual environment. In R. Sablating, J. Hemsley, P. Kammerer, E. Zolda, and J. Stockinger, editors, *Digital Cultural Heritage – Essential for Tourism, Proceedings of the 2nd International Conference EVA 2008*, pages 111–119. Austrian Computer Society, Aug. 2008.

[98] J. W. Modestino, R. W. Fries, and A. L. Vickers. Texture discrimination based upon an assumed stochastic texture model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(5):557–580, September 1981.

[99] D. Mohr and G. Zachmann. Segmentation of distinct homogeneous color regions in images. In W. G. Kropatsch, M. Kampel, and A. Hanbury, editors, *Proceedings of the 12th International Conference on Computer Analysis of Images and Patterns, CAIP 2007*, volume 4673 of *Lecture Notes in Computer Science*, pages 432–440. Springer, August 2007.

[100] F. J. C. Monteiro and A. C. Campilho. Watershed framework to region-based image segmentation. In B. Lovell, D. Laurendeau, and R. Duin, editors, *Proceedings of the 19th International Conference on Pattern Recognition, ICPR 2008*, pages 1–4. IEEE Computer Society, December 2008.

[101] K. Moustakas, D. Tzovaras, and M. G. Strintzis. A geometry education haptic VR application based on a new virtual hand representation. In *Virtual Reality*, pages 249–252, 2005.

[102] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, March 1957.

[103] F. Nielsen and R. Nock. On region merging: The statistical soundness of fast sorting, with applications. In *Proceedings of the Computer Vision and Pattern Recognition, CVPR 2003*, volume II, pages 19–26, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[104] Y. Ohta, T. Kanade, and T. Sakai. Color information for region segmentation. *Computer Graphics and Image Processing*, 13(3):222–241, July 1980.

[105] T. Ojala, T. Mäenpää, M. Pietikäinen, J. Viertola, J. Kyllönen, and S. Huovinen. Outex: A new framework for empirical evaluation of texture analysis algorithms. In *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume I, pages 701–706, 2002.

[106] T. Ojala and M. Pietikäinen. Unsupervised texture segmentation using feature distributions. *Pattern Recognition*, 32(477–486), 1999.

[107] G. Opriessnig. User-defined mapping functions and collision detection to improve the user-friendliness of navigation in a virtual reality environment. In E. Banissi, K. Börner, C. Chen, G. Clapworthy, C. Maple, A. Lobben, C. J. Moore, J. C. Roberts, A. Ursyn, and J. Zhang, editors, *Seventh International Conference on Information Visualization*, volume IV, pages 446–451, London, UK, July 2003. IEEE Computer Society.

[108] N. R. Pal and S. K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294, 1993.

[109] D. K. Panjwani and G. Healey. Markov random field models for unsupervised segmentation of textured color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(10):939–954, 1995.

[110] Y. Peng, T. Xing-Wei, and F. Ju-Fu. A unified model of GMRF and MOG for image segmentation. In *Proceedings of the International Conference on Image Processing, ICIP 2005*, volume 3, pages 1140–1143, September 2005.

[111] G. Peters and J. Kerdels. Image segmentation based on height maps. In W. G. Kropatsch, M. Kampel, and A. Hanbury, editors, *Proceedings of the 12th International Conference on Computer Analysis of Images and Patterns, CAIP 2007*, volume 4673 of *Lecture Notes in Computer Science*, pages 612–619. Springer, August 2007.

[112] J. C. Pichel, D. E. Singh, and F. F. Rivera. Image segmentation based on merging of sub-optimal segmentations. *Pattern Recognition Letters*, 27(10):1105–1116, 2006.

[113] M. Pietikäinen, T. Mäenpää, and J. Viertola. Color texture classification with color histograms and local binary patterns. In *Proceedings of the 2nd International Workshop on Texture Analysis and Synthesis, Texture 2002*, pages 109–112, June 2002.

