UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Universitat Politècnica de Catalunya
Teoria del Senyal i Comunicacions

This thesis is submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy (PhD)

Image Segmentation Evaluation and Its Application to Object Detection

---

by Jordi Pont Tuset

Advisor: Prof. Ferran Marques Acosta

Barcelona, December 2013

Abstract

The first parts of this Thesis are focused on the study of the supervised evaluation of image segmentation algorithms. Supervised in the sense that the segmentation results are compared to a human-made annotation, known as ground truth, by means of different measures of similarity. The evaluation depends, therefore, on three main points.

First, the image segmentation techniques we evaluate. We review the state of the art in image segmentation, making an explicit difference between those techniques that provide a flat output, that is, a single clustering of the set of pixels into regions; and those that produce a hierarchical segmentation, that is, a tree-like structure that represents regions at different scales from the details to the whole image.

Second, ground-truth databases are of paramount importance in the evaluation. They can be divided into those annotated only at object level, that is, with marked sets of pixels that refer to objects that do not cover the whole image; or those with annotated full partitions, which provide a full clustering of all pixels in an image. Depending on the type of database, we say that the analysis is done from an object perspective or from a partition perspective.

Finally, the similarity measures used to compare the generated results to the ground truth are what will provide us with a quantitative tool to evaluate whether our results are good, and in which way they can be improved. The main contributions of the first parts of the thesis are in the field of the similarity measures.

First of all, from an object perspective, we review the used basic measures to compare two object representations and show that some of them are equivalent. In order to evaluate full partitions and hierarchies against an object, one needs to select which of their regions form the object to be assessed. We review and improve these techniques by means of a mathematical model of the problem. This analysis allows us to show that hierarchies can represent objects much better with much less number of regions than flat partitions.

From a partition perspective, the literature about evaluation measures is large and entangled. Our first contribution is to review, structure, and deduplicate the measures available. We provide a new measure that we show that improves previous ones in terms of a set of qualitative and quantitative meta-measures. We also extend the measures on flat partitions to cover hierarchical segmentations.

The second part of this Thesis moves from the evaluation of image segmentation to its application to object detection. In particular, we build on some of the conclusions extracted in the first part to generate segmented object candidates. Given a set of hierarchies, we build the pairs and triplets of regions, we learn to combine the set from each hierarchy, and we rank them using low-level and mid-level cues. We conduct an extensive experimental validation that show that our method outperforms the state of the art in many metrics tested.

Acknowledgments

I would like to express my gratitude to my supervisor, Prof. Ferran Marques, for our enriching endless discussions, for the uncountable liters of red ink on our drafts, and for the perfect working environment he contributed to create.

A very special thanks goes to my colleagues, and friends, at D5-120, for our philosophical debates, baking competitions, exchanges of knowledge and skills, and for making the office a pleasant and comfortable place to work.

I must also acknowledge Albert and Josep, from whom I have learnt many things and discovered many tools; for setting up an awesome technical environment for *princesses* so we had to think only about research.

My sincerest gratitude goes also to Prof. Jitendra Malik and Dr. Pablo Arbeláez, for the opportunity they granted me to collaborate with their group.

I would also like to thank my parents, my sister, and the rest of my family for the support, love, guidance, and happiness they provided me not only through my thesis, but my entire life.

Last, but not least, I would like to say thanks to Mireia, my wife, for putting up with me and my mathematical models, my deadlines, my gadgets, my thesis,... for making life so easy and so enjoyable, for being there, no matter what.

I also acknowledge that this thesis would not have been possible without the financial assistance of the Image Processing Group (GPI) at UPC and the Spanish Government through the FPU grant and the projects CENIT-2007-1012 I3MEDIA and TEC2007-66858/TCM PROVEC.

Contents

1 Introduction

Humans see and understand the images they perceive effortlessly. Our eyes capture an image and our brain easily translates it into a useful source of information: what objects are in front of us?, where are we?, are we in trouble?, etc. Computer vision can be understood as a field that tries to emulate human vision, transforming an image into a useful source of information.

The first step of vision, capturing the image, has successfully been solved by computers, and in fact, they are much better than humans in this regard: one can easily take a picture of a subject far away using a telephoto lens or observe the movement of cells using a microscope.

Machines are not yet even close to human performance, however, when it comes to understanding this image. Computer vision can therefore be understood as the capability of machines to translate the set of pixels of an image into useful information.

In this context, image segmentation is one of the most basic and studied problems, yet it is among the ones that remain farther to be solved effectively. The simplest definition of image segmentation could be *the process of dividing an image into segments*, but this definition leaves many questions unanswered: What properties should these segments have? Should they be homogeneous in color? Should they represent the highest level of detail or just describe a higher interpretation of the image?

That is why image segmentation is often referred to as an ill-posed problem. This, however, has not prevented researchers to keep pushing during years and providing more and more advanced techniques of image segmentation. A question rapidly arises, however: how can researchers prove that their new algorithm performs better than the previous approaches in such an ill-posed problem?

Historically, many works *proved* the performance of their algorithms on a reduced set of example images and showed the qualitative results in the paper, which lacks any statistical significance. The results were usually described in terms such as "suffering from *oversegmentation*," or "being *undersegmented*." Given the ubiquity of these two terms in the literature and in this thesis, we devote Section 1.1.1 of this introduction to describe and discuss them.

Many other approaches to the evaluation of image segmentation have been presented since then: supervised/unsupervised, system-level, etc. Section 1.1.2 classifies them and describes their pros, cons, and particularities.

Parts I and II of this thesis are devoted to the study, review, and improvement of the supervised evaluation of image segmentation. This approach is dominant in the literature and it is based on measuring how similar the segmentation results are to a set of images segmented by humans, the so-called ground truth.

A good supervised evaluation depends, therefore, on two main parts: having good, large enough ground-truth databases; and comparing the results using the appropriate evaluation measure. Section 1.1.3 in this introduction is devoted to the former: it describes the main ground-truth segmentation databases publicly available, especially those used in this thesis. Parts I and II of this thesis are focused on the latter: review, study, comparison, and improvement of the measures themselves.

The last missing ingredient, once having good databases and measures, is a large set of image partitions to evaluate and test the properties of the measures. For that we select a set of state-of-the-art image segmentation algorithms, some with publicly-available implementations from other research groups and some other with own implementations from our group, and segment all the images in the used databases. Section 1.2 of this introduction describes the representative algorithms that we will use in the experiments of this thesis.

We divide them into those providing *flat* or *hierarchical* partitions. The former produce a single image partition for each parameterization and they represent different scales by varying some of these parameters. Special attention will be given to the latter, whose output is a pyramidal structure that represents regions at all scales.

Table 1.1 illustrates the structure of the first two parts of this thesis. First, we divide the evaluation measures studied depending on the perspective under which they evaluate segmentation (table columns), namely an object perspective or a partition perspective; that is, whether they evaluate how well the partitions represent single objects or full partitions. In practice, these two perspectives are related to the type of ground-truth they compare to: object-based ground truths such as PASCAL or partition-based ones such as BSDS500, and are described in Parts I and II, respectively. Each part contains an introduction and two chapters: one devoted to evaluate flat partitions and another focused on hierarchies.

	Part I Object perspective	Part II Partition perspective
Flat partitions	Chapter 2	Chapter 4
Hierarchies	Chapter 3	Chapter 5

Table 1.1: Structure of Parts I and II of this thesis

Apart from improving the evaluation techniques, Parts I and II allowed us to gain deep insights about the state of the art, especially for hierarchical techniques. Part III of this thesis builds upon these conclusions and translates the gained knowledge beyond the assessment itself, by proposing a new technique for object candidate generation called Multiscale Combinatorial Grouping (MCG) in Chapter 6. Specifically, MCG uses a varied set of hierarchies to produce a ranked set of segmented object candidates.

Finally, Chapter 7 concludes the thesis by providing an overall analysis of the contributions of each part and some future lines of research.

1.1 Image Segmentation Evaluation

1.1.1 Over- and Undersegmentation

Let us describe the two following intuitive and qualitative types of discrepancy between partitions, given that they are widely used when describing the quality of the results qualitatively:

- **Oversegmentation**: A partition is said to be oversegmented when it divides the actual regions, those in the ground-truth partition, into smaller subregions.
- **Undersegmentation**: When the regions of a partition overlap more than one actual region, we say that the result is undersegmented.

To illustrate these two ideas, Figure 1.1 shows a ground-truth partition (a) along with an oversegmented partition (b), an undersegmented result (c), and a partition affected by both problems (d). Note that the ground-truth partition has two regions, apart from the background.

(a) (b) (c) (d)

Figure 1.1: Over- and undersegmentation illustrative example: Example image with a ground-truth partition (a), an oversegmented partition (b), an undersegmented result (c), and a partition with both properties (d). The original image is taken from [AMFM11]

Figure 1.1(b) shows a purely **oversegmented** partition, that is, a partition that comes from dividing the original regions into smaller subregions. In other words, a purely oversegmented partition is *finer* than the ground truth partition, i.e., the partition gives a level of detail too high with respect to the ground truth. One of the objectives of the region-based image representation is to reduce the number of basic image elements to be handled from thousands of pixels to the minimum number of meaningful regions. Oversegmentation is clearly a problem under this point of view, since it entails that we are not capable of reducing the number of image representation parts as much as possible. In region-based motion estimation, for instance, oversegmentation can lead to global inconsistency due to a too local view of the object. If assessing the contours of the regions, oversegmentation preserves the ground-truth boundaries but it introduces many false contours. In object detection, there may be no region that sufficiently covers the actual object.

A purely **undersegmented** partition is shown in Figure 1.1(c), i.e., the partition comes from merging some of the regions in the ground truth. Looking it the other way around, in this case the ground-truth partition is finer than a purely undersegmented partition. In other words,

the representation is too coarse, so this entails that we are merging some of the meaningful parts of the image. In an object-retrieval scenario, for instance, undersegmentation can lead to missing the target object, since it has been merged with other objects or background parts. Regarding the boundaries, undersegmentation misses some of the boundaries of the ground truth.

Realistic results are usually affected by both problems, that is, they are never purely over- or undersegmented. Figure 1.1(d) shows an example of this case, where neither the ground truth is finer than the partition, nor the other way around.

1.1.2 Types of evaluation

As proposed in [ZFG08], image segmentation evaluation methods can be divided into the types presented in Figure 1.2.

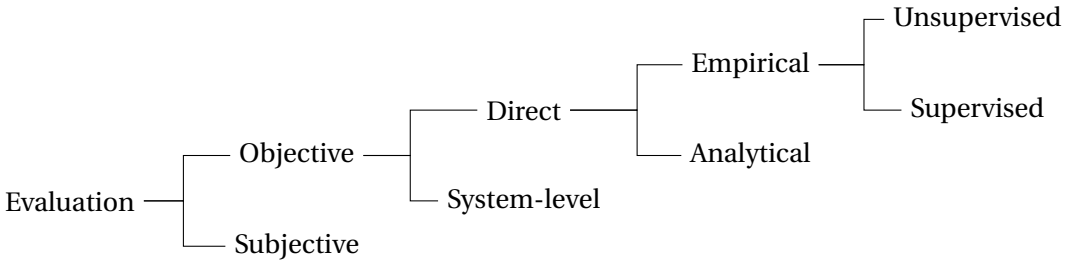


Figure 1.2: Segmentation evaluation methods division from [ZFG08]

This scheme introduces the factors that play an important role in image segmentation evaluation methods: level of subjectivity, whether the method assesses directly segmentation results or not, level of human interaction, etc. The proposed classification, however, simplifies the problem in such a way that introduces some mistakes that may mislead the user. Supervised methods, for instance, are classified as objective methods, while human ground truth segmentation creation is clearly subjective.

Any assessment method will try to model the subjective human perception, so we rearrange the methods in an axis depending on the proximity of the method to **human perception**. This axis could also be interpreted as the level of subjectivity involved in the evaluation. Subjectivity is usually criticized in an assessment measure for the lack of consistency it entails, but in the case of image segmentation this consistency is intrinsic to the definition of the problem. A measure should reflect the human subjectivity and at the same time try to minimize the effect of its inconsistency.

The rest of particularities of the evaluation methods are not always binary, so we have distributed them as pros and cons along with each method. Figure 1.3 shows the types of segmentation evaluation methods in the human perception axis, the higher the closer to human perception. On the left, in red, the weak points of the method are displayed, while on the right, in green, their strong points are depicted.

First of all, **subjective** methods are those in which results are assessed manually by a human. This methodology is, as the name states, subjective; but also expensive and slow. Being subjective is both the major drawback of this technique and its strongest point. While it can be unfair and biased, it may be the only one to truly capture the goodness of a method: in the

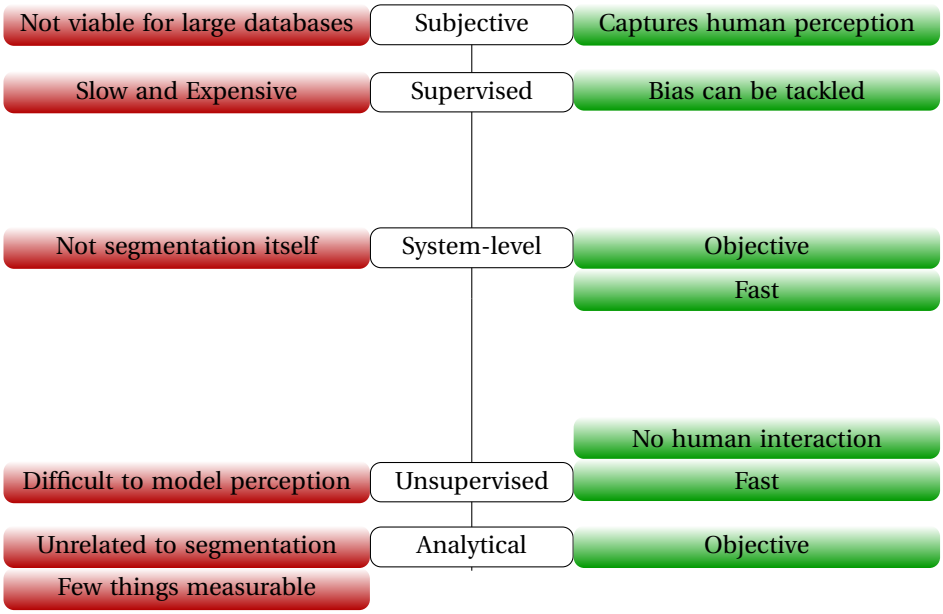


Figure 1.3: Level of human perception in the segmentation evaluation methods along with their pros and cons. The higher the method, the closer to human perception. On the left, in red, the weak points of the methods and on the right, in green, their strong points

end, any objective method will try to model the subjective human evaluation. The latter two, cost and slowness, are drawbacks that usually make this type of assessment feasible only for very reduced datasets, and therefore of low statistical significance, or even of no significance at all. In other words, one can always find a reduced set of images where a particular algorithm outperforms any algorithm.

Let us illustrate the particularities of the segmentation evaluation techniques by means of the example presented in Figure 1.4. It shows a simple image (a) along with the ideal, or true, partition (b) created by a human, and two different partitions (c) and (d). We would easily reach a consensus on considering (c) a better segmentation than (d). In other words, a subjective evaluation would give a better score to (c) than (d). As we will show below, however, the most straightforward and intuitive measures of segmentation quality would give (d) a better score, since, for instance, the number of discrepant pixels with the ideal partition is lower.

For realistic-sized databases that provide statistically significant results, subjective evaluation is impractical, and so some type of automation is mandatory. The more automated the measure, however, the more difficult to capture human perception. Consequently, a lot of effort has been put into finding measures that reflect or approximate the human perception of a good segmentation.

Decreasing a level in the axis of human perception in Figure 1.3, **supervised** methods are based on comparing the segmentation result against a manually segmented reference, also known as *ground truth*. Instead of manually assessing each result, and in order to accelerate the process, results are automatically compared to these references. On the plus side, human perception is still taken into account in the assessment process, so it is well ranked in the human perception axis. Creating a good and large-enough ground-truth database, however, is a slow, expensive, and arduous process.

(a) (b) (c) (d)

Figure 1.4: Illustrative example of the difficulty to define an objective measure of segmentation quality: (a) original image, (b) partition performed by a human, (c) and (d) two different partitions. The partition is depicted in red, while in the cases (c) and (d), the discrepancy with respect to (b) is depicted in dashed yellow lines. The original image is taken from [AMFM11].

In order to overcome the intrinsic subjectivity of human influence and its possible bias, [AMFM11] presents a ground truth database that is segmented more than once by different individuals, which provides priceless information to capture and mitigate the human subjectivity when assessing segmentation results.

Many measures of supervised methods are based on the number of misclassified pixels [CCR05, CCTCR09] of the partition with respect to the ground truth. Recalling the example in Figure 1.4, (d) would be considered better than (c) because the number of misclassified pixels in (c) is higher.

Other supervised methods could be based on comparing the features of the resulting regions with those of the actual ones: the more similar, the better. Depending on the features used, however, this type of evaluation can get close to a system-level analysis, since the techniques based on the features analyzed would be given more importance than others. Again, if, for instance, we compare the color mean of the ground-truth region with the color mean of the segmented region, the result in Figure 1.4 (d) would be better, since the misclassified pixels are closer in color to the reference than in (c). It is clear, therefore, that defining a good measure to compare the resulting partitions against the ground truth is crucial and not trivial.

There may be cases in which the available ground-truth databases are not suitable for the context in which the method has to be assessed and creating a tailored dataset has a cost too high or is a too slow procedure. A possible way to overcome this issue could be to evaluate the segmentation algorithm at a more global scope, and assess the results of complete systems based on the segmentation algorithms and compare their performance, i.e., an assessment at **system level**. In an object-retrieval scenario, for instance, it is much easier to evaluate whether the system has retrieved or missed a set of objects than to assess the quality of the underlying partitions.

Since human interaction is still needed to define the expected behavior of the analyzed system, we place the method in a medium level in the axis of Figure 1.3.

In favor of this type of assessment, segmentation is seldom the final objective itself, but it is

usually the starting point for other algorithms. It can be criticized, however, that results could be biased toward a specific finality and not reflect the generality of the segmentation problem.

Recalling the example of Figure 1.4, if, for instance, we evaluate the segmentation as a previous step of finding objects with respect to the shape of its borders, (c) would be considered better partition than (d). The user will have no guarantee, however, that in other applications the algorithm will perform better using this partition.

In a lower level of the human perception axis in Figure 1.3, there are the **unsupervised** methods, which do not require reference images nor human interaction. Instead, they measure how well a set of *desired* features are kept from the original image. However, the difficulty of formally defining the segmentation problem makes the objective measures to be hard to define and to justify that they represent the human criterion.

Many typical measures are based on the assumption that regions have to be internally uniform in some sense (not necessarily uniform in color), and different regions have to be distinct. This assumption, however, does not necessarily match human perception, since semantics play an important role in human image segmentation. The lack of human interaction may be therefore a weak point but it is also an advantage, since it makes unsupervised evaluation the only one suitable for automatic tuning of segmentation algorithms.

Recalling the example of Figure 1.4, if the color variance of the region was the measure to be minimized, again the better partition would not be the expected one, since the misclassified pixels in (d) are closer to the region mean than in (c).

The impossibility of capturing the nature of human perception mathematically places **analytical** methods in the lowest part of the human perception axis in Figure 1.3. They consist in analytically assessing the parts of the algorithms that can be mathematically proven without ambiguity, such as their complexity or their memory requirements. There is usually a trade-off between some of these measures and segmentation quality. For instance, different parameterizations of an algorithm may allow us to obtain good results but making the process slow, or to obtain less accurate outputs faster.

We believe, therefore, that the supervised approach to evaluate segmentation is the more appropriate. As we have introduced, in this case the measures used to compare the results to the ground-truth are the cornerstone of the analysis. Parts I and II of this thesis are therefore devoted to the study and comparison of the available measures to evaluate both flat and hierarchical segmentations in a supervised environment.

1.1.3 Ground-Truth Databases

As explained in Section 1.1.2, creating a sufficiently-large ground-truth segmentation database is time-consuming and expensive, so any publicly-available segmented image collection is priceless for assessing image segmentation algorithms properly. Following we present the publicly-available ground-truth segmentation databases that are used in this thesis, namely BSDS500, DCU, and PASCAL.

BSDS and DCU: One of the most valuable and used collection on generic image segmentation may be found in [AMFM11], known as **BSDS500**, that stands for Berkeley Segmentation Data Set 500. The database consists of 500 color images (extension of BSDS300 [MFTM01]) that cover a wide range of *natural* scenarios. They all have the same resolution of 481×321 or

321×481 pixels.

The subjects that segmented the images were asked to do the following: *Divide each image into pieces, where each piece represents a distinguished thing in the image. It is important that all of the pieces have approximately equal importance. The number of things in each image is up to you. Something between 2 and 20 should be reasonable for any of our images.*

The first property that stands out is that the number of regions in each partition is not fixed, in contrast with many object databases whose images consist mainly of a centered object. Each image is segmented by more than one individual (approximately five in mean), so the resulting database consists of 2696 image partitions. Image segmentation is an ill-posed problem, so having the same image segmented by different individuals allows to measure the consistency of the marked boundaries, as studied in [Mar03]. In other words, the intrinsic bias and inconsistency of a human-generated partition can somehow be mitigated, studied, and quantified.

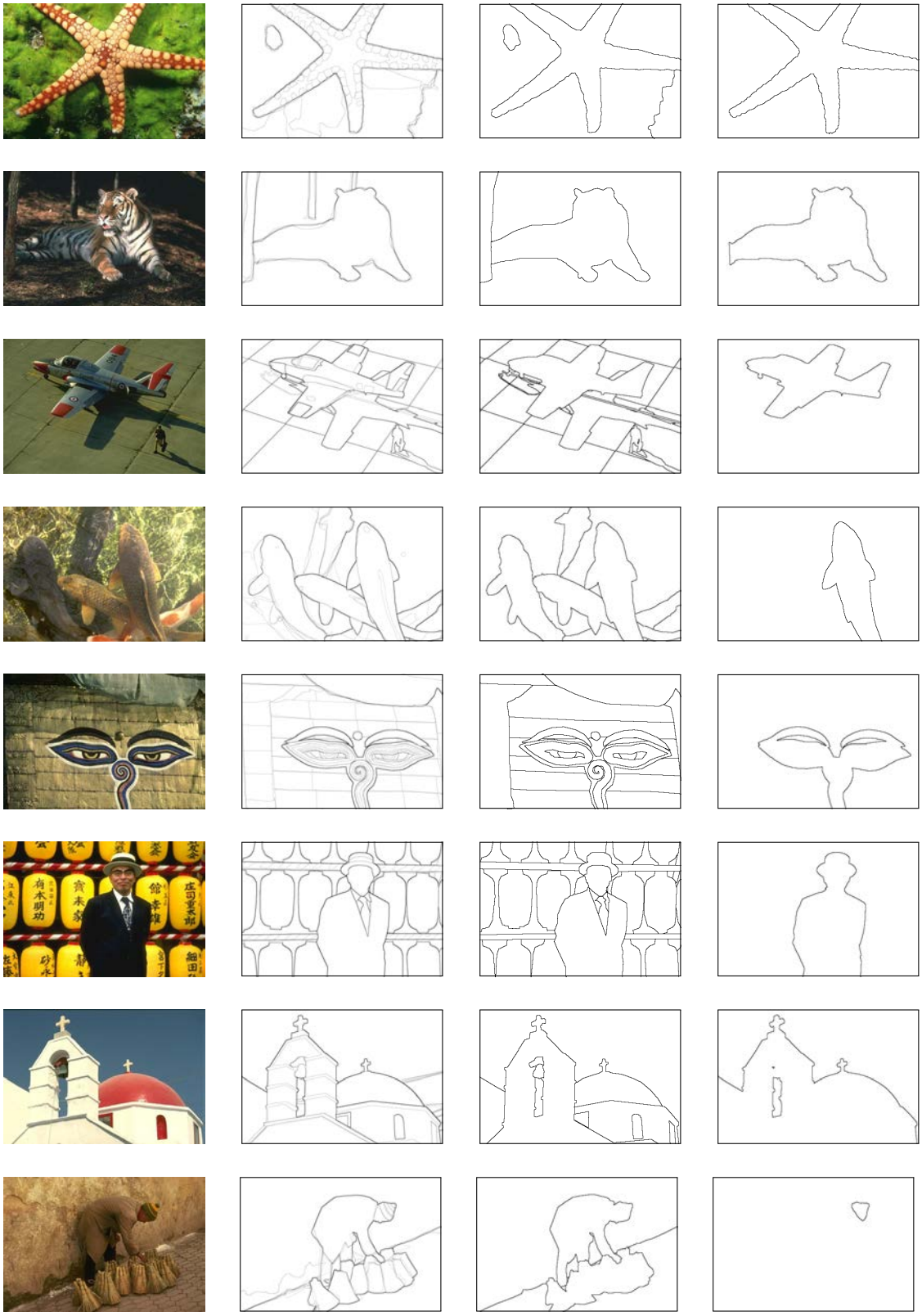
Built on a subset of BSDS500, the database in [MO10] consists of one hundred single-object partitions, or masks. Being an object segmentation better defined than a generic segmentation, the masks are aimed at the highest pixel accuracy at the image boundaries, since it was created to evaluate supervised segmentation algorithms, i.e., more precision was needed. We will refer to this database as **DCU**, that stands for Dublin City University, given that it was created in this university. To handle the ambiguity when marking a pixel that is close to the object boundary, the authors defined the following guideline: *Retain pixels that appear to contain some of the objects color along the object border, and that do not appear to be image compression artifacts.* The authors claim that using this methodology: *Each pixel along the border would be on average half-inside and half-outside the true form of the foreground object.*

To visually compare the precision in both databases, Figure 1.5 shows a detail of a partition in DCU (b), compared to one in BSDS500 (c), overlapped on the original image (a). In this example, it is clear that the result by DCU is much accurate than the one by BSDS500. In contrast, however, the lack of multiple partitions of the same image, the reduced set of partitions (100 in front of 2696), and being only object masks; are a plus for BSDS500.

(a) (b) (c)

Figure 1.5: Precision comparison between ground-truth databases: (a) Original image, (b) detailed view of the partition in [MO10], and (c) the same view of a partition in [AMFM11].

To get a flavor of the type of images and partitions of both data sets, Figure 1.6 shows a set of images from BSDS500 (column (a)). Column (b) is the representation of the added partitions



(a) (b) (c) (d)

Figure 1.6: Segmentation ground-truth examples from BSDS500 and DCU: (a) original image, (b) added BSDS500 partitions, (c) single BSDS500 partition, and (d) DCU mask

from all the annotators in BSDS500; being darker contours those on which most annotators agree, and brighter areas those boundaries marked only by one annotator. Column (c) shows one particular partition from BSDS500 and (d) the mask from DCU.

Notice that, in the four first rows, DCU partitions come from selecting and refining one of the BSDS500 regions. In other word, one of the BSDS500 partitions is the same than DCU but with an oversegmented background and a less precise boundary. In the three following rows, the BSDS500 partitions also oversegment the object itself. In the last row, DCU focused on an object of minor importance in the image.

PASCAL: The PASCAL Visual Object Classes challenges [EVGW+] were a series of competitions that were held each year from 2005 to 2012. It started as a classification and object detection challenge and it was yearly extended with object segmentation, action classification, etc. Every year, a set of annotated images was released as a training and validation set, and a new testing set was created and not made public until after the competition had finished.

In this work we are interested in the dataset corresponding to the segmentation competition. The dataset evolved from 2007 to 2012 and ended up consisting of 2913 images with 6934 objects segmented in the training and validation set (also referred to as *trainval* set). The objects segmented are classified into 20 different classes: person, animal (bird, cat, cow, dog, horse, sheep), vehicle (aeroplane, bicycle, boat, bus, car, motorbike, train), and indoor (bottle, chair, dining table, potted plant, sofa, tv/monitor).

Objects are annotated using a tri-map, instead of just a binary mask. The pixels marked as object are ensured to be foreground, the pixels marked as background are non-object, keeping a strip of at most 5 border pixels that can be either and do not affect the evaluation of the results. Figure 1.7 shows some examples of images in PASCAL along with their ground-truth annotation.

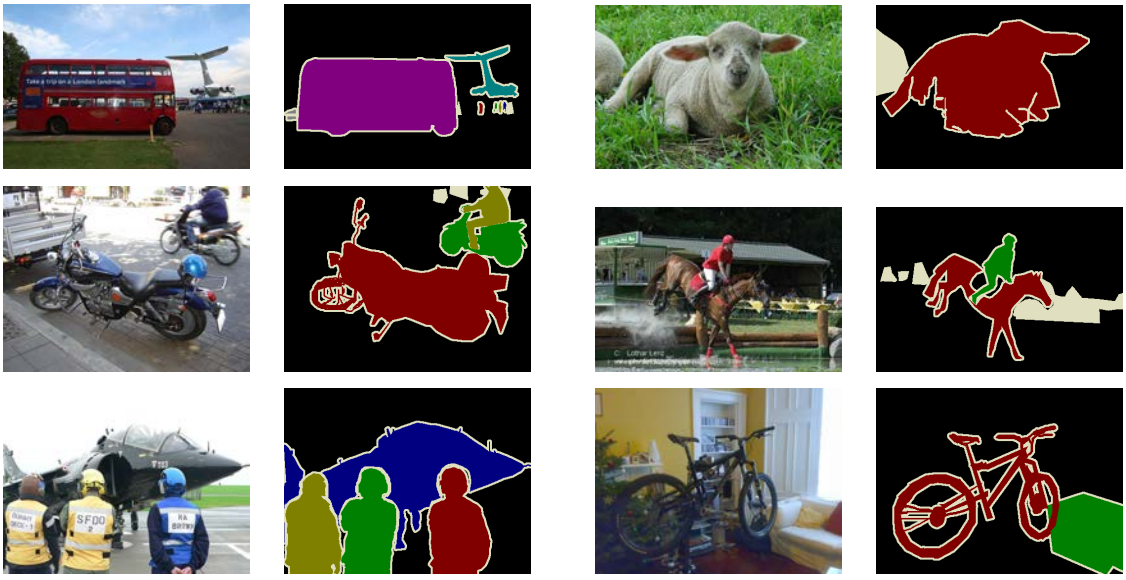


Figure 1.7: PASCAL examples images and annotations: colored areas refer to objects, black areas to background, and cream areas are the border pixels

1.2 State of the Art in Image Segmentation

We divide the state of the art into those methods whose output is a flat partition and a hierarchical structure. First of all, flat segmentation techniques are those whose output is a single flat partition, that is, a division of the image pixels set into some subsets, called regions. In order for these methods to represent objects at different scales, one tunes their parameters for the result to be coarser or finer, which generally entails having more or less regions.

Figure 1.8 depicts an example of an image from BSDS500 segmented by a state-of-the-art algorithm at two different levels of detail. As we can observe, the finer partition is not an over-segmentation of the coarser one, that is, the regions in the finer partitions do not come from dividing the ones in the coarser one.

Image Ground Truth 3 clusters 15 clusters

Figure 1.8: Flat segmentation technique: finer partitions are not necessarily a refinement of the coarser one

This observation is crucial, in the sense that the regions from partitions at different levels of detail cannot be directly related and thus they have to be processed separately. This is the main difference with respect to hierarchical segmentation techniques, in which the whole range of scales is represented in a single structure, and regions at different levels have some type of relationship. As we will show in different points of this thesis, this behavior is a differential advantage of the hierarchical structures with respect to the flat segmentation techniques.

Let us delve into the description of the hierarchical segmentation techniques. As introduced before, they contain partitions of the image at different levels of detail in a single structure. They are usually represented by means of a tree, where the root represents the whole image, the leaves are the regions at the highest level of detail, and a parent node represents the merging of all their children regions.

Let us start by describing how the hierarchies used in this thesis are constructed, as illustrated in Figure 1.9. On the left extreme of the diagram, we start by a partition at the maximum level of detail (pixels, super-pixels, etc.) and we represent each region as a node. Then, from left to right, we iteratively merge those sets of regions that are more similar according to a given criterion. Defining this criterion is the cornerstone of the hierarchy creation algorithm and will define the properties of the final result. Some examples in the literature include computing the distance between the color mean of the regions, learning distances between a set of region descriptors, or scanning the contour strength in the boundary between the regions.

Continuing in Figure 1.9, the new region formed at each merging step is depicted as the parent node of the merged regions. We will refer to the set of flat partitions formed at each step as merging-sequence partitions. The process stops when all regions are merged into the whole image, represented by the root of the tree. The right-most tree represents the full hierarchy.

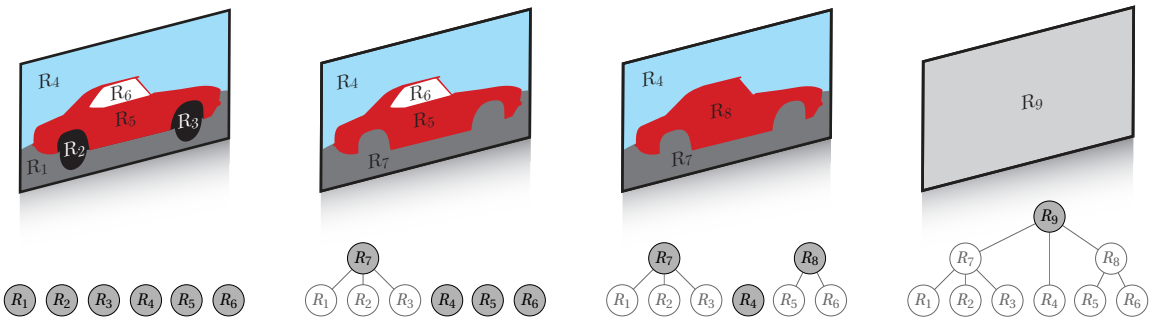


Figure 1.9: Schematic representation of the merging sequence of a hierarchy as a sequence of region trees

To get a better idea of how the full hierarchy looks like, Figure 1.10 depicts the same example but each node of the tree is now represented by the region it forms and the left column shows the merging-sequence partitions. Regions in the hierarchies aim at representing the objects in the image, from the smallest ones or the finest detail, to the biggest ones. As we can observe, however, there may be no single region representing the object of interest, that is, hierarchies can also suffer some degree of oversegmentation.

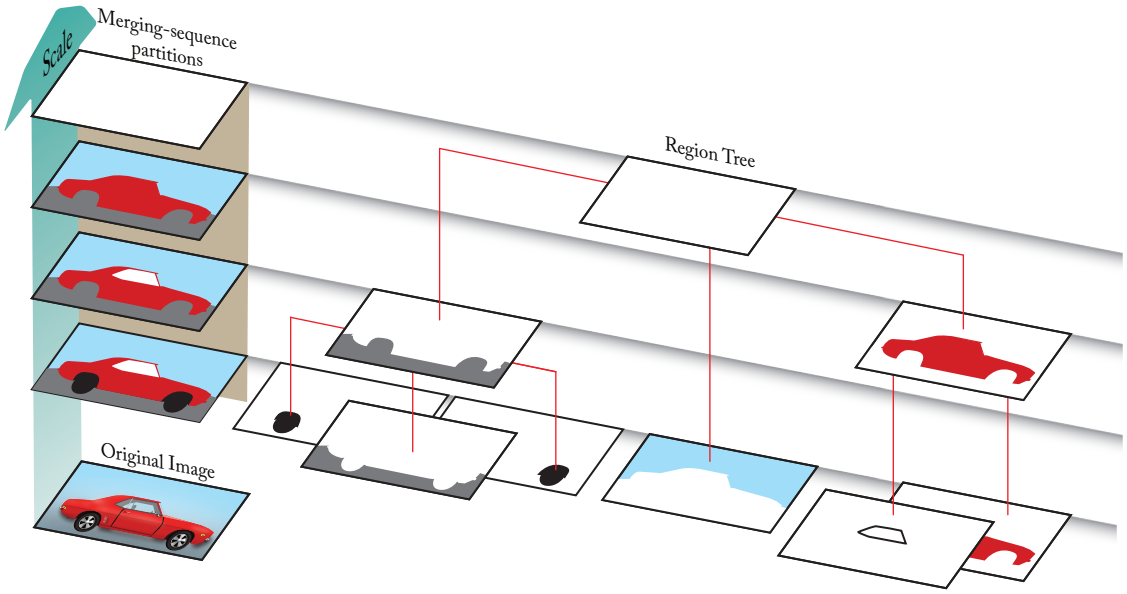


Figure 1.10: Graphical representation of a hierarchy as a region tree

The following paragraphs list the state-of-the-art flat segmentation techniques used in this thesis, the hierarchical algorithms, and the strategies that we will use as baselines, that is, *dummy* ways of dividing the image that we will use to normalize the performance of state-of-the-art algorithms. Please note that we are not implying that the seven selected techniques are the best performing methods, nor that the parameterizations used are optimal. We use these segmentation techniques as a tool to show the properties of the evaluation measures, so doing an exhaustive survey of the state-of-the-art in image segmentation is out of the scope of this thesis.

Flat Segmentation:

- **MeanShift** [CM02, CGM02]: Mean shift algorithm applied to image segmentation. We use the MATLAB® wrapper published in [Bag11] of the EDISON implementation [EDI02]. The parameters selected are the default ones in [Bag11] except `SpatialBandWidth`, `RangeBandWidth`, and `MinimumRegionArea`, which are swept for the number of regions in the partitions to be between 10 and 100.
- **NCut** [SM00]: Normalized Cuts applied to image segmentation. We use the publicly available implementation by the authors and sweep the number of regions in the partitions.
- **EGB** [FH04]: Efficient Graph-Based image segmentation. We use the publicly available implementation by the authors and sweep the parameters for the number of regions in the partitions to be between 10 and 100.

Hierarchical Segmentation:

- **gPb-UCM** [AMFM11]: Based on the seminal work of [NS96], gPb-UCM builds an Ultrametric Contour Map (UCM) on the globalized probability of boundary (gPb) contour detector. We use the publicly available implementation by the authors and we extract the partitions at different contour strengths.
- **ISCRA** [RS13a]: Region merging technique with trained similarities at different scales. The hierarchies are provided pre-computed by the authors, we extract the partitions at the 60 proposed levels, and sample some of them.
- **NWMC** [VMS08]: Based on the seminal work of [SG00], NWMC builds a Binary Partition Tree (BPT) that models the color of each region by its mean and takes contour complexity into account. We use a private own implementation and sample some partitions in the merging sequence.
- **IID-KL** [CM10]: Based on the seminal work of [SG00], IID-KL builds a Binary Partition Tree (BPT) that models the color of each region by means of its histogram. We use a private own implementation and sample some partitions in the merging sequence.

Baselines:

- **QuadTree**: The partitions are formed by recursively splitting the image into two rectangular halves, independently of the image content.
- **Random**: Starting from the superpixels provided by gPb-UCM (around 1200), we iteratively merge pairs of neighboring regions selected randomly.

Figure 1.11 shows an example partition from each of the segmentation techniques used in this thesis.

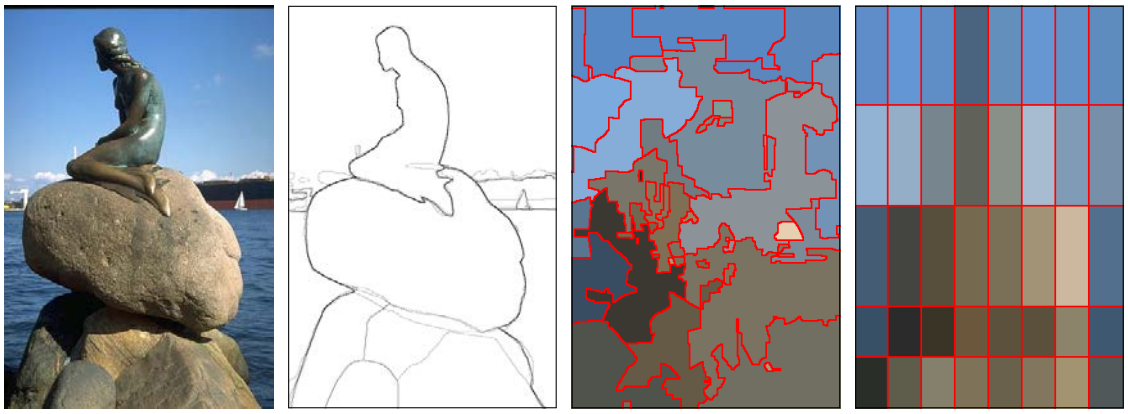
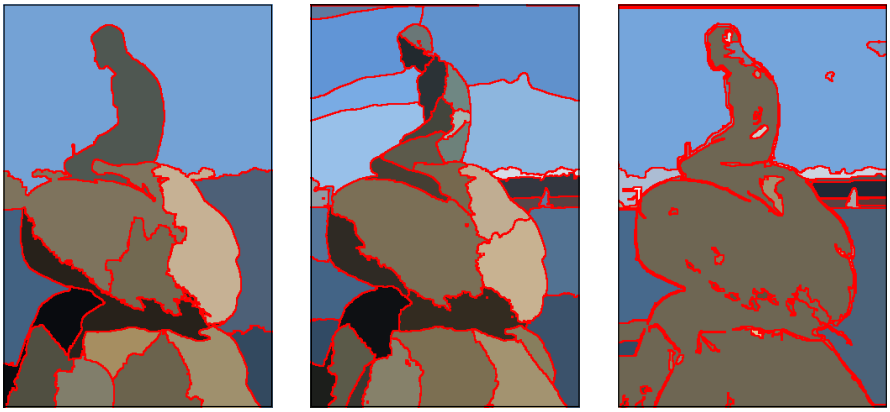
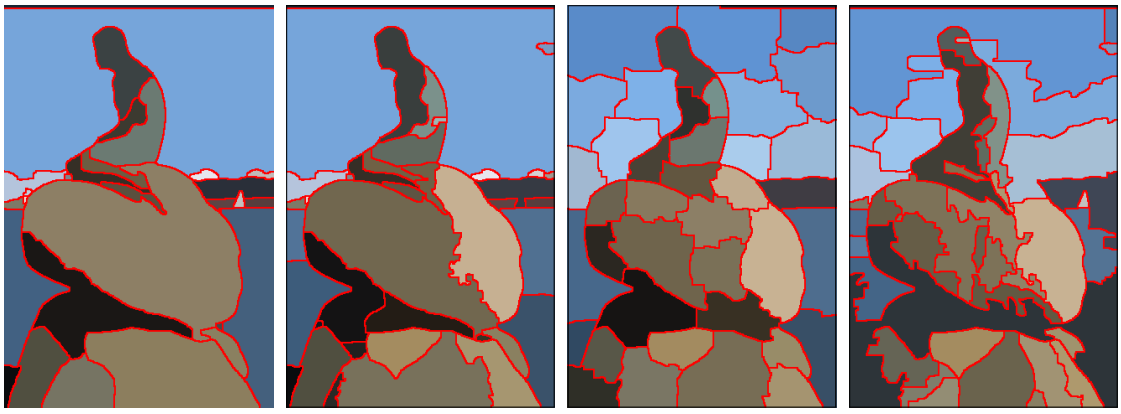


Image Ground Truth Random QuadTree



MeanShift NCut EGB



gPb-UCM ISCRA NWMC IID-KL

Figure 1.11: Qualitative examples of all segmentation techniques used in this thesis

Part I

Segmentation Evaluation From an Object Perspective

Introduction

One of the applications given to image segmentation is being a pre-processing step of object detection algorithms, given that they simplify the representation of an image to a reduced set of regions. It is reasonable, therefore, to think that part of the good or bad performance of these object detection algorithms can be attributed to the pre-processing algorithms, which motivates the evaluation of image segmentation algorithms in this context, i.e., in terms of their ability to serve as a pre-processing step to object detection and segmentation.

In other words, we want to evaluate how *well* and how *easily* can objects be formed by merging or selecting regions from a given image partition. By how *well* we mean which is the maximum object quality that can be achieved by selecting regions from a partition. By how *easily* we mean how many regions from this partition are needed to get to this quality, the less the better.

In practice, there will always be a trade-off between achievable quality and number of regions needed, so our approach is to evaluate the object quality that can be achieved by selecting a limited number of regions from a partition. In other words, we are evaluating image segmentation using the Pareto front as in [EMT02], in the plane of achievable quality vs number of regions. We differ from this work in the fact that we explicitly find the maximum achievable quality for any number of regions using combinatorial optimization techniques.

The approach is also coined as **upper-bound** object quality from a partition in [GWL06, GWL07] and it is used in works such as [GAV+08, WO10, AGBB12]. We differ from these works in the sense that, as we will show in the next section, we do actually find the *real* upper-bound achievable quality.

Chapter 2 is devoted to the evaluation of flat partitions from this perspective, while Chapter 3 extends the analysis to hierarchies. The remainder of this chapter surveys, analyzes, and compares the measures used to evaluate the quality of an object representation against an annotated database.

Object quality measures

An object in an image can be defined as a set of pixels that grouped together have some semantical meaning. Object detection consists in *roughly localizing* these sets in the image, or in other words, providing an approximate area where the object is likely to be, usually represented by a bounding box.

Object segmentation aims at providing the accurate shape of the object also, apart from its location. The intuitive way of representing the shape of a single object is by dividing the image pixels into two sets: those belonging to the object and those not. Single-object segmentation can be seen, therefore, as a two-class classification problem, aiming at dividing all the image pixels into either the *object* or *non-object* classes.

In this context, we can refer to the object class as *positive* pixels and the non-object class as negative pixels. Using this notation, given an automatic segmentation method m, its resulting single-object detection (also known as machine result) can be written as a division of the image pixel set I into two disjoint classes:

$$I = P_m \cup N_m$$

where P_m and N_m refer to positive and negative pixels, respectively, and the subscript stands for the method used. Equivalently, the ground-truth object can be denoted as:

$$I = P_{gt} \cup N_{gt}$$

The goal of any automatic algorithm is to achieve a perfect detection, i.e., $P_m = P_{gt}$, but if this is not the case, we focus on the following sets:

- **True positives**: Pixels that are detected as object and they are labeled as so in the ground truth: $TP = P_m \cap P_{gt}$.
- **False positives**: Pixels that are detected as object but they are not labeled as so in the ground truth: $FP = P_m \cap N_{gt}$.
- **False negatives**: Pixels that are classified as non-object but they are labeled as object in the ground truth: $FN = N_m \cap P_{gt}$, also known as *misses*.

Given these definitions, it is clear that our objective is to maximize the true positives while minimizing both the false positives and the false negatives. Figure 1.12 shows the Venn diagram illustrating these sets.

