Multimedia Tools and Applications manuscript No.
(will be inserted by the editor)

From Global Image Annotation to Interactive Object Segmentation

Xavier Giró-i-Nieto · Manuel Martos ·
Eva Mohedano · Jordi Pont-Tuset

Received: date / Accepted: date

Abstract This paper presents a graphical environment for the annotation of still images that works both at the global and local scales. At the global scale, each image can be tagged with positive, negative and neutral labels referred to a semantic class from an ontology. These annotations can be used to train and evaluate an image classifier. A finer annotation at a local scale is also available for interactive segmentation of objects. This process is formulated as a selection of regions from a precomputed hierarchical partition called Binary Partition Tree. Three different semi-supervised methods have been presented and evaluated: bounding boxes, scribbles and hierarchical navigation. The implemented Java source code is published under a free software license.

Keywords Interaction · Segmentation · Multiscale · Annotation · Hierarchical

1 Motivation

The large and growing amount of visual digital data acquired nowadays has raised the interest for systems capable of its automatic analysis from a semantic point of view. After a first generation of algorithms in which specific-case solutions were developed through an expert study of the problem (e.g. text or face recognition), it is a general trend in the computer vision community to try to develop generic solutions that can be easily adapted to a diversity

This work was partially founded by the Catalan Broadcasting Corporation through the Spanish project CENIT-2009-1026 BuscaMedia, and by Spanish project TEC2010-18094 MuViPro: “Multicamera Video Processing using Scene Information: Applications to Sports Events, Visual Interaction and 3DTV.”

Xavier Giró-i-Nieto
Campus Nord UPC (Mòdul D5), Jordi Girona 1-3
08034 Barcelona, Catalonia/Spain
Tel.: +34 93 401 57 69
Fax: +34 93 401 64 47
E-mail: xavier.giro@upc.edu

of domains. Pattern recognition techniques have been successfully applied to a broad range of applications in computer vision [1], especially in their supervised learning variant. This type of problems usually works with images and videos that significantly represent the problem that is to be solved. This dataset is split in two parts: a first one to train a classifier and a second one to evaluate the expected performance of the learnt model. If performance is good enough, this model can be used to automatically annotate large amounts of data. In order to both train and evaluate a classifier, it is necessary to previously annotate the dataset, a task that requires some kind of human interaction, whether explicit or implicitly.

Before training a classifier, pattern recognition problems require the extraction of features that map images into a space where decision boundaries can be estimated. Good features are those that confine the instances of each class to a portion of the feature space that does not overlap with the instances associated to the rest of the classes. In the case of image analysis, a first solution is to use features extracted after considering images at the *global* scale. This approach simplifies the manual annotation task as the expert only needs to decide whether the image represents or contains an instance of the target class. However, in those cases where instances appear in a specific part of the image, like in object detection problems, global scale annotation makes it more difficult to train good classifiers, as they need to discriminate which portions of the positively annotated images are actually related to the modelled class. In these situations, a *local* scale annotation provides better features for the classifier at the expense of a higher effort from the annotator, who must manually indicate the area of support of the instance. This task requires the introduction of a graphical user interface to assist users into the determination of these areas.

The annotation process does not only require selecting visual data but also associating it to a semantic class. If this class has a semantic meaning, as in most computer vision tasks, these semantics must be defined in an additional data structure. Ontologies are the most common solutions adopted by the scientific community as they define classes in a formal and structured manner [6]. Successful computer vision techniques not only base their results on the signal processing algorithms but also on semantic reasoning processed at a higher level [14] [15]. The use of ontologies introduces context in the analysis task and offers an opportunity to fuse image analysis with other modalities such as text and audio. For these reasons, annotation tools not only need to offer a workspace to select images and regions but must also provide mechanisms to handle ontologies.

This paper extends a previous work [16] where GAT (Graphical Annotation Tool) was introduced for the annotation of still images at the local scale. This original version has been improved with an integrated environment where annotations can be generated at both global and local scales. This core functionality has been complemented with a new perspective to train and evaluate image classifiers. Moreover, this current work proposes a novel interactive segmentation environment that has been evaluated in terms of accuracy and user time. GAT is addressed to an academic audience that can find in this software a solution to generate a ground truth of MPEG-7/XML [4] annotations, which can be later used to test their own classification algorithms.