[114] S. V. Pons, J. L. G. Rodríguez, and O. L. Vera Pérez. Active contour algorithm for texture segmentation using a texture feature set. In B. Lovell, D. Laurendeau, and R. Duin, editors, *Proceedings of the 19th International Conference on Pattern Recognition, ICPR 2008*, pages 1–4. IEEE Computer Society, December 2008.

[115] H. Qi and N. A. Diakides. Thermal infrared imaging in early breast cancer detection - a survey of recent research. In *25th Annual International Conference of the IEEE EMBS*, pages 448–452, 2003.

[116] A. F. R. Rahman and M. C. Fairhust. Multiple classifier decision combination strategies for character recognition: A review. *International Journal on Document Analysis and Recognition*, 5(4):166–194, July 2003.

[117] T. R. Reed and J. M. H. du Buf. A review of recent texture segmentation and feature extraction techniques. *CVGIP–Image Understanding*, 57(3):359–372, May 1993.

[118] M. A. Roula, A. Amira, A. Bouridane, P. Milligan, and P. Sage. A novel technique for unsupervised texture segmentation. In *Proceedings of the International Conference on Image Processing, ICIP 2001*, volume 1, pages 58–61. IEEE, October 2001.

[119] M. Rousson, T. Brox, and R. Deriche. Active unsupervised texture segmentation on a diffusion based feature space. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, CVPR 2003*, volume II, pages 699–704, Los Alamitos, CA, USA, June 2003. IEEE Computer Society.

[120] R. A. Ruddle. The effect of trails on first-time and subsequent navigation in a virtual environment. In *Virtual Reality*, pages 115–122. IEEE, 2005.

[121] P. K. Sahoo, S. Soltani, and A. K. C. Wong. Survey of thresholding techniques. *Computer Vision, Graphics and Image Processing*, 41(2):233–260, 1988.

[122] M. P. Sampat, M. K. Markey, and A. C. Bovik. Computer-aided detection and diagnosis in mammography. In A. C. Bovik, editor, *Handbook of Image and Video Processing*, chapter 10.4, pages 1195–1217. Academic Press, New York, 2. edition, June 2005.

[123] M. Sattler, R. Sarlette, and R. Klein. Efficient and realistic visualization of cloth. In *Eurographics Symposium on Rendering 2003*, June 2003.

[124] G. Scarpa and M. Haindl. Unsupervised texture segmentation by spectral-spatial-independent clustering. In Y. Tang, S. Wang, D. Yeung, H. Yan, and G. Lorette, editors, *Proceedings of the 18th International Conference on Pattern Recognition, ICPR 2006*, volume II, pages 151–154, Los Alamitos, August 2006. IEEE Computer Society.

[125] G. Scarpa, M. Haindl, and J. Zerubia. A hierarchical finite-state model for texture segmentation. In *International Conference on Acoustics, Speech and Signal Processing, ICASSP 2007*, volume I, pages 1209–1212, Los Alamitos, April 2007. IEEE.

[126] G. Scarpa, M. Haindl, and J. Zerubia. A hierarchical texture model for unsupervised segmentation of remotely sensed images. *Lecture Notes in Computer Science*, 4522:303–312, 2007.

[127] L. Shafarenko, M. Petrou, and J. Kittler. Automatic watershed segmentation of randomly textured color images. *IEEE Transactions on Image Processing*, 6(11):1530–1544, November 1997.

[128] C. W. Shaffrey, I. H. Jermyn, and N. G. Kingsbury. Psychovisual evaluation of image segmentation algorithms. In *Advanced Concepts for Intelligent Vision Systems, ACIVS 2002*, Ghent University, Belgium, September 2002.

[129] M. Sharma and S. Singh. Minerva scene analysis benchmark. In *Seventh Australian and New Zealand Intelligent Information Systems Conference*, pages 231–235. IEEE, November 2001.