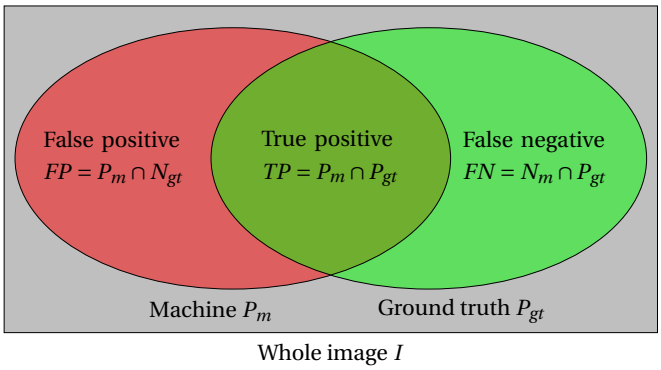


Figure 1.12: Sets involved in an object segmentation result with respect to a ground truth

The sets P_m and P_{gt} can be represented by means of a binary image, with positive pixels displayed in some color and the negative ones in white. Figure 1.13 shows an image from BSDS500 (a), along with a ground-truth object partition represented in green (b) and a possible machine result in red (c). The sets *TP*, *FP*, and *FN* are shown in different colors in (d).

F-Measure: Precision and Recall: A widely used and accepted pair of measures to assess a detection algorithm is the following:

- **Precision**: Measures the percentage of detected pixels that are actually true:

$$Precision = \frac{|TP|}{|P_m|} = \frac{|P_m \cap P_{gt}|}{|P_m|} \leq 1$$

(a) (b) (c) (d)

Figure 1.13: An example image (a), along with the ground truth object (b) and a sample machine result (c) represented in form of binary image. (In green P_{gt}, in red P_m, and in white N_{gt} and N_m.) The result of overlapping (b) and (c) is shown in (d), where the true positives (TP) are in black, the false positives (FP) in red, and the false negatives (FN) in green. The original image is from BSDS500 [AMFM11], and the ground-truth object partition from DCU [MO10]

- **Recall**: Measures the percentage of ground-truth positives that are actually detected:

$$Recall = \frac{|TP|}{|P_{gt}|} = \frac{|P_m \cap P_{gt}|}{|P_{gt}|} \leq 1$$

Our objective is to maximize both measures, but in general there is a trade-off between them. As an example, marking the whole image as object gives us the maximum true positives and zero false negatives, but in contrast gives us a high amount of false positives; or in other words, gives a perfect recall but a very low precision. Ideally $Precision = 1$ and $Recall = 1$, which is so iff $P_m = P_{gt}$, that is, in a perfect result.

In order to measure the trade-of between the two measures, the **F-measure** is defined as the weighted harmonic mean between precision and recall:

$$F_\beta = \left(1 + \beta^2\right) \frac{Precision \cdot Recall}{\beta^2 \cdot Precision + Recall}$$

where β allows us to weight precision and recall differently. If we give equal importance to both measures, then the usual F-measure is the harmonic mean between them:

$$F = F_1 = 2 \frac{Precision \cdot Recall}{Precision + Recall}$$

In terms of true and false detections, this measure can be rewritten as follows:

$$F = \frac{2|TP|}{2|TP| + |FN| + |FP|} \tag{1.1}$$

To the knowledge of the authors, this coefficient was first reported by Czekanowski in 1913 [Cze13], in the context of anthropology. Later, Dice used it in 1945 [Dic45] to compare the number of species in two samples, with respect to the shared species in both. He coined it as *coincidence index*. It was also used in the context of plant sociology by Sørensen in 1948 [Sø48]. Named after them, the coefficient is also known as *Czekanowski*, *Dice's*, or *Sørensen's coefficient*.

Following the example in Figure 1.13, counting the pixels in (d) we have that the precision and recall are:

$$Precision = \frac{|TP|}{|P_m|} = \frac{22215 \text{ pixels}}{22907 \text{ pixels}} \approx 0.97 \qquad Recall = \frac{|TP|}{|P_{gt}|} = \frac{22215 \text{ pixels}}{25069 \text{ pixels}} \approx 0.89$$

As could be guessed from the graphical representation, the result has a good precision, i.e., the 97% of the detected pixels are true object. However, recall is 89%, since we miss all the bar. As a measure of the trade-off, the F-measure is:

$$F = 2\frac{Precision \cdot Recall}{Precision + Recall} = 2\frac{0.97 \cdot 0.89}{0.97 + 0.89} \approx 0.93$$

$$F = \frac{2|TP|}{2|TP| + |FN| + |FP|} = \frac{2 \cdot 22215}{2 \cdot 22215 + 2854 + 692} \approx 0.93$$

Jaccard Similarity Coefficient: The Jaccard index was introduced in the context of plant sociology by Jaccard in 1901 [Jac01], and in the context of object segmentation it is defined as the *Intersection over Union* (*IoU*) between the machine and the ground-truth results:

$$J(P_m, P_{gt}) = \frac{|P_m \cap P_{gt}|}{|P_m \cup P_{gt}|} = \frac{|TP|}{|TP| + |FN| + |FP|} \tag{1.2}$$

In the PASCAL Visual Object Classes Challenge 2010 [EVGW+10] the Jaccard coefficient (called area of overlap a_0) is used to assess whether a particular object has been detected ($a_0 \geq 0.5$) or not ($a_0 < 0.5$). In the context of object detection in [MO10], object accuracy is measured by means of the same value, denoted as $\mathscr{A}_0$. The performance measure used in the salient object extraction evaluation in [GWL06, GWL07] is also J, although denoted as P. The work in [ME07] uses also this measure but it is denoted as *Overlap Score* (*OS*), or *spatial support score*. In [AMFM11] the Jaccard index is referred to as *overlap* and in [RFE+06], as *ratio of intersection*.

Spatial Quality Measure: The approach followed by the MPEG7 committee to assess object-based segmentation quality was the Spatial Quality Measure [WM97]:

$$SQM(P_m, P_{gt}) = \frac{|FP| + |FN|}{|P_{gt}|} = 1 - \frac{|TP| - |FP|}{|P_{gt}|}$$

This approach does not take into account the size of the assessed region $|P_m|$, so the behavior is not consistent for varying region sizes. This entails that, in the case of oversegmented results, for instance, a small region which overlaps with the ground truth on a hundred pixels and just one pixel lies outside of it would be penalized in front a bigger region with much more pixels overlapping the true negative region. As an advantage, this measure is additive, in the sense that the measure for the merging of two regions can be computed just adding the value for the two merged regions, which is not the case for the F measure or the Jaccard index.

Based on this measure, the work in [VMS99] and [VM04] presents a measure that is aimed at being perceptually weighted. They claim that false negatives are worse perceived by humans than false positives: a *halo* around the object does not affect as much as losing a part of it; so they penalize the former. This same idea was applied in [CP03] and [MC06] to present an almost identical measure. We believe that this type of approaches are too much application-dependent for general segmentation evaluation.

Equivalence between the Jaccard index and the F measure: Comparing the expression of the F measure (Equation 1.1) and the Jaccard coefficient (Equation 1.2), we can deduce the following equality:

$$\frac{F}{2-F} = \frac{\frac{2|TP|}{2|TP|+|FN|+|FP|}}{2-\frac{2|TP|}{2|TP|+|FN|+|FP|}} = \frac{2|TP|}{4|TP|+2|FN|+2|FP|-2|TP|} = J$$

That is, both measures are functionally related as: $J = \frac{F}{2-F}$.

Figure 1.14 plots the value of J as a function of F, in the range of interest $[0,1]$. Given that their relationship is a monotonically increasing function, any ranking between algorithms using any of the two functions would be the same. In other words, for the purpose of segmentation algorithm comparison, both measures are equivalent. Despite this simple equivalence, there exist works in the literature [SCF+12] that report results using both measures in parallel.

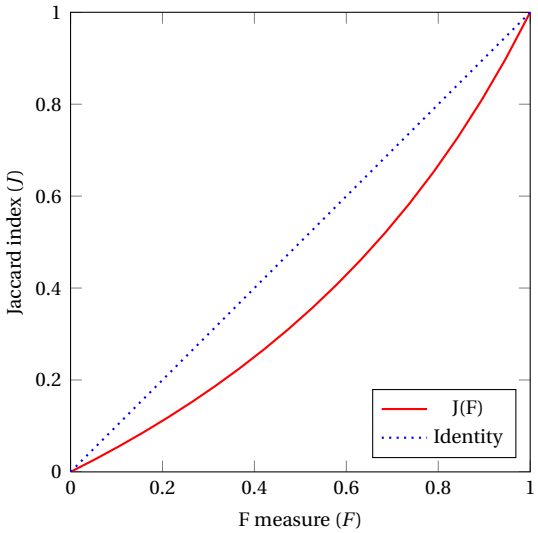


Figure 1.14: Comparison between the F-measure and the Jaccard index (J)

In this work we will mainly use the Jaccard coefficient, since it is more used in the literature, and therefore, comparing results will be easier. We believe, however, that the main reason why this measure was selected against the F-measure is aesthetic: the expression in terms of P_m and P_{gt} is more compact; although from a *detection* point of view, the F-measure is theoretically more justified in our opinion.

2 Flat Segmentation

2.1 Introduction

The previous chapter expounds on how an object is represented, interpreted, and how we evaluate its accuracy with respect to a ground-truth object. These measures can therefore be used to directly evaluate the output of an object detection or segmentation algorithm, and as motivated before, the Jaccard index will be our measure of choice.

As introduced before, our proposal is to evaluate image segmentation by finding its upper-bound performance with respect to annotated objects. There remains a question to be answered, however: given a partition and a ground truth, which is the upper-bound, or optimal, region selection? That is, which is the set of regions from a partition that merged together form the best object with respect to a ground truth?

This looks like a trivial question that is usually answered as: form the set by selecting those regions whose overlap with the ground-truth is higher that 0.5. In other words, select those regions that have more pixels *inside* the ground-truth object than *outside*. We will refer to this technique as the *local* baseline, and as we will show in the experiments, this technique does not achieve the optimal result in a significant number of cases.

Ge et al. [GWL06, GWL07] proposed an improved heuristic to try to reach the upper-bound performance but the technique does not provide the optimal results in a high number of cases either. Section 2.2 formalizes and describes these two baseline region selection techniques.

Section 2.3 mathematically formalizes the problem of finding the optimal region selection and proposes an efficient algorithm to solve it. We refer to this technique as **optimal region selection**.

We perform an extensive experimental validation. First, for the results to be completely unrelated to any segmentation technique, Section 2.4.1 performs an academic example consisting on selecting the regions from the manually-generated partitions of BSDS300 to form the objects on the same images from the DCU object database. To assess the techniques in a realistic scenario and evaluate the trade-off between quality and number of regions, Section 2.4.2 performs an extensive test using seven state-of-the-art segmentation algorithms on PASCAL 2012 and DCU.

2.2 Baseline Region Selection Techniques

This section describes the baseline techniques used in the literature to select the set of regions from a partition that merged together better represent a ground-truth object. Formally, we represent the partition to be assessed as a set of N regions R_i as follows: $P = \{R_1, \ldots, R_N\}$,

where $R_i \cap R_j = \emptyset$ for $i \neq j$, and $\cup_{i=1}^{N} R_i$ covers the whole image. As introduced before, P_{gt} is the ground-truth object region.

The most intuitive (and used in practice) technique considers as "part of the object" those regions that overlap with the ground truth at least in half the area of the region:

$$P_l = \bigcup \left\{ R_i \;\middle|\; \rho_l\left(R_i, P_{gt}\right) > 0.5 \right\} \qquad \rho_l\left(R_i, P_{gt}\right) = \frac{|R_i \cap P_{gt}|}{|R_i|},$$

where P_l refers to the positive pixels formed by the technique, which will refer to as **local**.

For very undersegmented results, i.e., partitions where the regions are much bigger than the ground-truth object, this technique does not ensure that any region will be selected. Selecting no region implies that $J = 0$, a value that affects significantly on the statistics of the results: in terms of optimizing J, this is clearly not an upper-bound of the performance of the algorithm.

In [GWL06, GWL07] the set of regions selected to form the best representation (P_{Ge}) is:

$$P_{Ge} = \bigcup \left\{ R_i \;\middle|\; \rho_{Ge}\left(R_i, P_{gt}\right) > 0.5 \right\} \qquad \rho_{Ge}\left(R_i, P_{gt}\right) = \max\left\{ \frac{|R_i \cap P_{gt}|}{|R_i|}, \frac{|R_i \cap P_{gt}|}{|P_{gt}|} \right\}.$$

The first term of the maximization selects those regions that are, at least, half-inside the ground-truth region, as it would seem sensible. The second term selects those regions that, although not being half-inside, cover at least 50% of the ground-truth region. This region selection technique has been used as an evaluation measure in works such as [GAV+08, WO10, AGBB12]. We will refer to this technique as **Ge** [GWL06].

This procedure does not guarantee that at least one region will be selected either. Figure 2.1 shows a ground-truth object, a circle in solid line, along with a partition in dotted lines. In this case, there is no region in the partition which covers more than half the ground-truth object, nor any one which is half inside it. This case corresponds to $J = 0$, while selecting any overlapping region would entail a better J.

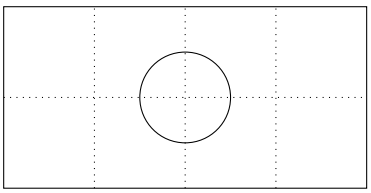


Figure 2.1: Example where the local and Ge [GWL06] strategies lead to $J = 0$

To ensure that at least one region is always selected, we could simply select the single region that maximizes J itself. Formally:

$$P_o = \begin{cases} \cup \{ R_i \,|\, \rho(R_i, P_{gt}) > 0.5 \} & \text{if } \max_i \rho(R_i, P_{gt}) > 0.5 \\ \arg\max_{R_i} J\left(R_i, P_{gt}\right) & \text{otherwise} \end{cases}$$

where ρ can be either ρ_l or ρ_{Ge}, and P_o is the selected set of regions.

It may be argued that the values of J obtained for the cases where the result is too undersegmented, and so no region is selected by the local or Ge strategies, should not be taken into account because they represent a *miss* of the segmentation algorithm, i.e., the partition is not capable of representing the object by means of its regions. This is indeed true, but letting the

value J be zero does not seem the correct approach to not take them into account. Instead, the region selection algorithm should always look for the best representation in terms of J and, a posteriori, set a threshold on J under which the partitions can be considered as misses. In realistic scenarios with small and challenging objects, this type of results are found in a significant number of cases and thus they may affect the final evaluation result if left to zero.

2.3 Optimal Region Selection Technique

This section presents an algorithm to actually find the optimal representation that can be formed merging regions from a partition. In other words, it finds the actual upper bound quality that can be achieved by combining the original regions. To do so, we model the problem mathematically and propose an efficient algorithm to solve it.

Formally, our objective is to optimize the Jaccard similarity between P_{gt} and $P_o = \cup_{j\in\mathscr{J}} R_j$, being the latter the representation formed by the optimal strategy, where $\mathscr{J}$ is the set of indices of the selected regions by this strategy. Recalling Equation 1.2 we have:

$$J\left(P_o, P_{gt}\right) = \frac{\left|P_o \cap P_{gt}\right|}{\left|P_o \cup P_{gt}\right|} = \frac{\left|\left(\bigcup_{j\in\mathscr{J}} R_j\right) \bigcap P_{gt}\right|}{\left|\left(\bigcup_{j\in\mathscr{J}} R_j\right) \bigcup P_{gt}\right|}$$

Regarding the numerator, we have:

$$\left|\left(\bigcup_{j\in\mathscr{J}} R_j\right) \bigcap P_{gt}\right| = \sum_{j\in\mathscr{J}} \left|R_j \bigcap P_{gt}\right| = \sum_{j\in\mathscr{J}} TP_j$$

where $TP_j = \left|R_j \cap P_{gt}\right|$ is the number of True Positive pixels from region j and the first equality holds given that $R_i \cap R_j = \emptyset$ for $i \neq j$.

Regarding the denominator, we can write:

$$\begin{aligned}\left|\left(\bigcup_{j\in\mathscr{J}} R_j\right) \bigcup P_{gt}\right| = \left|\bigcup_{j\in\mathscr{J}} \left(R_j \bigcap \overline{P_{gt}}\right) \bigcup P_{gt}\right| = \sum_{j\in\mathscr{J}} \left|R_j \bigcap \overline{P_{gt}}\right| + \left|P_{gt}\right| = \\ = \sum_{j\in\mathscr{J}} FP_j + \left|P_{gt}\right|\end{aligned}$$

where $FP_j = \left|R_j \bigcap \overline{P_{gt}}\right|$ is the number of False Positive pixels from region j. The Jaccard coefficient can the be written as:

$$J\left(P_o, P_{gt}\right) = \frac{\sum_{j\in\mathscr{J}} TP_j}{\sum_{j\in\mathscr{J}} FP_j + \left|P_{gt}\right|}$$

Let us define:

$$\begin{aligned}\mathbf{tp} &= \left(TP_1, \ldots, TP_N\right) = \left(\left|R_1 \cap P_{gt}\right|, \ldots, \left|R_N \cap P_{gt}\right|\right) \in \mathbb{N}^N \\ \mathbf{fp} &= \left(FP_1, \ldots, FP_N\right) = \left(|R_1 \cap \overline{P_{gt}}|, \ldots, |R_N \cap \overline{P_{gt}}|\right) \in \mathbb{N}^N \\ \mathbf{x} &= \left(x_1, \ldots, x_N\right) \in \{0,1\}^N\end{aligned}$$

where $x_j = 1$ if $j \in \mathscr{J}$, and 0 otherwise. Recall that N is the number of regions in the partition being assessed. Using this notation we can write the problem of finding the best J as $\mathscr{F}$:

$$\mathscr{F}: \qquad \underset{\mathbf{x}}{\text{maximize}} \;\; J = \frac{\left(\mathbf{tp}, 0\right) \cdot \left(\mathbf{x}, 1\right)^T}{\left(\mathbf{fp}, \left|P_{gt}\right|\right) \cdot \left(\mathbf{x}, 1\right)^T} \tag{2.1}$$

This problem is known as a **Linear Fractional Combinatorial Optimization** (LFCO) problem [Rad92], fractional programming problem [Din67], or ratio minimization problem [KBR07]. Kolmogorov [KBR07] proposes a generic methodology to solve this problem using parametric max-flow algorithms.

Alternatively, as presented by Radzik [Rad92], in this work we will use the methodology we derive below. For convenience, let us define $\mathbf{t} = (\mathbf{tp}, 0)$, $\mathbf{f} = (\mathbf{fp}, |P_{gt}|)$, and, abusing notation $\mathbf{x} = (\mathbf{x}, 1)$. First, $\mathscr{F}$ is equivalent to solving its dual problem $\mathscr{P}$:

$$\mathscr{P}: \text{ minimize } \delta, \text{ s.t. } \frac{\mathbf{t}\cdot\mathbf{x}^T}{\mathbf{f}\cdot\mathbf{x}^T} \leq \delta \quad \forall \mathbf{x}$$

Intuitively, the maximum value of a function is equal to its minimum upper bound for all $\mathbf{x}$. Given that $\mathbf{t}\cdot\mathbf{x}^T > 0$ and $\mathbf{f}\cdot\mathbf{x}^T > 0$, we can rewrite the problem as:

$$\mathscr{P}: \text{ minimize } \delta, \text{ s.t. } \left(\mathbf{t}\cdot\mathbf{x}^T\right) - \delta\left(\mathbf{f}\cdot\mathbf{x}^T\right) \leq 0 \quad \forall \mathbf{x}$$

Let δ^* be the optimal value of δ for $\mathscr{P}$. Let us define the following value:

$$h(\delta) = \max\left\{\left(\mathbf{t}\cdot\mathbf{x}^T\right) - \delta\left(\mathbf{f}\cdot\mathbf{x}^T\right)\right\}$$

Function $h(\delta)$ is convex, piecewise linear, decreasing, and δ^* is its only root. Problem $\mathscr{P}$ is therefore equivalent to finding the root of $h(\delta)$, i.e.:

$$\mathscr{R}: \qquad \text{solve} \quad h(\delta) = 0$$

The solution proposed in [Rad92] is to use Newton's method for solving $\mathscr{R}$, i.e., for finding the root of $h(\delta)$. Algorithm 1 describes the method to solve $\mathscr{F}$ by means of the Newton's method applied to $\mathscr{R}$. The same work proves that the algorithm runs in $O\left(N^3 \log N\right)$ iterations.

Algorithm 1 Newton's method for solving an LFCO problem

1: $\overline{\delta} \leftarrow 0$
2: **while** $h(\overline{\delta}) \neq 0$ **do**
3: maximize $(\mathbf{t} - \overline{\delta}\mathbf{f})\cdot\mathbf{x}^T$ ▷ Linear optimization problem
4: $\overline{\delta} \leftarrow \frac{\mathbf{t}\cdot\mathbf{x}^T}{\mathbf{f}\cdot\mathbf{x}^T}$
5: **end while**
6: $\delta^* \leftarrow \overline{\delta}$

From the practical point of view, as we will show in the experiments in the following section, this formulation as an LFCO and solving it via the Newton method provide us with an efficient algorithm to find the optimal region selection. The successive integer linear problems of the method are solved from Matlab using the IBM ILOG CPLEX optimizer.

2.4 Experimental Results

In this section we experimentally compare the two baseline techniques, namely local and Ge [GWL06], both with the improvement to handle cases with no region selected; with the optimal region selection strategy.

First, Section 2.4.1 shows the results of considering the manual partitions of BSDS500 as the machine result to evaluate, and assess them on the ground-truth objects of DCU. Section 2.4.2 presents a deeper analysis: an evaluation of seven state-of-the-art techniques and two baseline methods on DCU and PASCAL 2012.

2.4.1 Human Performance

This section provides a first overview of the different issues that may arise when selecting regions from a partition to form an object. To be independent of any segmentation method, we consider the multiple human-made partitions on the BSDS500 dataset as the machine results to evaluate. As ground-truth, we consider the 100 masks marked in the DCU database, since they are a subset of the BSDS500.

These experiments will provide us with a practical best achievable performance to normalize the state-of-the-art results, that is, we will be able to evaluate which is the amount of acceptable accuracy given by a human annotator. In other words, we are somehow assessing the human performance when segmenting images.

Table 2.1 shows, in the first row, the mean J value of the three strategies in this experiment. Although the differences in mean are not significant, the fact that the three strategies obtain different performances even in human-made partitions shows that they are not equivalent. In other words, neither the local nor the Ge strategy are capable of always finding the selection of regions that best fits the ground-truth object in terms of the Jaccard index.

	Local	Ge [GWL06]	Optimal
Mean J value	0.9264	0.9271	0.9273
Number of discrepant results	8	7	-
Worst J discrepancy	0.2878	0.0870	-

Table 2.1: Discrepancies between the region-selection strategies when assessing human performance

The second row of Table 2.1 shows the number of discrepant results (out of 541) of each technique with respect to the optimal strategy. These values corroborate the fact that the differences between the strategies in these experiments are not extremely significant.

Nevertheless, given that the degree of oversegmentation and undersegmentation of human partitions is not as high as in actual segmentation algorithms results, the experiment performed in this section cannot be considered as specially challenging for selecting the best set of regions. Therefore, it would be fair to focus on the worst-case scenario to compare the three strategies.

The third row of Table 2.1 shows the worst-case discrepancy between the Jaccard index of the two non-optimal strategies with respect to the optimal one. The worst discrepancy for both baseline strategies is significant, being the local technique considerably worse.

This results also gives us a hint of the challenge of defining the scale in image segmentation: should a method segment full objects or their parts? In the case of humans segmenting images, the mean number of selected regions in the optimal result is 5.4 out of 22 regions in the partition in mean; that is, humans in BSDS500 oversegmented the objects in DCU.

From a qualitative point of view, Figure 2.2 shows the selected set of regions using the three techniques for some of the cases of maximum discrepancy.

The two first rows correspond to degenerate cases where the partition is clearly undersegmented. In these cases, the local technique includes just the regions that are mainly inside the

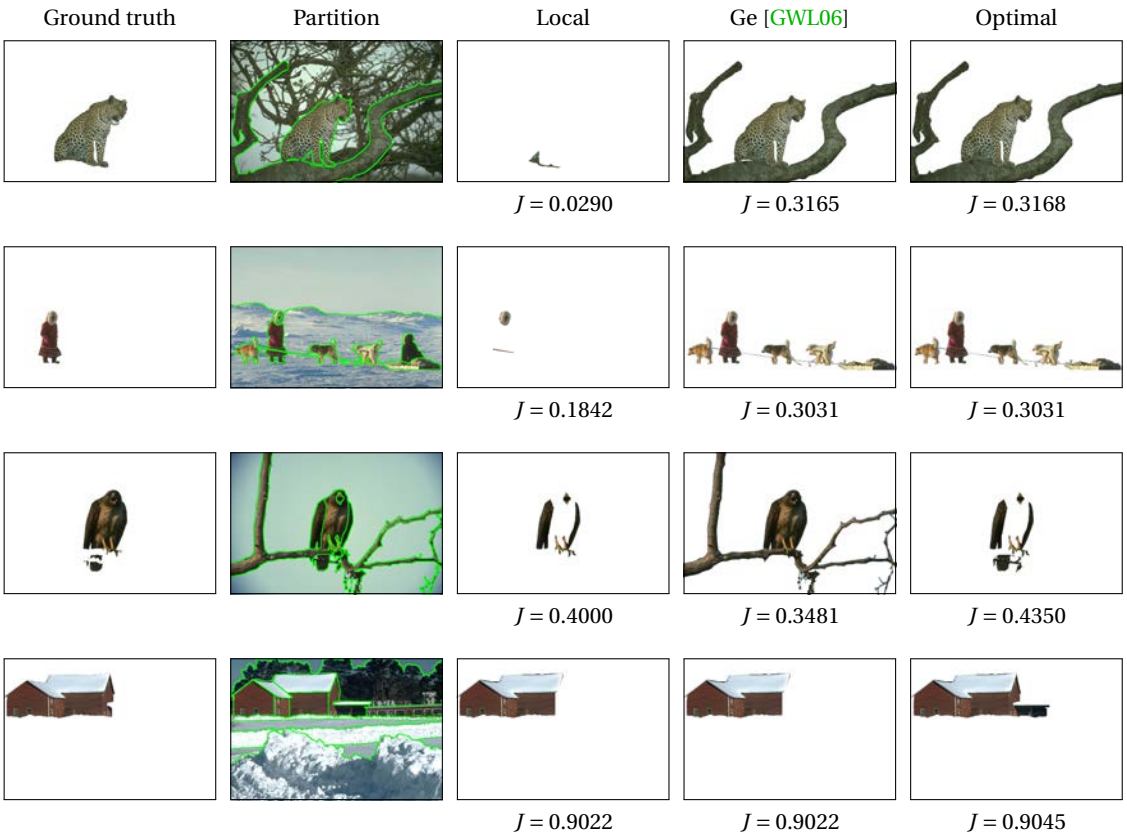


Figure 2.2: Illustration of the discrepant region selection for the three strategies

ground-truth object, while the technique by Ge, is capable of selecting the undersegmented region because it covers the majority of the object.

The third row represents the opposite case, where not selecting the undersegmented region is more favorable, so the local technique gets closer to the upper bound. In any case, these three first results can be understood as misses of the algorithm (very low J), so it can be argued that they can be ignored and thus they do not affect the result.

The last row, however, shows a case where the result is not affected by a severe undersegmentation and the selected regions achieve a good $J \approx 0.9$. Despite this, neither baseline technique achieves the upper-bound quality. In this case, however, the absolute difference in J is not significant.

To further explore the differences between the three strategies, the next section tests them on the larger database of PASCAL 2012 segmentation challenge, by evaluating a wide range of state-of-the-art segmentation techniques.

2.4.2 State-of-the-Art Comparison

In this section we assess the three strategies compared in this paper when selecting the set of regions from the following seven state-of-the-art segmentation techniques: gPb-UCM [AMFM11], ISCRA [RS13a], NWMC [VMS08], IID-KL [CM10], MeanShift [CM02, CGM02], NCut [SM00], EGB [FH04]. In the case of flat segmentation techniques, we sweep their parameters so that

the number of regions in the partitions covers a range between 10 and 100 approximately. In the case of hierarchical techniques, we extract some partitions in the merging sequence so that the number of regions is in the same range.

We segment the images in two datasets with object-based ground-truth: DCU and PASCAL 2012, the former on the full 100 objects and the latter on the 3427 objects of the validation set of the segmentation task. As presented in the previous section, the results will be up-bounded by the human performance estimated by the partitions in BSDS500 in the case of DCU. In order to somehow normalize the results, we also segment the images using two simple baselines: Quadtree and Random. Recall Section 1.2 for a description of these methods and all the state-of-the-art techniques.

We will first show the evaluation and comparison of all segmentation techniques in terms of the optimal region selection technique and then we will analyze the differences between the optimal techniques and the local and Ge strategies.

State of the art comparison by means of the optimal strategy: Figure 2.3 shows the comparison of all the techniques introduced above in terms of the upper-bound achievable quality from the object perspective. The horizontal axis represents the number of regions in the partition, which, as introduced above, has been set to sweep between 10 and 100 regions.

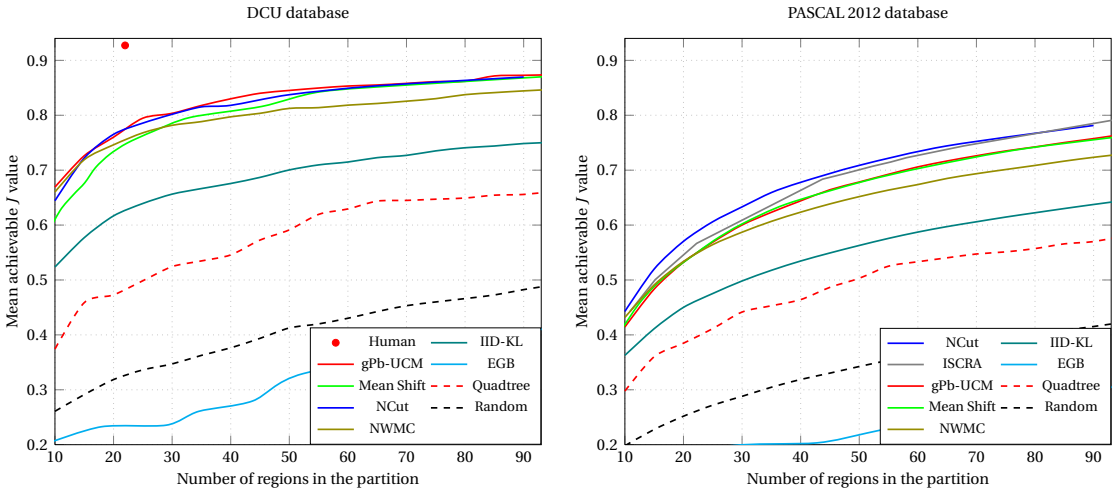


Figure 2.3: State-of-the-art comparison from the object perspective on DCU (left) and on PASCAL 2012 segmentation task (right). Several state-of-the-art techniques compared. Quality with respect to the number of regions in the partitions.

The first general comment is that, as expected for all methods, the more regions in the partition, the better achievable quality. Regarding the differences between databases, it is clear that PASCAL 2012 objects are more challenging than those in DCU.

The first unexpected result comes from EGB, being noticeably below the two baseline techniques, which is counter-intuitive and we will analyze further below. With the exception of EGB, the two baseline techniques are consistently worse than the state-of-the-art techniques. Comparing the two baselines, in this context the Quadtree is noticeably better than a random partition.

Figure 2.4 depicts some representative examples to shed some light into this behavior. The

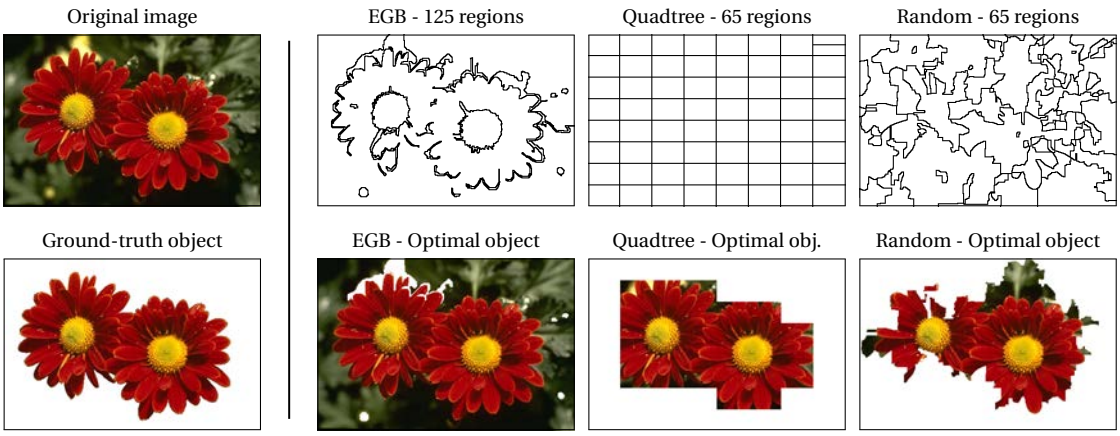


Figure 2.4: Representative examples of EGB with respect to the baseline methods

left-most column shows an image with its ground-truth object from DCU. The three columns on the right show the partition and optimal object from EGB and the two baselines (Quadtree and Random).

Looking at the partitions, it is clear that EGB is much better than the other two in the sense that almost all semantical meaning in the image is kept. From a *region-based* point of view, however, the partition is formed by tiny regions along the boundaries of the object (note that the EGB has 125 regions while the baselines just 65).

In contrast, the baseline partitions have nothing to do with the image content, but taking a look at the optimal objects, the EGB one is almost the whole image, while the baselines are capable of at least delineating the object approximately.

This is so because the EGB result keeps the true contours, but from the point of view of regions, the partition is heavily undersegmented. On the other hand, the Quadtree partition has nothing to do with the image contours, but since the regions are evenly distributed, we are able to get the approximate shape. To sum up, this object-based analysis **penalizes undersegmentation strongly**, causing the EGB results to be even worse than a random partition, and a Quadtree to be considerably better than a random partition.

As a side note, we are not implying that EGB cannot be parameterized better: as we will see in next chapters in this dissertation, we optimized the EGB parameters in terms of a *boundary-based* measure, which does not strongly penalize undersegmentation. Finding better parameters of EGB in terms of object-based measures is out of scope of this thesis.

At the upper part of the plot of DCU, humans obtain a quality of J =0.92 with 22 regions in the partitions in average, which is coherently much better than any automatic method. The gap with respect to 1 shows that image segmentation is an ill-posed problem in which not even humans agree 100%.

Regarding the rest of state-of-the art techniques, gPb-UCM and NCut are the best in DCU, while NCut and ISCRA in PASCAL. Again, these results may be related to the fact that gPb-UCM is trained on BSDS500, while ISCRA on PASCAL, so we could conclude that the safest bet is on NCut.

Figure 2.5 shows the same plot as Figure 2.3 but the horizontal axis is the number of selected

regions to form the optimal object, instead of the number of regions in the partition. This allow us to evaluate the number of regions in which each technique over-segments the objects of interest, regardless of the number of regions in which the background is divided. The plot shows that humans represent the objects of interest with around 5 regions in mean, while the automatic techniques do not start to saturate until 12 or 14 regions are selected.

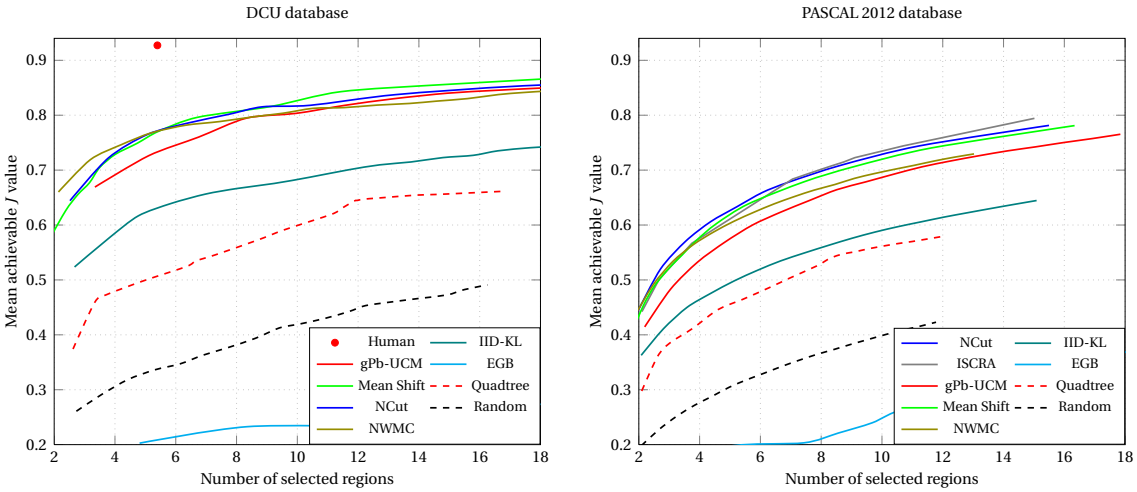


Figure 2.5: State-of-the-art comparison from the object perspective on DCU (left) and on PASCAL 2012 segmentation task (right). Several state-of-the-art techniques compared. Quality with respect to the number of selected regions.

Regarding the comparison of techniques, it is noticeable that gPb-UCM looses positions in the ranking. One possible explanation is that, at the same number of regions in a partition, the rest of techniques over-segment less the objects marked in the ground truth and instead divide the background more. We do not get deeper into the comparison of hierarchical techniques in this section because, as we will see in the next chapter, assessing them *directly*, i.e., without previously extracting a set of partitions from them, entails much better results.

Region selection techniques comparison: The plots shown until this point are obtained using the optimal region selection technique. Figure 2.6 evaluates the differences between this technique and the baseline technique Ge [GWL06] considered in terms of three *meta-measures*, all of them the lower the better. (The comparison with the local technique is almost identical to that of Ge, so we skip it.)

First, it shows the mean loss in achievable quality ( ) reflected by the baseline measures, that is, how far the results from Ge are from the *real* optimal value. This value is around 1% for all state-of-the-art methods in both databases.

Second, we have analyzed the variance of the results, measured by the increase of the standard deviation of the results by the baseline technique with respect to that of the optimal technique ( ). In this case, the increase gets to 6% in some cases, although for the best performing techniques this value remains very low. The differences are therefore not sufficient to significantly change the ranking between segmentation techniques.

Finally, we show the percentage of cases in which the region selection from the baseline tech-

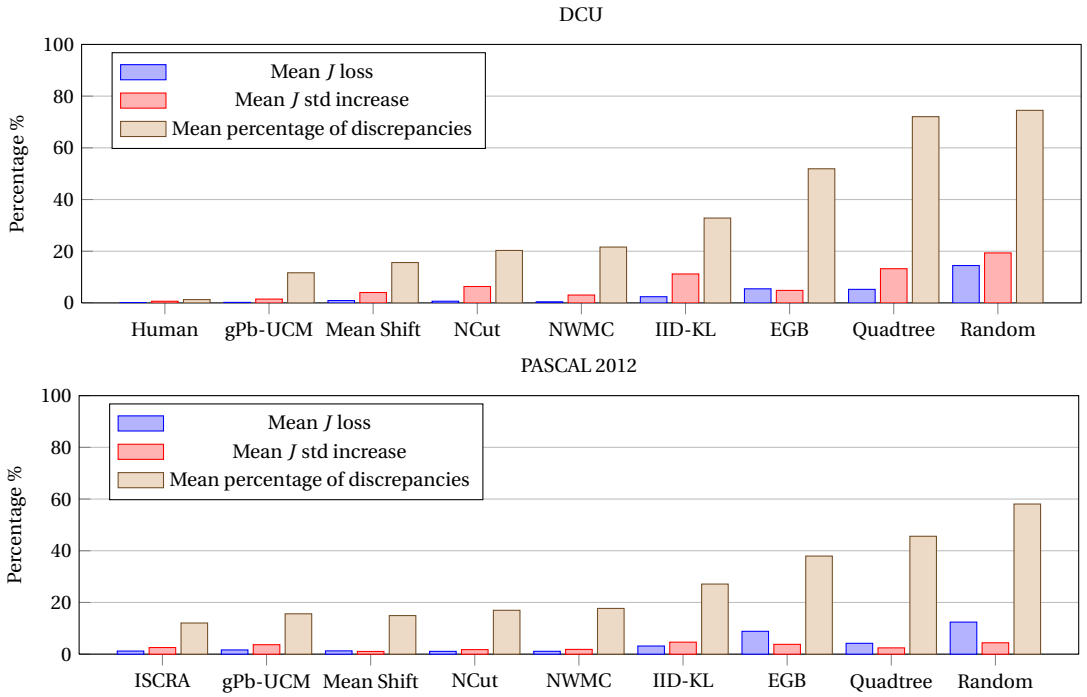


Figure 2.6: Mean loss of achievable quality, mean increase of the standard deviation of J, and mean percentage of discrepant results

nique is not the optimal one, that is, the number of discrepancies between the optimal and the baseline technique (▭). These values get to around 20% for some state-of-the-art techniques.

Overall, we can conclude that the baseline techniques do not reach the optimum region selection in a significant number of cases, but in these cases the difference in mean achievable quality is around 1% and the ranking between segmentation techniques does not vary significantly. To further assess whether these discrepancies are relevant in practice, we focus on the 5 best-performing segmentation techniques.

Figure 2.7 shows the frequency of cases at a given optimal J and loss in achievable J for the cases where the baseline techniques (both of them in the same plot) do not reach the optimum. In other words, each bi-dimensional bin counts the number of discrepant cases with a particular loss in achievable J and optimal J.

The first observation is that there is a clear boundary in the diagonal at 45 degrees. This is forced by the fact that the loss has to be always lower that the optimal J, because otherwise the baseline should give a negative result. We can also observe that the majority of discrepant cases lie in two spots. First, at the bottom-left of the plot, there are the *misses* of the algorithms, where the optimal J is almost 0 but so is for the local technique. Second, at the bottom-left of the plot, there are many results which are *good detections* of the algorithms and the baseline almost reaches the optimal result (recall that this plot only shows discrepant cases).

Interestingly, for results with $J > 0.5$, the higher the optimal J value, the lower the loss in achievable J. In other words, the baseline techniques tend to fail with worse results at *challenging* images where the segmentation technique does not represent the object correctly.

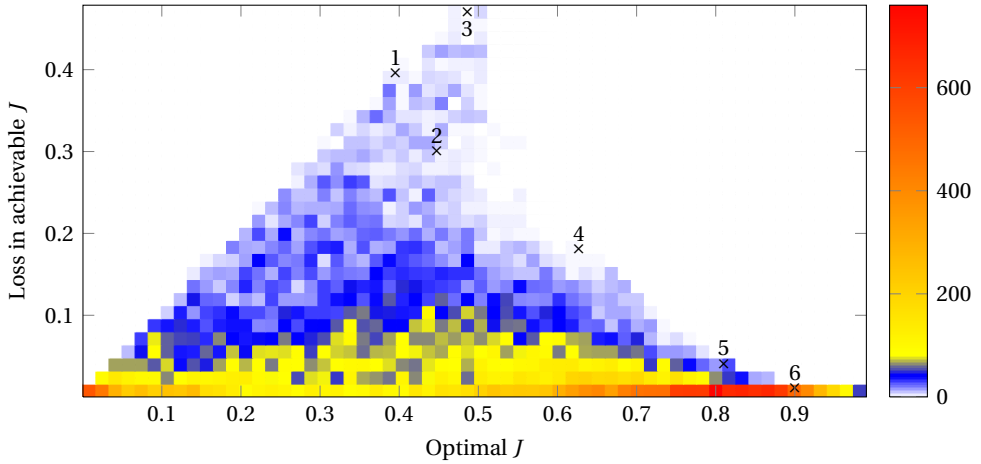


Figure 2.7: Bi-dimensional histogram of the discrepant cases in DCU and PASCAL 2012 for both the Ge and local baseline techniques, in terms of the quality of the optimal selection versus the loss when using the Ge selection technique. The marked points refer to the selected qualitative results in Figure 2.8.

Figure 2.8 show some particular examples at different positions in the plot (marked with a ×) to evaluate whether these discrepancies are qualitatively relevant. First, result (1) shows a case where the local get to the optimal while Ge does not, and result (3) the other way around. In between, in result (2) neither baseline technique gets to the optimum representation. These three first results, however, correspond to misses of the segmentation algorithm, and, as we saw in Figure 2.7, this kind of results are not the most common.

The last three results (4)-(6) correspond to acceptable detections of the segmentation technique, and in these cases, both baseline techniques give the same results, and the loss is less significant. The last result (6) is a representative of the most common discrepant case (hot spot in Figure 2.7), where the optimal region selection further improves a very good result.

Regarding the computational cost, the extra time spent (that is, without considering the steps that are common to all region selection strategies) by the optimal solution with respect to the local approach using MATLAB® is 0.40 ± 0.14 seconds, which is generally negligible with respect to the majority of processes that are involved in image segmentation.

To sum up, we have shown that the baseline techniques for region selection do not reach the optimal representation in a significant number of cases. In the cases where the segmentation algorithm performs poorly, the selected regions by the baselines can be significantly different than the optimum. The discrepant cases on partitions with an acceptable quality are the most frequent, although in these cases the difference in achieved J are reduced.

We believe, therefore, that using the baseline techniques can mislead the assessment. Adding the reduced computational load of computing the optimum, we believe that the optimal region selection proposed in this section is the best technique to evaluate flat segmentation algorithms from an object-based perspective.

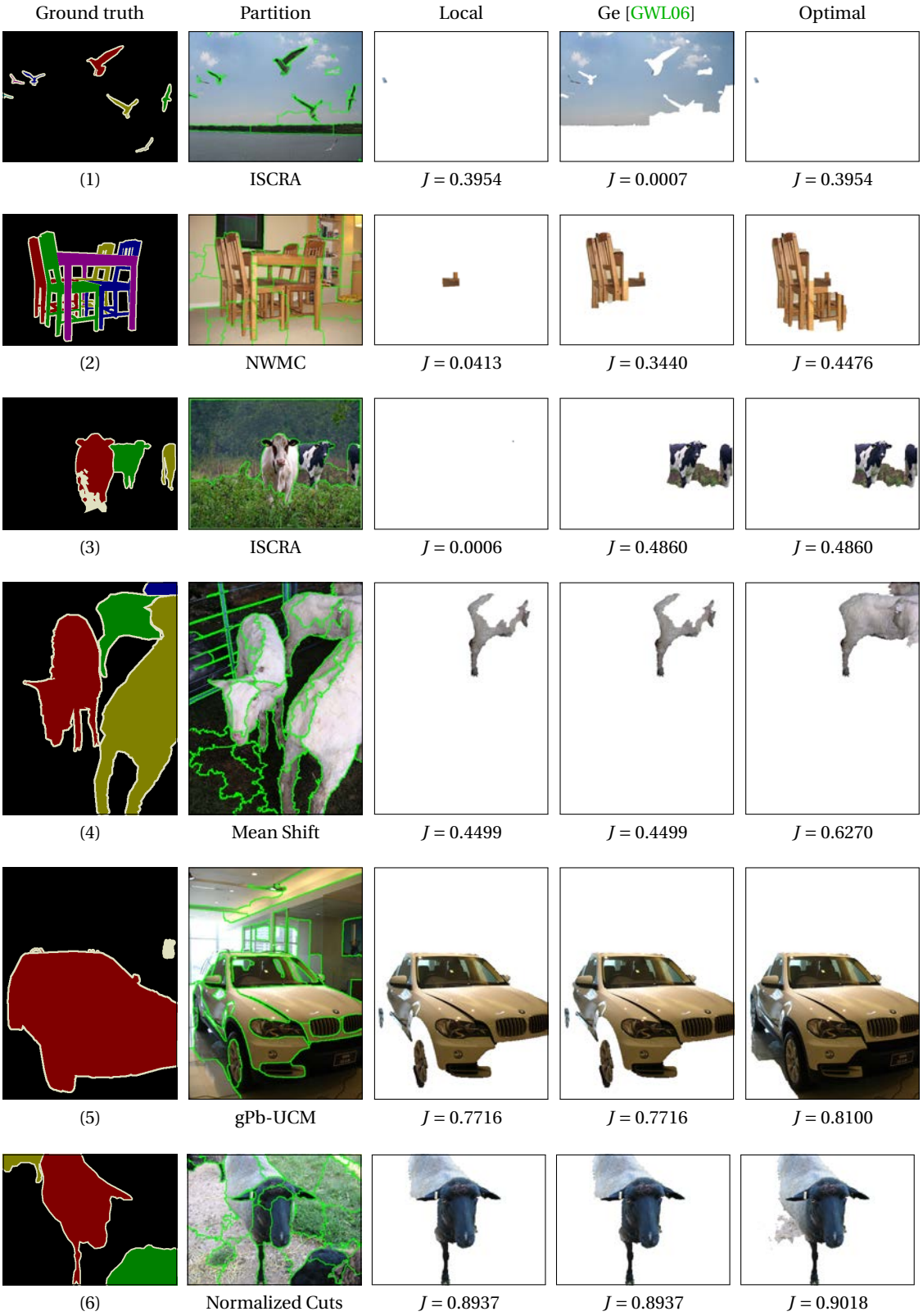


Figure 2.8: Graphical representation of the discrepant region selection for the three strategies on the marked points of Figure 2.7

3 Hierarchical Segmentation

3.1 Introduction

The previous section handles the evaluation of flat segmentation techniques from an object-based perspective. In that case, to handle objects at multiple scales, we sampled the parameters of the segmentation methods for the partitions to sweep a certain number of regions. The region selection was then performed on this sequence of flat partitions.

The final quality achieved in that perspective, therefore, depends on the particular partitions we extract from the hierarchies, apart from the quality of the hierarchies *per se*. The partitions in the merging sequence is a natural choice, but could other partitions entail better results? Working on hierarchies allows us to get rid of the sampling of the scale and to perform the region selection directly on the regions in the tree.

For algorithms that heuristically sample some fusions of multiples regions [ME07], this entails that they have one parameter less to tune and optimize, and it simplifies the region sampling. In practice, we will show that the number of regions needed to get a certain quality when selecting the regions directly from the hierarchy is much lower than when we first extract some flat partitions and we perform region selection on them.

From the point of view of evaluation, we can extract the achievable quality that can be reached by selecting a certain number of regions from a hierarchy, which in practice is the true upper bound that a selection method can achieve. In the case of flat partitions, this relationship was not as close since it was masked by the algorithm used to select the partition/scale in which we performed the region selection.

The first step of this chapter is to generalize the mathematical model to find the best region selection on flat partitions to handle a hierarchy (Section 3.2). As we will show in the experiments, this will allow us to find the upper-bound quality of the objects formed by a certain number of regions on a hierarchy, without having to tune any other parameter of scale.

To add diversity, some object segmentation techniques [RFE+06] combine the input from different image segmentation algorithms. Section 3.3 expands the upper-bound region selection model to consider the merging of regions from different hierarchies.

Finally, Section 3.4 presents the experiments performed on many state-of-the-art hierarchical segmentation algorithms to show the behavior of the evaluation model proposed. The experiments show that the partition/scale selection performed on the previous section indeed masks the actual quality that can be reached by a hierarchy, in the sense that the flat optimal is notably lower, for a given number of regions, than that obtained directly from the hierarchy. The experiments also show that selecting regions from multiple hierarchies effectively takes advantage of the diversity and significantly improves the upper-bound quality.

3.2 Optimal Region Selection From a Hierarchy

This section expands the optimal region selection technique presented in Section 2.3 to hierarchies. Recalling the representation of a hierarchy in Figure 1.10 and Figure 1.9 (page 12), our objective is to find the set of regions from a hierarchy that, merged together, best represent a ground-truth object. If we did not add any further requirement we could find this optimum by selecting the regions from the flat partition formed by the leaves of the tree.

One of the interests of the hierarchies, however, lies in its ability to represent objects at different scales. Ideally, regions in the upper part of the hierarchy should represent objects, and the regions below should refer to their parts. In realistic hierarchies, however, there is always some degree of oversegmentation, that is, the objects are usually not completely represented by a single region. We could then evaluate the quality of a hierarchy by the number of regions it needs to represent an object, or equivalently, we could measure the quality that a hierarchy can reach by merging up to a given number of regions.

Figure 3.1 shows an example of hierarchy (a), and three different objects formed by selecting regions from this hierarchy. First, (b) shows the single region in the hierarchy that better represents the car. As we can see, there is no single region that covers the whole car, so (c) shows the best object representation from regions in the hierarchy, in this case, 3 of them.

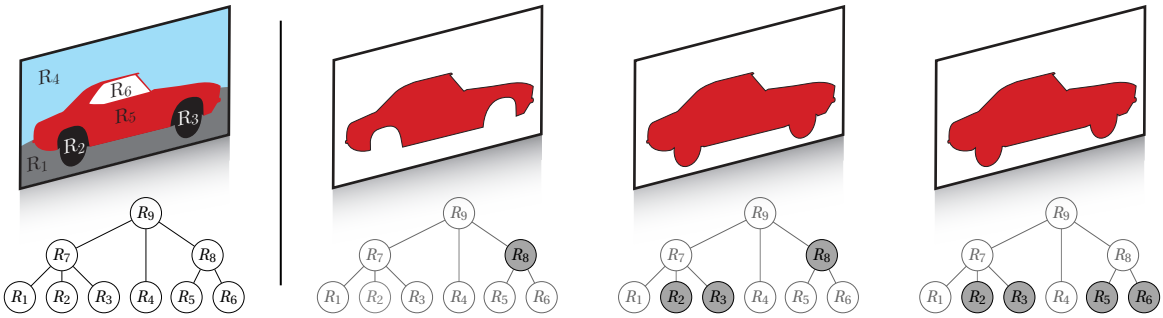


Figure 3.1: Examples of objects (b), (c), (d) formed by selecting regions from a hierarchy (a)

Finally, to show the relevance of the use of a hierarchy, (d) depicts the same optimal object but formed by 4 regions in the flat partition from the leaves of the hierarchy. This last representation could be obtained by the algorithms in the previous section, but we would not discover that we can achieve the same result with one region less. In realistic hierarchies, as we will see in the experiments in Section 3.4, this reduction is much more drastic, from hundreds of leave regions to only a few.

In contrast with the flat segmentation case, it is not straightforward to define heuristic techniques to find the best representation with a limited number of regions, equivalent to the baseline techniques from the previous section. Let us therefore model the search mathematically, generalizing the model proposed in the previous section.

Let H be a hierarchy formed by n_l leave regions L_i $(i = 1,\ldots,n_l)$. Let R_j $(j = 1,\ldots,n_r)$ be all the regions in the hierarchy, including the leaves $(R_i = L_i, i = 1,\ldots,n_l)$. In the hierarchy of Figure 3.1, $n_l = 6$ and $n_r = 9$.

For each leave L_i, let Ω_i be the set of indexes of its ancestor regions R_j, from $R_j = L_i$ to the root

of the tree. Note that the hierarchy is fully defined given L_i and Ω_i for all leaves. In Figure 3.1, for instance, $\Omega_1 = \{1,7,9\}$ and $\Omega_4 = \{4,9\}$.

Recalling the model for flat partitions in Equation 2.1, let us rewrite the previous definitions with the current notation:

$$\begin{aligned}\mathbf{tp} &= \left(\left|L_1 \cap P_{gt}\right|, \ldots, \left|L_{n_l} \cap P_{gt}\right|\right) \in \mathbb{N}^{n_l} \\ \mathbf{fp} &= \left(|L_1 \cap \overline{P_{gt}}|, \ldots, |L_{n_l} \cap \overline{P_{gt}}|\right) \in \mathbb{N}^{n_l} \\ \mathbf{x} &= \left(x_1, \ldots, x_{n_l}\right) \in \{0,1\}^{n_l}\end{aligned}$$

where P_{gt} is the set of positive pixels in the ground truth object, and $x_i = 1$ if we *select* the leave region L_i and 0 otherwise. The model for finding the best representation merging only leaves would then be:

$$\mathscr{F}: \qquad \underset{\mathbf{x}}{\text{maximize}} \;\; J = \frac{(\mathbf{tp}, 0) \cdot (\mathbf{x}, 1)^T}{(\mathbf{fp}, \left|P_{gt}\right|) \cdot (\mathbf{x}, 1)^T}$$

In order to include all the regions in the model, let us define $\mathbf{y} = \left(y_1, \ldots, y_{n_r}\right) \in \{0,1\}^{n_r}$, where y_j is a binary variable that takes a value of 1 if we select region R_j and 0 otherwise. Please note that $y_j = x_j$ for all $j <= n_l$, that is, we are duplicating variables for all leaves. This would not be necessary, but apart from clarifying the notation, it will be easier to generalize to the case of multiple hierarchies, as we will show in the next section.

We would like $\mathbf{y}$ to be the primary variables of the problem and $\mathbf{x}$ to be dependent variables from which the quality of the formed object is computed, as in the flat model. In other words, we would like the two sets of variables $\mathbf{x}$ and $\mathbf{y}$ to be linked to model the fact that selecting a region in the hierarchy forms the same object than selecting all the leaves below it.

If we achieved this linkage, we could modify $\mathscr{F}$ to model the search on the full hierarchy as:

$$\mathscr{G}: \qquad \underset{\mathbf{y}}{\text{maximize}} \;\; J = \frac{(\mathbf{tp}, \mathbf{0}, 0) \cdot (\mathbf{x}, \mathbf{y}, 1)^T}{(\mathbf{fp}, \mathbf{0}, \left|P_{gt}\right|) \cdot (\mathbf{x}, \mathbf{y}, 1)^T} \tag{3.1}$$

where the values of $\mathbf{y}$ do not affect the objective function. The J value would therefore be obtained as in the flat case from the leave regions.

Linking selected regions and selected leaves: Our objective is to add the needed constraints to Problem (3.1) so that for each $y_j = 1$, the values of x_i referring to the leaves L_i below R_j are forced to be $x_i = 1$. Intuitively, x_i must be 1 iff any y_j with $j \in \Omega_i$ is 1 (logical *"or"*).

For the sake of efficiency, we would like the resulting problem to have linear constraints. The linear constraints that are equivalent to the logical *"or"* previously introduced are:

$$x_i \leq \sum_{j \in \Omega_i} y_j \qquad\qquad i = 1 \ldots n_l \tag{3.2}$$

$$x_i \geq y_j \qquad \forall j \in \Omega_i \qquad\qquad i = 1 \ldots n_l \tag{3.3}$$

Intuitively, (3.2) forces x_i to be 0 if all y_j for $j \in \Omega_i$ are 0, that is, it forces L_i *not* to be selected if no ancestor is selected. Equivalently, (3.3) forces x_i to be 1 if any y_j is 1, that is, it forces L_i to be selected if any ancestor is selected.

Recalling the example of Figure 3.1, the values of $\mathbf{x}$ and $\mathbf{y}$ for the three selected objects (b), (c), (d) would be:

$$\text{(b)} \left| \begin{array}{l} \mathbf{x} = (0,0,0,0,1,1) \\ \mathbf{y} = (0,0,0,0,0,0,0,1,0) \end{array} \right. \quad \text{(c)} \left| \begin{array}{l} \mathbf{x} = (0,1,1,0,1,1) \\ \mathbf{y} = (0,1,1,0,0,0,0,1,0) \end{array} \right. \quad \text{(d)} \left| \begin{array}{l} \mathbf{x} = (0,1,1,0,1,1) \\ \mathbf{y} = (0,1,1,0,1,1,0,0,0) \end{array} \right.$$

Note that the values of $\mathbf{y}$ in (c) and (d) are different because they come from different selections of regions, but since they represent the same object, the values of $\mathbf{x}$ are the same.

Forcing realistic results: Up until this point, the model links the selected regions in the hierarchy with the correspondent leaves, but we have not imposed any constraint to the selected regions. This would allow the model to select, for instance, a region and it ancestor at the same time. This does not reflect a realistic behavior, since selecting the ancestor only would entail the same selected object.

Formally, we have to impose that only one region can be selected in each path from the leaves to the root, ensuring that no co-occurrence happens between nodes an their ancestors. Mathematically we model it as follows:

$$\sum_{j \in \Omega_i} y_j \leq 1 \qquad\qquad i = 1 \dots n_l \tag{3.4}$$

Limiting the number of selected regions: As introduced above, one of the interest of the analysis on hierarchies is the ability to limit the search to a certain number of selected regions n_s. This can be imposed directly on the model by adding the following constraint:

$$\sum_{j=1 \dots n_r} y_j = n_s \tag{3.5}$$

Final model: The final model for the region selection on a hierarchy, adding the constraints (3.2), (3.3), (3.4) and optionally (3.5) to the problem (3.1); is as follows:

$$\begin{aligned} \mathscr{G}: \quad & \underset{\mathbf{y}}{\text{maximize}} \quad J = \frac{(\mathbf{tp}, \mathbf{0}, 0) \cdot (\mathbf{x}, \mathbf{y}, 1)^T}{(\mathbf{fp}, \mathbf{0}, |P_{gt}|) \cdot (\mathbf{x}, \mathbf{y}, 1)^T} && && (3.6) \\ & \text{subject to} \quad x_i \leq \sum_{j \in \Omega_i} y_j && && i = 1 \dots n_l \\ & \qquad\qquad x_i \geq y_j \qquad \forall j \in \Omega_i && && i = 1 \dots n_l \\ & \qquad\qquad \sum_{j \in \Omega_i} y_j \leq 1 && && i = 1 \dots n_l \\ & \qquad\qquad \sum_{j=1}^{n_r} y_j = n_s \end{aligned}$$

3.3 Optimal Region Selection From Multiple Hierarchies

Let us consider a set of hierarchies built on the same image. The main difference of this scenario with respect to the single-hierarchy model is that we do not ensure anymore that the regions being selected will not overlap. This entails that we cannot model the true/false positives of the formed object as a linear combination of the true/false positives of the regions being merged nor the leaves below them.

To model the problem as a binary search problem, however, this would be desirable, so we have to modify the problem to find a representation where the union of regions is done on non-overlapping regions. As we will see, we achieve this goal by working on the intersection partition of the leaves partition of all hierarchies, that is, working on the coarsest partition whose regions are always contained in one leave region on each hierarchy.

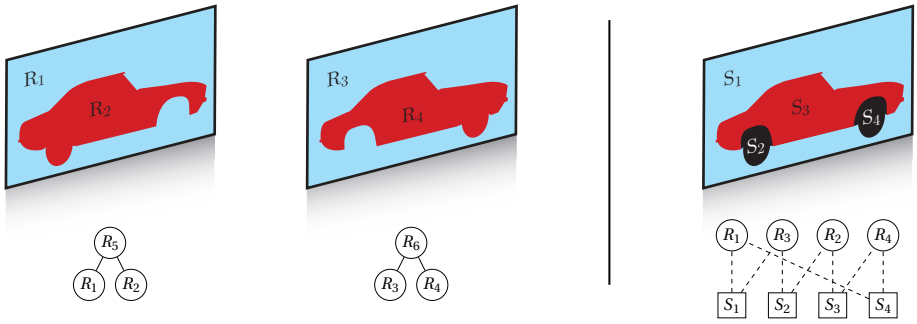


Figure 3.2: Region selection on multiple hierarchies: we work on the intersection partition (right) of the leaves partitions of the two hierarchies (left)

Let us illustrate the process with an example. Figure 3.2 shows two simple hierarchies and their leaves partitions (left), along with the intersection partition of the two leave partitions. The plot in dashed lines represents the relationship between the original regions R_j and the regions in the intersection partition S_i. For instance $S_2 = R_2 \cap R_3$ and $R_4 = S_3 \cup S_4$.

This representation allows us to compute the true/false positives of the merged object as a linear combination of the true/false positives of the regions in the intersection partition. As we did in the previous section, the next step is, given an object formed by a set $\{R_j\}$, find the set $\{S_i\}$ that forms the same object.

Formally, we have n_h hierarchies H_k with a total number of n_r regions (in the example of Figure 3.2, $n_r = 6$). As in the previous section, let y_j, $j = 1 \dots n_r$, be a binary variable that is 1 iff region R_j in the hierarchies is selected. Let the intersection partition be formed by n_π regions S_i. Again, x_i is a binary variable that indicates whether region S_i is selected, meaning that there exists a selected region R_j that intersect with S_i.

Figure 3.3 shows an example of an object formed by selecting regions from the example hierarchies in Figure 3.2. The values that the variables $\mathbf{x}$ and $\mathbf{y}$ would take are shown and the selected regions are highlighted. Note that the object cannot be formed by regions from only one of the two hierarchies, so we are taking advantage of the diversity in multiple hierarchies.

We define Ω_i^k as the set of indices of the regions R_j that intersect with S_i in hierarchy H_k. Note that this is equivalent to the set Ω_i in the previous section generalized to n_h hierarchies and *duplicating* the leaves as the intersection partition. Following the example in Figure 3.2, some of the sets defining the structure would be:

$$\Omega_1^1 = \{1,5\} \qquad \Omega_1^2 = \{3,6\} \qquad \Omega_4^1 = \{1,5\} \qquad \Omega_4^2 = \{4,6\}$$

As before, we must ensure that only one region is selected in the whole path from any leave to the root, and we should link the selected set $\{R_j\}$ with the corresponding set $\{S_i\}$. Generalizing Problem (3.6), we get to the following model for the optimal region selection on multiple hierarchies:

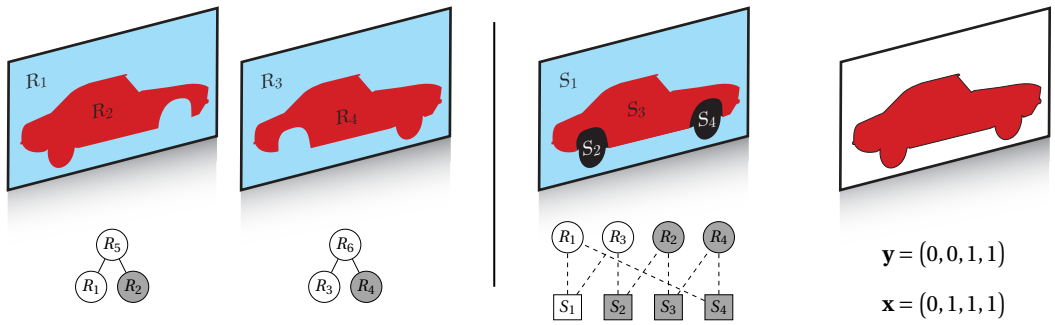


Figure 3.3: Region selection example on multiple hierarchies. The object on the right is formed by the union of non-overlapping regions on the intersection partition, although it is formed by two overlapping regions on the original hierarchies.

$$
\begin{aligned}
\mathscr{H}: \quad & \underset{\mathbf{y}}{\text{maximize}} \quad J = \frac{(\mathbf{tp}, \mathbf{0}, 0) \cdot (\mathbf{x}, \mathbf{y}, 1)^T}{(\mathbf{fp}, \mathbf{0}, |P_{gt}|) \cdot (\mathbf{x}, \mathbf{y}, 1)^T} \\
& \text{subject to} \quad x_i \leq \sum_{j \in \Omega_i^k} y_j \qquad\qquad i = 1 \dots n_\pi \qquad k = 1 \dots n_h \\
& \qquad\qquad x_i \geq y_j \qquad \forall j \in \Omega_i^k \qquad i = 1 \dots n_\pi \qquad k = 1 \dots n_h \\
& \qquad\qquad \sum_{j \in \Omega_i^k} y_j \leq 1 \qquad\qquad i = 1 \dots n_\pi \qquad k = 1 \dots n_h \\
& \qquad\qquad \sum_{j=1}^{n_r} y_j = n_s
\end{aligned}
\tag{3.7}
$$

3.4 Experimental Results

We perform the experiments on the subset of segmentation techniques from the previous section that provide a hierarchical output, namely ISCRA [RS13a], gPb-UCM [AMFM11], NWMC [VMS08], and IID-KL [CM10]; using the same parameters introduced before. The experiments are performed on the PASCAL 2012 segmentation task dataset and we sweep the maximum number of selected regions per object from 1 to 20, allowing also an unconstrained number of regions.

Flat versus hierarchical segmentation: Let us first analyze the differences between working directly on a hierarchy or sampling different partitions at different scales and then selecting the optimal regions. Figure 3.4 shows the results for gPb-UCM and NWMC (the rest of techniques report similar results and are omitted for the sake of readability). The dashed lines correspond to the optimal result on flat partitions and the solid line to that done directly on the hierarchies.

The first observation is that, for a given number of regions, the achievable quality working directly on hierarchies is extremely better than working on flat partitions extracted from the same hierarchy. This means that the process to extract partitions from a hierarchy masks the actual quality that can be achieved with it.

On top of that, if the two methods in the plot were to be judged by the curve on flat partitions,

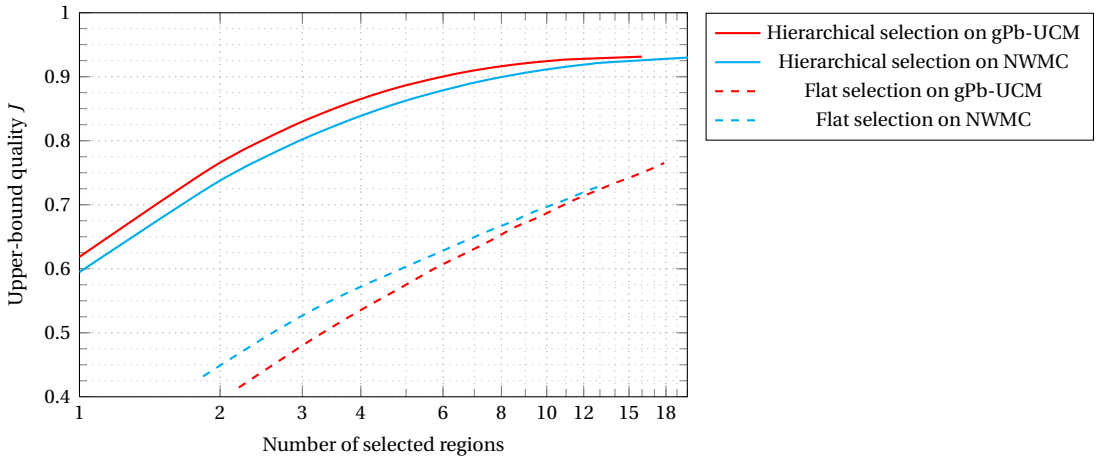


Figure 3.4: Analysis of hierarchies as flat partitions or as full hierarchies. As full hierarchies, their achievable quality at a given number of regions is notably higher.

one could conclude than NWMC is better than gPb-UCM. When selecting regions directly from the hierarchy, however, the ranking is exchanged. This is another proof of the impact that extracting flat partitions can have on the results.

Looking the results from another point of view, we need roughly five regions from the flat partitions to get the same achievable quality that can be reached by selecting only one region directly from the hierarchy. The impact on algorithms that explore the merging of only a limited number of regions [ME07] can thus be dramatic.

To further interpret the result, let us also compute the number of regions to process in comparable scenarios. Assuming a hierarchy with N leave regions, there are at most $2N-1$ regions in the hierarchy. The same hierarchy sampled at four scales would entail four flat partitions at N, $\frac{3}{4}N$, $\frac{1}{2}N$, and $\frac{1}{4}N$, which adds to $2.5N$ regions. Even in the case of only sampling at four scales, therefore, the number of regions to handle is higher in the case of working on flat partitions than when working directly on hierarchies.

To sum up, this experiment shows that hierarchies are much better suited as a pre-processing step to object detection and segmentation, in the sense that they can obtain objects of much better quality, formed by less regions, and searching for them in a smaller pool.

Hierarchical segmentation state-of-the-art comparison: The next step is to analyze the state of the art in hierarchical segmentation from the perspective of object detection. Figure 3.5 shows the achievable quality of the four state-of-the-art segmentation hierarchies analyzed. As baselines we evaluate a random hierarchy and a Quadtree. On the left, the plot shows the raw mean achievable quality of the hierarchies.

The final quality of the hierarchy depends both on the initial segmentation (leaves) and the creation process of the hierarchy. In order to evaluate only the hierarchy creation process, the plot on the right normalizes the achievable quality at each number of regions by the achievable quality on the leaves of that particular hierarchy. This way, all curves start at 1 and the decrease can only be attributed to mistakes in the creation of the hierarchy. The four state-of-the-art techniques do not change their relative behavior when normalized, given that they all start from leaves partitions of the same quality.

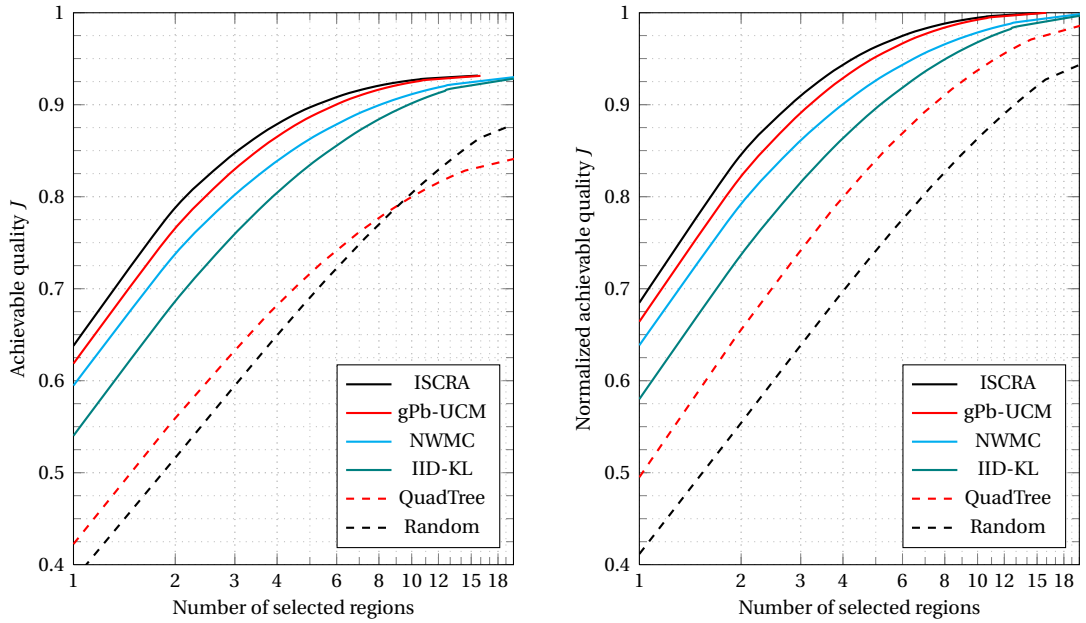


Figure 3.5: Mean achievable Jaccard index J, unnormalized (left) and normalized (right), with respect to the number of selected regions on state-of-the-art segmentation hierarchies. The number-of-regions axis is in logarithmic scale.

In the case of the baselines, in contrast, we do notice a significant difference: the random hierarchy is heavily penalized in front of the Quadtree. Recalling that the random tree starts at the gPb-UCM super-pixel partition, while the Quadtree starts on a uniform grid partition, this shows that by uniformly growing the rectangular regions, as done in the Quadtree, it is more probable to get an approximate location than by randomly growing good super-pixels. In other words, the Quadtree somehow uniformly samples the space of possible locations and shapes, so it usually makes a better guess than a random hierarchy based on super-pixels.

In the unnormalized plot, therefore, the random hierarchy performs very well when sampling the lower part of the hierarchy (when we allow a lot regions to be selected) thanks to the influence of the super-pixels. When we move up through the hierarchy (less regions), however, the randomness predominates. This leaves us to wondering what would be the result of combining good super-pixels *in a Quadtree way*, that is, somehow evolving by spatially uniformly merging the super-pixels.

The number of regions selected when no limit is set also provides valuable information about the hierarchies, since it reflects the number of regions needed to get the same quality that can be achieved by merging regions from the leaves. We also compute the number of regions needed to get to the 99% of this quality, to be more robust to cases where very small regions are selected.

The first two rows of Table 3.1 shows the number of regions needed to achieve 100% and 99% of the maximum quality on each hierarchy. Recalling that gPb-UCM, ISCRA, NWMC, IID-KL, and Random all start from a super-pixel partition of the same achievable quality (see Figure 3.5 left), the differences in the number of regions of the unconstrained case can only be attributable to the hierarchy creation process.

	gPb-UCM	ISCRA	NWMC	IID-KL	Quadtree	Random
Number of regions at 100%	15.98	**15.70**	21.67	22.69	27.61	45.37
Number of regions at 99%	7.21	**6.56**	9.55	10.56	12.96	14.81
Number of regions at 90%	3.21	**2.76**	4.02	5.40	7.71	11.61

Table 3.1: Mean number of regions needed to achieve 100%, 99%, or 90% of the maximum quality on a hierarchy

The first noticeable result is that the number of regions needed to get from 99% to 100% of the maximum quality is considerable. For the case of gPb-UCM, for instance, we reach 99% at 7.21 regions but we need more than two times this quantity (15.98) to get to 100%. This makes it even more important to evaluate the quantity at 99%, since the result will have much less variance.

Focusing therefore on the number of regions at 99%, we corroborate that ISCRA is the hierarchy that makes the most of the available quality at the leaves, in the sense that it is the one that achieves this maximum quality (99% of it to be precise) with the minimum number of regions (6.56 in mean).

The last row of Table 3.1 shows the number of regions needed to get to 90% of the maximum quality. It is noteworthy that with around three regions we get the 90% of the maximum quality. Handling triplets of regions is affordable in computational terms, so these results show that hierarchies are promising as a pre-processing step of object detection and segmentation algorithms.

Figure 3.6 shows some qualitative examples using the four state-of-the-art hierarchies and the two baselines. Each result shown is automatically selected as the one whose quality best matches the mean quality of the method, aiming at being as representative as possible.

Region selection on multiple hierarchies: Finally, let us evaluate the selection of regions from multiple hierarchies. To do so, we allow the optimal region selection algorithm to get regions from each of the four state-of-the-art techniques. Figure 3.7 (left) shows the mean achievable J obtained (——), along with that of the the four original hierarchies.

The plot shows that combining multiple hierarchies achieves notably better results than any of the single state-of-the-art segmentation hierarchies separately, especially for a low number of regions, showing that the four state-of-the-art techniques are indeed diverse. Obviously, this comes at the price of making the pool of regions four times larger.

We also look for the optimum region selection when combining the two baseline techniques. Figure 3.7 (right) shows the mean achievable J obtained (——), which again clearly outperforms the two original hierarchies. Comparing the right and left plot, we observe that the gain obtained on the combined SoA is comparable to that obtained on the baseline techniques, showing that the improvement is due to the diversity itself, independently of the quality of the original techniques.

To further evaluate the combination of multiple hierarchies, Figure 3.8 plots the percentage of regions that are selected from each of the combined hierarchies, that is, it shows where do the regions come from in the case of the state-of-the-art combined result (left) and in the case of baseline combined hierarchies (right).

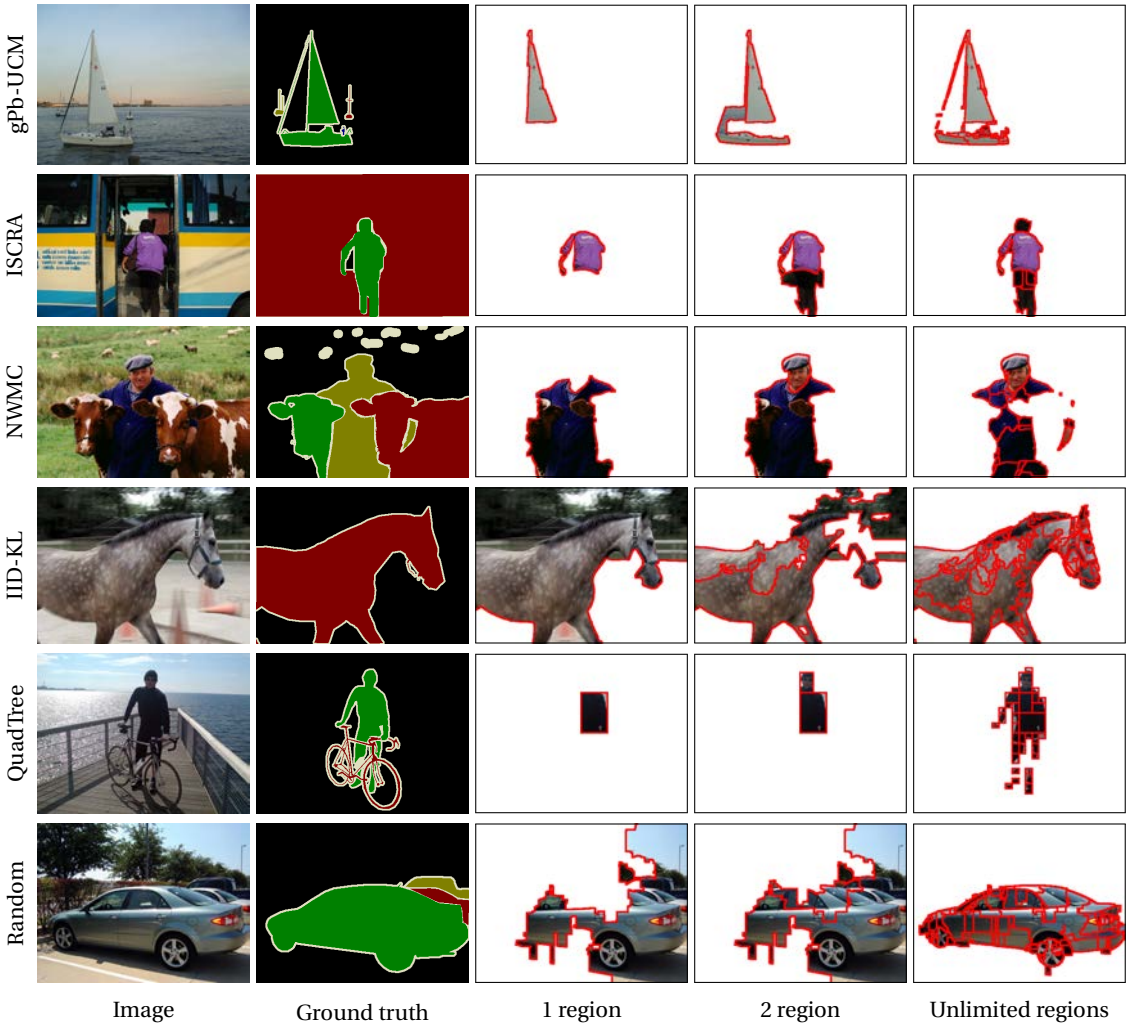


Figure 3.6: Qualitative results of region selection on hierarchies

In the case of the state-of-the-art combination, gPb-UCM is clearly the hierarchy from which more regions are selected, even when only one region is selected. Recalling that ISCRA had better performance in terms of region selection, a possible interpretation is that ISCRA is slightly worse in many of the results and notably better in some objects that gPb-UCM misses. This way, it is possible the mean achievable quality getting only one region from ISCRA is better, although there are more objects in which gPb-UCM is slightly better than ISCRA.

In the case of baseline combination, the plot shows that, the more regions we allow to get, the more are selected from the random hierarchy. Again, this is coherent with the fact that the latter starts from good super-pixels while QuadTree is based on a regular rectangular grid. When only a few regions are selected, in contrast, QuadTree gets the majority of the regions, meaning that at higher levels of the hierarchy, a uniform grid is much better than random mergings.

Let us finalize this analysis with some qualitative results. Figure 3.9 shows three examples from selecting an unconstrained number of regions from the four SoA hierarchies. For each of them we show the original image, the object ground-truth, the object form by combining regions from the four hierarchies, and the regions coming from each of the hierarchies.

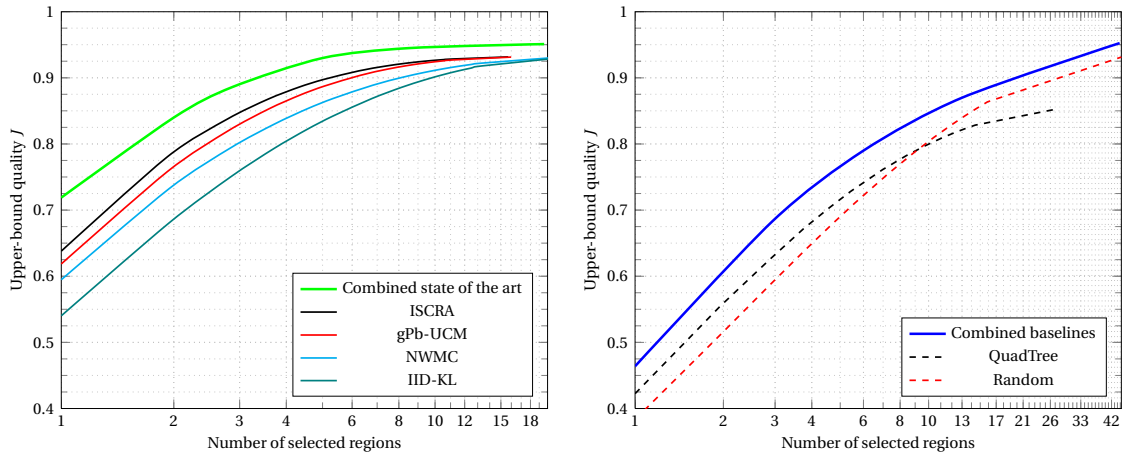


Figure 3.7: Multi-hierarchy region selection analysis: Best achievable quality when merging regions from four state-of-the-art segmentation hierarchies (left) or from two baseline techniques (right) to form PASCAL objects

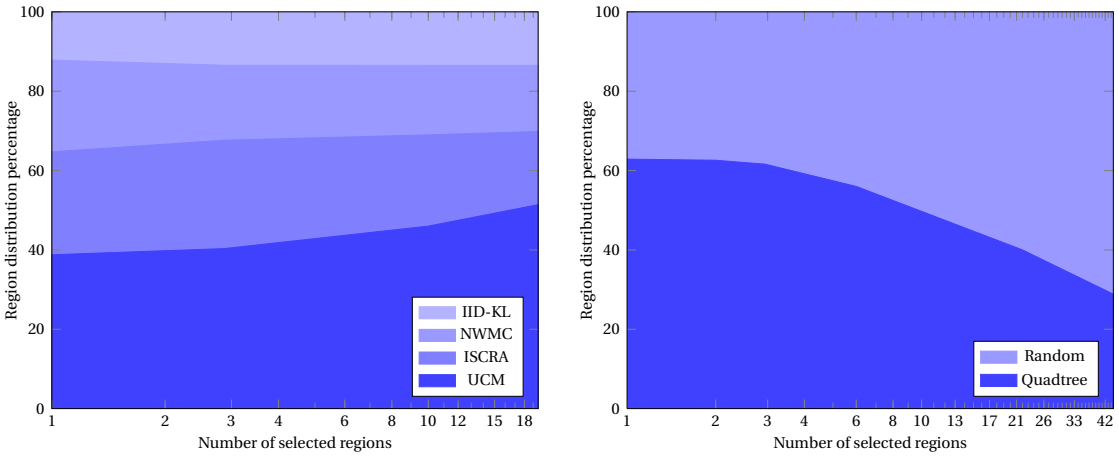


Figure 3.8: Multi-hierarchy region selection analysis: Selected region distribution among the multiple SoA (left) or baseline (right) hierarchies combined

Figure 3.10 show qualitative results from the baseline hierarchies combination when selecting three regions. We can observe that there are results in which all regions are selected from QuadTree (two first rows), all are obtained from Random (third row), or, as in the majority of cases, the Random regions refine little areas of the QuadTree regions.

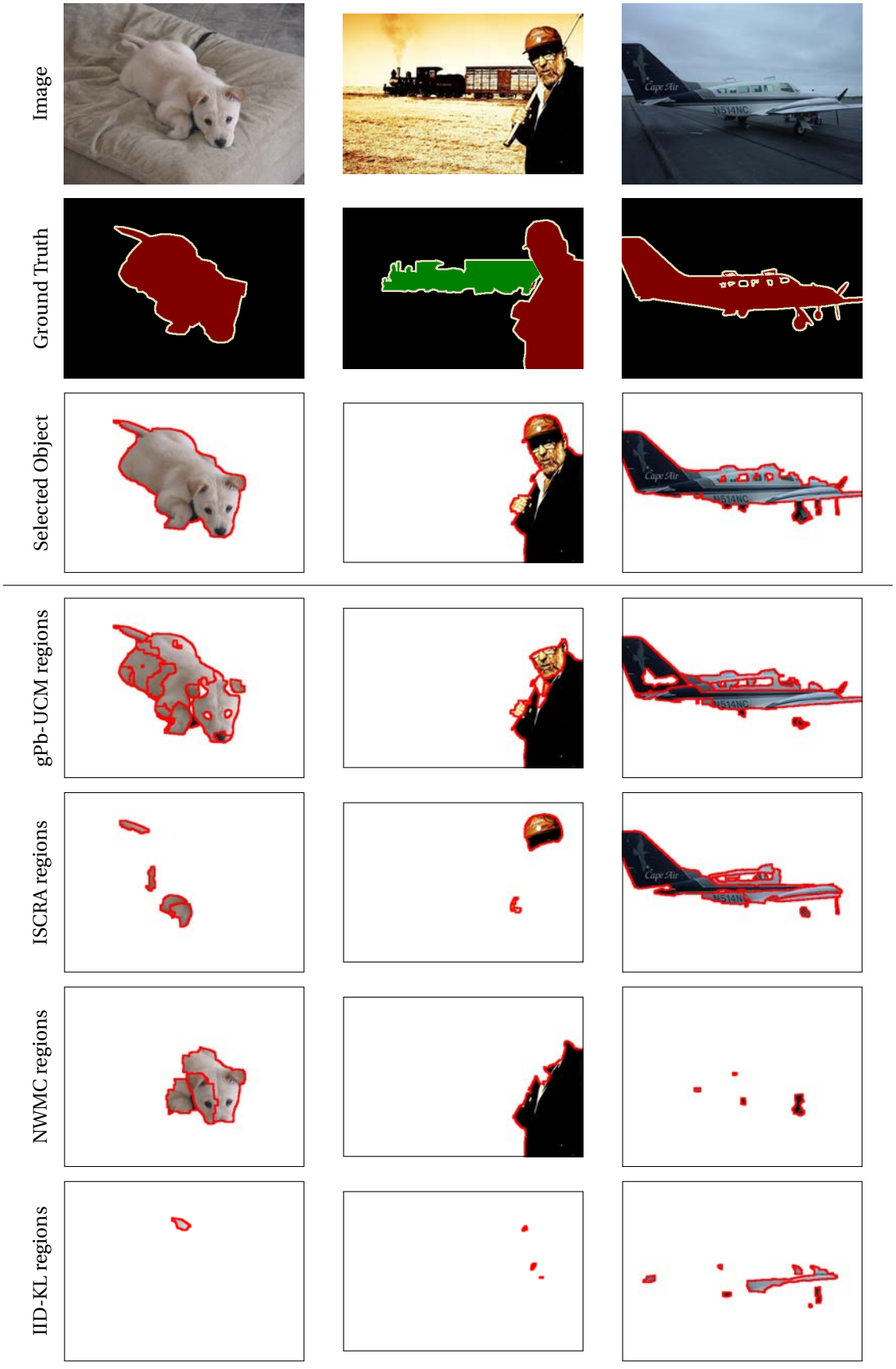


Figure 3.9: Qualitative results of region selection on multiple state-of-the-art segmentation hierarchies

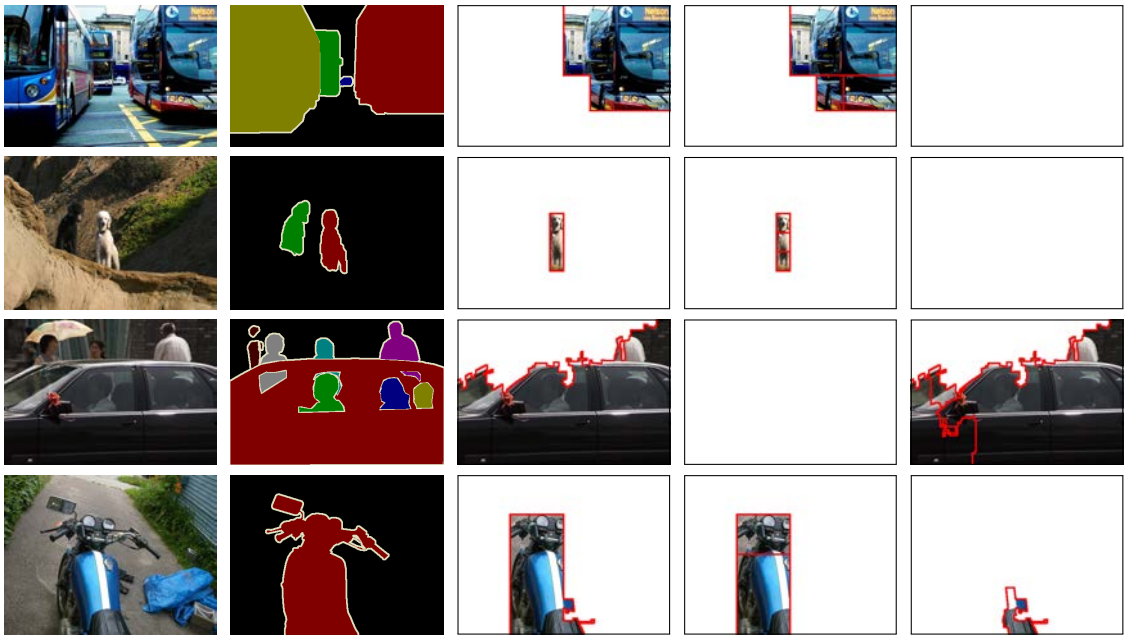


Figure 3.10: Qualitative results of region selection on multiple baseline hierarchies

Part II

Segmentation Evaluation From a Partition Perspective

Introduction

The previous section is focused on the evaluation of the quality of segmentation algorithms, both flat and hierarchical, in terms of how well they perform as a pre-processing step of object detection and segmentation techniques. Evaluating a generic image segmentation algorithm in the specific environment of object detection can be justified by the wide use of this type of algorithms and the availability of large databases that are annotated only at object level, or even only at the bounding box level.