[130] E. Sharon, M. Galun, D. Sharon, R. Basri, and A. Brandt. Hierarchy and adaptivity in segmenting visual scenes. *Nature*, 442(7104):719–846, June 2006.

[131] J. R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In M. C. Lin and D. Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer-Verlag, May 1996. From the First ACM Workshop on Applied Computational Geometry.

[132] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[133] J. F. Silverman and D. B. Cooper. Bayesian clustering for unsupervised estimation of surface and texture models. *IEEE Transactions on Pattern Analysis and Machine Inteldigence*, 10(4):482–495, 1988.

[134] M. Singh and S. Singh. Spatial texture analysis: A comparative study. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume I, pages 676–679, Washington, DC, USA, August 2002. IEEE Computer Society.

[135] J. L. Sobral. Optimised filters for texture defect detection. In *International Conference on Image Processing*, volume III, pages 565–568, 2005.

[136] B. Sumengen, B. S. Manjunath, and C. S. Kenney. Image segmentation using curve evolution and region stability. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume II, pages 965–968, Washington, DC, USA, August 2002. IEEE Computer Society.

[137] Z. Tan and N. H. C. Yung. Image segmentation towards natural clusters. In B. Lovell, D. Laurendeau, and R. Duin, editors, *Proceedings of the 19th In-*

ternational Conference on Pattern Recognition, ICPR 2008, pages 1–4. IEEE Computer Society, December 2008.

[138] C. W. Therrien, T. F. Quatieri, and D. E. Dudgeon. Statistical model-based algorithms for image analysis. *Proceedings of the IEEE*, 74(4):532–551, April 1986.

[139] M. K. Titsias and A. Likas. Class conditional density estimation using mixtures with constrained component sharing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):924–928, 2003.

[140] T. Tweed and S. Miguet. Automatic detection of regions of interest in mammographies based on a combined analysis of texture and histogram. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume 2, pages 448–452, Los Alamitos, CA, USA, August 2002. IEEE Computer Society.

[141] P. Vácha and M. Haindl. Image retrieval measures based on illumination invariant textural MRF features. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval, CIVR 2007*, pages 448–454, New York, NY, USA, 2007. ACM Press.

[142] J. Vila, B. Beccue, and S. Anandikar. The gender factor in virtual reality navigation and wayfinding. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences, HICSS 2003*, volume 4, pages 101–107, Washington, DC, USA, 2003. IEEE Computer Society.

[143] X. Wang, X. Ding, and C. Liu. Optimized Gabor filter based feature extraction for character recognition. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume IV, pages 223–226, Los Alamitos, CA, USA, August 2002. IEEE Computer Society.

[144] Z. Ye and C.-C. Lu. Wavelet-based unsupervised SAR image segmentation using hidden Markov tree models. In R. Kasturi, D. Laurendeau, and C. Suen, editors,

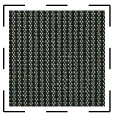
Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002, volume II, pages 729–732, Los Alamitos, CA, USA, August 2002. IEEE Computer Society.

[145] J. Žára, P. Chromý, J. Čížek, K. Ghais, M. Holub, S. Mikeš, and J. Rajnoch. A scaleable approach to visualization of large virtual cities. In *Proceedings Fifth International Conference on Information Visualisation*, pages 639–644, Los Alamitos, CA, USA, 2001. IEEE Computer Society.

[146] H. Zhang, J. E. Fritts, and S. A. Goldman. Image segmentation evaluation: A survey of unsupervised methods. *Computer Vision and Image Understanding*, 110(2):260–280, May 2008.

[147] J. Zhang, T. Tan, and L. Ma. Invariant texture segmentation via circular Gabor filters. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002*, volume II, pages 901–904, Los Alamitos, CA, USA, August 2002. IEEE Computer Society.

[148] Y. J. Zhang. Evaluation and comparison of different segmentation algorithms. *Pattern Recognition Letters*, 18:963–974, 1997.



Index