Representing an image as a set of regions instead of plain pixels, however, has many applications beyond object detection, such as image coding [MRHM12], image filtering [SG00], contour detection [AMFM11, PTM10], etc. As a generic representation, image partitions should therefore be assessed also as a whole, *intrinsically*, independently of any final application they can be given.

In this context, the Berkeley Segmentation Dataset (BSDS500) [AMFM11] plays a central role, since it contains 500 images, each of them segmented by more than one individual. What still lacks consensus is how to evaluate the quality of a segmented image with respect to the annotated ground-truth partition. In other words, which measures should we use to compare the agreement between two image partitions?

In comparison with object detection as presented in previous sections, we are generalizing the measures from a detection framework (positive/negative class), to a generic clustering environment; where it is not as straightforward to define a quality measure. This justifies, in part, the current absence of consensus in this regard.

Chapter 4 sheds some light into the generic evaluation of flat image segmentation algorithms. We review and structure the measures in the literature, we propose a new measure from a novel perspective, and we define a way to quantitatively and qualitatively compare the performance of the evaluation measures, which we refer to as meta-measures.

As in the previous part of this dissertation, we then generalize the results to evaluate hierarchical segmentation techniques, but in the context of generic partitions, in Chapter 5. We will introduce the added challenges of evaluating hierarchies in this context and propose new techniques to solve them.

4 Flat Segmentation

4.1 Introduction

This chapter is focused on the supervised evaluation of flat image partitions, that is, the assessment of the quality of image partitions by measuring the agreement between them and the ground-truth partitions of the Berkeley Segmentation Dataset (BSDS500) [AMFM11]. In this context, the evaluation measures that help researchers understand the weak and strong points of their algorithms is of paramount importance.

The first contribution of this chapter (Section 4.2) is to **survey and structure** a wide set of evaluation measures available in the literature. To do so, we show that measures can be classified depending on the interpretation of an image partition they are based on. The most obvious one is to interpret an image partition as a clustering of the set of pixels, so any generic measure to evaluate clustering algorithms can be applied in this context. We can also interpret segmentation as a *detection* problem, aiming at telling apart the pixel contours that are true boundaries from those that are not. Finally, we can also cast the problem to a two-class clustering of the set of all pairs of pixels: those pairs belonging to the same region, and those coming from different regions.

Many of the most used evaluation measures, however, are limited to provide a single number, that is, given a pair of partitions (machine-generated and ground truth) they give us a single number that somehow reflects the degree of agreement between both. In the field of object detection assessment, Hoiem et al. [HCD12] refer to these measures as *performance summary measures* and they stress that the results should be evaluated beyond performance summary measures in order to "help understand how one method could be improved." In other words, researchers need better feedback from the evaluation than a single number.

Back to segmentation assessment, the precision-recall curves for boundaries [MFM04] are good examples of tools that provide richer feedback than the F-measure used as summary. Moreover, as pointed out by [AMFM11], in addition to boundary-based measures, region-oriented measures should be considered when assessing segmentations. However, the current region-based measures are limited to summary ones [UPH07, Mei05, MFM04, AMFM11, KDNS94, HD95, Don00, CCR05, PTM12, PZ12].

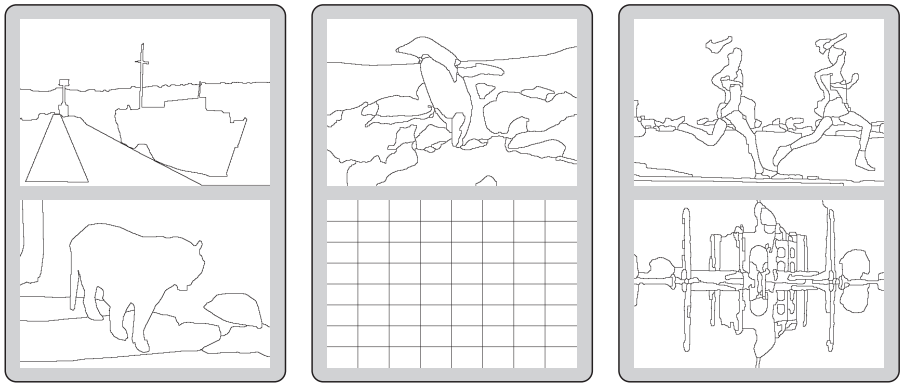
The second contribution of this chapter (Section 4.3) is a region-based precision-recall environment for the assessment of image segmentation. Inspired by [HJBJ+96, HCD12] and by the fact that parts of objects are important clues for object detection [FGMR10], we present the **precision-recall for objects and parts**, which is based on classifying the regions into object and parts candidates.

Summary measures also play a role in performance comparison and researchers have a large

list to choose from, thus the question that now arises is how to compare the goodness of an evaluation measure. In other words, we should define a *meta-measure* to compare the evaluation measures. The principle of a meta-measure is to assume a plausible hypothesis about the segmentation evaluation and analyze how well measures match this hypothesis.

Some previous works based their claims on qualitative meta-measures, that is, showing the behavior of the measures on particular qualitative examples [CCR05, UPH07]. The third contribution of this chapter (Section 4.4) is to propose **two new qualitative meta-measures** that generalize those available in the literature. Extensive quantitative meta-measures, however, are desirable.

The first approach to an extensive quantitative meta-measure was proposed in [Mar03]. The hypothesis in this work was that measures should be able to discriminate between two pairs of human-marked partitions coming from different images (for instance, the two partitions in Figure 4.1.a). In an annotated database with multiple partitions per image, the quantitative meta-measure was defined as the number of same-image partition pairs that the measure judges as less similar than other pairs of partitions coming from different images. [JMIB06] presented a comparison of some measures in terms of this meta-measure.



(a) (b) (c)

Figure 4.1: Examples of the meta-measure principles: How good are the evaluation measures at distinguishing these pairs of partitions?

The fourth contribution of this chapter (Section 4.5) is to present **two new quantitative meta-measures**. Moreover, instead of basing our hypotheses only on human-made partitions, we extend the analysis to partitions from six State-of-the-Art (SoA) segmentation techniques.

The first assumption is that measures should be capable of distinguishing SoA partitions from those obtained without taking into account the content of the image. In this case, we use the Quadtree as a baseline, as in previous experiments of this thesis. The meta-measure is then defined as the number of results from SoA algorithms that are judged better than the quadtree. As a qualitative example, we assess how well a measure distinguishes between partitions like those in Figure 4.1.b.

As a second meta-measure, we assume that any measure should be able to distinguish a partition obtained by a SoA method on a given image from a partition obtained by the same method but on a different image, as the two partitions shown in Figure 4.1.c. The meta-measure in this case is defined as the number of cases in which the measure correctly judges the same-image partition as better.

Finally, Section 4.6 presents the experimental validation of this chapter. We first compare all surveyed measures using the three meta-measures. We show that the two precision-recall measures (boundary- and objects-and-parts-based) have outstanding results as summary measures with respect to the rest of measures, while providing richer information for researchers to interpret the results. We further interpret these two precision-recall frameworks by comparing some SoA segmentation algorithms and show some qualitative results illustrating the differences between the two frameworks.

4.2 Measure Review and Structure

The state-of-the-art measures can be classified depending on the image partition interpretation on which they are based. The most common interpretation is as a clustering of the pixel set into a number of subsets or regions, A partition can also be interpreted as a two-class clustering of the set of pairs of pixels, with some pairs linking pixels from the same region and others linking pixels from different regions. Finally, a partition can be represented as a two-class clustering of the pixel contours into boundaries and non-boundaries. Figure 4.2 illustrates these three different partition interpretations.

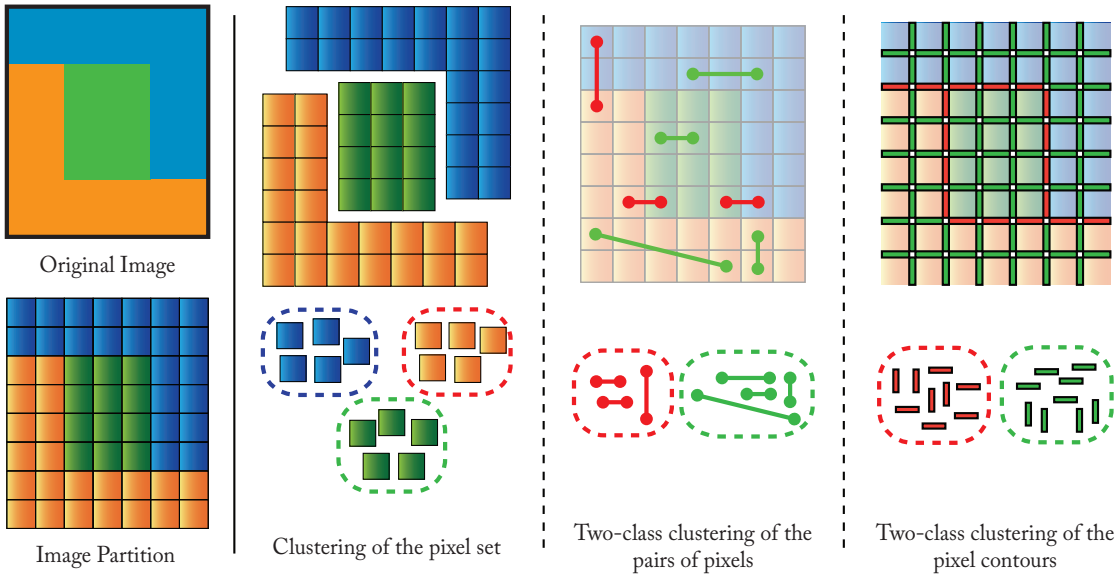


Figure 4.2: The three interpretations of an image partition: clustering of the pixel set, two-class clustering of the pairs of pixels, and two-class clustering of the pixel contours

The contributions of the following sections are to review, de-duplicate, and discuss about the main measures found under each of these interpretations, keeping the notation from the original papers where possible. As we will show below, many measures can be interpreted as generalizations of simple measures such as the F measure, the Jaccard index, or precision-recall. In this chapter, we will refer to these measures as **base measures**.

4.2.1 Pixel-set clustering interpretation

An image can be seen as a set of pixels and so image segmentation as dividing them into clusters (regions). A straightforward way to define a measure to compare two whole partitions could therefore be to generalize the base measures (Jaccard, precision-recall, etc.) via a

region-to-region comparison. A simple question arises, however: which should be the matching between regions from the two partitions to be compared?

Interpreting each region of a partition as a node of a graph, and two partitions as a bipartite graph, a base measure could be used to set the weight of the edges between nodes. A global similarity measure can therefore be defined as the total weight of a bipartite graph, as a generalization of a base measure for each edge. Under this perspective, we explain three strategies found in the literature to define which edges of the bipartite graph to consider: A *local* strategy (Section 4.2.1.1), a *greedy* strategy (Section 4.2.1.2), and the *optimal* strategy (Section 4.2.1.3). We show in each section that several previously proposed assessment measures are the generalization of the base measures via these strategies.

Apart from computing the global weight of the graph, an interesting and intuitive approach also using base measures is explained in Section 4.2.1.4, which *counts instances* of correct detections, and over- and under-segmentations.

An alternative to perform any matching between regions is to assess the intersection of both partitions, i.e., the partition with the minimum number of regions that is a refinement of both regions, as done in the measures in Section 4.2.1.5.

4.2.1.1 Local region matching

The most direct way to generalize a base measure to a pair of partitions S, S' is to compare each region R in S with its best local *match* R' in S', and then to generalize the measure by averaging the region-to-region results. In other words, to compute the weight of a bipartite graph whose edges are selected locally. Formally, being M a base measure and d the global measure, this strategy can be written as:

$$d\left(S,S'\right)=\frac{1}{n}\sum_{R\in S}|R|\cdot\max_{R'\in S'}M\left(R,R'\right) \tag{4.1}$$

where n is the number of image pixels. Note that these measures will not be symmetrical.

Below, some well-known measures used in the literature and presented under different notations and points of view are shown to be equivalent to Equation 4.1 with M being one of the three following base measures: *precision*, F measure, or Jaccard index.

Generalizing *precision*: The **directional Hamming distance** from one partition S to another S', $D_H\left(S\Rightarrow S'\right)$, was introduced in [KDNS94], further analyzed in [HD95] and used in [FMnR+02]. First, each region $R'_i\in S'$ is associated with a region $R_i\in S$ such that $\left|R'_i\cap R_i\right|$ is maximal. Note that this assignment is not necessarily injective. Then, the directional Hamming distance is defined as:

$$D_H\left(S\Rightarrow S'\right)=\sum_{R'_i\in S'}\sum_{R_j\neq R_i}\left|R'_i\cap R_j\right|$$

Applying the following equality:

$$\left|R'_i\right|=\sum_{R_j\in S}\left|R'_i\cap R_j\right|=\left|R'_i\cap R_i\right|+\sum_{R_j\neq R_i}\left|R'_i\cap R_j\right|,$$

recalling that R_i was the region that maximized the intersection with R'_i and obviating the sub-indexes, we get the following alternative expression:

$$D_H\left(S\Rightarrow S'\right)=n-\sum_{R'\in S'}\max_{R\in S}\left|R'\cap R\right| \tag{4.2}$$

where n is the number of pixels of the image.

From another point of view, the work in [CCR05] presents a distance between partitions called **asymmetric partition distance**. The authors intuitively describe it as the minimum number of pixels that have to be deleted from one partition S to be finer than the other (S'). The following property of d_{asym} is presented in the paper as the way to compute it efficiently:

$$d_{asym}\left(S',S\right) = n - \sum_{R\in S} \max_{R'\in S'} \left|R\cap R'\right|$$

Noteworthy, comparing this expression to Equation 4.2, we have that the asymmetric partition distance and the directional Hamming distance are exactly the same value (the order of the partitions is important): $d_{asym}\left(S',S\right) = D_H\left(S'\Rightarrow S\right)$.

This same expression allows us to further understand this measure formally. If we redefine it as a similarity measure and normalize it to range from 0 to 1, we have:

$$s_{asym}\left(S',S\right) = 1 - \frac{1}{n} d_{asym}\left(S',S\right) = \frac{1}{n}\sum_{R\in S} \max_{R'\in S'} \left|R\cap R'\right| = \frac{1}{n}\sum_{R\in S} |R| \max_{R'\in S'} \frac{\left|R\cap R'\right|}{|R|}$$

If S' is viewed as ground truth, the following expression reveals that this measure consists in generalizing the base measure *precision*.

$$s_{asym}\left(S',S\right) = \frac{1}{n}\sum_{R\in S} |R| \cdot \max_{R'\in S'} Prec\left(R,R'\right) \tag{4.3}$$

The asymmetric partition distance is closely related to the *projection number* $p_S\left(S'\right)$ presented in [Don00] in the context of graph clustering since $d_{asym}\left(S,S'\right) = n - p_S\left(S'\right)$. In [JMIB06], this same value is denoted as $a\left(S,S'\right)$.

In [Don00], Van Dongen proved that the measure $d_{vD}(S,S') = 2n - p'_S(S) - p_S(S')$ is a metric Recalling the relationship introduced in the previous paragraph, we have that:

$$d_{vD}(S,S') = D_H\left(S'\Rightarrow S\right) + D_H\left(S\Rightarrow S'\right)$$

In [HD95], the normalized version of this metric is referred to as *normalized Hamming distance*.

Generalizing the F measure: In the context of text document clustering, [LA99] presented a measure to compare clusters that consists in generalizing the base measure F to the whole clustering via the local matching between clusters. In the context of image segmentation, this methodology can be described as:

$$L\left(S,S'\right) = \frac{1}{n}\sum_{R\in S} |R| \cdot \max_{R'\in S'} F\left(R,R'\right)$$

To our knowledge, this measure has not been used in the context of image segmentation.

Generalizing the Jaccard index: In [AMFM11], one of the measures presented is the *covering* of a partition S by a partition S'. It corresponds to the generalization of the Jaccard index (the authors refer to it as *overlap*) via the local-region-matching strategy. Formally, the **segmentation covering** of a partition S by a partition S' can be rewritten as:

$$\mathscr{C}\left(S'\rightarrow S\right) \quad = \quad \frac{1}{n}\sum_{R\in S} |R| \cdot \max_{R'\in S'} J(R,R')$$

4.2.1.2 Greedy region matching

The work in [MH01] presented[1] a methodology to match regions (in the original work, clusters) between two partitions aiming at maximizing the global coincidence between all the matched regions. The algorithm starts by matching the pair of regions with maximum coincident pixels. Then, it iteratively matches the pair of regions with maximum coincident pixels that have not been matched before, in order for the matching to be injective. The measure is defined as:

$$H\left(S,S'\right)=\frac{1}{n}\sum\left\{\left|R\cap R'\right|\middle|R',R\text{ matched}\right\}$$

Interestingly, this measure is symmetrical and, the same way as s_{asym} (Equation 4.3), is a generalization of the base measure *precision* but performing a greedy region matching.

4.2.1.3 Optimal region matching

The intuitive step further to match the regions in two partitions is to compute an assignment that globally maximizes some measure between the pairs of matched regions.

This idea was presented in the area of genetics [AF99], but the algorithm proposed run in exponential time, so it was not feasible for a large number of clusters like in image segmentation.

In [Gus02], the authors present a way to compute this value in polynomial time. To do so, the problem of finding the partition-distance is modeled as finding the matching in a bipartite graph that maximizes its weight, where each node represents a region and the weight of the edges is the number of coincident pixels between each pair of regions. This maximization is efficiently solved using the Hungarian algorithm [Kuh55].

In [CCR05], this measure is denoted as **symmetric partition-distance** (d_{sym}) and it is shown that it is equivalent to the minimum number of pixels that must not be taken into account for two partitions to be identical. Almost at the same time, the work in [JMIB06] presented exactly the same measure naming it **bipartite-graph-matching** (*BGM*) distance. In the context of clustering comparison, [Mei05] defines this measure as **classification error distance** (d_{CE}) and proves some interesting theoretical properties.

Formally, let us assume we have a bipartite graph $G=\left(S,S',S\times S'\right)$, i.e., a graph whose nodes are all the regions from the two partitions S and S' and its edges join any region from S to any region in S'. A *matching* of G is a subset $V\subset S\times S'$ of pairwise non-adjacent edges, i.e., no two edges share a common node. Using this notation, the bipartite-graph-matching distance can be written as:

$$BGM\left(S,S'\right)=\max_{V}\sum_{(R,R')\in V}\left|R\cap R'\right|$$

where V is a matching of G. Again, like in s_{asym} and H, it consists in generalizing the base measure *precision* to the whole partition but now via an optimal matching. It is proven to be a metric in [CCR05].

4.2.1.4 Counting instances

The work in [HJBJ+96] presented the set of measures, known as Hoover measures, to assess and compare range image segmentation algorithms, although they are equally valid for gen-

[1] Although [MH01] first used the measure, we find clearer the definition given in [Mei03] by the same author

eral image segmentation. Given a partition to be assessed and a ground truth partition, the intuitive idea is to *count* the number of correct detections, the instances of under- and over-segmentation, and the missed and noise regions. These instances are defined by means of two base measures: *precision* and *recall*.

Given a threshold $0.5 < T \leq 1$, and the two partitions S (machine) and S' (ground truth), the instances are defined as follows.

A pair of regions $R \in S$ and $R' \in S'$ are considered as an instance of **correct detection** if precision and recall are above T:

$$\frac{\left|R \cap R'\right|}{|R|} > T \quad \text{and} \quad \frac{\left|R \cap R'\right|}{|R'|} > T$$

Note that in the original work, these values are not referred to as precision and recall, but the inequalities are equivalent.

A region $R' \in S'$ and a set of regions $R_1, \ldots, R_k \in S$ are considered as an instance of **oversegmentation** if each R_i's precision and the overall recall are above T:

$$\frac{\left|R_i \cap R'\right|}{|R_i|} > T \quad i = 1 \ldots k \quad \text{and} \quad \frac{\left|(\cup R_i) \cap R'\right|}{|R'|} > T$$

i.e., if there is a set of regions *mostly inside* a ground-truth region that *covers* a large part of this region.

A region $R \in S$ and a set $R'_1, \ldots, R'_k \in S'$ are considered as an instance of **undersegmentation** if:

$$\frac{\left|R \cap (\cup R'_i)\right|}{|R|} > T \quad \text{and} \quad \frac{\left|R \cap R'_i\right|}{\left|R'_i\right|} > T \quad i = 1 \ldots k$$

i.e., if a region covers a set of ground-truth regions.

A region $R \in S'$ or $R \in S$ that is not involved in any of the former classifications is an instance of **missed** or **noise** region, respectively.

Sweeping the possible values of the threshold T we have a curve of evolution of each measure that allows us to assess the method at different tolerances. In [MPB04] the Area Under this Curve (AUC) is considered as a global indicator of performance for each measure.

Although not much attention has been given to these measures, possibly due to the dependence on a threshold T and the fact that not a single measure but several ones are proposed, they are the only ones that are defined using the intuitive terms in which segmentation discrepancies are usually described.

4.2.1.5 Working with the intersection partition

Given two partitions S and S', the intersection partition $S \cap S'$ is the partition with the minimum number of regions that is a refinement of the two. Formally, $S \cap S' = \{R \cap R' | R \in S, R' \in S'\}$.

Bidirectional consistency error: In [Mar03], the consistency of the BSDS300 human partitions is analyzed by means of two measures *GCE*, *LCE*, aiming at being robust against different granularities of the scene interpretation. As the author points out, these measures are not suitable for general-purpose image segmentation evaluation. The same work proposes a measure

that is not transparent to oversegmentation: the bidirectional consistency error (*BCE*), which can be reinterpreted in terms of base measures as follows:

$$BCE(S,S') = 1 - \frac{1}{n}\sum_{R,R'} \left|R \cap R'\right| \min\left\{Prec\left(R,R'\right), Rec\left(R,R'\right)\right\}$$

Information-theoretic measures: The work in [Mei03] introduced a new point of view to clustering assessment measures based on information-theoretic results that can also be applied to partitions. The author defines a discrete random variable taking N values that consists in randomly picking any pixel in the partition $S = \{R_1, \ldots, R_N\}$ and observing the region it belongs to. Assuming all the pixels equally probable to pick, the probability of each outcome of the random variable is: $P(i) = \frac{|R_i|}{n}$, where n is the number of pixels in the image. The entropy of this random variable can be understood as the uncertainty in picking a pixel from the partition, thus the *entropy associated with a partition* S is defined as:

$$H(S) = -\sum_{i=1}^{N} P(i)\log P(i) = -\frac{1}{n}\sum_{R\in S} |R| \log \frac{|R|}{n}$$

Similarly, when having two partitions S and $S' = \{R'_1, \ldots, R'_{N'}\}$, the joint probability of a pixel belonging to a cluster i in S and to i' in S' is defined as: $P(i,i') = \frac{\left|R_i \cap R'_{i'}\right|}{n}$, and so the mutual information between partitions S and S' as:

$$I\left(S,S'\right) = \frac{1}{n}\sum_{R\in S}\sum_{R'\in S'} \left|R \cap R'\right| \log \frac{n\left|R \cap R'\right|}{|R||R'|}$$

This value is interpreted as the uncertainty of the region where a pixel belongs in S' knowing the region it belongs in S. (Zero if equal partitions.)

The **variation of information** is defined in [Mei03] as: $VI(S,S') = H(S) + H(S') - 2I(S,S')$. In [Mei05], the author shows many theoretical properties of this measure such as being a metric.

Given the nature of this measure, no expression in terms of base measures has been found, although its expression in terms of the intersection partition:

$$VI\left(S,S'\right) = 2H(S\cap S') - H(S) - H(S'), \tag{4.4}$$

gives us an intuition of the rationale behind this measure. It compares the uncertainty of the intersection partition with respect to the original uncertainties: if S and S' are not similar, the intersection partition will have many more regions, so more uncertainty.

It is proven that $VI(S,S')$ is bounded by $2\log\left(\max\left(|S|,|S'|\right)\right)$, so the **normalized variation of information** is defined as:

$$NVI\left(S,S'\right) = \frac{VI\left(S,S'\right)}{2\log\left(\max\left(|S|,|S'|\right)\right)}$$

Certain similarities can be found between the Mirkin metric [Mir96], defined as:

$$d_M\left(S,S'\right) = \sum_{R\in S}|R|^2 + \sum_{R'\in S'}\left|R'\right|^2 - 2\sum_{R\in S}\sum_{R'\in S'}\left|R\cap R'\right|^2$$

and Equation 4.4. Furthermore, the Mirkin metric is equivalent to the Rand index, as explained in the following section.

4.2.2 Pairs-of-pixels interpretation

An image partition can be viewed as a classification of all the pairs of pixels into two classes: pairs of pixels belonging to the same region, and pairs of pixels from different regions. Recalling Figure 4.2, the pairs of pixels belonging to the same region are depicted in red and those belonging to different regions in green.

This way, any base measure applied to this point of view can be interpreted as a segmentation assessment measure. Formally, let $I = \{p_1, \ldots, p_n\}$ be the set of pixels of the image and consider the set of all pairs of pixels:

$$\mathscr{P} = \{ (p_i, p_j) \in I \times I \,|\, i < j \} \tag{4.5}$$

Note that $|\mathscr{P}| = \binom{n}{2} = n(n-1)/2$. Given two partitions S and S', we divide $\mathscr{P}$ into four different sets, depending on where a pair (p_i, p_j) of pixels fall [Mei03]:

$\mathscr{P}_{11}$: in the same region both in S and S',
$\mathscr{P}_{10}$: in the same region in S but different in S',
$\mathscr{P}_{01}$: in the same region in S' but different in S,
$\mathscr{P}_{00}$: in different regions both in S and S'.

4.2.2.1 Rand index

The Rand index, originally defined in [Ran71] as a clustering evaluation measure, arises naturally in this context:

$$RI(S,S') = \frac{|\mathscr{P}_{00}| + |\mathscr{P}_{11}|}{|\mathscr{P}|}$$

This measure *counts* the pairs of pixels that have coherent labels for the two partitions being compared, with respect to the number of possible pairs of pixels.

The work in [BHEG02] proved the equivalence between the Mirkin metric (Section 4.2.1.5) and the Rand index:

$$RI(S,S') = 1 - \frac{d_M(S,S')}{2 \cdot |\mathscr{P}|}$$

which interestingly links two different points of views of a partition.

When more than one ground-truth partition is provided for each image, [UH05] proposed the **Probabilistic Rand Index** (*PRI*), which aims at "accomodating refinement *only* in regions that human segmenters find ambiguous and penalizing differences in refinement elsewhere." As shown in the Appendix of [UPH07], the feasible computation of *PRI* consists in averaging the Rand Index *RI* between the assessed partition and each ground-truth partition. An improved version of the measure called **Normalized Probabilistic Rand** (NPR) index is presented in [UPH07].

4.2.2.2 F measure for regions

Defining the objective of image segmentation as detecting those pairs of same-region pixels and applying the base measures precision and recall to this point of view, the measures presented in [Mar03] as precision-recall for regions arise naturally:

$$Prec_r = \frac{|\mathscr{P}_{11}|}{|\mathscr{P}_{11}| + |\mathscr{P}_{10}|} \qquad Rec_r = \frac{|\mathscr{P}_{11}|}{|\mathscr{P}_{11}| + |\mathscr{P}_{01}|}$$

As a summary measure, we propose the **F measure for regions**:

$$F_r = 2\frac{Prec_r \cdot Rec_r}{Prec_r + Rec_r}$$

4.2.3 Boundary-map interpretation

All measures above could be applied to any clustering algorithm, no matter the nature of the elements being classified. In fact, the majority of the indices presented come from the application of general-clustering assessment measures to image segmentation.

Image pixels, however, are spatially distributed in the image plane, and so the concept of neighborhood arises naturally. Therefore, an image partition with connected components can be unambiguously defined by their boundaries, i.e., a bijection could be made between all the possible image partitions and all the possible closed boundary maps.

This assertion justifies that image segmentation can be assessed via the partition boundaries without loss of information. Let us denote as B the set of boundaries between pixels, i.e., the set of *segments* between pixels in the image or the set of pixel contours. Recalling the definition of $\mathscr{P}$ in Equation 4.5 as the set of pairs of pixels in the image, we define the set of pairs of neighboring pixels as $\mathscr{N} \subset \mathscr{P}$. One can define a bijection between B and $\mathscr{N}$ linking each segment to the pair of pixels at each side of the segment.

Using this notation, boundary detection can be understood as a two-class clustering of B, dividing the segments into those being boundaries and those not. Recalling Figure 4.2, the boundary pixel contours are depicted in red, and the non-boundary pixel contours in green.

Applying any base measure to this interpretation can therefore be used to assess image segmentation, as done in [MO10] with the Jaccard index in the context of semi-automatic image segmentation. As the authors pointed, however, unnoticeable shifts in the position of the boundaries can make the measure be almost zero, making the measure unusable.

To be robust to boundary localization, a common approach is to match the boundaries of one segmentation to the boundary segments of the other and then compute a global measure based on this matching. Similarly to the region-matching approaches presented in Section 4.2.1, in the case of pixel-boundaries assignment, one can perform the matching under a local approach (Section 4.2.3.1), using a greedy technique (Section 4.2.3.2), or finding the globally optimum modeling the problem as a maximum-weight bipartite-graph matching (Section 4.2.3.3).

This section does not get deep into the expressions of the measures because, in contrast to the previous sections, the original works already present the expressions in terms of base measures.

4.2.3.1 Local pixel-boundaries assignment

This approach consists in matching each boundary segment of a partition with the closer one of the other partition. Each of the pairs of segments matched are compared by some measure of dissimilarity and then a global measure is obtained by adding up the contributions of each pair.

In [HD95], the distances between pairs of matched segments defined the *distance signatures* of the pair of partitions. A good segmentation entails a low mean and variance of the signature. A high variance indicates that there are some outliers (e.g. leaks in region growing techniques).

A similar approach is presented in [MO10] as *fuzzy Jaccard index*, where the similarity of the segment pairs is defined as their distance weighted by a Gaussian function. Other measures within this context can be found in [HD95, EJ09].

4.2.3.2 Greedy pixel-boundaries assignment

In the approach presented in [BKD01] the matching was forced to be injective aiming at creating a matching globally better. Apart from that, the ground truth included some "don't-care" areas: textured zones of the images in which edges can be ignored. Each edge-detector result gives a point in the (%*False Positives*, %*False Negatives*) plane. The plot of all the points for different parameterizations of the edge detector algorithm gives the so-called *Receiver Operating Characteristic* (ROC) curve.

4.2.3.3 Optimal pixel-boundaries assignment

The first work considering the optimal matching between boundaries is [LH00]. Although the work in [BKD01] is aware of the existence of this technique, the authors claim that the differences between the optimal technique and their greedy approach are not significant.

Good acceptance was reached in [Mar03], where the authors claim that Precision-Recall curves better reflect the performance of edge detectors than ROC curves. The F measure is used as an indicator of the trade-off between the two values, which we will refer to in this work as F_b. The maximum-weight bipartite-graph matching presented in [LH00] is improved in [Mar03] to take edge orientation into consideration and the problem is *sparsified* in order to gain computational efficiency. All pairs of ground-truth and machine-generated boundary pixels that are farther than a predefined threshold are directly discarded as a possible detection.

4.2.4 Structure overview

Table 4.1 summarizes the proposed structure of the main state-of-the-art measures studied in this thesis.

4.3 New Measure: F Measure for Objects and Parts

In the context of image segmentation evaluation, precision-recall curves for boundaries [Mar03, MFM04] are a boon for researchers. They statistically reflect, for instance, that an algorithm is providing too coarse segmentations (low recall, high precision) or instead its results are too fragmented (low precision, high recall).

As pointed out by [AMFM11], however, region benchmarks are also needed apart from the boundary benchmarks when assessing image segmentation. Region benchmarks, however, are currently limited to summary measures as the ones reviewed in Section 4.2.1. (Note that in the vocabulary used in this thesis, region-based measures are the ones based on the interpretation of a partition as a clustering of the set of pixels.)

Partition Interpretation	Measure Representative	References	Notation
Pixel-set clustering	Directional Hamming distance	[HD95, CCR05]	D_H
	van Dongen distance	[Don00]	d_{vD}
	Segmentation covering	[AMFM11]	$\mathscr{C}$
	Bipartite graph matching	[JMIB06, CCR05]	*BGM*
	Bidirectional consistency error	[Mar03]	*BCE*
	Variation of information	[Mei05]	*VoI*
Pairs-of-pixels classification	Probabilistic Rand index	[Ran71, UPH07]	*PRI*
	Precision-Recall for regions	[Mar03]	P_r, R_r
Boundary map	Precision-Recall for boundaries	[LH00, Mar03]	P_b, R_b

Table 4.1: Measure structure overview for the three interpretations of an image partition

This section presents a new region benchmark that goes beyond the summary measures: the precision-recall curves for objects and parts. Motivated by the fact that image segmentation is increasingly being used as a preliminary step for object detection [ME07, AHG+12], we propose to assess segmentation under this perspective, that is, we interpret regions in a partition as potential object candidates, and classify them as correct or not depending on their overlap with the ground-truth regions. Similarly, we interpret regions in an oversegmentation as parts of objects, if merged together can form an object of the ground truth (inspired by [HJBJ+96] in range image segmentation evaluation).

Precision and recall are then computed as the weighted fraction of candidates with respect to the total number of regions, that is, part candidates are only *partially counted*.

Formally, let $S=\{R_1,\ldots,R_N\}$ be an image partition and $\{G_k\}$ a set of ground-truth partitions of the same image. We consider the set $G=\{R'_1,\ldots,R'_M\}$ of all the regions in $\{G_k\}$. For each pair of regions $R_i \in S$, $R'_j \in G$ we compute the relative overlaps as:

$$O_S^{ij} = \frac{|R_i \cap R'_j|}{|R_i|} \qquad O_G^{ij} = \frac{|R_i \cap R'_j|}{|R'_j|}$$

We define an *object threshold* γ_o and a *part threshold* $\gamma_p < \gamma_o$ and classify the regions in both partitions as described in Algorithm 2, where "$\leftarrow$" means that a region is classified only if it previously did not have a more favorable classification.

Let oc and oc' be the number of object candidates in S and G, respectively (note that they can differ, given that G can be formed by more than one partition and thus a region in S can be matched as object with more than one region in G), and pc and pc' the number of part candidates. Regarding the fragmentation candidates, we compute the percentage of the object that could be formed from the matched parts. Formally, we define the amount of fragmentation $fr(R_i)$ of a region $R_i \in S$ as the addition of the relative overlaps of the part candidates matched to R_i:

$$fr(R_i) = \sum_j \left\{ O_G^{ij} \text{ s.t. } O_S^{ij} > \gamma_o \right\} \tag{4.6}$$

$fr'(R'_j)$ is defined equivalently for G. The global fragmentation fr and fr' is computed adding the amount of fragmentation among all fragmentation candidates of S and G, respectively. Figure 4.3 shows a toy example to illustrate the proposed classification and measures.

Algorithm 2 Region candidates classification

for all $R_i \in S$, $R'_j \in G$ **do**
  if $O_S^{ij} > \gamma_o$ **and** $O_G^{ij} > \gamma_o$ **then**
    $R_i, R'_j \leftarrow$ Object candidates
  else if $O_S^{ij} > \gamma_p$ **and** $O_G^{ij} > \gamma_o$ **then**
    $R_i \leftarrow$ Fragmentation candidate
    $R'_j \leftarrow$ Part candidate
  else if $O_S^{ij} > \gamma_o$ **and** $O_G^{ij} > \gamma_p$ **then**
    $R_i \leftarrow$ Part candidate
    $R'_j \leftarrow$ Fragmentation candidate
  else
    $R_i, R'_j \leftarrow$ Noise
  end if
end for

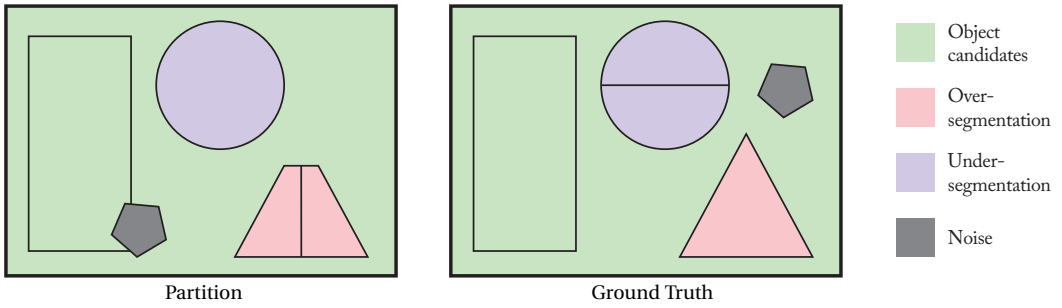


Figure 4.3: Toy example: Classification of the regions into object and part candidates. The rectangles are classified as object candidates, despite not fully overlapping. The partition circle is a fragmentation candidate with a fragmentation of 1 (parts cover it totally), and the ground-truth half-circles are parts candidates. The opposite holds for the triangles, but in this case the fragmentation is 0.9. Both pentagons are classified as noise.

We then define the **precision-recall for objects and parts** as follows:

$$P_{op} = \frac{oc + fr + \beta\, pc}{|S|} \qquad R_{op} = \frac{oc' + fr' + \beta\, pc'}{|G|} \tag{4.7}$$

Intuitively, in a completely oversegmented result, the recall would be high but the precision very low. Conversely, a completely undersegmented result (one single region) would entail a high precision but very low recall. As a summary measure, we propose to use the F measure (F_{op}) between P_{op} and R_{op}.

Figure 4.4 shows two realistic examples were we classify the regions from a human-made partition when compared to another human-made partition as ground-truth.

4.4 Qualitative Meta-Measures

This section and the following are about how to compare the goodness of the segmentation evaluation measures. The objective is therefore not to tell which segmentation algorithm to

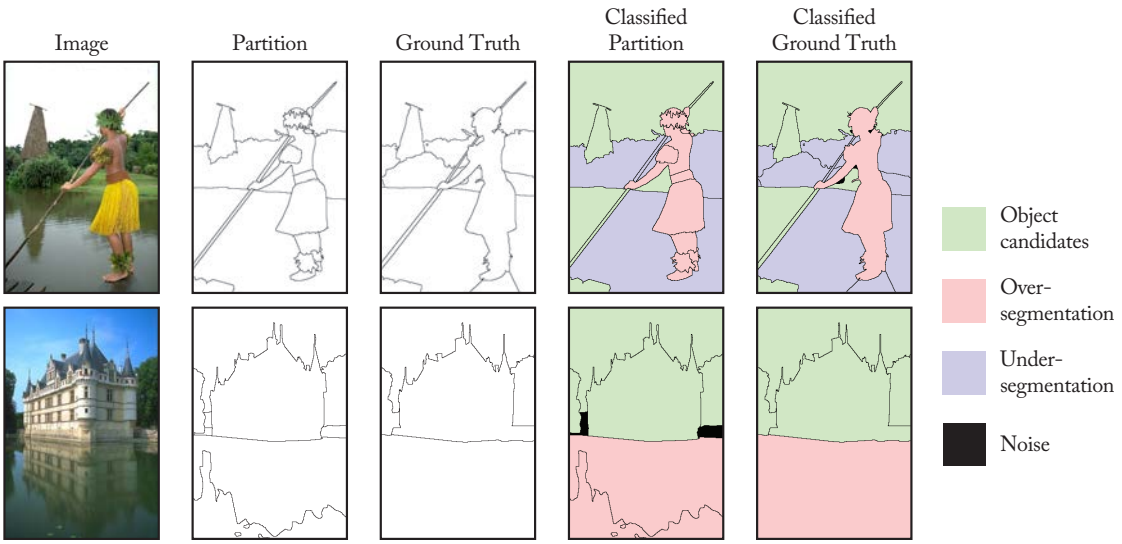


Figure 4.4: Realistic example: Classification of the regions into object and part candidates, the latter separated into over- and under-segmentation.

use, but which evaluation measures better summarize the quality of these algorithms. To distinguish these two analyses, we will refer to the metrics used to compare segmentation measures as *meta-measures*.

This section presents two qualitative meta-measures to compare the behavior of the evaluation metrics, while the next section will present some quantitative meta-measures. As introduced before, a meta-measure consists in making an assumption about the results and measuring how well each metric reflects this assumption. Particularly, the qualitative meta-measures we propose consist in two partition pairs whose discrepancies can be described intuitively, so the expected behavior of the assessment measure is clear. We will then analyze how coherent the measures are with the expected behavior.

These pairs are, in turn, representative enough of the discrepancies found in real-world comparisons. The two qualitative meta-measures we propose are: over- and under-segmentation (Section 4.4.1), and boundary misplacement (Section 4.4.2).

4.4.1 Over- and Undersegmentation

This section is devoted to present a qualitative meta-measures to analyze how the different metrics cope with over- and undersegmentation, or, in other words, how these effects are reflected on the measures. The evaluation consists in assessing whether the measures match the qualitatively correct result.

Let us assume a ground-truth partition S' consisting of a single rectangular region of size 1, and a partition S to be assessed consisting of two regions of size α and $1-\alpha$, with $\alpha \in (0,1)$ (Figure 4.5). Note that this is an example of oversegmentation. To analyze undersegmentation we can simply compute the similarity changing the order in which we consider the partitions.

The expected behavior from an assessment measure would be the following. It is clear that $\alpha = 0$ and $\alpha = 1$ are perfect results, so the measure should be 1 for these values. Intuitively,

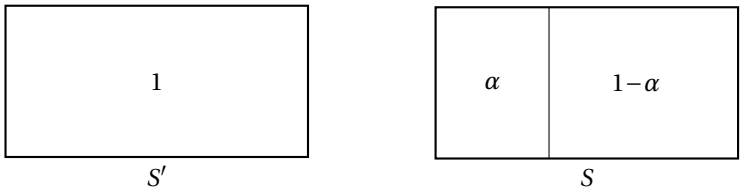


Figure 4.5: Partitions to assess the effect of the area distribution in oversegmented results

values of α close to this perfect result should be almost correct, since it can be interpreted as noise that can be easily filtered in following steps.

For values of α not close to 0 or 1, the result should be penalized because it means that there is a difference in the scale interpretation, that is, the result has been oversegmented. The area distribution in an oversegmented result should not, however, affect the measure, since it depends on the image content and thus should not reflect a better or worse result. In other words, a result with $\alpha = 0.5$ is not necessarily worse than with $\alpha = 0.7$, since the oversegmented region can actually be formed by two equal parts.

4.4.2 Boundary Misplacement

Let S' be a ground-truth partition consisting of two equal regions of size 0.5, and a partition S to be assessed consisting of two regions of size α and $1-\alpha$, with $\alpha \in (0.3, 0.5)$. The values of α are limited to a range around 0.5 since, for farther values, the effect would not be described as boundary misplacement. Figure 4.6 shows the two partitions S and S' used in this section.

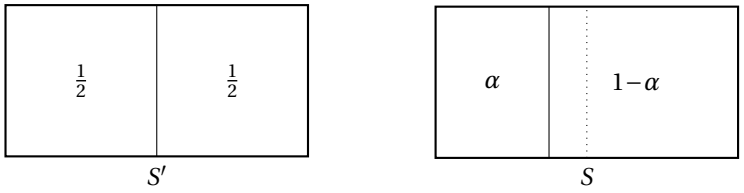


Figure 4.6: Partitions to assess the effect of the boundary misplacement

As previously, let us describe the expected behavior. Intuitively, small displacements of the actual boundary should not be penalized, since they are possibly caused by blurred boundaries or ground-truth inaccuracies. If the marked boundary is placed far from the actual one, there should be a penalization because of a missed boundary, but it should not depend on the distance to the true contour, since this distance will depend on the image content.

In [UPH07], this example is used to claim the goodness of the probabilistic Rand index measure in a *softer* manner than in our analysis: they expect the measure to be *almost flat* in an area of uncertainty while our analysis forces the measure to be constant in this area.

4.5 Quantitative Meta-Measures

A meta-measure analysis must rely on accepted hypotheses about the segmentation results and assess how coherent the measures are with such hypotheses. The qualitative meta-measures presented in the previous section are a useful tool to interpret the results, but a qualitative evaluation, on a large enough database so it can be statistically significant, is also desirable.

As an example of quantitative meta-measure, an accepted hypothesis can be the human judgment of quality of some particular examples. The meta-measure is then defined as a quantization of how coherent the evaluation measures are with this judgment, as done in works such as [UPH07, CCR05].

To provide statistically significant results, however, one must go beyond a handful of examples and provide a quantitative analysis on an annotated database. The remainder of this section explains one meta-measure already published in the literature (Sec. 4.5.1) and presents two new meta-measures (Sec. 4.5.2 and 4.5.3).

The two new meta-measures differ significantly from the previously presented in the sense that, instead of being based only on human-made partitions, we base our analysis on a large set of partitions made by state-of-the-art segmentation techniques.

4.5.1 Swapped-Image Human Discrimination

Given an image, there is no unique valid segmentation, since it depends on the perception of the scene, the level of details, etc. In order to cope with this variability, the Berkeley Segmentation Dataset (BSDS300 [MFTM01] and BSDS500 [AMFM11]) consists of a set of images each of them manually segmented by more than one individual.

The hypothesis behind the first meta-measure is that an evaluation metric should be able to tell apart the ground-truth partitions coming from two different images. In other words, given a pair of ground-truth partitions from BSDS500, a measure should be able to tell whether they come from the same image (thus differences are an acceptable refinement) or different images (unacceptable discrepancies).

As first proposed by [Mar03] to evaluate the coherence of BSDS300, given an evaluation measure m, we compute the Probability Density Function (PDF) of the values of m for all the pairs of partitions in BSDS500, grouped in two classes: those coming from different images and those from the same one. Figure 4.7 shows the PDFs for these two types of pairs of partitions using the F_b measure.

A simple classifier was then defined setting a threshold on the measure to discriminate the two types of pairs. The **Swapped-Image Human Discrimination** (SIHD) meta-measure is defined as the percentage of correct classifications of that classifier, that is, the sum of the area under the curve above and below the threshold for the same-image and different-image pairs, respectively. (In the original work, the authors reported the Bayes Risk.)

As qualitative examples, Figure 4.7 depicts four pairs of partitions as representatives of the type of mistakes and correct classifications using F_b.

4.5.2 SoA-Baseline Discrimination

One of the reasons why SIHD can be criticized is the fact that it is based only on human-made partitions, that is, it does not show how measures handle the *real-world* discrepancies found between SoA segmentation methods. This section and the following are devoted to present two meta-mesures based on SoA segmentation results.

The hypothesis on which we base the meta-measure presented in this section is that evaluation measures should be able to distinguish between (i) partitions obtained by any SoA

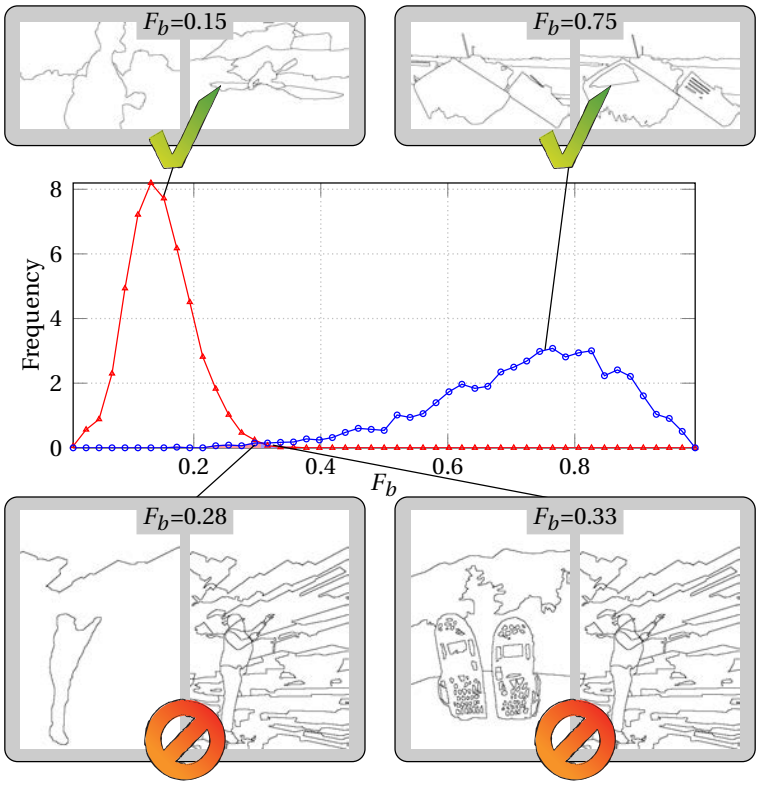


Figure 4.7: Distribution of F_b values for the same-image pairs of partitions (—○—) and different-image pairs (—△—). In gray rectangles, four representative pairs of partitions: a pair of correctly classified as different image (up-left) and as same image (up-right); and a pair incorrectly classified as different image (down-left) and as same image (down-right).

segmentation method on a given image and (ii) partitions obtained regardless of the image, that is, partitions that are created without taking into account the content of the image. These partitions are interpreted as a baseline, that is, the results that could be obtained *by chance*.

As in [AMFM11], we use a quadtree as baseline. In particular, we build the hierarchical partitions starting from the whole image and iteratively dividing the regions into four equal rectangles. Figure 4.1.b shows an example of partition obtained by a SoA method and by a quadtree.

For each of the techniques considered as SoA segmentation methods, we compute the number of images in the dataset in which an evaluation measure correctly judges that the baseline result is worse than the SoA generated partition. We refer to the resulting meta-measure as **SoA-Baseline Discrimination** (SABD), and it is defined as the global percentage of correct judgments for a given measure.

Figure 4.8 shows an example of a correct and an incorrect judgment by a segmentation evaluation measure of the quality of a baseline result with respect to a SoA result.

4.5.3 Swapped-Image SoA Discrimination

Segmentation evaluation measures are often used to adjust the parameters of a segmentation technique. They are therefore used to compare different partitions created by the same

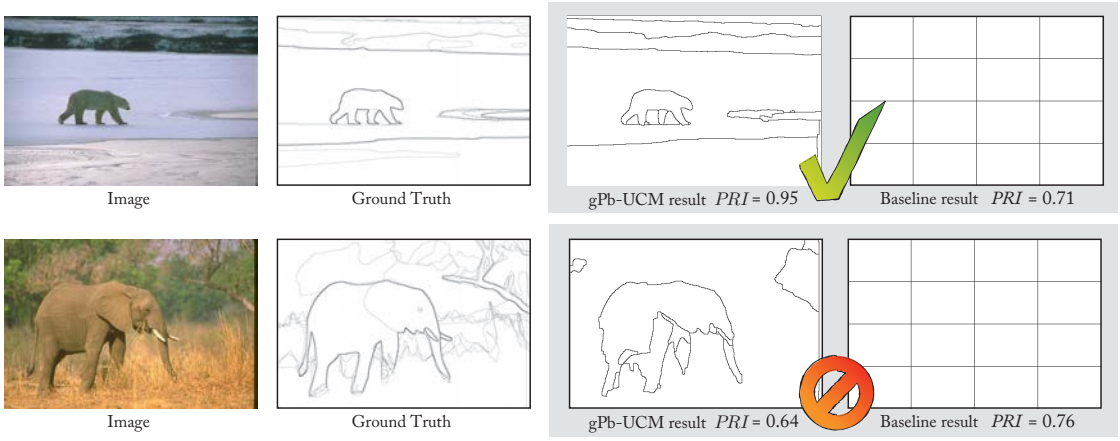


Figure 4.8: SABD illustrative example. Correct and incorrect judgments by a segmentation evaluation measure.

algorithm. To incorporate this type of comparisons to the meta-measures, we compare (i) the results created by a SoA segmentation technique with (ii) the results created by that same algorithm but on a different image.

In other words, we compare the ground-truth of a certain image with two results obtained using the same algorithm and parameterization: (i) one segmentation of that same image and (ii) one of a different image. The hypothesis in this case is that the evaluation measures should judge that the same-image result is better than the different-image one. In the example of Figure 4.1.c, the measure should judge that the first partition is better than the second one compared both with the ground-truth of the first one. In this meta-measure, evaluation measures have to tackle the potential bias of the SoA methods towards their specific type of results.

For each SoA segmentation technique, we compute the number of images in the dataset in which an evaluation measure correctly judges that the same-image SoA result is better than the different-image one. We define the meta-measure **Swapped-Image SoA Discrimination** as the percentage of results in the database, for all the SoA methods, that the measures correctly discriminates.

Figure 4.9 shows an example of a correct and an incorrect judgment by a segmentation evaluation measure of the quality of a SoA result judged on the same versus a different image.

4.6 Experimental Results

This section presents the experimental validation of the measures and meta-measures proposed in this thesis. Section 4.6.1 shows the comparison of all evaluation measures in terms of the proposed qualitative meta-measures, while Section 4.6.2 shows the quantitative meta-measures results. As a result of the analysis, we propose the two best performing measures F_b and F_{op} to be used in tandem. Section 4.6.3 analyzes the state-of-the-art techniques in terms of the precision-recall curves of these two measures and presents some experiments to further analyze their differences.

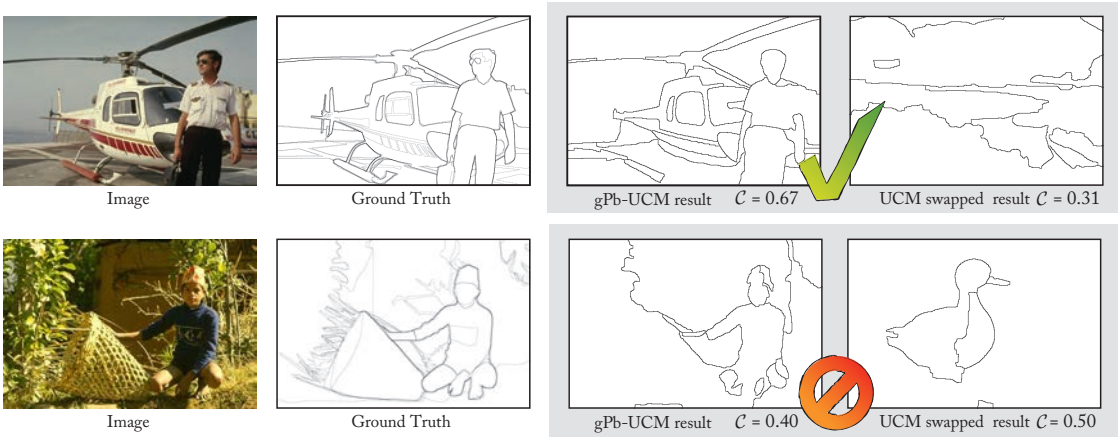


Figure 4.9: SISD illustrative example. Correct and incorrect judgments by a segmentation evaluation measure.

4.6.1 Qualitative Meta-Measures

Over- and Under-Segmentation: Figure 4.10 plots the value of the studied measures, in the case of over- and undersegmentation, for the whole range of valid $\alpha \in (0,1)$ to compare their behavior with the expected result described in Section 4.4.1.

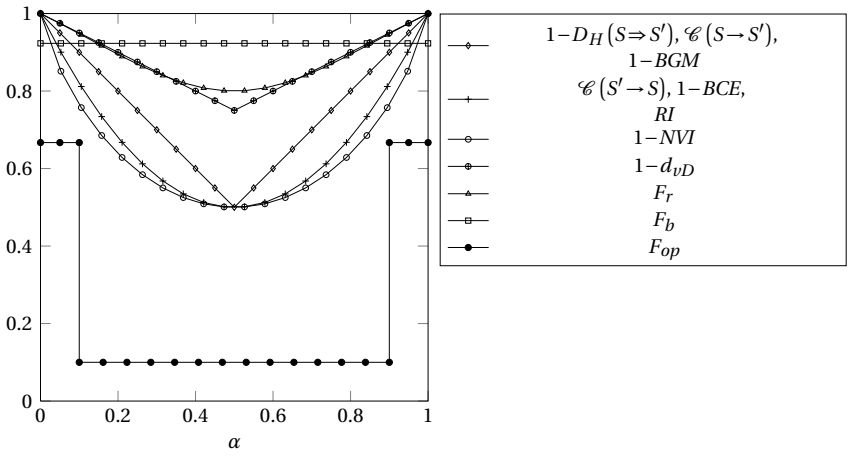


Figure 4.10: Two-region oversegmentation meta-measure results. The horizontal axis represents α, one of the region sizes, and the vertical axis represents the values of the measures.

First of all, there is a group of measures (six different plots) that form a convex function, strictly monotonically decreasing from $\alpha = 0$ to 0.5 and the other way around from 0.5 to 1. In other words, the closer the area of any of the oversegmented regions to the actual whole region, the better the result. As presented before, this behavior is intuitively desirable for values of α very close to 0 or 1, since they reflect that one of the regions can be classified as a very small unnoticeable noise region.

For values around $\alpha = 0.5$, however, there is no clear interpretation of why this behavior should persist, i.e., no easy justification can be found to the result for $\alpha = 0.5$ being worse than the result for $\alpha = 0.7$, since this depends on the semantical content of the image.

The boundary-based measure F_b (Figure 4.10 —▫—) penalizes the oversegmented results independently of the value of α, since it considers the boundary as a false positive. This behavior matches the perception that for values of α close to 0.5 the measure should be constant, but in contrast, the penalization is the same even for α very close to 0 or 1.

The behavior of the last measure assessed F_{op} (Figure 4.10 —•—) is closer to the expected one, since it is constant in the middle ranges of α, and penalizes less for values closer to 0 and 1. Note that the fact that the values of this measure are much lower than the rest of measures is irrelevant, since we use the measure to rank results so we could scale the values with no further influence. F_{op} is therefore, the one that best matches the a-priori intuition presented formerly.

Boundary misplacement: Figure 4.11 plots the values of the evaluation measures to analyze how they behave when a boundary is misplaced in the range of $\alpha \in (0.3, 0.5)$, and how well they match the expected behavior presented in Section 4.4.2

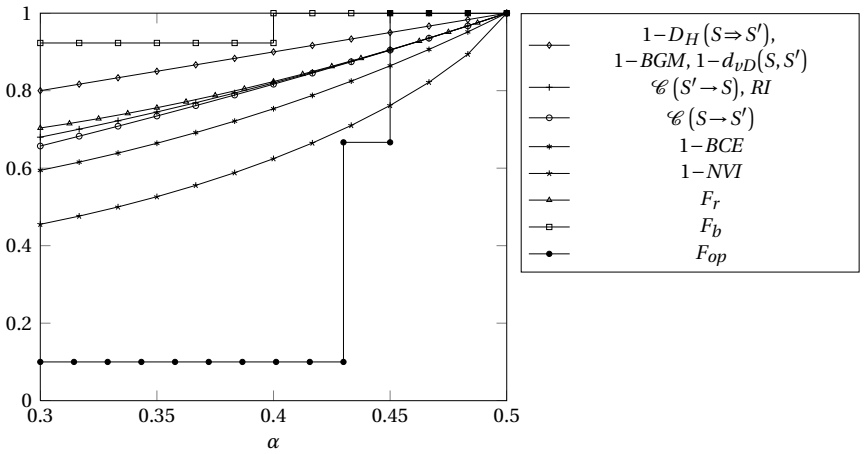


Figure 4.11: Boundary misplacement effect. The horizontal axis represents α and the vertical axis the values of the measures.

The most common behavior observed in this case is a strictly monotonic decrease of the similarity with respect to the distance from the detected boundary to the true contour. In other words, all measures, except F_b and F_{op}, consider the result the worse the farther the boundary from the actual contour, which does not match the intuition discussed previously.

In this test we expect the measure to be *flat* both for close boundaries (perfect result) and for far boundaries (misplaced). The behaviors of F_b (Figure 4.11 —▫—) and F_{op} (Figure 4.11 —•—) both match this intuition and the minor difference between them lies in a larger transition between both cases. The former has a single step between a perfect result and a misplaced one, which we believe has no relevance.

Global comparison: The two first columns of Table 4.2 show the results of the qualitative meta-measure evaluation on each of the assessed metrics.

4.6.2 Quantitative Meta-Measures

The state of the art of segmentation to compute the meta-measures is represented in this section by gPb-UCM [AMFM11], EGB [FH04], Mean Shift [CM02], Normalized Cuts [SM00],

Measure	Qual. Meta-Meas.		Quant. Meta-Meas.			
	Overseg.	Bound.Mis.	SIHD	SABD	SISD	Global
F_b	✓	✓	99.5	95.6	100.0	98.4
F_{op}	✓	✓	98.4	94.2	97.5	96.7
NVI	✗	✗	96.9	87.5	97.7	94.0
$\mathscr{C}(S \rightarrow \{G_i\})$	✗	✗	93.1	86.0	95.3	91.5
d_{vD}	✗	✗	95.1	86.9	90.1	90.7
$D_H(S \Rightarrow \{G_i\})$	✗	✗	78.5	91.3	98.8	89.5
BCE	✗	✗	93.3	78.9	95.4	89.2
BGM	✗	✗	90.7	81.6	92.0	88.1
PRI	✗	✗	77.7	88.8	93.7	86.7
$\mathscr{C}(\{G_i\} \rightarrow S)$	✗	✗	91.3	77.4	90.1	86.3
F_r	✗	✗	77.0	84.2	97.1	86.1
$D_H(\{G_i\} \Rightarrow S)$	✗	✗	73.0	92.1	76.5	80.5

Table 4.2: Measure comparison in terms of quantitative and qualitative meta-measures

NWMC [VMS08], and IID-KL [CM10]. As baseline, we use a Quadtree. All methods are assessed at the Optimal Dataset Scale (ODS) [AMFM11] with respect to each evaluation measure, that is, using the parameters that entail the best value of the measure in mean on the whole training set of BSDS500.

The parameter values of the newly proposed measure are: $\gamma_o = 0.95$, $\gamma_p = 0.25$, and $\beta = 0.1$. They have been trained on the training set of BSDS500, by optimizing the global meta-measure described below. Note that this optimization would not have been feasible without quantitative meta-measures.

Table 4.2 shows the three meta-measure results for the test set of BSDS500, as well as a global summary meta-measure. Given that each meta-measure represents a percentage of correct results, we define the global meta-measure as the global percentage of correct results.

In global terms, F_b and F_{op} are the two top-ranked summary measures. On top of that, they both provide much richer information in form of precision-recall curves. Adding that the two measures are the only ones with a good result in terms of the proposed qualitative meta-measures, we believe the tandem F_b-F_{op} should be the evaluation measures of choice.

Regarding the computational cost of the measures, the mean time for image to compute the distances to the multiple-partition ground truth of BSDS500 is 3.79 ± 2.06 s for F_b and at least one order of magnitude lower for the rest of measures. In particular, F_{op} takes 0.078 ± 0.020 s. In scenarios where the time constraints are tight, therefore, F_{op} would be the recommended measure.

4.6.3 Precision-Recall Frameworks:

This section tests the proposed tandem of measures to compare a large set of state-of-the-art segmentation techniques. Apart from the six methods used in the meta-measure computation, we also evaluate ISCRA [RS13a], from which only the pre-computed partitions on the test set are publicly available.

Figure 4.12 shows the boundary and objects-and-parts precision-recall curves for the seven SoA segmentation methods studied, the quadtree baseline, and the human performance. Prior to the assessment of segmentation techniques, let us focus on the comparison of the two evaluation frameworks.

It is noticeable that the human baseline performance (human assessed on a different image) for F_b is 0.21, which could be interpreted as F_b being too lax. In this same direction, the baseline boundary precision for F_b is between 0.2 and 0.3, that is, any result, no matter how wrong it is, will be judged as providing at least a 0.2 precision.

While in the case of F_{op} the human baseline is correctly downgraded to 0.05 (as well as the swapped-image results), then the surprising fact is that human performance is as low as 0.56 (0.81 in F_b), which could entail that F_{op} is too strict.

Although the dynamic range is a little higher in F_b (0.60 versus 0.51), the gap between the best method and humans is much higher in F_{op} (0.08 versus 0.21). In other words, F_{op} gives more resolution at the places where improvements over the SoA would be placed.

Regarding the comparison among segmentation techniques, both frameworks confirm that there are two techniques (gPb-UCM and ISCRA) with outstanding results with respect to the rest of SoA segmentation techniques. gPb-UCM is consistently better in terms of boundary localization, while ISCRA outperforms gPb-UCM from the point of view of regions and parts.

The advantages of *going beyond* the summary measures are also clear on these plots. For instance, the summary F_b measure of quadtree (0.41) judges this technique close to NWMC (0.55), but in the precision-recall curves it is clear that quadtree is much worse. Similarly,

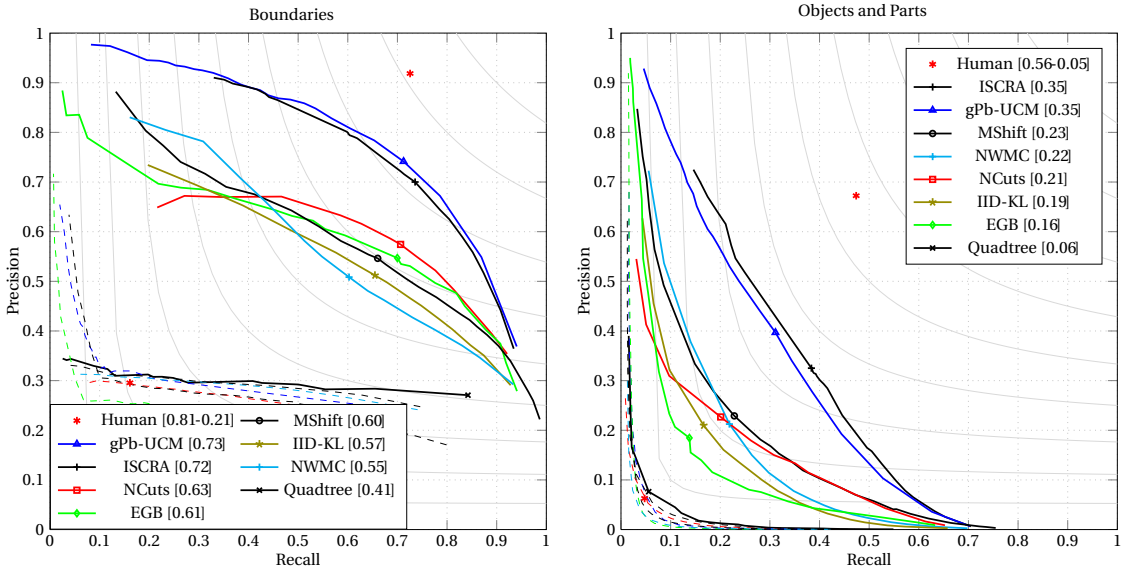


Figure 4.12: Precision-Recall curves for boundaries (left) and for objects and parts (right). The solid curves represent the seven SoA segmentation methods and the quadtree (see legends). In dashed lines with the same color, the SoA techniques assessed on a swapped image. The marker on each curve is placed on the Optimal Dataset Scale (ODS). The isolated red asterisks refer to the human performance assessed on the same image and on a swapped image. In the legend, the F measure of the marked point on each curve is presented in brackets.

judging by F_b, NWMC would be discarded with respect to NCuts for instance, but if we are interested in low recall rates it could be of interest.

As common points between the two measures, NCuts is judged as being much better at high recall rates than at low ones and conversely, NWMC is much better at high precision rates. The measures are coherent also in the fact that human results have a better precision than recall. As one of the main discrepant points, however, EGB is judged as the fourth best technique by F_b while being the worse for F_{op}. To further analyze this behavior, Figure 4.13 shows an image and the associated ground truth. The EGB result (a) consists of thin long regions that surround the object but do not close to create the regions of interest. The assessment value of this result is $F_b = 0.62$ and $F_{op} = 0.05$. From a region-based point of view, this type of results is correctly penalized by F_{op} and not by F_b, since as a contour detector the result is correct.

We further compare the measures qualitatively by creating two academic examples (Figure 4.13 (b) and (c)) that show the complementary behavior, that is, examples where the F_{op} behavior is not intuitive. First, partition (b) is composed of two boxes completely included on the objects of interest. F_{op} interprets them as part candidates, since they are completely included in the objects and cover a significant part of them, so it does not penalize the partition significantly. On the other hand, F_b penalizes the result because the boxes contours do not overlap with the true boundaries.

If we slightly increase the size of the boxes, however, making the contours overlap but having a small part of the boxes outside of the object, the situation is changed: the boxes are not considered parts anymore ($F_{op} = 0.04$) and the boundary measure does not judge the results as being very bad ($F_{op} = 0.28$). As a side note, this last qualitative comparison between F_{op} and F_b made us notice that the use of only a Quadtree as baseline is in clear favor of F_b in terms of SABD (SoA-Baseline Discrimination), in the sense that F_{op} will have a harder time telling them apart. We believe that adding the random tree to SABD would have been fairer to F_{op}.

To sum up, both measures are complementary, thus we propose them in tandem as the tool of choice for image segmentation evaluation.

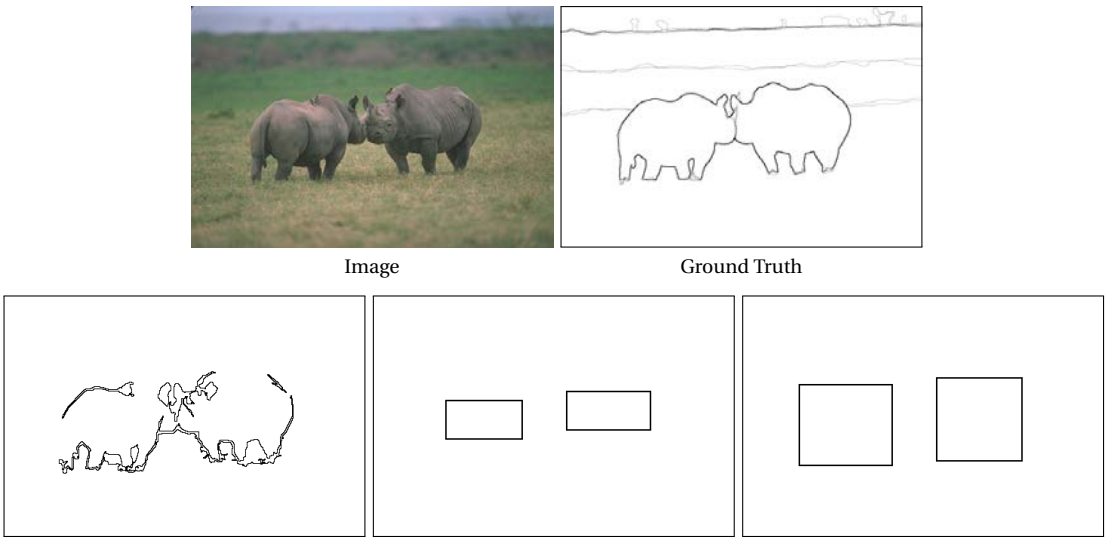


Figure 4.13: Qualitative comparison of F_b and F_{op}: extreme examples where the behavior of one of the measures is not the expected

5 Hierarchical Segmentation

5.1 Introduction

As we did in the first part of the thesis in the case of an object-based analysis, this section generalizes the evaluation of image segmentation algorithms to hierarchies, in this case from the perspective of full partitions. Recalling that a hierarchical region-based image representation is a structured set of image partitions from the most detailed ones (more regions) to the coarsest ones (less regions), the first approach to assessing the representation directly could be to compare *all* the partitions represented in the hierarchy and to assess their quality with respect to their number of regions, for instance.

As we prove in Section 5.1.1, however, the number of image partitions represented in a hierarchy can grow exponentially with the number of leaves of the tree, thus making it unfeasible to assess all partitions exhaustively by brute force.

Similarly to the case of an object-based measure, we could try to find the *upper-bound performance* of the representation, that is, to assess the partition that *best* matches the ground truth sweeping all possible scales. The usual approach in the literature [AMFM11, CM10, VMS08] in this direction consists in assessing the N partitions in the *merging sequence*, expecting them to be representatives of the achievable quality of the hierarchy. As we will show in the experiments, however, there is no guarantee that this set of partitions is even close to the best representation in the hierarchy for each scale.

Let us illustrate some of these concepts by the example hierarchy in Figure 5.1 (already introduced in Chapter 3). The four partitions form the merging-sequence partition set. The usual evaluation of the hierarchy would therefore assess these four partitions and their quality evolution is used as representative of the quality of the whole hierarchy.

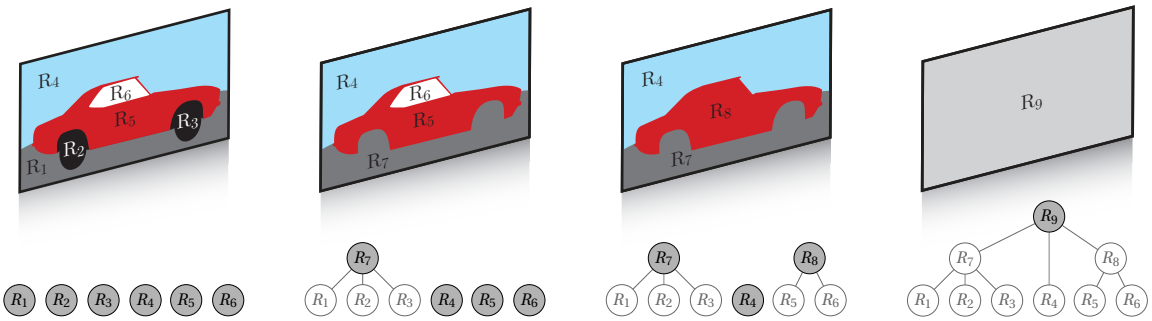


Figure 5.1: Hierarchy creation process: From left to right, the set of most-similar neighboring regions are merged at each step. Below, the hierarchy representation depicted by a tree, where the region formed from the merging of some segments is represented as the parent of the respective nodes.

The set of partitions from the merging sequence, however, is a small part of all partitions represented in a hierarchy. As we will show in Section 5.1.1, a hierarchy with N leaves has N partitions in the merging sequence while having up to $2^{\frac{N}{2}}$ partitions in the full hierarchy. Figure 5.2 shows an example partition represented in the hierarchy but not in the merging sequence of the hierarchy introduced above.

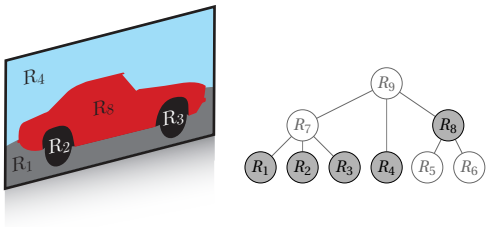


Figure 5.2: Partition represented in the hierarchy but not in the merging sequence

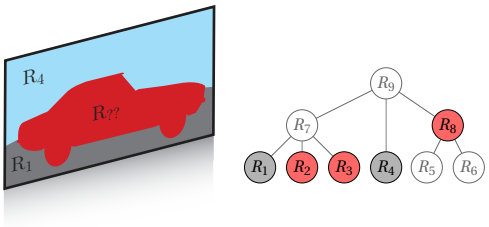


Figure 5.3: Partition formed by merging leave regions but not in the hierarchy

In sight of the cardinality of the set of partitions in a hierarchy, one may wonder why hierarchies are useful, or in other words, why not working directly on the leaves partition. The answer is that a hierarchy does indeed drastically reduce the space of possible partitions: the number of partitions that can be formed by merging regions from the leaves partition is the Bell number [BT10] $B_N \sim \left(\frac{0.792n}{\ln(n+1)}\right)^n$, which is considerably larger than 2^N. Figure 5.3 shows a partition not represented in the hierarchy introduced above. In it, the red region is not part of the hierarchy, that is, there is no node that represents this region; although it can be formed by merging leave regions.

Figure 5.4 represents the three sets of partitions involved in a hierarchy: the merging-sequence partition set, the set of partitions in the hierarchy, and the full set of partitions formed by merging leaves of the hierarchy.

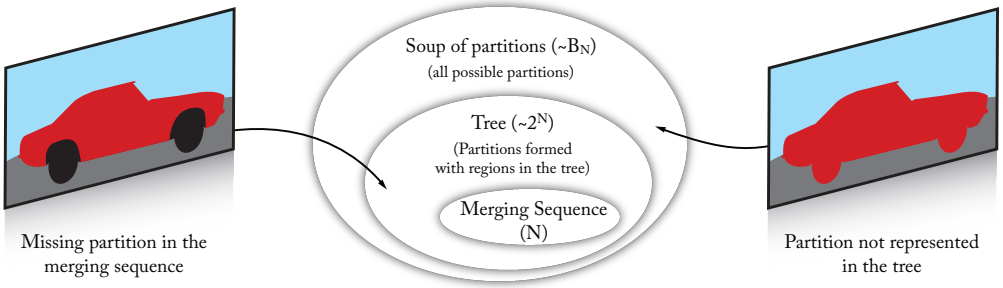


Figure 5.4: Representation of the sets of partitions in a hierarchy, from the ones in the merging sequence, to the whole soup of partitions that can be formed by merging leave regions

A hierarchy is therefore a tool that reduces and structures the space of partitions of an image. This way, the problem of finding a partition is reduced to the problem of finding a cut [PS13], or a pruning [SG00, HWG07], or a slice [XWC13] of the tree.

Since the partitions selected by all these works that make use of hierarchies are potentially not in the merging sequence, we believe that representing the quality of a hierarchy by the quality of the very reduced representative set of partitions in the merging sequence is not a correct approach. Our proposal is instead to extend the evaluation to all partitions represented in a

hierarchy, by computing the *upper-bound performance* in this set. In other words, to look for the best partition represented in a tree with respect to a given measure.

From a practical point of view, the upper-bound performance represents the best any method based on hierarchies can do, that is, it is a normalization factor that can tell researchers whether they are making the most of the hierarchies, so they must improve the hierarchy to advance; or on the contrary there are already good partitions in the hierarchy thus there is still room for improvement in the search algorithm.

The following sections are focused on finding the best partition in a hierarchy or in the whole soup of partitions with respect to different measures. Given the particularities of each of them, all sections in this chapter include their own experimental validation.

First, Section 5.2 finds the upper bound with respect to the directional Hamming distance (see Section 4.2.1). As we will show, some of the properties of this measures makes it possible to find the upper bound by means of dynamic programming, taking advantage of the structure of the hierarchy.

Since the experiments on the previous chapter have shown that this is not among the best performing measures, Section 5.3 describes how to compute the upper-bound performance in terms of F_b. Given the properties of this measure, we must model the problem mathematically, again as an LFCO.

Recalling the sets of partitions in Figure 5.4, Section 5.4 expands the search of the best partition in terms of F_b beyond the partitions represented in a tree, and find the upper-bound performance on the whole soup of partitions that can be formed from the leaves of the tree. This would allow us to evaluate the loss in achievable quality due to working on superpixels instead of on the full set of pixels.

Finally, Section 5.5 presents the model to find the best partition represented in a tree with respect to the newly-proposed F measure for objects and parts F_{op}.

5.1.1 Cardinality of the Set of Partitions in a Hierarchy

This section proves that the number of partitions represented in a hierarchy can grow exponentially. To simplify the analysis we focus on binary hierarchies, that is, hierarchies represented by trees with at most two children per node, although the hierarchies handled are not necessarily binary.

Let N be the number of leaves of a binary tree or, in image segmentation terms, the number of regions of the original partition on which the tree is built. Following the definitions in [CLRS01], the *depth* of a node is the number of nodes from it to the root, both inclusive. The *height* of the tree is the maximum depth of its leaves, and it is said to be *height-balanced* (in short, balanced) when the depth of the leaves differ at most by 1.

Lemma 5.1. *The number of nodes of a binary tree with N leaves is $M = 2N - 1$.*

Proof. Let us imagine we mark the nodes of the tree by the following procedure: starting from the N leaves marked, we mark any node with two marked sons and un-mark its sons. We repeat the process until only the root is marked. Each step of this process marks a new region and reduces the number of nodes still to be marked of the tree by one. Since originally there

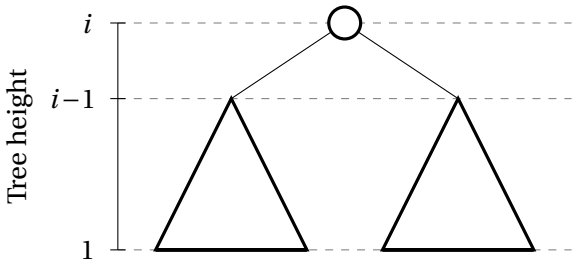


Figure 5.5: Recursive decomposition of a BPT. The vertical axis refers to the height of the corresponding tree. When two height-balanced subtrees of height $i-1$ are merged, the resulting tree has height i

are N marked nodes, and at the end of the process just one, the number of steps from the leaves to the whole image is exactly $N-1$. Consequently, $M = N + (N-1) = 2N-1$. □

Lemma 5.2. *The maximum number of nodes at depth exactly d in a binary tree is 2^{d-1}.*

Proof. By induction, for $d = 1$, there is only $1 = 2^0$ root. If at level d there are at most 2^{d-1} nodes, at level $j = d+1$, since the tree is binary, there will be at most $2^{d-1} \cdot 2 = 2^d = 2^{j-1}$. □

Lemma 5.3. *If all the leaves of a binary tree have exactly the same depth d, then $d = \log_2(N)+1$, or, equivalently $N = 2^{d-1}$.*

Proof. Given that all the leaves have the same depth, each level will be complete, so Lemma 5.2 applies. Being M the number of nodes in the tree, counting them from the root:

$$M = 2^0 + 2^1 + 2^2 + \cdots + 2^{d-1} = 2^d - 1$$

Applying Lemma 5.1:

$$2N - 1 = 2^d - 1 \quad \Rightarrow \quad d = \log_2(2N) = \log_2(N) + 1$$ □

Theorem 5.4. *Let $\mathscr{P}$ be the set of all possible partitions that can be extracted from a binary hierarchy with all leaves at depth d, and let $|\mathscr{P}|$ be its cardinality. Being N the number of regions in the original partition (leaves), then: $|\mathscr{P}| \geq 2^{N/2}$.*

Proof. We will proof the result by induction on the height of the tree h. For $h = 2$, $N = 2$, $|\mathscr{P}| = 2 \geq 2^{2/2} = 2$.

Let us assume that the result is true for $h = i-1$, i.e., for $N = 2^{i-2}$ (recall Lemma 5.3). A tree of height $h = i$ can be seen as the merging of the roots of two trees of height $h = i-1$, as depicted in Figure 5.5.

Then we will have that each partition in the left tree can be combined with any partition in the right tree forming a new partition. Adding the new root, it follows that:

$$|\mathscr{P}_i| = |\mathscr{P}_{i-1}|^2 + 1 \geq |\mathscr{P}_{i-1}|^2$$

Applying the induction hypothesis:

$$|\mathscr{P}_i| \geq |\mathscr{P}_{i-1}|^2 \geq \left(2^{\frac{2^{i-2}}{2}}\right)^2 = 2^{(2^{i-2})} = 2^{\frac{2^{i-1}}{2}}$$

So it follows that: $|\mathscr{P}| \geq 2^{N/2}$. □

In other words, the number of partitions that can be extracted from a hierarchy with all leaves at the same depth **grows exponentially** with the number of regions of the original partition. For a generic hierarchy, therefore, the brute-force assessment would be unfeasible at least in the worst-case scenario. To get a flavor of the dimensionality, for a balanced tree constructed on only 64 regions, the number of different partitions that can be extracted is higher than four thousand millions.

5.2 Upper-Bound of a Local Measure: Directional Hamming

This section presents the algorithm to compute the upper-bound performance in terms of the Directional Hamming D_H distance (see Section 4.2.1). In other words, we want to find the partition represented in the hierarchy that minimizes this measure with respect to a ground truth.

Intuitively, recalling Figure 5.5, the main idea behind our approach is to compute the best representation of k regions at height i as combinations of j and $k-j$ regions on each subtree of height $i-1$. Starting at height 1, only a representation with $k=1$ regions is possible, and then at each increase in height, only the best representations of the subtrees have to be explored, avoiding all the suboptimal representations.

This is a classical instance of problem to solve via dynamic programming, for which we need it to have a so-called *optimal substructure*, that is, we need that the global optimum can be formed via the local optimum of the subtrees. As we will show below, a sufficient condition is that the evaluation measure is *local*. Let us describe the algorithm formally.

Definition 1. *Given a partition $P = \{R_j\}$, and a ground truth partition GT, an assessment measure $m(P, GT)$ is said to be **local** when there exists another measure m^l such that:*

$$m(P, GT) = \sum_j m^l(R_j, GT)$$

Let $t(R_j)$ be the height of the subtree rooted at node R_j. Let us denote the regions of a tree as R_1,..., R_N, ..., R_{2N-1}, numbered in increasing order of t, i.e.:

$$1 = t(R_1) = t(R_2) = \cdots = t(R_N) \leq \cdots \leq t(R_{2N-1})$$

Note that, under this condition, R_1, ..., R_N are the leaves and R_{2N-1} is the root. Note also that, in the case of hierarchies built by sequential merging, the merging order fulfills this requirement.

Definition 2. *We define s_k^i as the value of the addition of m^l on the best representation by means of exactly k regions from the subtree below R_i.*

Following this definition, our objective is to compute s_k^{2N-1}, that is, the upper-bound performance of the whole tree for $k = 1 \ldots N$ regions. Formally, Algorithm 3 describes the procedure to find the upper-bound performance of a tree with respect to a local measure m, where il and ir are the indices of the *left* and *right* sons of the node R_i, respectively; and $t'(R_i)$ is the number of leaves in the subtree below R_i, so we have that $t'(R_j) = t'(R_{il}) + t'(R_{ir})$.

Algorithm 3 Upper-bound tree assessment by dynamic programming

for $i = 1,\ldots,2N-1$ **do**
  $s_1^i \longleftarrow m^l(R_i, GT)$
  if $i > N$ **then**
    for $k = 2,\ldots,t'(R_{il}) + t'(R_{ir})$ **do**
      $s_k^i \longleftarrow +\infty$
    end for
    for $p = 1,\ldots,t'(R_{il})-1$ **do**
      for $q = 1,\ldots,t'(R_{ir})-1$ **do**
        $s_{p+q}^i \longleftarrow \min\left\{s_{p+q}^i, s_p^{il} + s_q^{ir}\right\}$
      end for
    end for
  end if
end for

Note that we are assuming that the assessment measure is an *error* measure or a *distance*, in the sense that the lower the better, since we compute the minimum of the measure in the leaves of each node. We should change to the maximum if the measure is a *similarity*, i.e., the higher the measure, the better.

Lemma 5.5. *The number of regions at height h in a balanced tree is exactly* $N2^{1-h}$.

Proof. At height 1 there are N leaves and, at each height increase, the number of regions is divided by 2. □

Theorem 5.6. *Algorithm 3 complexity is* $O\big(N(\log_2 N)^2\big)$.

Proof. The number of relevant s_k^i updates in Algorithm 3 are:

$$T(N) = 2N + 1 + \sum_{i=N+1}^{2N-1} t'(R_{il}) \cdot t'(R_{ir})$$

In the worst-case scenario, a height-balanced tree, the number of leaves below each region at height h is exactly h, that is, $t'(R_{il})$ is equal to the height of R_{il}. Let us rewrite the summation by levels of height, that is, from $h = 2$ to $\log_2 N + 1$. The number of regions exactly at height h is $N2^{1-h}$ (Lemma 5.5), so the total number of operations is:

$$\begin{aligned} T(N) &= 2N + 1 + \sum_{h=2}^{\log_2 N+1} h \cdot h \cdot N2^{1-h} \leq \\ &\leq 2N + 1 + N \sum_{h=2}^{\log_2 N+1} h \\ &= 3N + \frac{1}{2}(\log_2 N+1)(\log_2 N+2) = O\big(N(\log_2 N)^2\big) \end{aligned}$$

□

That is, Algorithm 3 takes advantage of the hierarchical structure to find the upper-bound performance efficiently, which was not feasible by a brute-force analysis.

We will use the directional Hamming distance D_H as evaluation measure in this case, given that it is a local measure. To prove it, let us recall the definition of D_H:

$$D_H(P, GT) = \sum_{R \in P} \max_{R' \in GT} \left| R \cap R' \right|$$

We can trivially define the local version of D_H as:

$$D_H^l(R_j, GT) = \max_{R' \in GT} \left| R_j \cap R' \right|$$

Then, we have that $D_H(P, GT) = \sum_j D_H^l(R_j, GT)$, thus fulfilling Definition 1 of a local measure.

Experimental Validation

The remainder of this section is devoted to comparing the quality of the merging-sequence partitions with that of the optimal partitions. In other words, we evaluate how *far* from the optimum are the merging-sequence partitions in terms of D_H. We use the NWMC hierarchies [VMS08], starting at 300 regions (leaves), built on the 500 images from BSDS500 [AMFM11].

First, to illustrate how a typical result looks like, Figure 5.6 plots the evolution of the error D_H on the trees built on three example images. As expected by definition, the quality obtained via the upper-bound performance technique is always better than the merging sequence analysis. Qualitatively, the evolution of the merging-sequence curve is stepped, which shows that, at some points, although increasing the number of regions, the technique is not capable of finding a better representation.

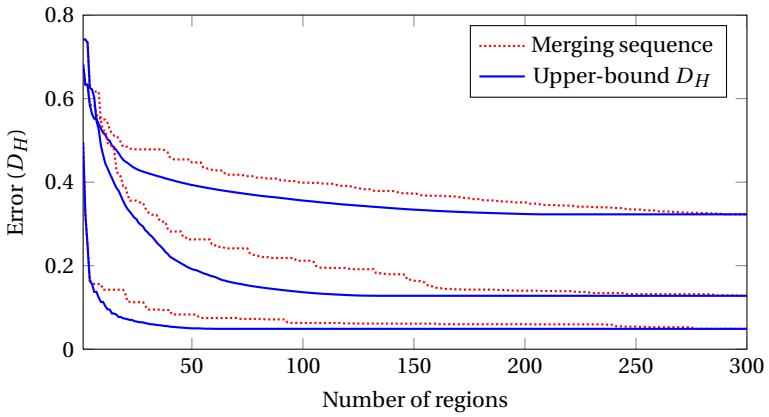


Figure 5.6: Merging sequence versus upper-bound D_H for three example trees

To make the results statistically significant, Figure 5.7 shows the mean of D_H for the whole set of 500 images of BSDS500, compared against the 2696 ground-truth partitions defined on these images. Please note that there is not training involved neither in the tree creation nor in the analysis, so it is fair to use the whole BSDS500 as testing.

The differences between the strategies are indeed kept in mean, thus showing that measuring the quality on the merging sequence does not provide the upper-bound performance.

To get a better quantitative idea of the relevance of the differences, Figure 5.8 plots the mean and variance of the percentage increase of the error of the merging sequence analysis with respect to the upper-bound performance.

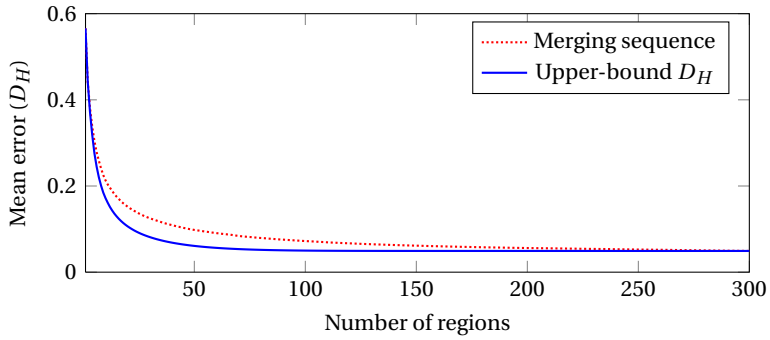


Figure 5.7: Merging sequence versus upper-bound D_H performance error mean in BSDS500

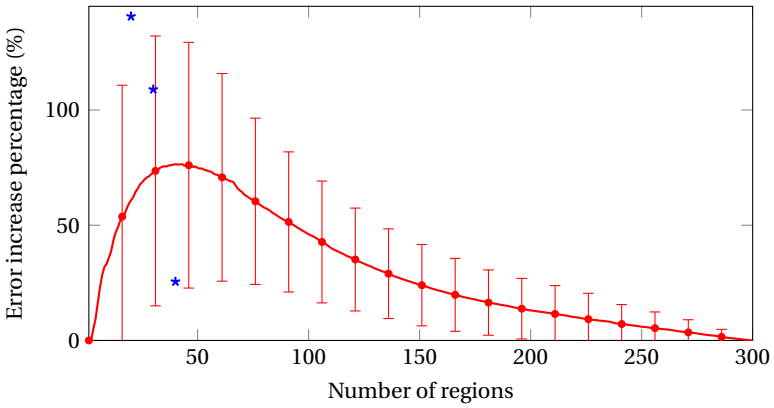


Figure 5.8: Analysis of the error increase percentage of merging sequence with respect to upper-bound-performance error in terms of D_H. Blue stars show the values for the three example images of Figure 5.9.

As a sanity check, the error both for 300 regions and 1 region is the same using both strategies, since there is only one partition with 300 regions (all the leaves of the tree), as well as with only one regions (the whole image).

The error percentage increase is maximum at around 40 regions, where it reaches 76%. This value is clearly significant and could mislead the assessment performed when comparing different techniques or tuning a segmentation algorithm.

The high variance observed in the error percentage increase makes it even more crucial to use the upper-bound performance, since it means that the merging-sequence result is not consistent between images, and therefore it cannot be considered as a bias that does not affect the ranking of the results.

Finally, Figure 5.9 shows three examples of partitions from the ground truth, the merging sequence, and the upper-bound performance analysis in terms of D_H. The error increase percentage for these three examples is plotted in blue stars in Figure 5.8, selected at three levels of percentage of error increase. These examples graphically corroborate that the differences can indeed be significant.

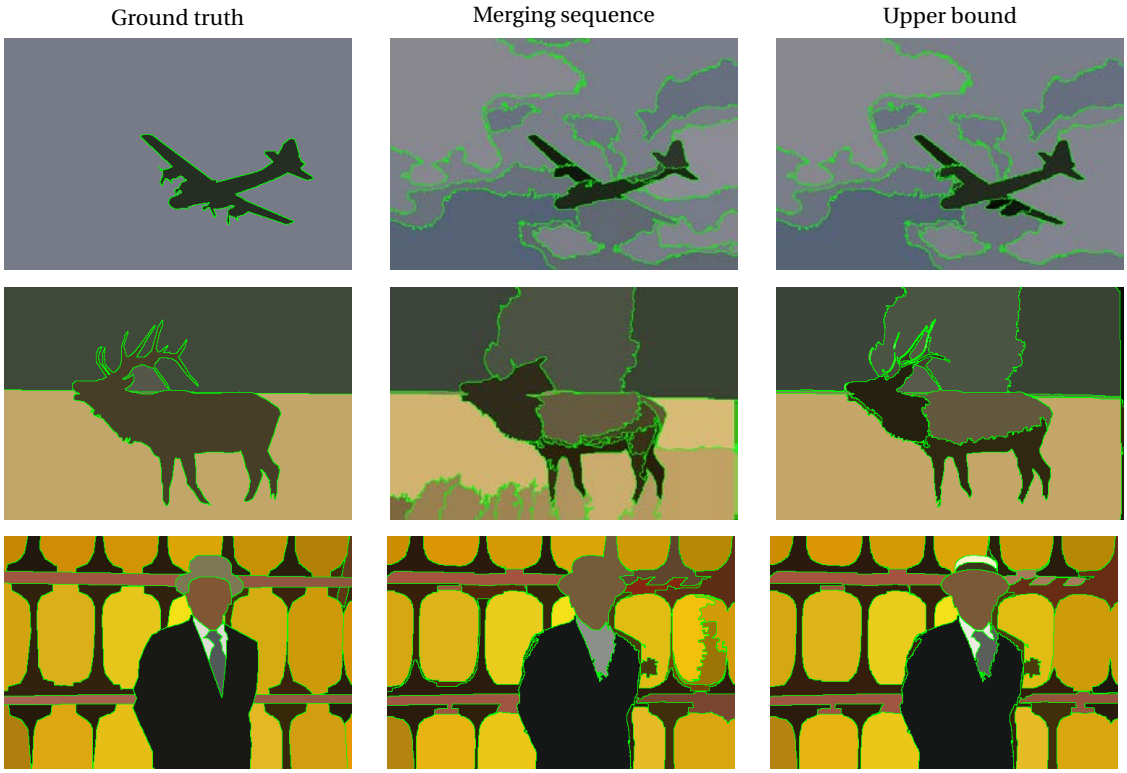


Figure 5.9: Three example partitions with 20, 30, and 40 regions, respectively. First column: one of the ground-truth partition from BSDS500, second column: partition obtained from the merging sequence, and third column: partition reaching the upper-bound performance

5.3 Upper-Bound of a Global Measure: F Measure for Boundaries

The previous section shows how to compute the upper-bound performance based on any local measure, that is, any measure that can be decomposed as the addition of another measure computed locally. This section tackles the computation of the upper-bound performance with respect to the boundary-based F measure F_b (see Section 4.2.3).

The computation of F_b, however, is based on a global optimal matching between the set of boundary pixels of the partition to be assessed and those of the ground truth; so it is not a local measure. This makes it impossible to use dynamic programming to find the optimum.

To make the problem feasible, we propose an algorithm that performs a local matching between the ground truth and each of the *pieces* of region boundaries of the hierarchy. To combine them into a global optimum (it is not directly the addition as in the previous case), we model the problem as an LFCO, which combines the results of the local matchings into a global optimum. This allows us to efficiently find the upper bound of the optimal global matching for any represented partition.

Apart from the computation of the upper-bound performance *per se*, we believe that modeling these problems on hierarchies may inspire other researcher to adapt the approach or tools used to other practical problems, as in [XWC13].

Formally, let P_0 be the partition on which a hierarchy H is built and $R_1 \dots R_n$ its regions. Let $R_{c1}, R_{c2}, \dots, R_{cn_i}$ be the children regions merged at step i to form the parent region R_{pi}, for

$i = 1, \ldots, n_m$ (n_m is the number of merging steps to build H). We define σ_i, $i = 1, \ldots, n_m$, as the common boundary between the regions that are merged at step i of the merging sequence. Figure 5.10 shows a simplified hierarchy creation, where the common boundaries are highlighted.

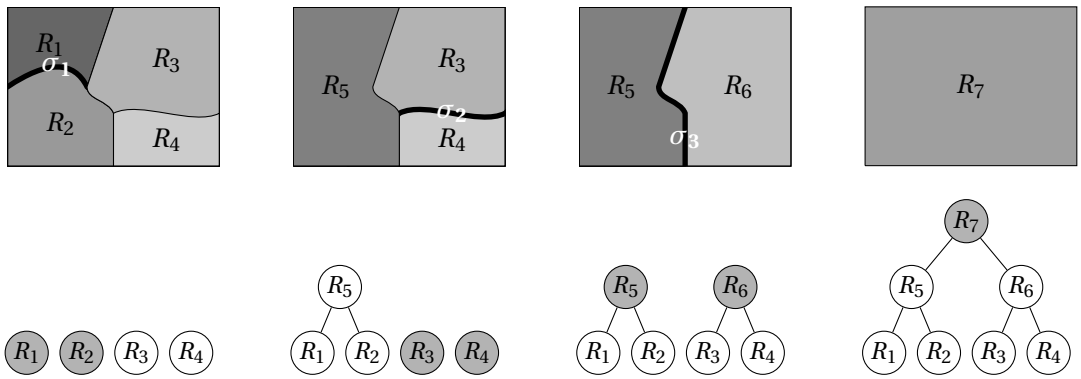


Figure 5.10: Hierarchy creation process with the regions being merged at each step highlighted and the common boundary between them marked

Let $\mathscr{P}$ be the set of all partitions represented in the hierarchy H. Any partition $P \in H$ can be unequivocally described by the set of σ_i that forms its boundaries. Let $\mathbf{p} \in \{0,1\}^{n_m}$ be a binary vector such that $p_i = 1$ if the boundaries of P contain σ_i. Recalling Figure 5.10, the set of merging-sequence partitions can be identified by the vectors: $\mathbf{p} = (1, 1, 1)$, $(0, 1, 1)$, $(0, 0, 1)$, and $(0, 0, 0)$.

This way, one can define a bijection between the set of partitions $\mathscr{P}$ and a subset $\chi \subset \{0,1\}^{n_m}$. Our approach to find the partition that entails the best matching relies on modeling the problem as a binary search in χ and solving it using computationally feasible techniques.

Specifically, we will model the upper-bound partition selection as a Linear Fractional Combinatorial Optimization (LFCO) problem [Rad92]:

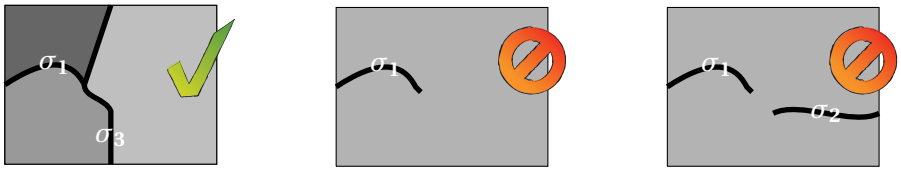
$$\text{LFCO:} \quad \underset{\mathbf{x}}{\text{maximize}} \ \frac{\mathbf{t} \cdot \mathbf{x}^T}{\mathbf{f} \cdot \mathbf{x}^T} \qquad \text{s.t. } \mathbf{x} \in \chi \subset \{0,1\}^{n_m} \tag{5.1}$$

being $\mathbf{f}, \mathbf{t} \in \mathbb{R}^{n_m}$ and all the constraints that define χ linear.

Section 5.3.1 explores the constraints that have to be imposed to vector $\mathbf{p}$ in order for the corresponding partition to be valid within the hierarchy (that is, define χ) and how to make them linear. Next, Section 5.3.2 presents how the F_b of a partition with respect to a ground truth can be obtained from $\mathbf{p}$ in the form of an LFCO as in Equation 5.1. Finally, Section 5.3.3 adds the needed constraints to be able to force a certain number of regions in the partitions.

5.3.1 Forcing the partition to be in the hierarchy

Not all combinations of boundaries σ_i form a valid partition of the hierarchy and thus not all $\mathbf{p} \in \{0,1\}^{n_m}$ correspond to feasible solutions of our problem. Recalling the previous example hierarchy, Figure 5.11 shows the partition corresponding to $\mathbf{p} = (1,0,1)$, which is a valid partition in the hierarchy (although not in the merging sequence), and the representation corresponding to $\mathbf{p} = (1,0,0)$ or $\mathbf{p} = (1,1,0)$, which are clearly not partitions at all.



$\mathbf{p} = (1,0,1)$ $\mathbf{p} = (1,0,0)$ $\mathbf{p} = (1,1,0)$

Figure 5.11: Valid values of $\mathbf{p}$ correspond to partitions in the hierarchy. Invalid representations of $\mathbf{p}$ do not correspond to partitions.

The intuitive idea we want to model to not allow the invalid representations is that if the boundary between some merged regions is not in the partition, the boundaries between any set of their children cannot be in the partition either.

Formally, let $\Sigma_i = \{i_j | j = 1 \dots n_i\}$ be the indices of the boundaries σ_{i_j} between children regions of the parent regions that define σ_i. In the example of Figure 5.10, for σ_3, $\Sigma_3 = \{1,2\}$, because R_1, R_2, R_3, and R_4 are children of R_5 and R_6 that form σ_3.

Then, if the regions that form σ_i are merged ($p_i = 0$), all the pairs of regions that form the boundaries indexed by Σ_i must also be merged ($p_{i_j} = 0$). Formally $p_i = 0 \Rightarrow p_{i_j} = 0 \quad \forall\, i_j \in \Sigma_i$, or equivalently the following constraints:

$$p_i = 1 \quad \text{or} \quad \sum_{i_j \in \Sigma_i} p_{i_j} = 0 \tag{5.2}$$

The binary search problem we are modeling will be much more efficient to solve if it is linear. The following linear constraint is equivalent to Equation 5.2:

$$\sum_{i_j \in \Sigma_i} p_{i_j} \leq K\, p_i \tag{5.3}$$

where K is a *sufficiently large* constant, which in our problem can be set to n, the number of regions in the hierarchy. To conclude, the set of partitions represented in the hierarchy H can be identified with the set:

$$\chi = \left\{ \mathbf{p} \in \{0,1\}^{n_m} \middle| \sum_{i_j \in \Sigma_i} p_{i_j} \leq n\, p_i \right\}.$$

In the sequel, any partition P in the hierarchy H will be identified by its corresponding binary vector $\mathbf{p} \in \chi$.

5.3.2 Upper-bound partition selection as an LFCO

For a given partition $P \in H$ ($\mathbf{p} \in \chi$), let TP be the set of matched boundary pixels with the boundary pixels of a ground truth partition, i.e. true positives, and FP the false positives set. We can write that:

$$|TP| = \sum_{i=1}^{n_m} p_i\, |\sigma_i^m| \qquad |FP| = \sum_{i=1}^{n_m} p_i\, |\sigma_i^u|$$

where $\sigma_i = \sigma_i^m \cup \sigma_i^u$ is a division of the boundary pixels between *matched* and *unmatched*.

The first approach we propose is to perform a single matching between the boundary pixels of the original partition P_0 and those of the ground-truth partition, and define σ_i^m and σ_i^u as those sets of pixels of σ_i matched or unmatched, respectively.

Defining $\boldsymbol{\sigma}^m = \left(|\sigma_1^m|,\ldots,|\sigma_{n_m}^m|\right) \in \mathbb{N}^{n_m}$, $\boldsymbol{\sigma} = \left(|\sigma_1|,\ldots,|\sigma_{n_m}|\right) \in \mathbb{N}^{n_m}$, the problem of finding the partition in the hierarchy with the best F_b with respect to the ground truth can be written as:

$$\mathscr{F}: \ \underset{\mathbf{p}}{\text{maximize}} \ F_b = 2\frac{(\boldsymbol{\sigma}^m,0)\cdot(\mathbf{p},1)^T}{(\boldsymbol{\sigma},|\sigma_{gt}|)\cdot(\mathbf{p},1)^T}, \quad \text{subject to} \quad (5.3)$$

where σ_{gt} is the set of contours of the ground-truth partitions. As wanted, this problem is a *Linear Fractional Combinatorial Optimization* (LFCO) problem and can be solved efficiently via the algorithm proposed in [Rad92]. The remainder of this section is devoted to present the limitation of this approach: the ground-truth multi-matching and the solution we propose.

Ground-truth multi-matching: The previous matching strategy presents the following problem: the matching is carried out at the level of the original partition P_0, assuming it is optimal for all possible combinations of pieces of boundaries in the hierarchy. More precisely, this approach assumes that the sets σ_i^m and σ_i^u do not depend on the partition $\mathbf{p}$ being analyzed; or in other words, that the optimal matching for any partition $\mathbf{p}$ can be obtained from the initial matching on P_0.

But this is not fully correct. In order to illustrate the problem, Figure 5.12 depicts an example partition (a) and the correspondent ground truth (b). If pixels are matched globally, as presented in the previous section, let us assume that all pixels in σ_1 are matched to all $|\sigma_{gt}| = M$ ground-truth pixels. Then, we would have that $|\sigma_1^m| = M$ and $|\sigma_2^m| = 0$, that is, no pixel in σ_2 would be matched at the level of P_0. When computing the number of matched ground-truth pixels for the partition identified by $\mathbf{p} = (0,1)$, we would find that the number of matched pixels is $\boldsymbol{\sigma}^m \cdot \mathbf{p} = 0$, but the right portion of the ground-truth boundary should be matched to σ_2, that is, the correct result should be $\boldsymbol{\sigma}^m \cdot \mathbf{p} = M/2$.

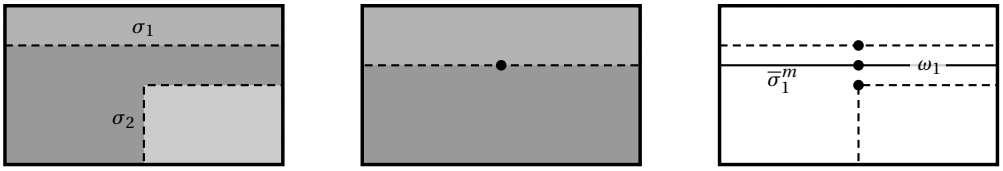


Figure 5.12: Ground-truth multi-matching representation: (a) Partition being assessed, (b) ground truth, (c) both partitions overlaid. The points are plotted to highlight the division of the boundary pixels into sets

The approach we propose to solve this issue is to perform n_m matchings between the pixels of the ground-truth partition and those of each σ_i, and define σ_i^m and σ_i^u as the sets of pixels locally matched or unmatched, respectively. In other words, some pixels of the ground truth can be matched with more than one boundary segment, and thus we call it multi-matching.

Formally, once performed the n_m matchings between the ground-truth boundary pixels and each σ_i, each boundary pixel of the ground truth may be matched to boundary pixels of some σ_i (from 1 to n_m) of the partition. Understanding the set of indices of each σ_i involved in the multi matching as a *signature* of each of the ground-truth boundary pixels, we divide these ground-truth boundary pixels into groups of equal signature.

This way, for instance, we will have a set of unmatched pixels, n_m sets of single-matched pixels which we will denote as $\overline{\sigma}_i^m$ (see Figure 5.12), and the rest will have more than one index in

the signature. Intuitively, we will count a ground-truth boundary pixel as matched only if any of the σ_i in its signature is in the partition but not counting it more than once.

To do so, and in order to have a compact modeling, let us group the set of ground-truth boundary pixels with equal signature and define the set as ω_j. Moreover, let us assume we have m different multiple-index sets of pixels ω_j with signatures $\Omega_j = \{s_1^j, \dots, s_k^j\}$. For instance, the pixels in the set ω_1 of the example of Figure 5.12 (see (c)) are each of them multi-matched to pixels in σ_1 and σ_2, then their signature is $\Omega_j = \{1,2\}$.

Let $\mathbf{q} \in \{0,1\}^m$ be a vector such that $q_j = 1$ if the set of pixels in ω_j should be considered as matched. The value of q_j is function of the values in the signature, that is, $q_j = 1$ if any $p_{s_i^j} = 1$ and 0 otherwise. Mathematically:

$$q_j = p_{s_1^j} \text{ or } p_{s_2^j} \text{ or } \cdots \text{ or } p_{s_{k_j}^j}$$

The equivalent linear constraints that define this equation are:

$$q_j \leq \sum_{s \in \Omega_j} p_s \tag{5.4}$$

$$q_j \geq p_s \qquad \forall\, s \in \Omega_j \tag{5.5}$$

Let us define $\overline{\boldsymbol{\sigma}} = \left(|\overline{\sigma}_1^m|, \dots, |\overline{\sigma}_{n_m}^m|\right) \in \mathbb{N}^{n_m}$ be the vector of single-matched number of ground-truth boundary pixels for each σ_i, and $\boldsymbol{\omega} = (|\omega_1|, \dots, |\omega_m|) \in \mathbb{N}^m$ the vector of the number of ground-truth boundary pixels with equal signature. Then, the problem $\mathscr{F}$ can be rewritten as:

$$\begin{aligned} \mathscr{F}: \ \underset{\mathbf{p},\mathbf{q}}{\text{maximize}} \ & F_b = 2\frac{\left(\overline{\boldsymbol{\sigma}}, \boldsymbol{\omega}, 0\right) \cdot \left(\mathbf{p}, \mathbf{q}, 1\right)^T}{\left(\boldsymbol{\sigma}, \mathbf{0}, |\sigma_{gt}|\right) \cdot \left(\mathbf{p}, \mathbf{q}, 1\right)^T} \\ \text{subject to} \ & (5.3), (5.4), (5.5) \end{aligned} \tag{5.6}$$

which, as wanted, fulfills the form of an LFCO as in Equation 5.1, identifying $\mathbf{x} = \left(\mathbf{p}, \mathbf{q}, 1\right)$ as the binary-valued variable of the problem.

5.3.3 Sweeping the number of regions

Problem 5.6 finds the optimal single partition in terms of F_b so, in other words, it finds the **upper-bound Optimal Image Scale** (ubOIS) partition, defined as the extension of the Optimal Image Scale (OIS) [AMFM11] on the full set of partitions in a hierarchy. Given that a hierarchy represents a collection of partitions of varying number of regions, it would also be desirable to explore the **upper-bound Optimal Dataset Scale** (ubODS) from sweeping a varied range of number of regions, equivalently to the Optimal Dataset Scale (ODS), which is limited to the merging-sequence partitions. To do so, we add the following constraint, that forces the result to have a specific number of regions:

$$\sum_{i=1}^{n_m} p_i = N$$

and sweep all the values of N between 1 and n_m.

5.3.4 Experimental Validation

We compare the upper-bound partition selection technique against the merging-sequence partition analysis on three SoA and one baseline hierarchies. As state-of-the-art hierarchies

we use gPb-UCM [AMFM11], NWMC [VMS08], and IID-KL [CM10]. As a baseline, we use the Random hierachy.

The trees are built on the 200 test images of the BSDS500 [AMFM11] and each of them is compared with each of the multiple ground-truth partitions available and the result averaged. In order for the comparison to be fair, the base partition P_0 on which the tree is built is the same for the four techniques: the one obtained with the gPb-UCM at 100 regions.

The upper-bound partition selection algorithm is implemented in MATLAB, and the optimization itself of the LFCO is done by the IBM ILOG CPLEX Optimizer (free of charge for academic use), which is called directly by the MATLAB code.

In turn, the boundary matching code used in all the experiments has been obtained from [MFM04]. Note that the original code represents the boundaries of a partition in the *pixel grid*, that is, as a mask in which the pixels swept by the boundaries moved half pixel up and left are activated, which leads to an ambiguous representation. This ambiguity can be solved using the *contour grid* [NMPG00], which we use in our code. The numerical impact of this change of representation is not significant but the code obtained is much simpler and more readable.

5.3.5 ODS and OIS

Figure 5.13 shows the mean ODS, OIS, ubODS, and ubOIS F_b values on the BSDS500 test set for the four compared hierarchies. In terms of hierarchies comparison, gPb-UCM presents better results than the rest of hierarchies. The main objective of this section, however, is not to compare the hierarchies themselves, but the partition selection techniques on which the assessment is based.

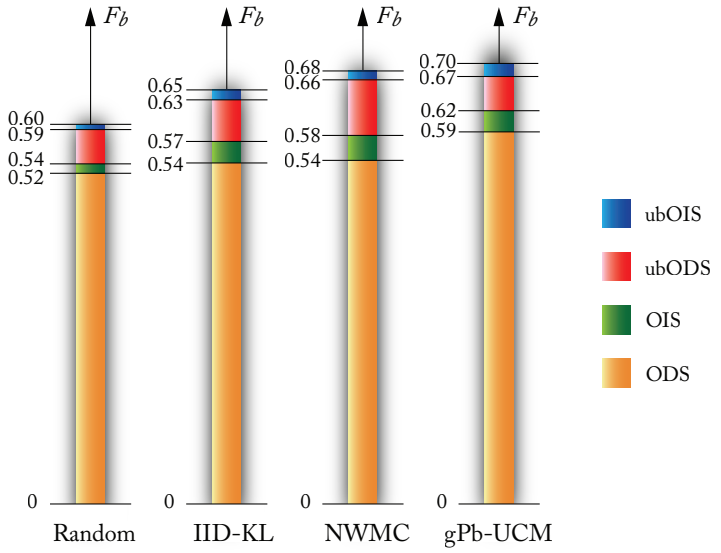


Figure 5.13: Analysis of the F_b values limited to the merging sequence (ODS, OIS) and computed on the full set of partitions represented in a tree (ubODS and ubOIS)

Regarding the comparison between ODS and OIS, the latter is coherently higher than the former both in the merging sequence and the upper-bound analysis. An improvement is also observed between the merging-sequence and the upper-bound techniques both on ODS and OIS, which is, again, coherent with the theory.

Moreover, what really makes the difference between comparison techniques is their relative values, that is, how well the assessment discriminates the quality between the different hierarchies. In particular, good assessment techniques should be able to correctly discriminate between a random tree and the other techniques. To evaluate this aspect, Figure 5.14 shows the relative values of ODS and OIS, that is, assigning 0 to the random tree, 1 to gPb-UCM, and scaling the rest of the values accordingly.

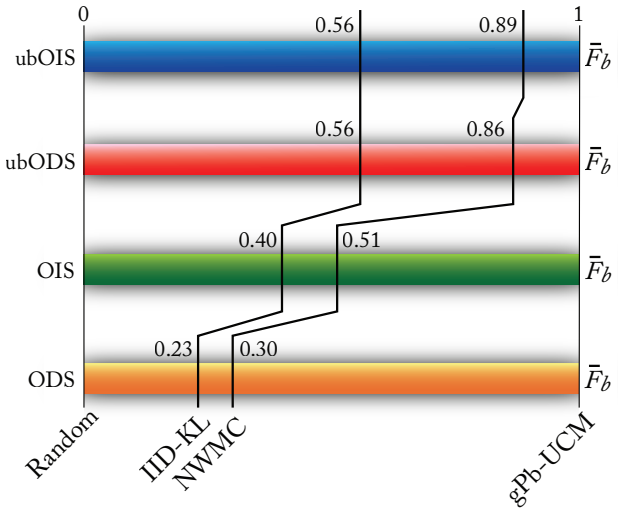


Figure 5.14: Relative ODS and OIS F_b values for the merging-sequence and upper-bound partition selection techniques

If the measurement techniques were equivalent, the relative values should not change, but there are significant differences in these relative values, meaning that the conclusions extracted from the assessment can vary depending on the criterion used.

As introduced previously, a desirable property of the assessment techniques is a high discrimination of the random tree. In other words, it is obvious that the random trees must be far away from any *real* hierarchy. Under this point of view, the upper-bound assessment provides much better behavior. As an example, the IID-KL tree is much closer to the random tree than to the gPb-UCM tree for ODS, while for the upper-bound ODS (ubODS), the IID-KL tree is halfway between the two, which is much more coherent with the expected results.

The improvement obtained in the OIS case with respect to ODS highlights the relevance of the *alignment* algorithm on the results obtained. By alignment we mean how the results on each image are put in correspondence to average result in the database. (Typical examples are the number of regions or the UCM strength of the partitions.) The same way, the improvement of the upper-bound analysis ubODS and ubOIS with respect to ODS and OIS is an indicator of the impact of the partition selection algorithm on the assessment, that is, which partitions we select from the tree affects the final result of the evalaution significantly.

Being the upper-bound and the merging-sequence assessment not equivalent could also entail changes in the ranking order of the hierarchies quality. Focusing on the sorting of the algorithm quality for the two top-rated hierarchies gPb-UCM and NWMC, in 529 of the 1800 cases studied (9 parameterizations on 200 images), the ranking provided by the merging sequence analysis is not coherent with the one provided by the upper-bound. In other words, different partition selection strategies can lead to different decisions with respect to which is the best hierarchy based on a supervised assessment.

5.3.6 Upper-bound precision-recall curves

As in the case of flat partitions, precision-recall curves give us much richer information than the summary measures (in this case OIS, ODS, ubOIS, and ubODS).

Figure 5.15 shows the precision-recall curves for the four hierarchies studied in this section. In the figure of the left, the points have been obtained using the merging-sequence partition selection whereas, in the figure of the right, the proposed upper-bound partition selection has been used.

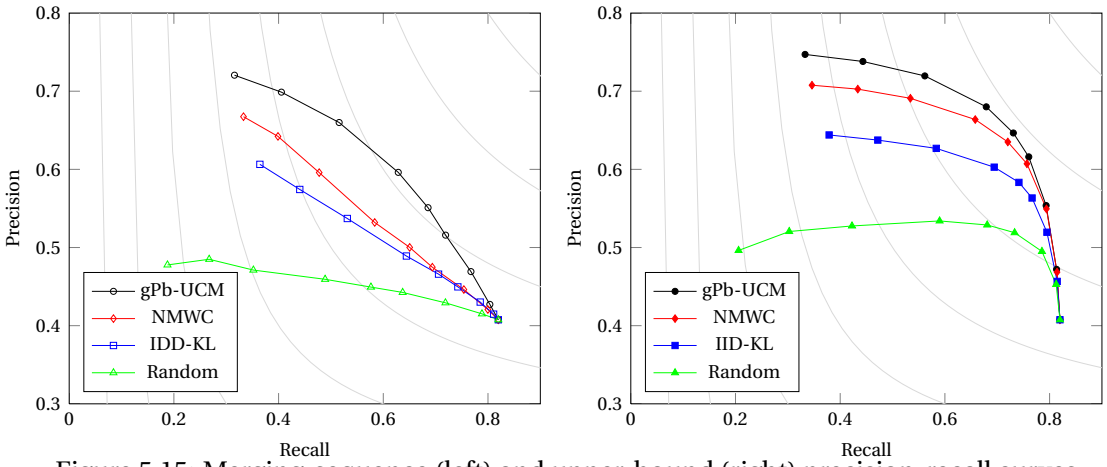


Figure 5.15: Merging-sequence (left) and upper-bound (right) precision-recall curves

In the range of interest of the hierarchies, that is, the range of better F_b, similarly to the results of the previous section, the upper-bound precision-recall curves better discriminate between the random hierarchy and the rest of trees. In the range of higher number of regions, close to the leaves of the tree, the different curves are much closer than in the merging sequence, which reflects that, in this range, the original partition is more influent than the hierarchy itself. Note that, coherently, all curves meet in the point corresponding to 100 regions, because each tree contains only one partition with the maximum level of detail: P_0.

To better visualize the differences between the precision-recall curves and their upper-bound equivalents, Figure 5.16 shows them both in the same axis, leaving the IDD-KL tree out for the sake of clarity of the plot.

Note that, for each type of hierarchy, both curves start from the same point at high number of regions and tend to converge for few regions. In the middle range, corresponding also to the better F_b values, the gain obtained with the upper-bound assessment is much more relevant for the NWMC tree than for the rest of trees, which reinforces the possibility that different partition selection techniques can lead to discrepant results.

5.3.7 Computational cost

Although a supervised assessment is usually performed offline, a reduced computational cost is of paramount importance. The faster the method is, the larger the datasets in which researchers will tune and test their algorithms can be, which results in a more solid research. This section compares the computational cost of the proposed evaluation techniques (ubODS and ubOIS) in front of ODS and OIS.

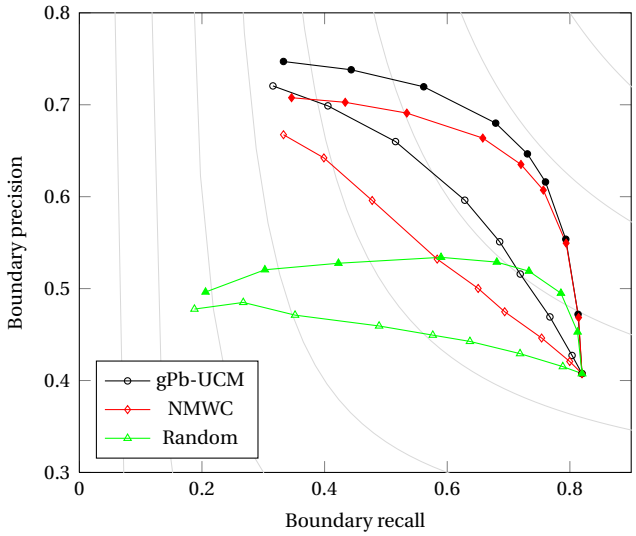


Figure 5.16: Merging sequence (unfilled markers) versus upper-bound (filled markers)

ODS requires to compute the boundary matching for k different number of regions, thus the whole time is k times the cost of a boundary matching, which we will refer to as $t(BM)$. OIS requires the ODS computation and to find the F_b maximum for each image, so the cost is also $k \cdot t(BM)$. The cost of ubODS is one boundary multi-matching $t(BMM)$ and k LFCO optimizations $t(LFCO)$. Finally, the cost of ubOIS is $t(BMM) + t(LFCO)$, since one single optimization finds the optimal number of regions, thus not needing the computation of ubODS.

The mean values obtained in the experiments for each of these processes are: $t(BM) = 0.34\,s$, $t(BMM) = 0.64\,s$, and $t(LFCO) = 0.21s$. The mean time spent for each technique is therefore $t(ODS) = t(OIS) = k \cdot 0.34\,s$, $t(ubODS) = 0.64 + k \cdot 0.21\,s$, and $t(ubOIS) = 0.85\,s$. Figure 5.17 shows the relative cost of the upper-bound techniques with respect to the cost of the merging-sequence ones. The higher the number of samples k, the lower the relative cost of the upper-bound techniques. The computation of ubODS is approximately 25% faster than ODS and OIS, while ubOIS, thanks to the fact that a single optimization is enough, is considerably faster.

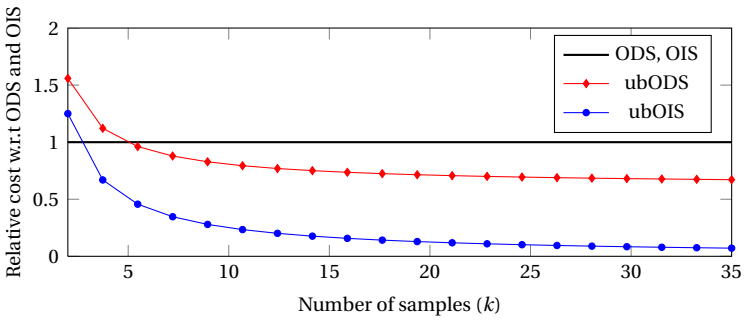


Figure 5.17: Computational cost analysis for varying number of samples (k)

5.3.8 Worst-discrepancy graphical results

To get a qualitative idea of the type of discrepancies between the region selection techniques, Figure 5.18 shows the most discrepant example of partition selection on the gPb-UCM tree for 6, 10, and 20 regions selected, that is, the three results whose partition selected in the merging sequence is more dissimilar with the upper-bound one.

The differences observed between the two strategies are visually relevant. In the first and second columns (6 and 10 regions), the merging sequence analysis obviates the main object of interest or part of it, while in the upper-bound partition selection the object is present in the selected partition. In the last column (20 regions) the upper-bound selection is capable of highlighting the higher importance of the background object with respect to the background as in the ground truth.

To sum up, the upper-bound partitions (Figure 5.18.d) represent the quality of the tree much better than those of the merging sequence (Figure 5.18.c), or in other words, the region selection masks the actual achievable quality of the tree.

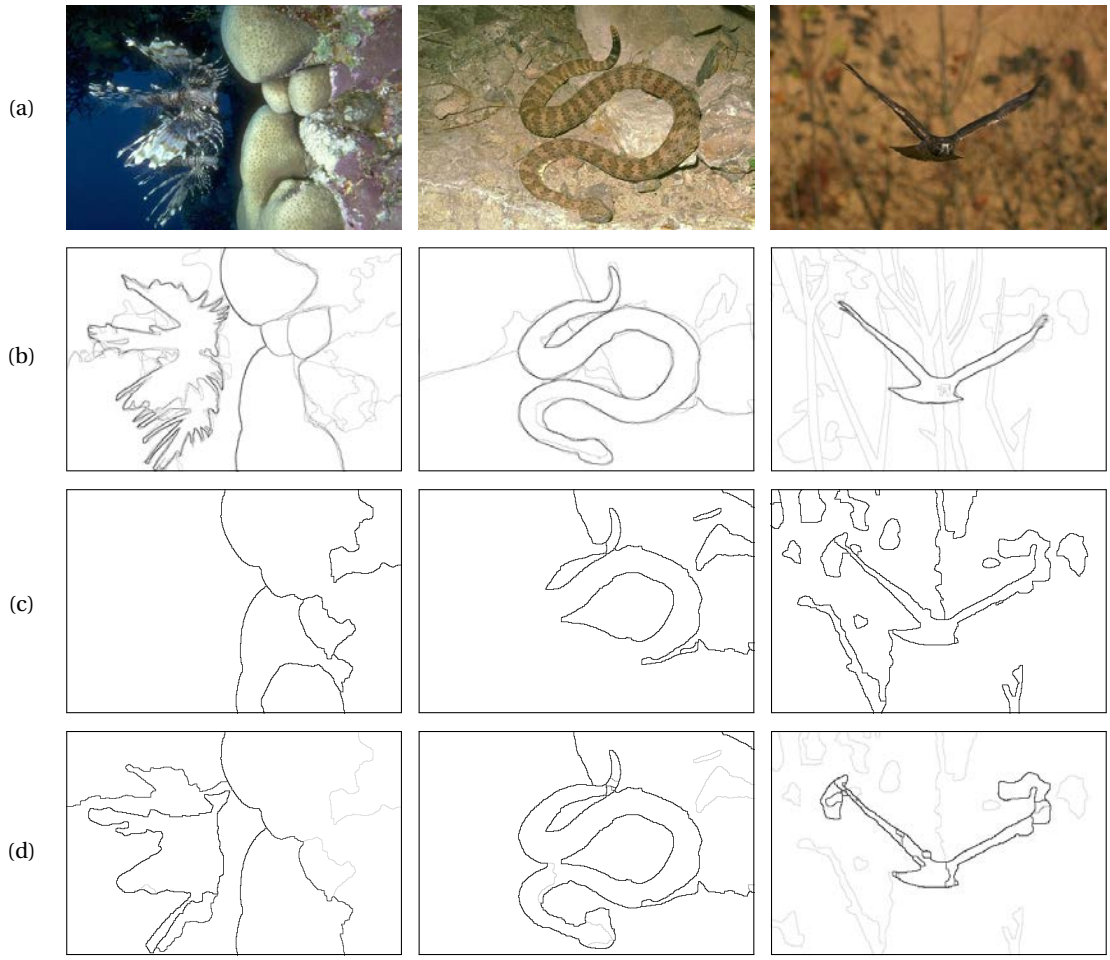


Figure 5.18: Worst-case results on gPb-UCM trees: (a) images of the BSDS500 test set, (b) their multiple ground-truth partitions, (c) partitions selected from the merging sequence with 6, 10, and 20 regions, respectively, and (d) the upper-bound partitions with the same number of regions

5.4 Upper-Bound on the Full Soup of Partitions: F Measure for Boundaries

The previous section computed the upper-bound F measure for boundaries F_b on the set of partitions that can be extracted from a hierarchy. Recalling Figure 5.4, the search space within a hierarchy is a subset of the whole set of partitions that can be formed by merging regions from the leaves partition.

If we start the hierarchy from pixels, this set contains all possible partitions, but handling a hierarchy with such a number of regions is not feasible in practice. The common approach is instead to start our hierarchies from super-pixels such as SLIC [ASS+10] or the Oriented Watershed Transform (OWT) on the gPb boundary detector as done in [AMFM11].

The question that now may arise is therefore, how much performance are we potentially losing by not starting from pixels? In practice, boundary recall is used (e.g. [VdBBR+12]), given that it represents the number of boundaries we are missing, and in fact gPb-UCM has almost full boundary recall at their leaves. This, however, does not reflect what would be the precision that we would have to get that recall.

One may argue that we can greedily assign each super-pixel to the region which a higher overlap with it. But then again, if for the easiest task of region selection to form objects we proved that this approach does not get to the optimum representation in a significant number of cases (see Section 2.4.2), how would we be sure that we are actually getting the real upper bound?

Given the high boundary recall that current super-pixels have, the differences would not be significant in practice. Motivated by the fact that, as shown in the previous section, these models can serve as inspiration for other researchers to apply them in their practical problems, we believe it is worth the effort despite the direct application does not seem promising.

Formally, let P be the partition we want to assess, and $R_1,\ldots,R_n$ its regions. Let $\sigma_1,\ldots,\sigma_b$ the boundaries between each pair of neighboring regions in the partition. Observe that, since we are handling the partition at a single scale, the number of pieces of boundaries will be much higher than in the hierarchical case.

Figure 5.19(a) shows an example partition with the corresponding pieces of boundary. Note that each σ_i always begins and ends in a junction, where three or four pieces of boundary meet (including the borders).

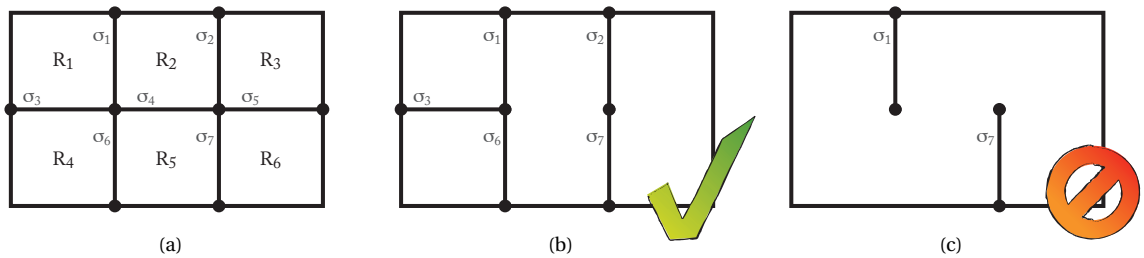


Figure 5.19: (a) Original partition P, (b) valid partition formed by merging regions in P, and (c) boundary representation that do not correspond to a valid partition

Let $\mathbf{p} = (p_1,\ldots,p_b)$ be a binary vector where $p_i = 1$ if, and only if, the piece of boundary σ_i *is*

in the partition. The objective function and the multi-matching constraints will be exactly the same than in the previous section, so the first approach to modeling the problem of finding the upper-bound F_b in flat partitions is:

$$\mathscr{G}: \underset{\mathbf{p},\mathbf{q}}{\text{maximize}} \; F_b = 2\frac{(\overline{\boldsymbol{\sigma}},\boldsymbol{\omega},0)\cdot(\mathbf{p},\mathbf{q},1)^T}{(\boldsymbol{\sigma},\mathbf{0},|\sigma_{gt}|)\cdot(\mathbf{p},\mathbf{q},1)^T} \quad \text{subject to} \quad (5.4),(5.5) \tag{5.7}$$

where $\mathbf{q}$ is the vector of the multi-matched pieces of boundary, and we have removed the constraint (5.3) to force the partition to be in the hierarchy.

What is missing, therefore, is to add the needed constraints to limit the search space to that of the valid partitions formed by merging regions from the super-pixel partition. To get the intuition behind the problem, Figure 5.19(b) and (c) show a correct partition, and an invalid situation that could be represented by the current model. These two situations correspond to the following $\mathbf{p}$ of the model: (a) $\mathbf{p} = (1,1,1,0,0,1,1)$ and (a) $\mathbf{p} = (1,0,0,0,0,0,1)$.

In sight of the example, the first approach to limiting the search is to impose constraints on each of the junctions of the original partition P. Figure 5.20 shows all possible situations in terms of number of incident boundaries on each junction.

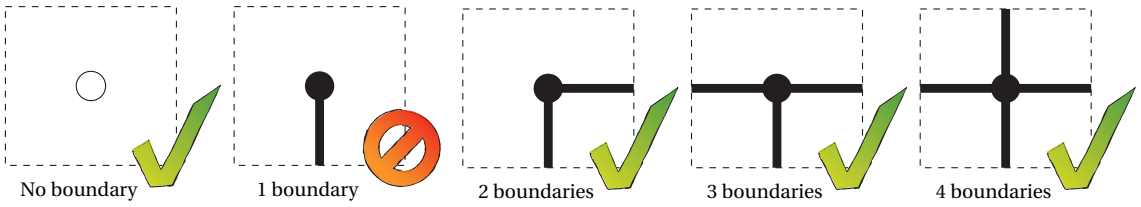


Figure 5.20: Possible situation in each junction with respect to the number of *activated* incident boundaries. The only invalid situation is at 1 incident boundary.

Intuitively, therefore, we should avoid that the number of σ_i that *touches* each junction J_k is 1. To model this behavior, we define z_k, $k = 1,\ldots,nj$ (nj is the number of junctions), as a binary variable indicating whether junction J_k *is* in the final partition. Let Φ_k be the set of indices i of the σ_i incident to J_k, then we can write:

$$\sum_{i\in\Phi_k} p_i \neq 1 \qquad k = 1\ldots nj$$

Transforming this constraint to inequalities we can relate it to the junction variables z_k:

$$\sum_{i\in\Phi_k} p_i \leq 4z_k \qquad k = 1\ldots nj \tag{5.8}$$

$$\sum_{i\in\Phi_k} p_i \geq 2z_k \qquad k = 1\ldots nj \tag{5.9}$$

Getting back to the *lateral* motivation of this section of investigating about the modeling on image partitions, the current formulation allows us to mathematically model the existence or not of each of the junctions in the final partitions. This may have applications to algorithms extracting depth from images, for instance, since junctions are important cues in this regard.

To double check the results, we reconstruct the boundary map that a given $\mathbf{x}$ represents and compute the corresponding partition by means of a region labeling algorithm. We then com-

pute the boundary map of that partition, and in principle we should get back the same boundary representation. Performing tests on BSDS500, however, we found that many boundary maps where not correct.

Figure 5.21 (a) shows a simple example where the current model fails, in the sense that the boundary map obtained (b) does not represent any valid partition, although the junction constraints are correctly fulfilled.

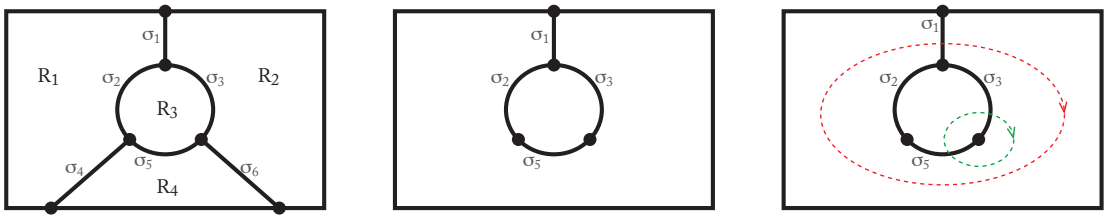


Figure 5.21: Example where the model with constraints only on the junctions can lead to an invalid result (b). We detect them by computing paths that cross only one boundary (c), in red. Correct paths cross a different number of boundaries (in green).

Intuitively, we must constrain, therefore, that no circular path without self-crossings overlaps with exactly one boundary, as illustrated in Figure 5.21 (c), in red. Note that this constraint is a generalization of the junction constraints, by considering the path around the junction itself, in green. To model this behavior mathematically, let us give some definitions.

We can interpret the boundaries σ_i as edges of a planar simple graph and the junctions J_k as its vertices. We refer to this graph as the boundary graph of the partition. In this situation, the partition regions are represented by the faces of the graph.

The dual of this graph is the so-called Region Adjacency Graph (RAG), in which vertices represents regions, edges represent boundaries between pairs of regions, and faces represent junctions. Figure 5.22 represents the boundary graph and its dual RAG of the previous example. Please note that we are ignoring the outer boundaries and the *external* regions for better readability but they should also be represented in the graph.

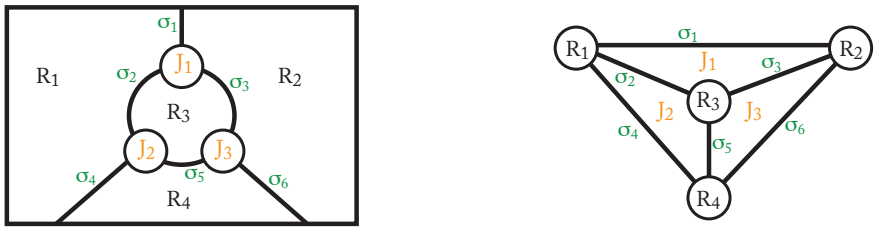


Figure 5.22: Boundary graph and its dual RAG

In it we see that the paths we were describing on the image plane correspond exactly to cycles in the RAG. In the case of the red path in Figure 5.21, the corresponding cycle in the RAG would be $R_1 - R_2 - R_4$. Note also that the faces of the RAG correspond to junctions and the edges around that face correspond to the incident boundaries to that junction.

The model should impose, therefore, that for any cycle in the RAG, the number of activated boundaries σ_i is different from 1, that is, it should impose constraints (5.8) and (5.9) for any

cycle. In this particular example, for instance, we should impose that:

$$p_1 + p_2 + p_3 \neq 1 \qquad p_1 + p_3 + p_4 \neq 1 \qquad p_2 + p_3 + p_4 \neq 1 \qquad p_1 + p_2 + p_4 \neq 1$$

where the fourth constraint would not allow the invalid representation shown in Figure 5.21.

We have therefore, at least mathematically, solved the problem, if we are able to find *all* the cycles in the RAG. The problem in practice is that the number of cycles grows significantly. Since each cycle corresponds to two constraints, this approach is not feasible.

Our proposal is therefore to solve the problem ignoring the cycle constraints, check if the result is valid by region labeling, and if it is not, find a cycle in the representation with exactly one activated boundary. Then add the constraint that will avoid that particular representation and iterate the process until we find the correct optimal representation.

We use the Depth-First Search (DFS) to find a set of cycles that could be used as a *base* to find all cycles. By computing XORs between their incidence vectors, we can build the full set of cycles and find the one that does not fulfill the constraints.

In order to speed this process up, we take advantage of the fact that we are looking for a cycle with exactly one activated boundary to discard many cycles beforehand and reduce the search space.

Experimental Validation

The rationale behind the upper-bound boundary F measure on flat partitions is to extend the evaluation to all partitions that can be formed by the leaves of a hierarchy, in order to quantify the loss in achievable quality when reducing the search space to those partitions represented in such a hierarchy.

In this case we use the SLIC superpixels [ASS+10] to perform the test, since they are a potential tool to be used as leaves of the trees and allow us to easily tune the number of regions in the partitions. In order to be in a controlled environment we start by setting the partitions to have only 60 super pixels, although we acknowledge that SLIC may have some degree of under segmentation at that level.

Recalling that our goal is to find the best partition with respect to a ground-truth that can be formed by merging regions from an original partition, we will compare the results obtained by the optimal algorithm with a greedy baseline consisting in assigning each region to the ground-truth region with the highest overlap. We will segment the 200 images in the test set of BSDS500 and compare the result to the 1063 partitions in it.

The quality obtained by the greedy method is $F_b = 0.65 \pm 0.11$, while the optimal method achieves a much better $F_b = 0.77 \pm 0.06$. The optimal result is better than the baseline in a 99.3% of the cases. In exchange, the optimal technique takes around 54 seconds per image while the greedy takes around 2 seconds, both in an unoptimized Matlab code. So in other words, the optimal technique achieves significantly better partitions in terms of F_b in exchange of being computationally expensive.

The algorithm performs 11.8 iterations in mean that add newer constraints to discard incoherent results between the modeled boundary map and the actual contours. When it finds those incoherences, it needs to find incorrect cycles in the regions adjacency graph. To do so,

we iteratively test XOR combinations of *base* cycles found by the depth-first search algorithm. We are not aware of any theoretical prove that an incorrect cycle will be found in a bounded number of iterations, but in practice the distribution of number of base cycles needed is as follows: 1% one cycle, 88.5% two cycles, 10.1% three cycles, and 0.4% four cycles. In the 1063 cases there was no need to go beyond four cycles.

On these experiments, therefore, it seems that the upper-bound F_b on flat partitions allows us to discover better partitions, but let us analyze some qualitative results. Figure 5.23 shows five representative examples with the image, one of its human annotation, the SLIC partition evaluated, and the best partition according to the optimal and the greedy strategies.

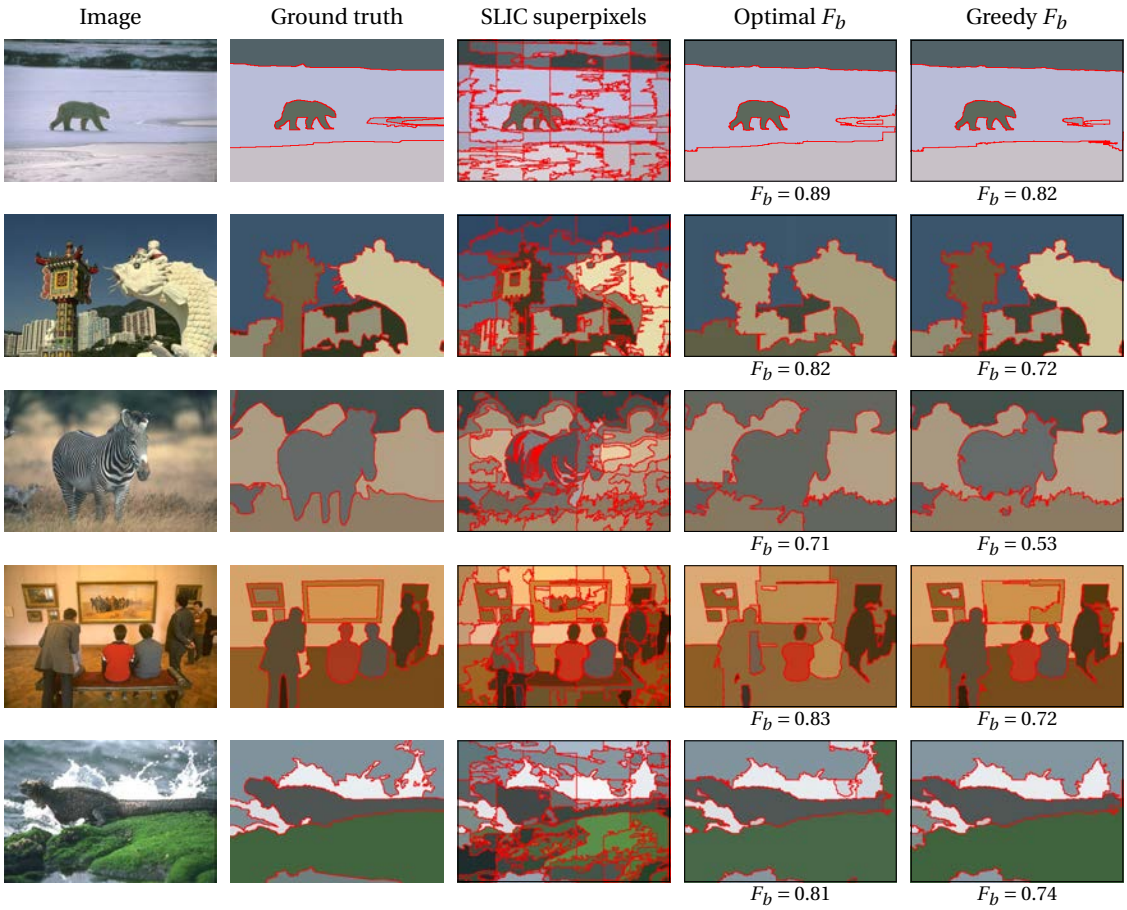


Figure 5.23: Qualitative results comparing the optimal and greedy upper-bound F_b on flat partitions

We observe that, although the achieved F_b is consistently better in the optimal case, in general the partition obtained using the greedy algorithm seems more correct qualitatively. Our interpretation is that we are *pushing* the measure too hard, and it reaches complex combinations of regions to sweep all possible contours. These solutions, however, usually suffer from severe oversegmentation. The greedy solution, in contrast, does not match as many contours but it suffers much less oversegmentation, which makes the solutions more qualitative pleasant. This result reinforces one of the conclusions of the previous chapters: we should use both region and boundary-based measures to evaluate image segmentation.

Realistic superpixel algorithms, however, usually create partitions with at least one order of

magnitude more regions than the partititons tested in this section (60 superpixels). We tested our algorithm on realistic partitions and it failed to converge in a significant number of cases, in which the algorithm to find an incorrect cycle ended up stuck.

To solve this issue we could design a better heuristic to find the incorrect cycle faster, but given the poor qualitative results obtained in the test examples, we started wondering whether it was a path worth exploring.

5.5 Upper-Bound of a Global Measure: F Measure for Objects and Parts

This section is devoted to extending the F measure for objects and parts, presented in Section 4.3, to hierarchies. We want to find the partition that maximizes the similarity with the ground truth in terms of F_{op}, that is, we want to find the upper-bound performance of the hierarchy with respect to this measure.

Recalling the model in Section 5.3, where we find the best partition in the hierarchy in terms of a boundary-based measure, we defined a vector $\mathbf{p}$ whose values reflected whether the ith boundary remained in the partition. Since the measure we are using in this section is region-based, we need a binary variable to reflect whether region i is in the final partition. Let us therefore first adapt the model to the region-based point of view.

Let $R_1,\ldots,R_{nr}$ be the nr regions in the hierarchy, being $R_1,\ldots,R_{nl}$ the nl leaves and $m_1\ldots,m_{nm}$ be the nm mergings that form the hierarchy. Let x_i be a binary variable that indicates whether region R_i *is* in the selected partition, and let z_j indicate whether merging j is not performed, that is, whether the boundaries between the merged regions at step j are in the partition (equivalent to variables p_i in Section 5.3).

We impose the constraints on Equation (5.3) to variables z_j to force a valid partition representation. The next step is to relate variables $\mathbf{x}$ and $\mathbf{z}$, that is, for each partition represented by a $\mathbf{z}$ we need to find the regions R_i that form the partition. Let ia and id be the indices of the mergings that make R_i *appear* and *disappear*, that is, the first and the last mergings where R_i *is* in the merging-sequence partitions. In the case of leave regions, we have that $x_i = z_{id}$, that is, region R_i will be in the partition if, and only if, merging id has not been performed. In the case of the root, assuming it is the last region R_{nr}, we need merging ia to be performed, that is $x_i = 1 - z_{ia}$. Figure 5.24 depicts a part of a hierarchy and the related variables.

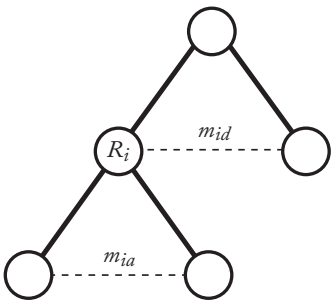


Figure 5.24: Hierarchy illustration that relates region R_i with the indices of the steps in the merging sequence in which R_i appear (ia) and disappear (id)

In the case of non-leave non-root regions, for R_i to be in the partition we need merging *ia* to be done and *id* not to be done, so we need:

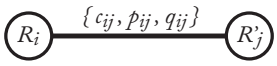
$$x_i = (1 - z_{ia}) z_{id} \qquad i = nl{+}1, \ldots, nr{-}1$$

Linearizing this equality leads us to the following linear constraints:

$$x_i \leq z_{id} \tag{5.10}$$
$$x_i \geq z_{id} - z_{ia} \tag{5.11}$$
$$x_i \leq 2 - z_{id} - z_{ia} \tag{5.12}$$

Once we have modeled the search on the hierarchy from a region-based point of view, let us model the measure itself. Recalling the definition of F_{op} in Section 4.3, the main idea is to match pairs of regions between the partition and the ground truth and classify them into object candidates, part candidates, or fragmentation terms. To represent this mathematically, given a region R_i in the hierarchy and a region R'_j in the ground truth that overlap, we define three binary variables c_{ij}, p_{ij}, q_{ij} that indicate whether R_i is an object candidate (c_{ij}=1) matched to R'_j, if R_i is a part candidate (p_{ij}=1) of R'_j, or the other way around, R'_j is a part candidate (q_{ij}=1) of R_i.

Each pair of intersecting regions R_i-R'_j is therefore described by the three binary variables:



The first step for these variables to correctly model the behavior in our measure is to force them to be zero if the corresponding overlap levels (O_S^{ij} and O_G^{ij} defined in Section 4.3) are not fulfilled (lower than γ_o or γ_p). If the overlap levels are fulfilled, we do not impose any constraint on the binary variables c_{ij}, p_{ij}, and q_{ij} letting the optimization algorithm *decide* whether that region should be an object candidate, a part candidate, or should not even be in the selected partition.

For instance, for regions R_i-R'_j to be classified as object candidates we must have $O_S^{ij} > \gamma_o$ and $O_G^{ij} > \gamma_o$ (see Algorithm 2). To enforce this behavior we add the following constraints:

$$\begin{aligned} O_S^{ij} &> \gamma_o \, c_{ij} \\ O_G^{ij} &> \gamma_o \, c_{ij} \end{aligned} \tag{5.13}$$

which force the constraints to be fulfilled if $c_{ij} = 1$ and impose no constraint if $c_{ij} = 0$. The constraints for the part and fragmentation candidates are defined accordingly:

$$\begin{aligned} O_S^{ij} &> \gamma_o \, p_{ij} & \qquad O_S^{ij} &> \gamma_p \, q_{ij} \\ O_G^{ij} &> \gamma_p \, p_{ij} & \qquad O_G^{ij} &> \gamma_o \, q_{ij} \end{aligned} \tag{5.14}$$

We finally enforce that each region has at most one classification at a time, and only if that region is selected; which is fulfilled by the following constraint:

$$c_{ij} + p_{ij} + q_{ij} \leq x_i \tag{5.15}$$

To find the expression of the objective function, let us first recall the expression of F_{op} from Section 4.3:

$$P_{op} = \frac{oc + fr + \beta \, pc}{|S|} \qquad R_{op} = \frac{oc' + fr' + \beta \, pc'}{|G|}$$

where $|S|$ and $|G|$ are the number of regions in the partition and in the ground truth, respectively. We first need, therefore, to find the expression of all *counts* involved as linear expressions from the variables defined. By abuse of notation, let ij be the single index of the ni intersecting pairs of regions, then we have:

$$oc = oc' = \sum_{ij=1}^{ni} c_{ij} \qquad pc = \sum_{ij=1}^{ni} p_{ij} \qquad pc' = \sum_{ij=1}^{ni} q_{ij}$$

$$fr = \sum_{ij=1}^{ni} O_G^{ij} p_{ij} \qquad fr' = \sum_{ij=1}^{ni} O_S^{ij} q_{ij}$$

And the number of regions in the partition can be obtained by:

$$|S| = \sum_{i=1}^{nr} x_i$$

If we substitute all the expressions to get the F_{op} expression we get to a Quadratic Linear Fractional Optimization (QFCO) problem because there appear crossed terms between the expression above. We optimize the arithmetic mean A_{op} between P_{op} and R_{op} instead of the harmonic mean F_{op}, getting to the following objective function:

$$A_{op} = \frac{1}{2}\big(P_{op} + R_{op}\big) = \frac{1}{2|S||G|}\Big[|S|\big(oc + fr + \beta pc\big) + |G|\big(oc' + fr' + \beta pc'\big)\Big]$$

Given that $|S|$ is a linear combination of the variables, A_{op} is again a QFCO in this formulation. To make it linear, we have to impose the number of regions beforehand (that is, compute the ubODS) by imposing:

$$|S| = \sum_{i=1}^{nr} x_i = N_r \tag{5.16}$$

This way, the final optimization problem to find the ubODS in terms of A_{op} is the following linear optimization problem:

$$\mathcal{K}: \underset{\mathbf{x},\mathbf{z},\mathbf{c},\mathbf{p},\mathbf{q}}{\text{maximize}} \; A_{op} = \frac{1}{2N_r|G|}\Big[N_r\big(oc + fr + \beta pc\big) + |G|\big(oc' + fr' + \beta pc'\big)\Big] \tag{5.17}$$
$$\text{subject to} \quad (5.13), (5.14), (5.15), (5.16)$$

To get the ubOIS we sweep the number of regions N_r and combine the per-image optima.

Experimental Validation

The algorithm to find the best partition in terms of the F measure for objects and parts is essentially a 4-class classification problem on the set of pairs of intersecting regions, that is, each pair of regions can be either an object match, a part candidate, a fragmentation candidate, or none of them. To model it as a binary problem we have therefore included three binary variables and some constraints per pair. Realistic problems have thousands of regions and thus the final model ended up being too big to solve in a reasonable time or with a reasonable amount of resources.

My advisor usually reminds me of the saying that *when you only have a hammer everything looks like a nail* and I believe in this case this is especially true: the problem of finding the optimal F_{op} in hierarchies does not fit anymore in the binary problem modeling used in the

previous sections of this thesis. We should, therefore, explore other approaches to solve the problem.

At this point of the thesis, however, we had the great opportunity to collaborate with Prof. Jitendra Malik's computer vision group at the University of California, Berkeley; working with Dr. Pablo Arbelaez in applying segmentation hierarchies to object detection and segmentation, specifically on segmented object candidate generation. We decided, therefore, to prioritize this new field of research to the detriment of finishing and polishing this last part of hierarchical segmentation evaluation. The findings on object candidate generation are described in the next chapter and the open doors are further discussed in the future work.

Part III

Object Candidates Generation

6 Object Candidates Generation

6.1 Introduction

Up until this point of the thesis, we have studied how to evaluate image segmentation techniques, both flat and hierarchical, comparing them both against annotated objects and full partitions. As a consequence, we have also gained knowledge about the state-of-the-art segmentation techniques.

In particular, one of the main conclusions of Chapter 3 is that working directly on hierarchies allowed us to represent objects with a certain quality using many less regions than when working on flat partitions. On top of that, we showed that the diversity coming from different hierarchies pushed the achievable quality even higher than when working on a single hierarchy, and merging up to three regions achieved the 90% of the maximum quality on some hierarchies.

Building on these three conclusions, this section goes beyond the pure supervised assessment of segmentation techniques and proposes **Multiscale Combinatorial Grouping** (MCG): an algorithm for object candidate generation. Given a set of varied hierarchies as input, MCG proposes a ranked set of segmented object candidates, that is, a set of object masks with an associated likelihood of being an object. MCG produces the candidates by learning how to combine up to three regions from a set of seven hierarchies. As we will show in the following sections, MCG outperforms the state of the art both in terms of object segmentation and localization, and in computational cost.

In contrast to some other approaches, MCG is class independent, in the sense that we do not particularize the algorithm for different classes of objects. In turn, MCG provides fully-segmented candidates, in contrast to only localizing the candidates in an image window.

The results shown in this chapter are the fruit of a collaboration with Professor Jitendra Malik's computer vision group at the University of California, Berkeley; working mainly with Dr. Pablo Arbeláez. The core of this chapter has been produced by Jordi Pont-Tuset, with the advise of both Dr. Arbeláez and Prof. Marques. The hierarchies on which this part of the thesis is based were proposed by Dr. Arbeláez and the rest of Berkeley's group and are briefly described in Section 6.3. The same section evaluates them in the terms proposed in previous chapters of this thesis.

The remainder of the chapter is organized as follows. First, Section 6.2 describes the state of the art in class-independent object candidate generation. Second, Section 6.3 briefly describes and evaluates the new hierarchies, proposed by Dr. Arbeláez and the rest of Berkeley's group in the framework of our collaboration, on which MCG is based. Then, Section 6.4 explains the core of MCG, that is, how we propose a set of ranked segmented object candidates

from the varied set of hierarchies described previously. Finally, Section 6.5 presents an extensive experimental validation of MCG on the database PASCAL 2012.

6.2 State of the Art

Class-independent methods that generate object hypotheses (candidates) can be divided into those whose output is an image window and those that generate segmented candidates.

Among the former, Alexe et al. [ADF12] propose an *objectness* measure to score randomly-sampled image windows based on low-level features computed on the superpixels of [FH04]. They introduce the *superpixels straddling* measure based on [FH04] to test whether it is likely for an object to lie within a window. Our candidates are formed from combinations of superpixels, so the *straddling* is always zero.

Van de Sande et al. [vdSUGS11] present a selective window search based on segmentation. Starting with the superpixels of [FH04] for a variety of color spaces, they produce a set of segmentation hierarchies by region merging, which are used to produce a set of object candidate windows. While we also take advantage of different hierarchies to gain diversity, we leverage multiscale information rather than different color spaces. Furthermore, in contrast to [ADF12, vdSUGS11] we focus on the finer-grained task of object extraction, rather than on window selection.

Among the methods that produce segmented candidates, Carreira and Sminchisescu [CS12] hypothesize a set of placements of fore- and background seeds and, for each configuration, solve a Constrained Parametric Min-Cut (CPMC) problem to generate a pool of object hypotheses. Endres and Hoiem [EH10] base their category-independent object proposals on an iterative generation of a hierarchy of regions, based on the contour detector of [AMFM11] and occlusion boundaries of [HEH11]. Kim and Grauman [KG12] propose to match parts of the shape of exemplar objects, regardless of their class, to detected contours by [AMFM11]. They infer the presence and shape of a candidate object by adapting the matched object to the computed superpixels. The three approaches perform some type of graph-based segmentation on each of the candidates proposals, which adds a certain computational burden. Our approach avoids this computation by taking a purely combinatorial approach, which allows us to generate a much larger pool of candidates very efficiently.

Recently, two works proposed to train a cascade of classifiers to learn which sets of regions should be merged to form objects. [RS13b] produces full region hierarchies by iteratively merging pairs of regions and adapting the classifiers to different scales. [WT13] specializes the classifiers also to size and class of the annotated instances to produce object candidates. We differ from these approaches in the fact that we do not specialize to different scales or classes of objects, making us less prone to overfitting.

Malisiewicz and Efros [ME07] took one of the first steps towards combinatorial grouping, by running multiple segmenters with different parameters and merging up to three adjacent regions. Arbeláez et al. [AHG+12] took another step, by considering hierarchical segmentations at three different scales and combining pairs and triplets of adjacent regions from the two coarser scales to produce object candidates. We also perform a combinatorial merging of adjacent pairs and triplets of regions but we enrich the set of candidates by combining candidates from varied hierarchies.

A substantial difference between our approach and previous work is that, instead of relying on pre-computed hierarchies or superpixels, we propose a unified approach that produces and groups high-quality multiscale regions. With respect to the combinatorial approaches of [ME07, AHG+12], our main contribution is to develop efficient algorithms to explore a much larger combinatorial space by taking into account a set of object examples, increasing the likelihood of having complete objects in the pool of candidates. Our approach has therefore the flexibility to adapt to specific applications and types of objects, and can produce candidates at any trade-off between their number and their accuracy.

6.3 Multiscale Hierarchical Segmentation

The multiscale hierarchical segmentation algorithm proposed by Dr. Arbeláez, Prof. Malik, and their group consists in building an improved version of gPb-UCM on multiple scales of the image, aligning them, and combining them in a single multiscale hierarchy. The resulting hierarchy keeps both the small details from the hierarchies built on the higher-resolution images, while containing the coarse structures of the lower-resolution ones. Specifically, the multiscale hierarchy is built from six single-scale hierarchies coming from images at 0.25, 0.5, 0.75, 1, 1.25, and 1.5 times the original resolution. In this thesis we will refer to the multiscale hierarchy as UCB-Multi and the hierarchies at each scale as UCB-Single, to make it clear that these new hierarchies are not a contribution of this thesis and instead were proposed by the University of California, Berkeley (UCB).

The remainder of this section is devoted to evaluate these hierarchies in the terms proposed in previous chapters of this thesis, in order to validate the approach taken in the following sections. First, Figure 6.1 extends the study performed in Section 4.6 (Figure 4.12) by adding UCB-Multi. It shows the precision-recall curves from a partition perspective on BSDS500, both from the point of view of boundaries and from objects and parts. UCB-Multi (—○—) consistently outperforms the state of the art at all scales and on both measures.

From an object perspective, Figure 6.2 extends the study presented in Section 3.4 (Figure 3.5). The left-hand plot shows the upper-bound achievable quality when selecting regions directly from the hierarchies. UCB-Multi is the hierarchy with best achievable quality at all ranges, especially for more than 3-4 regions.

The right-hand plot shows the normalized quality, as the percentage of the achievable quality at the leaves that is kept at a given number of regions. In this case, ISCRA outperforms UCB-Multi, meaning that UCB-Multi starts on a partition with very good quality, but from there the quality decreases faster in percentage than that of ISCRA. In this regard, ISCRA hierarchies heavily rely on learning the hierarchy creation process, adapting the algorithm to different scales. Since we do not have access to the code but only the pre-computed hierarchies provided by the authors, we cannot evaluate whether ISCRA is over-fit on the PASCAL validation set. On the other hand, UCB-Multi has been trained on the BSDS500 training set and no further adjustment is performed.

This same behavior is also observed in Table 6.1, in terms of the number of regions to get to a certain percentage of the maximum quality, which only depends on the hierarchy structure, not on the leaves partition quality. We corroborate that ISCRA makes the most out of the original quality by needing a region less to get to the 99% of the maximum quality. The fact that UCB-Multi gets to the 90% in 3 regions corroborates a decision taken in the next section.

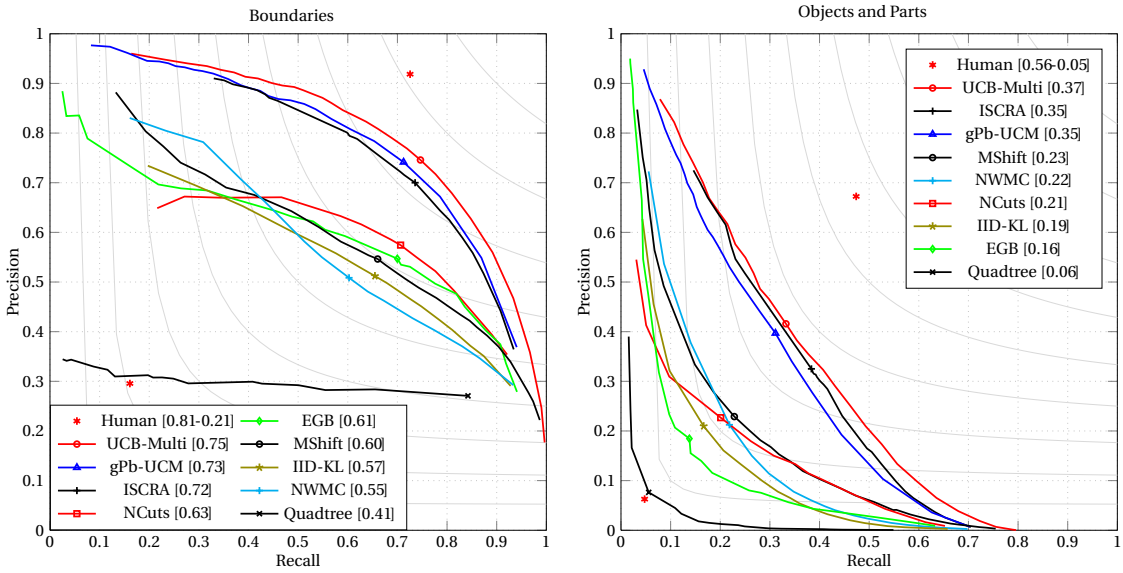


Figure 6.1: Precision-Recall curves for boundaries (left) and for objects and parts (right) on BSDS500. The curves represent the eight SoA segmentation methods and the quadtree. The marker on each curve is placed on the Optimal Dataset Scale (ODS). The isolated red asterisks refer to the human performance assessed on the same image and on a swapped image. In the legend, the F measure of the marked point on each curve is presented in brackets.

	UCB-Multi	gPb-UCM	ISCRA	NWMC	IID-KL	Quadtree	Random
Number of regions at 100%	21.33	15.98	**15.70**	21.67	22.69	27.61	45.37
Number of regions at 99%	7.47	7.21	**6.56**	9.55	10.56	12.96	14.81
Number of regions at 90%	3.03	3.21	**2.76**	4.02	5.40	7.71	11.61

Table 6.1: Mean number of regions needed to achieve 100%, 99%, or 90% of the maximum quality on a hierarchy

Let us finally evaluate the region selection from the hierarchies at multiple scales proposed. We allow the optimal region selection algorithm to get regions from each of the six single-scale hierarchies. Figure 6.3 (left) shows the mean achievable J obtained when combining regions from the six scales (——), only from UCB-Multi (——), and from each of the six scales separately (——).

First, the plot shows that UCB-Multi obtains a better result than any of the scales separately, showing that it is capable of keeping the properties from each scale. Second, combining multiple hierarchies achieves notably better results than any of the single-scale hierarchies separately, especially for a low number of regions, showing that the six techniques are indeed diverse, at the price of making the pool of regions six times larger. Despite this larger pool, however, the maximum level of quality is reached at only 17 regions, which is comparable to the original hierarchies.

With respect to combining the four state-of-the-art hierarchies as done in Section 3.4, the combination of the six scales of UCB-Single (——) outperforms the combined state-of-the-art (——) consistently at all ranges, corroborating the high diversity obtained from UCB-Single at multiple scales.

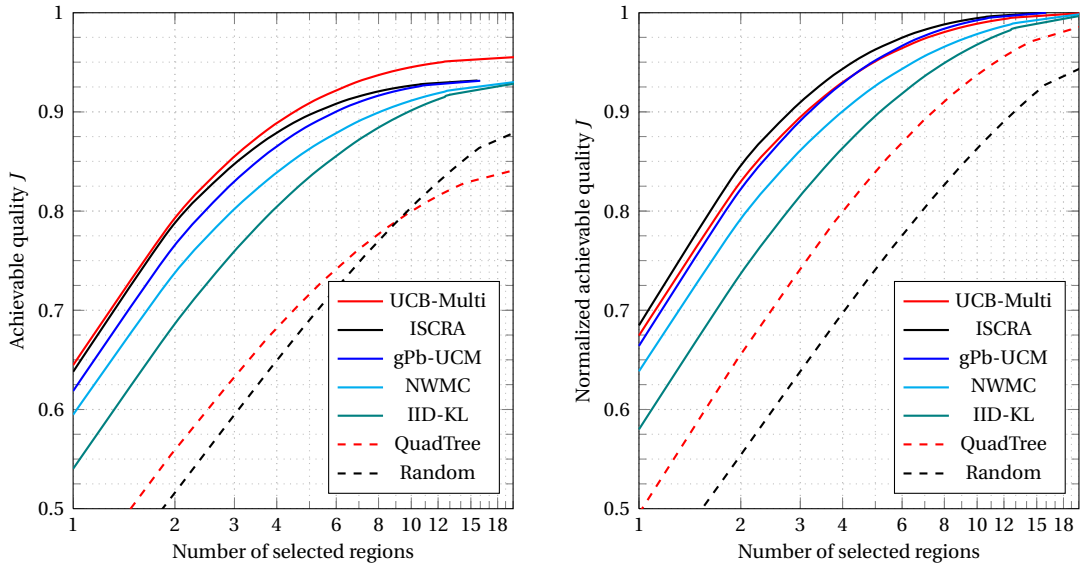


Figure 6.2: Mean achievable Jaccard index J, unnormalized (left) and normalized (right), with respect to the number of selected regions on state-of-the-art hierarchies. The number-of-regions axis is in logarithmic scale.

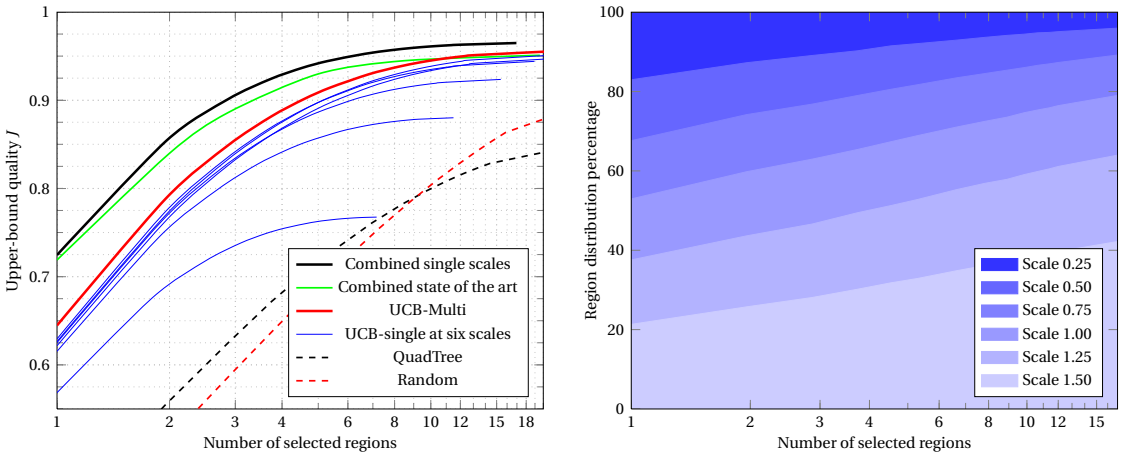


Figure 6.3: Multi-hierarchy region selection analysis. Best achievable quality when merging regions from the hierarchies at six scales to form PASCAL objects. Left: Achievable Jaccard index J with respect to the number of selected regions. Right: Selected region distribution among the six scales.

The right-hand plot of Figure 6.3 shows the percentage of regions that are selected from each of the scales, that is, it shows from which hierarchy the regions come from in the combined result (——). The plot shows that the algorithm selects regions from all scales, almost equally distributed when just one region is selected. The more regions we allow, the more of them are selected from the fine scale and the less from the coarser one, which is indeed coherent with the intuition.

To sum up, UCB-Multi obtains the best results from a partition perspective at all ranges and on both measures. At object level, it reaches the best achievable quality on a single hierarchy;

and when combining the six UCB-Single hierarchies, the achievable quality is better that the rest of the state of the art combined.

6.4 Combinatorial Grouping of Candidates

One can interpret the regions of a hierarchy as ranked candidates of objects or their parts. Ranked in the sense that the higher in the hierarchy (e.g. more UCM strength), somehow the more probable to be an object or a part; and objects and parts as shown by our newly-proposed measure that evaluates precisely this aspect.

To account for object fragmentation, that is, to recover those objects that have been divided into more than one region even in the hierarchy, one can also consider pairs and triplets of these regions as object candidates [ME07, AHG+12]. This is critical in objects with very contrasted parts, such as humans, for instance, where the trousers, the shirt, and the head usually do not have a similar color.

Given the diversity that we have shown that multiple hierarchies provide, our proposal is to create a large pool of candidates with a very high achievable quality by a combinatorial grouping of candidates from a ranked lists coming from diverse hierarchies. Specifically, we consider the singletons, pairs, and triplets of regions from the six individual scales (UCB-Single) and the multiscale hierarchy (UCB-Multi) as 21 lists of ranked candidates (recall that with triplets of regions we achieved the 90% of the maximum quality in the hierarchy).

As we will see in the experiments, the full joint set of candidates has around 15 million candidates per image and so it is highly redundant. Our next goal is therefore to reduce it, and we propose to do it by keeping only the top N_i candidates from each ranked list L_i, which results in a total number of candidates $N_c = \sum N_i$.

At training time, we would like to find, for different values of N_c, the number of candidates from each list $\overline{N}_i$ such that the joint pool of N_c candidates has the best achievable quality. We frame this learning problem as a Pareto front optimization [Ehr05] with two conflicting objective functions: number of candidates and achievable quality; the more candidates, the higher the achievable quality. In other words, we are interested in the sets of candidates that achieve a certain quality with the minimum number of candidates.

At test time, we select a working point on the Pareto front, represented by the $\{\overline{N}_i\}$ values, based either on the number of candidates N_c we can handle or on the minimum achievable quality our application needs; and we combine the $\overline{N}_i$ top candidates from each ranked list. As shown in the experiments in the following section, this allows us to reduce the initial pool of 15 million candidates to 5 thousand in exchange of only reducing the achievable quality from 0.88 to 0.84.

To further reduce the number of candidates below this point, while pushing the achievable quality as high as possible, we train a regressor to learn to rank the final set of candidates. Below we provide more details about each part of the algorithm.

6.4.1 Efficient learning

Formally, assuming R ranked lists L_i, an exhaustive learning algorithm would consider all possible values of the R-tuple $\{N_1, \ldots, N_R\}$, where $N_i \in \{0, \ldots, |L_i|\}$; adding up to $\prod_1^R |L_i|$ pa-

rameterizations to try, which is intractable in our setting.

Figure 6.4 shows the diagram of our algorithm at training time. To reduce the dimensionality of the learning step, we start by selecting two ranked lists L_1, L_2 (green curves) and we sample each list at S levels of number of candidates (green dots). We then scan the full set of S^2 different parameterizations to combine the candidates from both. Each of these sets is analyzed in the plane of number of candidates-achievable quality, so the full combination can be assessed as S^2 points in this plane (blue dots in the right-hand plot).

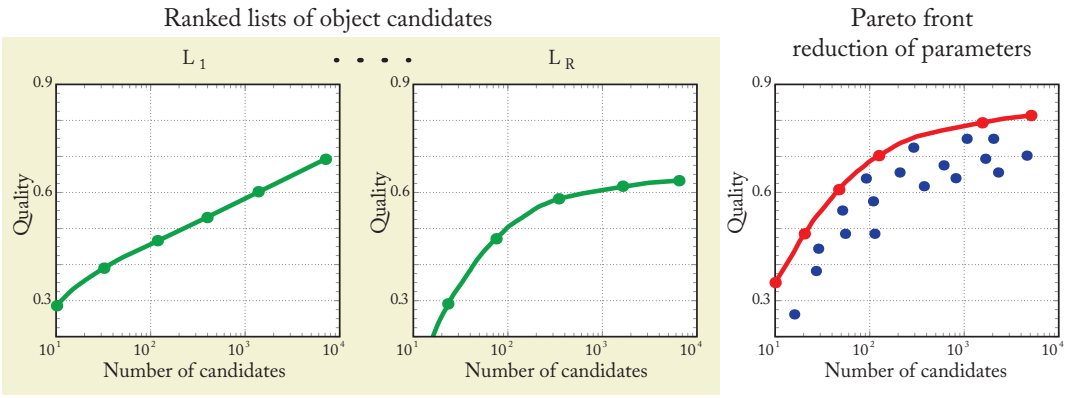


Figure 6.4: Training the combinatorial generation of candidates using the Pareto front: two ranked lists

The key step of the optimization consists in discarding those parameterizations whose quality point is not in the Pareto front (red curve), i.e., those parameterizations that can be substituted by another with better achievable quality with the same number of candidates, or by one with the same achievable quality with less candidates. As in the original ranked lists, we sample the Pareto front to S points, each of which correspond to a different set of candidates. The S points in the Pareto front can therefore be combined as the S point in a ranked list, so we can iteratively combine a new ranked list to the resulting Pareto points.

Figure 6.5 shows the general diagram of the process. We iterate the process until all the ranked lists are combined, and at the end each point in the final Pareto front corresponds to a particular parameterization $\{N_1,\ldots,N_R\}$. We test different orders in the combination of the ranked lists but not significant difference is observed. The final Pareto curve provides a set of solutions of the trade-off between number of candidates and achievable quality. At test time, we choose a point on this curve (black star), either at a given number of candidates N_c or at the achievable quality we are interested in, and combine the $\{\overline{N}_1,\ldots,\overline{N}_R\}$ top candidates from each ranked list. The number of sampled configurations using the proposed algorithm is $(R-1)S^2$, that is, we have reduced an exponential problem (S^R) to a quadratic one. As it will be shown in the experiments, this reduction makes the problem feasible while not reducing the achievable quality significantly.

6.4.2 Regressed Ranking of Candidates

Up until this point we have tackled the reduction of candidates (from millions to thousands in practice) while keeping the achievable quality as high as possible. To further reduce their number below this point while keeping more quality, we deduplicate candidates and we train a regressor from low-level features.

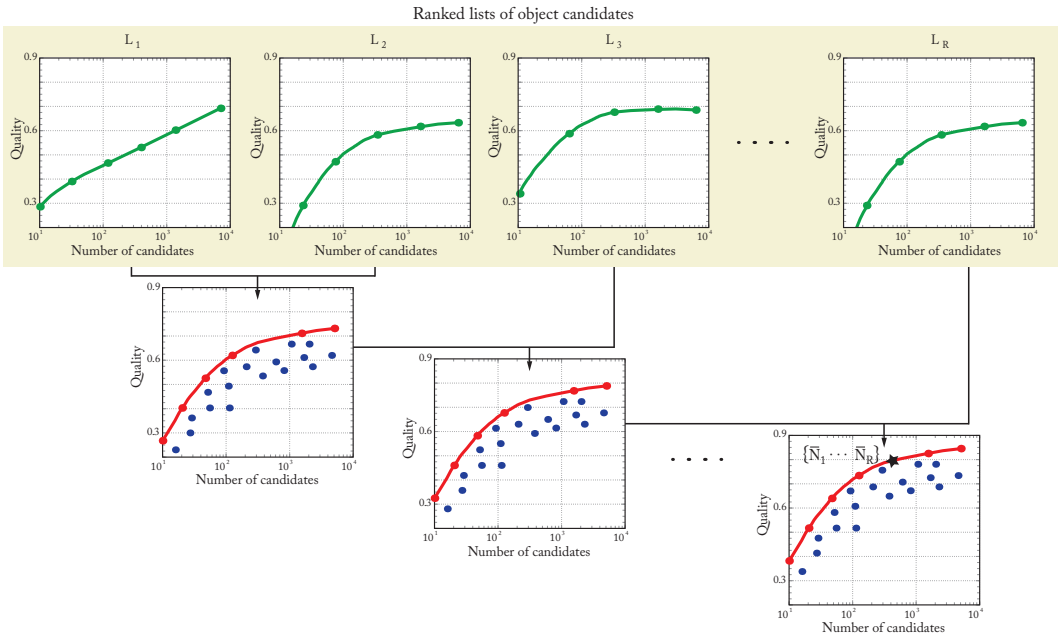


Figure 6.5: Training the combinatorial generation of candidates using the Pareto front: iterative combination

More specifically, at training time we gather the candidates at the selected Pareto front point and we filter them by overlap, that is, we discard all those candidates whose overlap with another one is above a certain threshold. Once deduplicated, we compute the best overlap of each candidate with the ground-truth objects, which we aim at learning from low-level features.

Since the candidates proposed are all formed by a set of regions from a hierarchy, we focus on features that can be computed efficiently in a bottom-up fashion. This way, we can precompute all the features for the original regions and efficiently calculate the features for the candidates in a few operations. We compute the following features:

- **Size and location**: Area and perimeter of the candidate; area, position, and aspect ratio of the bounding box; and the area balance between the regions in the candidate.

- **Shape**: Perimeter (and sum of contour strength) divided by the squared root of the area; and area of the region divided by that of the bounding box.

- **Contours**: Sum of contour strength at the boundaries, mean contour strength at the boundaries; minimum and maximum UCM threshold of appearance and disappearance of the regions forming the candidate.

Once we have the features for all candidates and the quality with respect to the ground truth, we train a Random Forest using these features to regress the object overlap with the ground truth. We tune the Random Forest learning on half training set and validate on the other half. For the final results, we train on the training set and evaluate our candidates on the validation set of PASCAL 2012.

At test time, if we let the ranking be directly the regressor output, we would have many very similar candidates with high ranking. In order to diversify the candidates spatially, that is, in

order not to have many similar objects in correlative positions, we use the Maximum Marginal Relevance measures as in [CS12]. Intuitively, we iteratively add new candidates by balancing their good regressed quality and their low overlap with already added candidates.

6.5 Experimental Results

Evaluation Measures: We assess the generation of candidates in terms of achievable quality with respect to the number of candidates proposed, that is, the quality we would have if an oracle selected the best candidate among the pool of object hypotheses. As a measure of quality of a specific candidate with respect to an annotated object, we will consider the Jaccard index J, as in Part I of this thesis. The experiments are performed on the segmentation challenge of PASCAL 2012, which consists of 2913 images with 6934 segmented objects in the training and validation set.

When computing the overall quality for the whole database, we will consider two metrics. First, we define the Jaccard index at class level (J_c) as the mean over classes of the Jaccard index of all pixels of each class (the segmentation accuracy of PASCAL). In other words, we consider all pixels of all objects of a given class as a single *inter-image* object and compute the per-class Jaccard index. Second, to avoid the bias of J_c towards methods focusing on large objects, we define the Jaccard index at object level (J_o) as the mean best Jaccard index for all the objects in the database (also Best Spatial Support score (BSS) [ME07]).

Learning Strategy Evaluation: This section estimates the loss in performance due to not sweeping all possible values of $\{N_1,\dots,N_R\}$ via the efficient learning strategy proposed. To do so, we will compare it with the full combination on a reduced problem to make the latter feasible. Specifically, we combine the 7 ranked lists coming from the singletons at all scales, instead of the full 21 lists, and we limit the search to 20 000 candidates, allowing us to discard all those solutions above this value, and thus speeding the process up.

In this situation, the mean loss in achievable quality along the full curve of parameterization is $J_o = 0.0002$, with a maximum loss of $J_o = 0.004$ (0.74%). In exchange, our proposed learning strategy on the full 21 ranked lists takes about 5 minutes to compute on the training set of PASCAL 2012, while the limited full combination takes 4 days. We estimate that the complete full combination would take months.

Combinatorial Grouping: We extract the lists of candidates from the six scales and the multiscale hierarchy, for singletons, pairs, and triplets of regions, leading to $R = 21$ lists, ranked by the minimum UCM strength of the regions forming each candidate.

Table 6.2 shows the number of candidates and achievable quality when combining the full 7 ranked lists on up to 1, 2, or 3 regions per candidate. Both the achievable quality and the number of candidates notably increase when considering pairs and triplets, proving that the original hierarchies indeed over-segment some objects.

Figure 6.6 shows the Pareto front evolution of J_c with respect to the number of candidates for up to 1, 2, and 3 regions per candidate (7, 14, and 21 lists, respectively). In other words, each point of these curves represents the quality of the resulting set of merging the top $\{N_1,\dots,N_{21}\}$ candidates from each hierarchy. As baselines, we plot the raw singletons from UCB-Multi, gPb-UCM, and Quadtree.

The improvement of considering the combination of all 1-region candidates from the 6 UCB-Single scales and the UCB-Multi (——) with respect to the raw UCB-Multi (——) is significant,

Number of regions per cand.	1	≤2	≤3
Number of candidates (N_c)	21 775	352 022	15 211 858
Jaccard at object level (J_o)	0.73	0.84	0.88
Jaccard at class level (J_c)	0.75	0.85	0.89

Table 6.2: Maximum achievable quality and number of candidates per image on PASCAL 2012 segmentation training set for the full combination of up to $n = 1,2,3$ regions per candidate

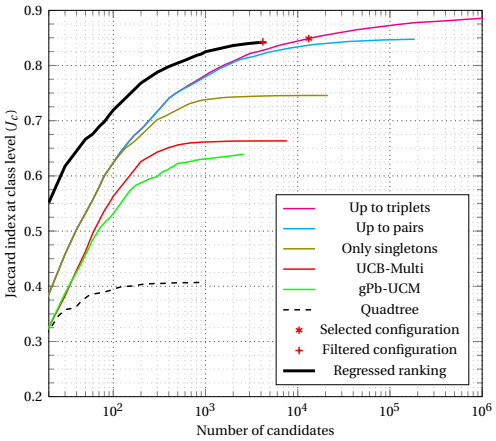


Figure 6.6: Object candidates achievable quality on PASCAL 2012 training set

which corroborates the diversity obtained from hierarchies at different scales. In turn, the addition of 2- and 3-region candidates noticeably improves the achievable quality, which is coherent with the upper-bound evaluation presented in Chapter 3.

The red asterisk (*) marks the selected configuration $\{\overline{N}_1,\ldots,\overline{N}_{21}\}$ we choose (black star in Figure 6.5), and the red plus sign (+) represents the set of candidates after de-duplicating those pairs whose overlap is above $J = 0.95$. The candidates at this point (3 620 in mean with $J_c = 0.84$) are the ones that are ranked by the learnt regressor, the result of which is plotted in black (——).

In summary, the full set of candidates (i.e., combining the full 21 lists) contains 15 211 858 candidates per image, achieving a quality of $J_c = 0.88$ in the validation set of PASCAL 2012. MCG allows us to reduce the number of candidates to 3 536, in exchange of decreasing the achievable J_c to only 0.84. The regressed ranking allows us to further reduce the number of candidates below this point while keeping the achievable quality as high as possible.

Comparison with State of the Art: We compare our results against [WT13, ADF12, KG12, CS12, AHG+12, vdSUGS11, EH10], using the implementations from the respective authors. Figures 6.7 and 6.8 show the achievable quality of all methods on the validation set of the segmentation task of PASCAL 2012 and PASCAL 2010, respectively. Please note that the techniques that provide a ranked set of candidates are represented as a line, where each point represents the set formed by the *nc* first candidates in the ranking; while the techniques providing a set of candidates without a ranking are represented as a single point.

The plot on PASCAL 2010 is included for us to be able to compare to Scapel [WT13], given that the latter is not reproducible on PASCAL 2012 due to the lack of some needed window loca-

tions. We plot the raw regions of UCB-Multi, ISCRA, gPb-UCM, and QuadTree as baselines. To compare fairly with Selective Search [vdSUGS11], we adapted their boxes to the multiscale hierarchy by computing the deepest subtree included in the window.

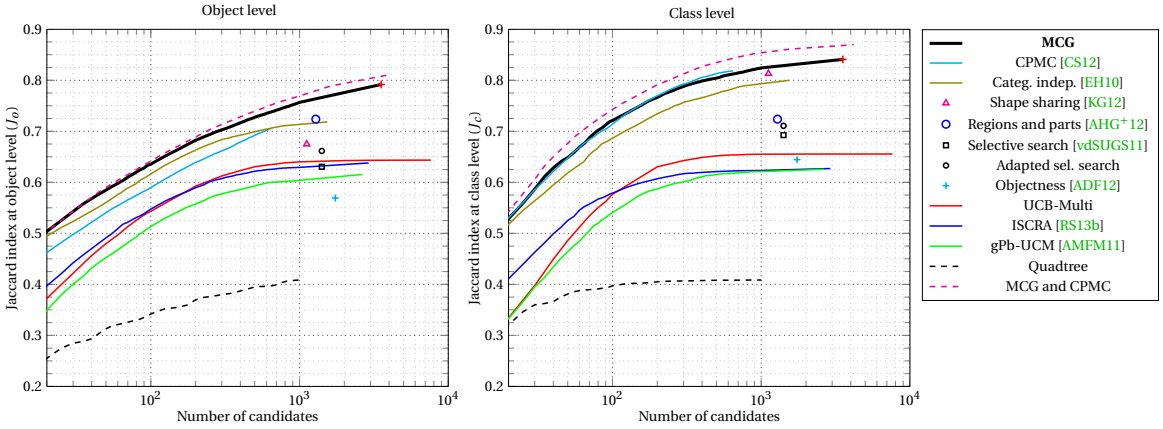


Figure 6.7: Achievable quality with respect to the number of candidates on PASCAL VOC12

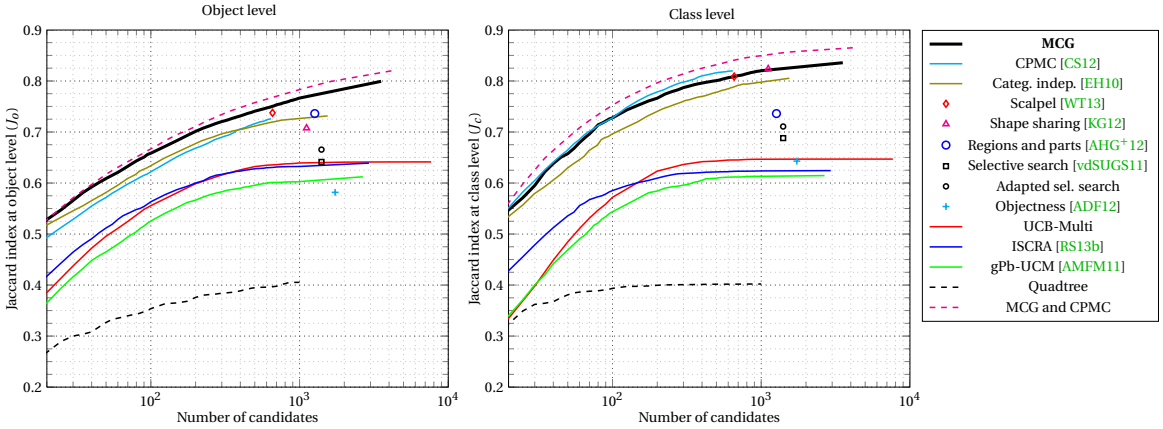


Figure 6.8: Achievable quality with respect to the number of candidates on PASCAL VOC10, in order to include Scalpel [WT13]

At object level (J_o), MCG candidates outperform the state of the art at all regimes, while at class level (J_c), MCG practically achieves the same quality than CPMC. The loss of competitiveness of CPMC (and shape sharing) at object level shows that they are biased toward large objects, since missing small objects practically do not affect J_c; while J_o gives the same weight to all objects, regardless of their size. In contrast, Regions and Parts [AHG+12] or the raw UCB-Multi and ISCRA cover all sizes uniformly, since they do not lose performance from J_c to J_o.

In sight of the comparison with CPMC, one may argue that MCG is a refined version of CPMC *just* improving on small objects. To rebut this statement, we evaluate the complementarity between MCG and CPMC by computing the Pareto front of combining the two sets of ranked candidates; that is, we evaluate the sets of candidates corresponding to combining some candidates from MCG and CPMC. The curve obtained (dashed magenta - - -), shows a significant improvement, being of around $J_c = 0.025$ at 650 candidates per image. In other words, MCG candidates improve CPMC not only by detecting the small objects but also by proposing better candidates than CPMC in some large objects.

To further show the complementarity of both methods, Figure 6.9 shows a scatter plot of the quality of all objects in PASCAL 2012 validation set, where we observe the lack of a strong correlation between the quality of both methods on each object. Furthermore, objects in the upper triangle are more common than in the lower triangle, meaning that it is more common that MCG achieves a better quality than CPMC. In particular, there are not many objects missed by MCG and correctly covered by CPMC.

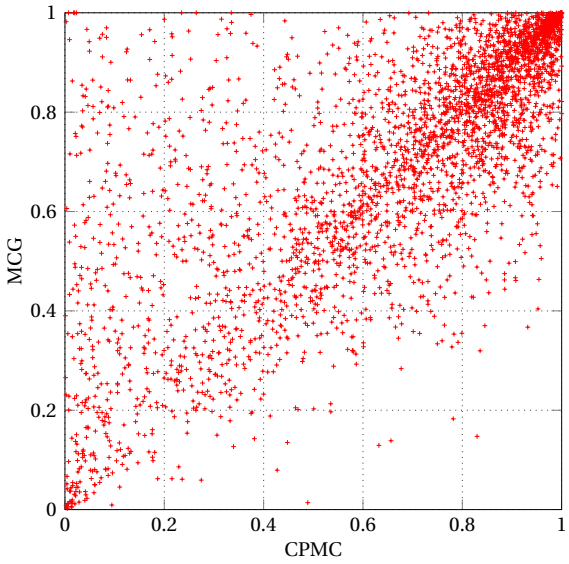


Figure 6.9: MCG vs CPMC: Comparison of object candidates achievable quality among 300 candidates on the 3 427 objects of PASCAL VOC12 segmentation validation set. Pearson correlation coefficient: 73.7%.

To extend the comparison with the state of the art, Table 6.3 shows the quality (J_o) on each of the 20 PASCAL classes at two different number of candidates (100 and 1100), comparing MCG with those state-of-the-art techniques that are relevant at each number of candidates. MCG outperforms all techniques on the two regimes at the global level and on the majority of classes.

	N_c	Plane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	MBike	Person	Plant	Sheep	Sofa	Train	TV	Global
MCG	1100	**80.8**	47.7	**85.9**	**77.9**	**68.4**	78.9	**67.9**	**91.4**	**68.5**	**86.0**	79.4	**87.7**	**80.8**	73.1	**73.1**	**71.0**	**83.8**	83.7	78.8	**86.3**	**76.0**
[EH10]	1100	75.1	**49.1**	80.7	68.8	62.8	76.4	63.3	89.4	64.6	83.0	80.3	83.7	78.4	**78.0**	66.9	66.2	69.5	82.0	**84.3**	81.8	71.6
[AHG+12]	1100	74.4	46.6	80.5	69.4	64.6	73.5	61.2	89.0	65.1	80.5	78.4	85.2	77.2	70.6	67.9	68.8	73.5	81.6	75.8	82.0	71.4
[KG12]	1100	73.8	40.6	75.8	66.7	52.7	**79.7**	50.6	91.2	59.2	80.2	**80.7**	87.4	79.0	74.7	62.1	54.6	65.0	**84.6**	82.4	79.5	67.4
[vdSUGS11] ad.	1100	68.3	39.6	70.6	64.8	58.0	68.2	51.8	77.6	58.2	72.6	70.4	74.0	66.2	59.9	59.8	55.4	67.7	71.3	68.6	78.7	63.1
[vdSUGS11]	1100	55.9	34.6	56.0	62.0	55.1	76.2	50.4	70.7	52.4	67.7	65.0	67.4	58.2	59.9	55.2	53.5	59.5	68.5	70.8	85.7	58.9
MCG	100	72.0	40.4	**78.5**	**64.7**	**54.7**	**69.0**	47.5	**85.0**	53.2	**75.2**	**68.2**	**82.5**	**73.6**	64.2	**58.9**	52.7	**72.6**	**73.1**	68.9	**73.3**	**63.6**
[EH10]	100	70.6	**40.8**	74.8	59.9	49.6	65.4	**50.4**	81.5	**54.5**	74.9	68.1	77.3	69.3	**66.8**	56.2	**54.3**	64.1	72.0	71.6	69.9	61.7
[CS12]	100	**72.7**	36.2	73.6	63.3	45.4	67.4	39.5	84.1	47.7	73.2	64.0	81.1	72.2	64.3	52.8	42.9	62.2	72.9	**74.3**	69.5	59.0

Table 6.3: Per-class and global Jaccard index at object level (J_o) at 1100 (upper part) and 100 (lower part) candidates per image. Class-level (J_c) results show an equivalent behavior.

MCG Candidates as Object Windows: The previous evaluation methodology mixes how well MCG localizes the potential objects as well as how good the delineated shapes are. This section evaluates MCG purely in terms of object localization. To do so, we transform our candidates to image windows by computing their bounding boxes, and evaluate how well they overlap with the bounding boxes of the ground-truth objects. This way we are comparing fairly against the methods whose output are image windows, as Selective Search [vdSUGS11].

Figure 6.10 shows the percentage of ground-truth windows whose overlap (Jaccard index) with one of the proposed candidates is above a certain threshold. MCG outperforms Selective Search at all levels of overlap at the same number of candidates per image (1411 c/i). On our full set of candidates per image (3536 c/i), the results are even better at all ranges.

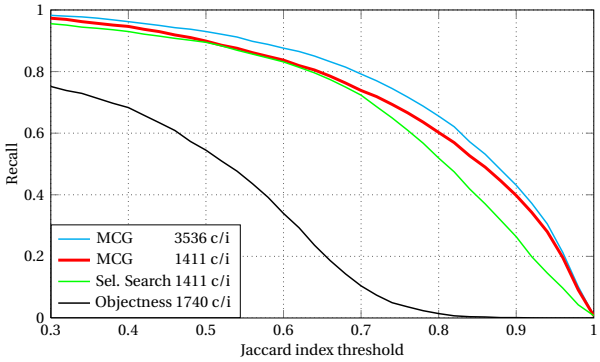


Figure 6.10: MCG as bounding boxes: each candidate and each object in the ground truth are transformed to its bounding box. Percentage of ground-truth bounding boxes that we would *detect* (recall) at different thresholds of Jaccard index J.

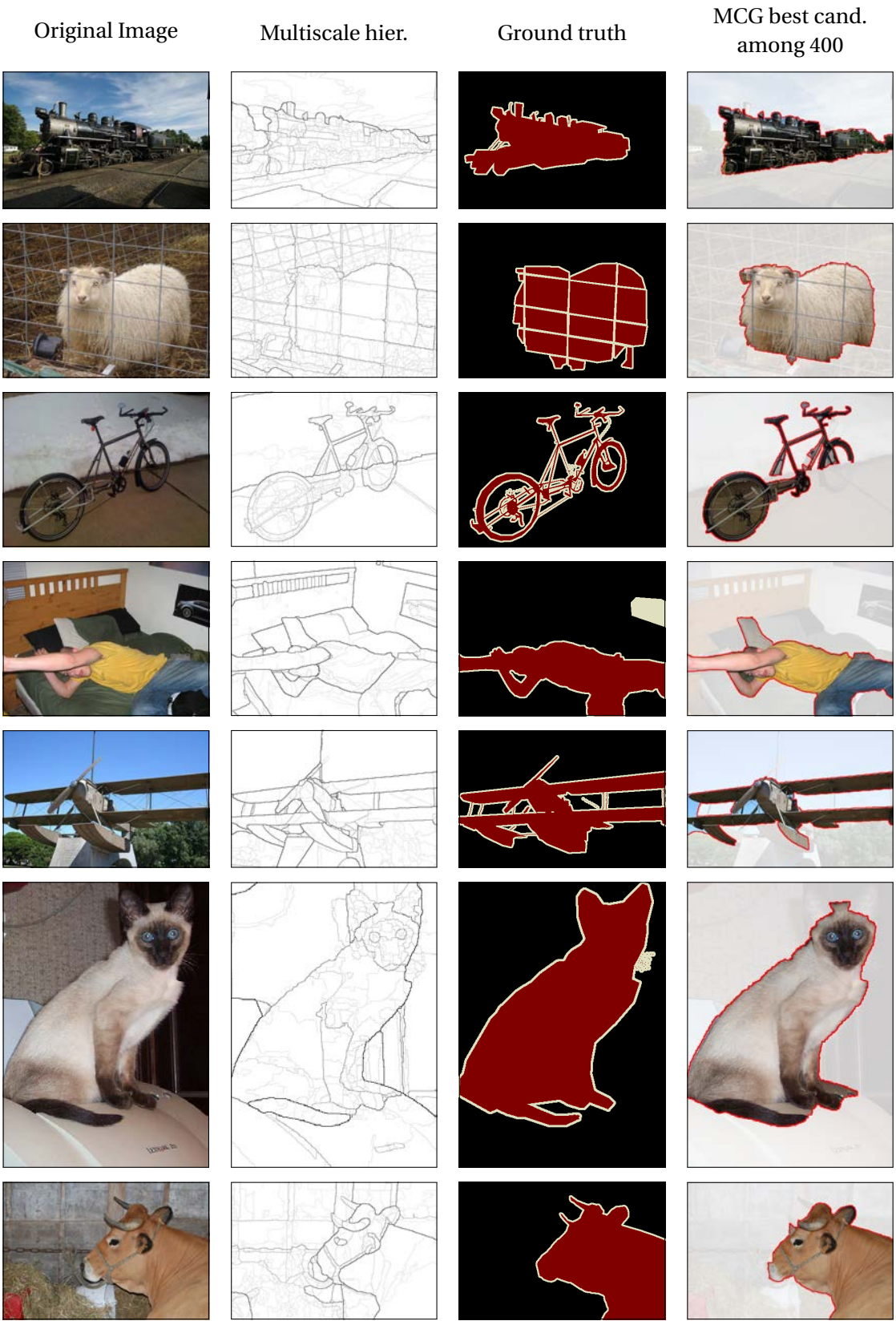
Table 6.4 shows the per-class localization results, which shows that MCG outperforms selective search in terms of object localization in 15 out of 20 classes. Please note that this per-class table is computed for a threshold of 0.5, which, as seen in Figure 6.10, is precisely the one for which selective search is closer to MCG.

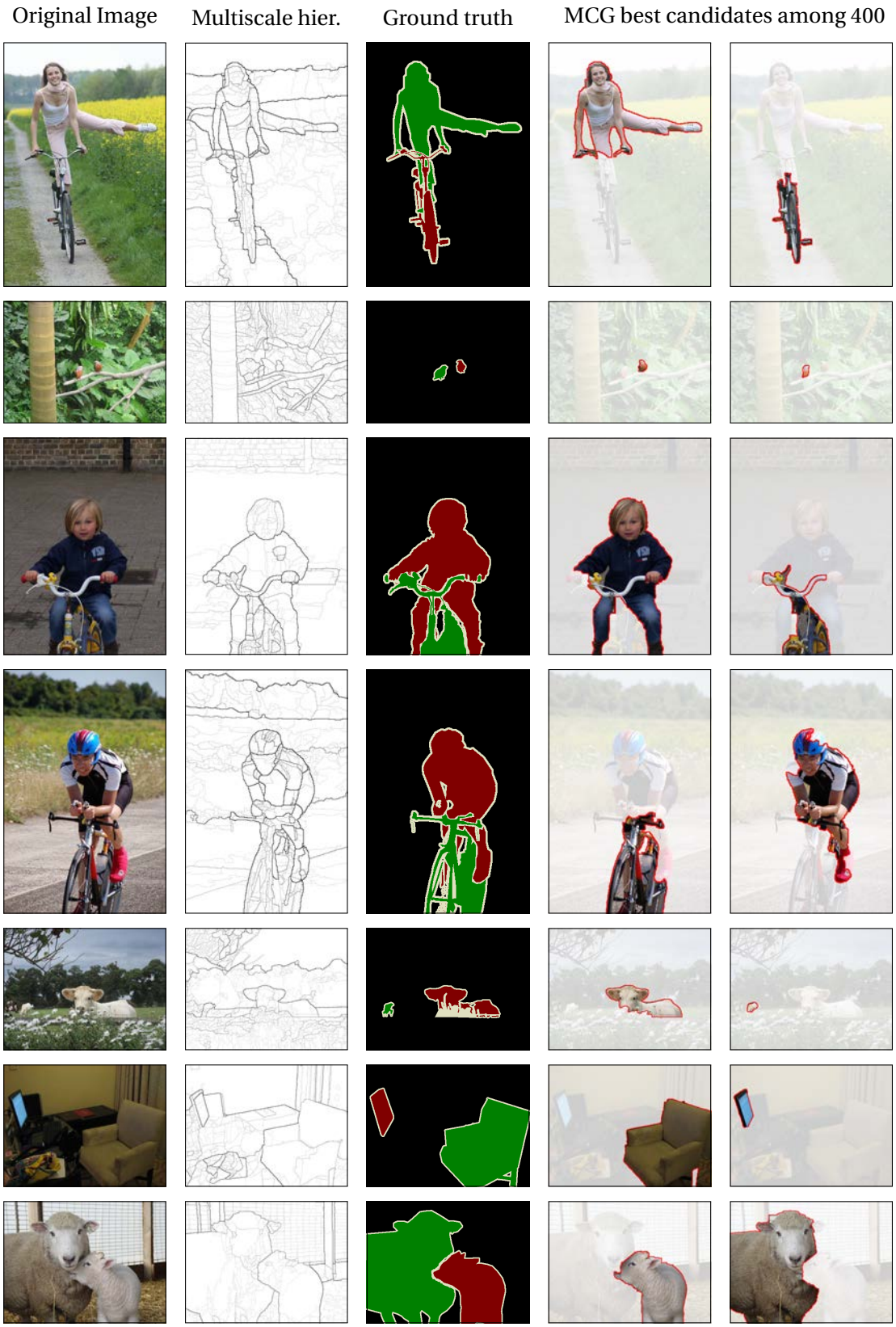
	N_c	Plane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	MBike	Person	Plant	Sheep	Sofa	Train	TV	Global
MCG	3537	97.3	96.1	96.4	93.5	74.1	97.4	78.7	100.0	96.8	98.5	100.0	100.0	99.0	100.0	89.7	93.6	92.2	100.0	100.0	100.0	93.0
MCG	1411	**97.3**	92.2	**96.4**	**92.6**	**67.9**	**95.7**	**72.3**	**100.0**	92.3	**97.7**	**100.0**	**100.0**	**98.1**	99.0	**85.4**	86.0	**90.8**	97.2	**98.9**	**98.0**	**89.9**
[vdSUGS11]	1411	95.5	**93.2**	95.7	87.0	67.9	94.8	72.3	100.0	**95.5**	97.0	98.8	100.0	96.2	**100.0**	84.6	**90.1**	83.7	**100.0**	97.8	96.9	89.5
[ADF12]	1740	45.5	65.0	64.3	44.4	43.8	62.9	35.3	66.7	53.0	54.5	50.0	74.0	74.0	69.9	52.3	48.0	52.9	57.5	58.1	57.1	54.5

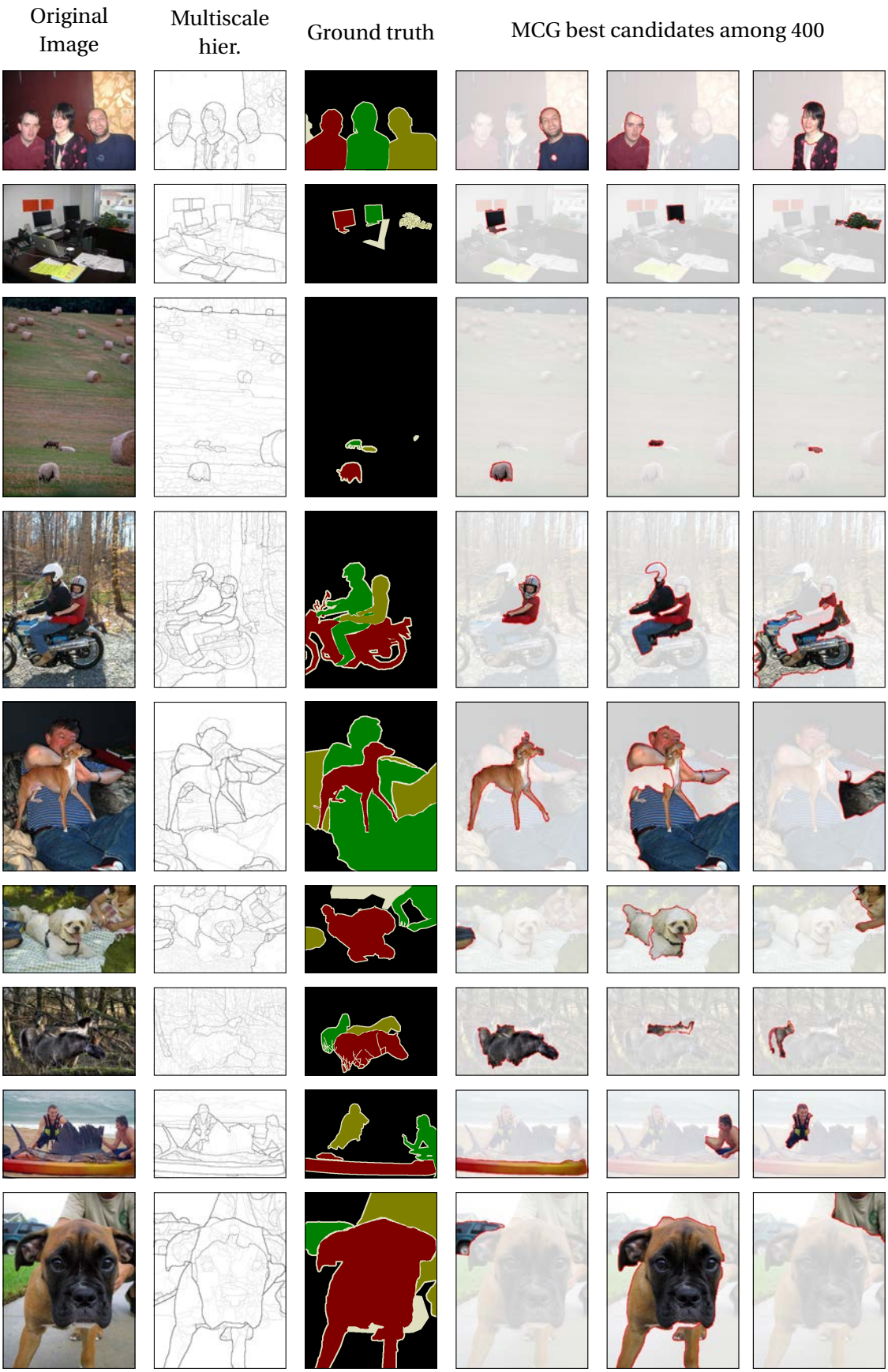
Table 6.4: MCG as bounding boxes: Per-class and global recall at Jaccard index 0.5

MCG Speed: The combinatorial grouping takes 12±2 seconds to generate 3536 candidates per image on the validation set of PASCAL 2012, limited to one thread. Taking advantage of the hierarchical structure used is at the core of such a good performance. The equivalent process of CPMC to generate 650 candidates on the same machine takes 92±44 s.

MCG Qualitative Results: The following pages show some qualitative results of MCG. For some images on PASCAL 2012 validation set, we show the original image, the UCB-Multi hierarchy in which we base the generation of candidates, the ground-truth objects on that image, and the best candidate among a pool of 400 candidates for each object.


Original Image
Multiscale hier.
Ground truth
MCG best cand. among 400


Original Image
Multiscale hier.
Ground truth
MCG best candidates among 400


Original Image
Multiscale hier.
Ground truth
MCG best candidates among 400

7 Conclusions and Future Work

The first part of this Thesis has been devoted to analyze the assessment of image segmentation algorithms from an object perspective. To compare a partition or a hierarchy against an object ground-truth, we have evaluated the upper-bound performance that can be reached by selecting regions from them. We have reviewed how previous algorithms looked for this best representation and prove that none of them could find the actual optimum. To do so, we have modeled the problem mathematically as a Linear Fractional Combinatorial Optimization (LFCO) problem and solved it via successive linear optimization problems on iterations of the Newton algorithm.

We have conducted extensive experimental validation on the Pascal 2012 segmentation task and the DCU database. We have performed the region selection algorithms on a representative set of state-of-the-art algorithms, both flat and hierarchical, as well as on baseline segmentations to normalize the performance. Comparing the region selection techniques, we have shown that the non-optimal techniques in the literature fail at finding the upper-bound performance in a significant number of cases, and we have further analyzed the type of differences obtained.

From the point of view of comparing segmentation techniques, we have shown that hierarchies are capable of representing the objects with much less number of regions for the same achievable quality, or achieving a much better result with the same number of regions, than the flat segmentation techniques. We have also shown that selecting regions from a varied set of hierarchies can further improve the results significantly. We plan to submit the findings of this part of the Thesis on the upcoming European Conference on Computer Vision (ECCV) 2014.

The second part of the Thesis has been focused on the evaluation of image segmentation from a partition perspective. We have first extensively reviewed the state of the art in similarity measures, by structuring them into three interpretations of an image partition, deduplicating some previous work and homogenizing notation. We have presented a new precision-recall measure, filling the gaps we have noticed in the measure structure.

To compare the quality of the measures, we have proposed a set of qualitative and quantitative meta-measures, based on extensive ground-truth datasets and a large representation of state-of-the-art segmentation techniques. We have shown that the pair of precision-recall measures, both boundary-based and the newly proposed for objects and parts, outperforms the rest of the measures in terms of the presented meta-measures.

We have compared the state-of-the-art in terms of these two proposed measures, which has allowed us to further study the behavior, properties, and differences between the two selected measures. We have made publicly available a code to easily evaluate image segmentation in

terms of all the state-of-the-art evaluation measures. These findings were published in:

- J. Pont-Tuset and F. Marques, *Measures and meta-measures for the supervised evaluation of image segmentation*, in Computer Vision and Pattern Recognition (CVPR), 2013.

We have then extended some of the studied measures to hierarchies by finding the best represented partition in it in terms of a given measure. A first approach consists in extending the directional Hamming distance, which although not being among the best ones in terms of meta-measures, can be extended by means of dynamic programming algorithms, as published in:

- J. Pont-Tuset and F. Marques, *Upper-bound assessment of the spatial accuracy of hierarchical region-based image representations*, in International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2012.

The next step has been to extend the two best-performing measures. We have first focused on the boundary-based precision-recall curves, which has entailed an LFCO model with multi-matching constraints. We have experimentally shown that finding the upper-bound measure in this case better reflected the quality of the hierarchies and that the measure behaved more accordingly to the intuition. This extension was published in:

- J. Pont-Tuset and F. Marques, *Supervised assessment of segmentation hierarchies*, in European Conference on Computer Vision (ECCV), 2012.

We have also found the upper-bound boundary-based precision-recall measure among all partitions that can be formed by merging super-pixels, which ended in a model that iteratively adds cycle constraints. This model is not efficient enough to be applied to realistic partitions with a high number of superpixels, so the future work in this regard is to improve the algorithm or to tackle the problem from another type of mathematical model.

We have finally extended the proposed precision-recall for objects and parts to hierarchies but, again, the model using combinatorial optimization is not well suited for hierarchies of realistic sizes. Future work can also be focused on finding new approaches to allow us to generalize the new measure to hierarchies.

The final part of this Thesis has been devoted to the application of image segmentation to object detection, and in particular to the generation of segmented object candidates. The algorithm takes the hierarchies computed at multiple scales to gain diversity, and combines singletons, pairs, and triplets of regions from all of them. It then learns the optimal way of combining the candidates from each hierarchy in a Pareto optimization framework and it finally ranks the candidates using low- and mid-level features to produce a final ranked list of candidates.

We have performed an extensive validation against an exhaustive representation of the state-of-the-art and have shown that our candidates outperform previous methods in a variety of measures and points of view. We have submitted the findings of this final part to the upcoming conference in Computer Vision and Pattern Recognition (CVPR) 2014.

The first steps we performed on using segmentation hierarchies were published in:

- J. Pont-Tuset and F. Marques, *Contour detection using binary partition trees*, in International Conference on Image Processing (ICIP), 2010.

We have also collaborated with other researchers in close topics to this thesis and published

the results in the following conferences and journals:

- X. Giró-i-Nieto, M. Martos, E. Mohedano, and J. Pont-Tuset, *From global image annotation to interactive object segmentation*, in Multimedia Tools and Applications, 2013.
- X. Giró-i-Nieto, C. Ventura, J. Pont-Tuset, S. Cortes, and F. Marques, *System architecture of a web service for content-based image retrieval*, in ACM International Conference on Image and Video Retrieval, 2010.

Bibliography

[ADF12] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari, *Measuring the objectness of image windows*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **34** (2012), 2189–2202. 108, 116, 117, 119

[AF99] Anthony Almudevar and Chris Field, *Estimation of single-generation sibling relationships based on DNA markers*, Journal of Agricultural, Biological, and Environmental Statistics **4** (1999), no. 2, 136–165. 58

[AGBB12] S. Alpert, M. Galun, A. Brandt, and R. Basri, *Image segmentation by probabilistic bottom-up aggregation and cue integration*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **34** (2012), no. 2, 315 –327. 17, 24

[AHG+12] P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik, *Semantic segmentation using regions and parts*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012. 64, 108, 109, 112, 116, 117, 118

[AMFM11] Pablo Arbeláez, Michael Maire, Charless C. Fowlkes, and Jitendra Malik, *Contour detection and hierarchical image segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **33** (2011), no. 5, 898–916. 3, 6, 7, 8, 13, 19, 20, 28, 40, 51, 53, 57, 63, 64, 68, 69, 72, 73, 77, 83, 89, 90, 95, 108, 117

[ASS+10] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk, *SLIC superpixels*, Tech. Report 149300, EPFL, 2010. 95, 98

[Bag11] Shai Bagon, http://www.wisdom.weizmann.ac.il/ bagon/matlab.html, 2011. 13

[BHEG02] Asa Ben-Hur, Andre Elisseeff, and Isabelle Guyon, *A stability based method for discovering structure in clustered data*, Pacific Symposium on Biocomputing, 2002, pp. 6–17. 61

[BKD01] Kevin Bowyer, Christine Kranenburg, and Sean Dougherty, *Edge detector evaluation using empirical ROC curves*, Computer Vision and Image Understanding **84** (2001), no. 1, 77–103. 63

[BT10] Daniel Berend and Tamir Tassa, *Improved bounds on bell numbers and on moments of sums of random variables*, Probability and Mathematical Statistics **30** (2010), no. 2, 185–205. 78

[CCR05] J. S. Cardoso and L. Corte-Real, *Toward a generic evaluation of image segmentation*, IEEE Transactions on Image Processing **14** (2005), no. 11, 1773–1782. 6, 53, 54, 57, 58, 64, 68

[CCTCR09] Jaime S. Cardoso, Pedro Carvalho, Luís F. Teixeira, and Luís Corte-Real, *Partition-distance methods for assessing spatial segmentations of images and videos*, Computer Vision and Image Understanding **113** (2009), no. 7, 811 – 823. 6

[CGM02] C.M. Christoudias, B. Georgescu, and P. Meer, *Synergism in low level vision*, IEEE International Conference on Pattern Recognition, 2002. 13, 28

[CLRS01] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to algorithms*, The MIT Press, 2001. 79

[CM02] D. Comaniciu and P. Meer, *Mean shift: a robust approach toward feature space analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **24** (2002), no. 5, 603 –619. 13, 28, 72

[CM10] F. Calderero and F. Marques, *Region merging techniques using information theory statistical measures*, IEEE Transactions on Image Processing **19** (2010), no. 6, 1567–1586. 13, 28, 40, 73, 77, 90

[CP03] P.L. Correia and F. Pereira, *Objective evaluation of video segmentation quality*, IEEE Transactions on Image Processing **12** (2003), no. 2, 186–200. 20

[CS12] Joao Carreira and Cristian Sminchisescu, *CPMC: Automatic object segmentation using constrained parametric min-cuts*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **34** (2012), no. 7, 1312–1328. 108, 115, 116, 117, 118

[Cze13] Jan Czekanowski, *Zarys metod statystycznych w zastosowaniu do antropologii [An outline of statistical methods applied in anthropology]*, Prace Towarzystwa Naukowego Warszawskiego **5** (1913). 19

[Dic45] Lee R. Dice, *Measures of the amount of ecologic association between species*, Ecology **26** (1945), no. 3, pp. 297–302. 19

[Din67] W. Dinkelbach, *On nonlinear fractional programming*, Management Science **13** (1967), no. 3, 492–498. 26

[Don00] Stijn Dongen, *Performance criteria for graph clustering and markov cluster experiments*, Tech. Report INS-R0012, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, The Nederlands, 2000. 53, 57, 64

[EDI02] EDISON code, http://coewww.rutgers.edu/riul/research/code/EDISON/index.html, 2002. 13

[EH10] Ian Endres and Derek Hoiem, *Category independent object proposals*, European Conference on Computer Vision (ECCV), 2010. 108, 116, 117, 118

[Ehr05] Matthias Ehrgott, *Multicriteria optimization*, Springer, 2005. 112

[EJ09] Francisco J. Estrada and Allan D. Jepson, *Benchmarking image segmentation algorithms*, International Journal on Computer Vision (IJCV) **85** (2009), 167–181. 63

[EMT02] Mark Everingham, Henk Muller, and Barry Thomas, *Evaluating image segmentation algorithms using the pareto front*, European Conference on Computer Vision (ECCV), 2002. 17

[EVGW+] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*, http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 10

[EVGW+10] ———, *The PASCAL Visual Object Classes Challenge 2010 Results*, http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html, 2010. 20

[FGMR10] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan, *Object detection with discriminatively trained part-based models*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **32** (2010), no. 9, 1627–1645. 53

[FH04] Pedro F. Felzenszwalb and Daniel P. Huttenlocher, *Efficient graph-based image segmentation*, International Journal of Computer Vision **59** (2004), 2004. 13, 28, 72, 108

[FMnR+02] Jordi Freixenet, Xavier Muñoz, D. Raba, Joan Martí, and Xavier Cufí, *Yet another survey on image segmentation: Region and boundary information integration*, European Conference on Computer Vision (ECCV), 2002. 56

[GAV+08] Lutz Goldmann, Tomasz Adamek, Peter Vajda, Mustafa Karaman, Roland Mörzinger, Eric Galmar, Thomas Sikora, Noel E. O'Connor, Thien Ha-Minh, Touradj Ebrahimi, Peter Schallauer, and Benoit Huet, *Towards fully automatic image segmentation evaluation*, International Conference on Advanced Concepts for Intelligent Vision Systems, 2008, pp. 566–577. 17, 24

[Gus02] Dan Gusfield, *Partition-distance: A problem and class of perfect graphs arising in clustering*, Information Processing Letters **82** (2002), no. 3, 159–164. 58

[GWL06] Feng Ge, Song Wang, and Tiecheng Liu, *Image-segmentation evaluation from the perspective of salient object extraction*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006. 17, 20, 23, 24, 26, 27, 28, 31, 34

[GWL07] Feng Ge, Song Wang, and Tiecheng Liu, *New benchmark for image segmentation evaluation*, Electronic Imaging **16** (2007), no. 3, 033011.1–16. 17, 20, 23, 24

[HCD12] D. Hoiem, Y. Chodpathumwan, and Q. Dai, *Diagnosing error in object detectors*, European Conference on Computer Vision (ECCV), 2012. 53

[HD95] Qian Huang and B. Dom, *Quantitative methods of evaluating image segmentation*, IEEE International Conference on Image Processing, vol. 3, 1995, pp. 53–56 vol.3. 53, 56, 57, 63, 64

[HEH11] Derek Hoiem, AlexeiA. Efros, and Martial Hebert, *Recovering occlusion boundaries from an image*, International Journal of Computer Vision **91** (2011), no. 3, 328–346 (English). 108

[HJBJ+96] Adam Hoover, Gillian Jean-Baptiste, Xiaoyi Jiang, Patrick J. Flynn, Horst Bunke, Dmitry B. Goldgof, Kevin Bowyer, David W. Eggert, Andrew Fitzgibbon, and Robert B. Fisher, *An experimental comparison of range image segmentation algorithms*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **18** (1996), 673–689. 53, 58, 64

[HWG07] Lu Huihai, J. C. Woods, and M. Ghanbari, *Binary partition tree analysis based on region evolution and its application to tree simplification*, IEEE Transactions on Image Processing **16** (2007), no. 4, 1131–1138. 78

[Jac01] Paul Jaccard, *Étude comparative de la distribution florale dans une portion des alpes et des jura*, Bulletin de la Société Vaudoise des Sciences Naturelles **37** (1901), 547–579. 20

[JMIB06] Xiaoyi Jiang, Cyril Marti, Christophe Irniger, and Horst Bunke, *Distance measures for image segmentation evaluation*, EURASIP Journal on Applied Signal Processing **2006** (2006), 1–10. 54, 57, 58, 64

[KBR07] V. Kolmogorov, Y. Boykov, and C. Rother, *Applications of parametric maxflow in computer vision*, IEEE International Conference on Computer Vision (ICCV), 2007. 26

[KDNS94] T. Kanungo, B. Dom, W. Niblack, and D. Steele, *A fast algorithm for MDL-based multi-band image segmentation*, Tech. report, IBM Reasearch Division, RJ 9754 (84640), 1994. 53, 56

[KG12] J. Kim and K. Grauman, *Shape sharing for object segmentation*, European Conference on Computer Vision (ECCV), 2012. 108, 116, 117, 118

[Kuh55] H. W. Kuhn, *The Hungarian method for the assignment problem*, Naval Research Logistic Quarterly **2** (1955), 83–97. 58

[LA99] Bjornar Larsen and Chinatsu Aone, *Fast and effective text mining using linear-time document clustering*, ACM International Conference on Knowledge Discovery and Data Mining, 1999, pp. 16 – 22. 57

[LH00] G. Liu and R.M. Haralick, *Assignment problem in edge detection performance evaluation*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2000. 63, 64

[Mar03] David Martin, *An empirical approach to grouping and segmentation*, Ph.D. thesis, EECS Department, University of California, Berkeley, Aug 2003. 8, 54, 59, 61, 63, 64, 68

[MC06] Fernando Monteiro and Aurélio Campilho, *Performance evaluation of image segmentation*, Lecture Notes in Computer Science, vol. 4141, 2006, pp. 248–259. 20

[ME07] Tomasz Malisiewicz and Alexei A. Efros, *Improving spatial support for objects via multiple segmentations*, British Machine Vision Conference, 2007. 20, 35, 41, 64, 108, 109, 112, 115

[Mei03] Marina Meilă, *Comparing Clusterings by the Variation of Information*, Annual Conference of Computational Learning Theory, 2003. 58, 60, 61

[Mei05] ———, *Comparing clusterings: an axiomatic view*, International Conference on Machine Learning, 2005. 53, 58, 60, 64

[MFM04] D. Martin, C. Fowlkes, and J. Malik, *Learning to detect natural image boundaries using local brightness, color, and texture cues*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **26** (2004), no. 5, 530–549. 53, 63, 90

[MFTM01] 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*, IEEE International Conference on Computer Vision (ICCV), 2001. 7, 68

[MH01] Marina Meilă and David Heckerman, *An experimental comparison of model-based clustering methods*, Machine Learning **42** (2001), 9–29. 58

[Mir96] Boris Mirkin, *Mathematical classification and clustering*, Kluwer Academic Press, 1996. 60

[MO10] Kevin McGuinness and Noel E. O'connor, *A comparative evaluation of interactive segmentation algorithms*, Pattern Recognition **43** (2010), 434–444. 8, 19, 20, 62, 63

[MPB04] Jaesik Min, M. Powell, and K.W. Bowyer, *Automated performance evaluation of range image segmentation algorithms*, IEEE Transactions on Systems, Man, and Cybernetics **34** (2004), no. 1, 263 – 271. 59

[MRHM12] M. Maceira, J. Ruiz-Hidalgo, and J.R. Morros, *Depth map coding based on a optimal hierarchical region representation*, 3DTV-Conference, 2012, pp. 1–4. 51

[NMPG00] P. Nunes, F. Marques, F. Pereira, and A. Gasull, *A contour-based approach to binary shape coding using a multiple grid chain code*, Signal Processing: Image Communication **15** (2000), no. 7-8, 585–599. 90

[NS96] Laurent Najman and Michel Schmitt, *Geodesic saliency of watershed contours and hierarchical segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **18** (1996), no. 12, 1163–1173. 13

[PS13] Guillem Palou and Philippe Salembier, *Hierarchical video representation with trajectory binary partition tree*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013. 78

[PTM10] J. Pont-Tuset and F. Marques, *Contour detection using binary partition trees*, IEEE International Conference on Image Processing, 2010, pp. 1609–1612. 51

[PTM12] ———, *Supervised assessment of segmentation hierarchies*, European Conference on Computer Vision (ECCV), 2012. 53

[PZ12] Bo Peng and Lei Zhang, *Evaluation of image segmentation quality by adaptive ground truth composition*, European Conference on Computer Vision (ECCV), 2012. 53

[Rad92] T. Radzik, *Newton's method for fractional combinatorial optimization*, Symposium on Foundations of Computer Science, 1992, pp. 659–669. 26, 86, 88

[Ran71] William Rand, *Objective criteria for the evaluation of clustering methods*, Journal of the American Statistical Association **66** (1971), no. 336, 846–850. 61, 64

[RFE+06] B.C. Russell, W.T. Freeman, A.A. Efros, J. Sivic, and A. Zisserman, *Using multiple segmentations to discover objects and their extent in image collections*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006. 20, 35

[RS13a] Zhile Ren and Gregory Shakhnarovich, *Image segmentation by cascaded region agglomeration*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013. 13, 28, 40, 73

[RS13b] Zhile Ren and Gregory Shakhnarovich, *Image segmentation by cascaded region agglomeration*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013. 108, 117

[SCF+12] Avan Suinesiaputra, Brett R. Cowan, J. Paul Finn, Carissa G. Fonseca, Alan H. Kadish, Daniel C. Lee, Pau Medrano-Gracia, Simon K. Warfield, Wenchao Tao, and Alistair A. Young, *Left ventricular segmentation challenge from cardiac mri: A collation study*, Proceedings of the Second International Conference on Statistical Atlases and Computational Models of the Heart: Imaging and Modelling Challenges (Berlin, Heidelberg), STACOM'11, Springer-Verlag, 2012, pp. 88–97. 21

[SG00] P. Salembier and L. Garrido, *Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval*, IEEE Transactions on Image Processing **9** (2000), no. 4, 561–576. 13, 51, 78

[SM00] Jianbo Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **22** (2000), no. 8, 888–905. 13, 28, 72

[Sø48] Thorvald Sørensen, *A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons*, Biologiske Skrifter **5** (1948), 1–34. 19

[UH05] Ranjith Unnikrishnan and Martial Hebert, *Measures of similarity*, IEEE Workshop on Application of Computer Vision, 2005. 61

[UPH07] R. Unnikrishnan, C. Pantofaru, and M. Hebert, *Toward objective evaluation of image segmentation algorithms*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **29** (2007), no. 6, 929–944. 53, 54, 61, 64, 67, 68

[VdBBR+12] Michael Van den Bergh, Xavier Boix, Gemma Roig, Benjamin de Capitani, and Luc Van Gool, *SEEDS: superpixels extracted via energy-driven sampling*, European Conference on Computer Vision (ECCV), 2012. 95

[vdSUGS11] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders, *Segmentation as selective search for object recognition*, IEEE International Conference on Computer Vision (ICCV), 2011. 108, 116, 117, 118, 119

[VM04] P. Villegas and X. Marichal, *Perceptually-weighted evaluation criteria for segmentation masks in video sequences*, IEEE Transactions on Image Processing **13** (2004), no. 8, 1092–1103. 20

[VMS99] P. Villegas, X. Marichal, and A. Salcedo, *Objective evaluation of segmentation masks in video sequences*, Workshop on Image Analysis for Multimedia Interactive Services, 1999. 20

[VMS08] V. Vilaplana, F. Marques, and P. Salembier, *Binary partition trees for object detection*, IEEE Transactions on Image Processing **17** (2008), no. 11, 2201–2216. 13, 28, 40, 73, 77, 83, 90

[WM97] M. Wollborn and R. Mech, *Procedure for objective evaluation of VOP generation algorithms*, MPEG Committee, Document ISO/IEC JTC1/SC29/WG11 M2704, 1997. 20

[WO10] Hongzhi Wang and John Oliensis, *Generalizing edge detection to contour detection for image segmentation*, Computer Vision and Image Understanding **114** (2010), no. 7, 731 – 744. 17, 24

[WT13] David Weiss and Ben Taskar, *Scalpel: Segmentation cascades with localized priors and efficient learning*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013. 108, 116, 117

[XWC13] Chenliang Xu, Spencer Whitt, and Jason J Corso, *Flattening supervoxel hierarchies by the uniform entropy slice*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013. 78, 85

[ZFG08] Hui Zhang, Jason E. Fritts, and Sally A. Goldman, *Image segmentation evaluation: A survey of unsupervised methods*, Computer Vision and Image Understanding **110** (2008), no. 2, 260–280. 4