The rest of the paper is structured as follows. Section 2 reviews some of the related work in the field of semantic annotation of still images, both at the local and global scales. Section 3 presents the basic workflow with GAT, an overview of the different parts that are described in the remain of the paper. Section 4 presents the graphic interface proposed to manually annotate images at a global scale and how these annotations are used in this same tool to train and evaluate image classifiers. Section 5 focuses on the interactive segmentation of objects to generate local annotations, proposing and assessing three different selection modes. Finally, Section 6 draws the conclusions and provides instructions about how to download and test this tool.

2 Related Work

The manual annotation of images is a time-consuming task that has been an intense research area for the last decade [8,11]. There exists a variety of solutions that have explored topics such as crowd-sourcing, usability, interactive segmentation, and ontology management.

At the global scale, the TRECVID evaluation campaign used the IBM Efficient Video Annotation (EVA) tool [24] to annotate the presence of certain concepts in video shots. This web-based tool painted the box around the video key-frames with one color (green, red or white) to visually code the associated label (positive, negative or neutral). The user could change the initial red frame assigned by default by clicking on the keyframes. This code of colors has been adopted in this work to indicate the labels at the global scale, although the selection mechanism has been modified to provide more flexibility to the user. Another web-based solution [5] addressed the multi-class problem in a semi-automatic system where the annotation tool placed images on row panels depending on a suggested label. In this case, the user could visually identify the outliers and edit the labels with a simple drag-and-drop mechanism between panels. At the local scale, an on-line interface was developed by the LabelMe project [21] to collect a large amount of object silhouettes. Users drew a polygon around the object, which provided a local but somewhat rough annotation of it. The user also introduced a free textual label that was mapped onto the WordNet ontology [9].

A popular strategy for obtaining crowd-sourced annotations is through online games. The Extra Sensory Perception (ESP) game [2] collected textual labels at the global scale by showing the same image to a pair of players. Players were prompted to enter words related to the shown image and, when an agreement was obtained between different players, they were rewarded with points. The label was considered correct by the authors when different pairs agreed on a word. This idea was extended to the local scale in the Name-It-Game [23], where objects were outlined by a *revealer* player and had to be predicted by a second *guesser* player upon a gradual appearance of the selected object. This interface combined freehand and polygonal segmentations, and the considered concepts were extracted from the WordNet ontology.

The main drawback of web-based tools and games is that they need setting up a server, a task that may require advanced technical skills. Although this architecture is appropriate for a collaborative annotation effort, it poses problems when simpler configurations are preferred. GAT has been developed as a multi-platform desktop tool to facilitate its adaptation from third-part users. However, the source code is also prepared to work with a remote repository, as reported in [17].

There exist other desktop solutions apart from GAT. M-OntoMat-Annotizer [19] is a region-based annotation tool that combines multimedia and domain-specific ontologies. This software contains a segmentation engine that lets users associate concepts to selected sets of region. The tool is also capable to extract low level visual descriptors and generate MPEG-7 descriptions that contain both perceptual and semantic information. The MPEG-7 data format has also been adopted by GAT, as it offers a formal language to represent content at both low and high level. However, M-OntoMat-Annotizer provides a single interface for both global and local annotations, and it requires an individual processing of each image. GAT, instead, facilitates the annotation at the global scale, with a dedicated perspective based on thumbnails and selection tools for the fast labeling of images.

3 Workflow

GAT provides four different perspectives aimed at guiding the user during the different stages of the annotation. Figure 1 offers an overview of them as well as the input and output data associated to each of them. The user can jump at any moment from one perspective to another through dedicated icons located in the toolbar.

After launching GAT, the *Instances explorer* is presented. This perspective allows a quick overview of the instances already annotated so, at launch time, it will appear empty. At this point the user can either load an annotation previously saved in disk or select an ontology to be associated to a new annotation. In the latter case, a floating window will appear prompting the user with three possible options: exploring the file system to load an existing ontology, read the ontology from a remote URL, or creating a new one from scratch. The last option will show a new panel with a simple ontology editor, where classes can be added, removed, and renamed. This editor can be accessed again in the future during the annotation. Any new ontology must be saved in a file so that new annotations can refer to it.

Once the annotation is initialized, the next stage corresponds to the visual labelling of images. This stage requires changing to the *Collection Annotator* perspective. This perspective is populated with the thumbnails of the images selected by the user from a local directory. The user can directly label images

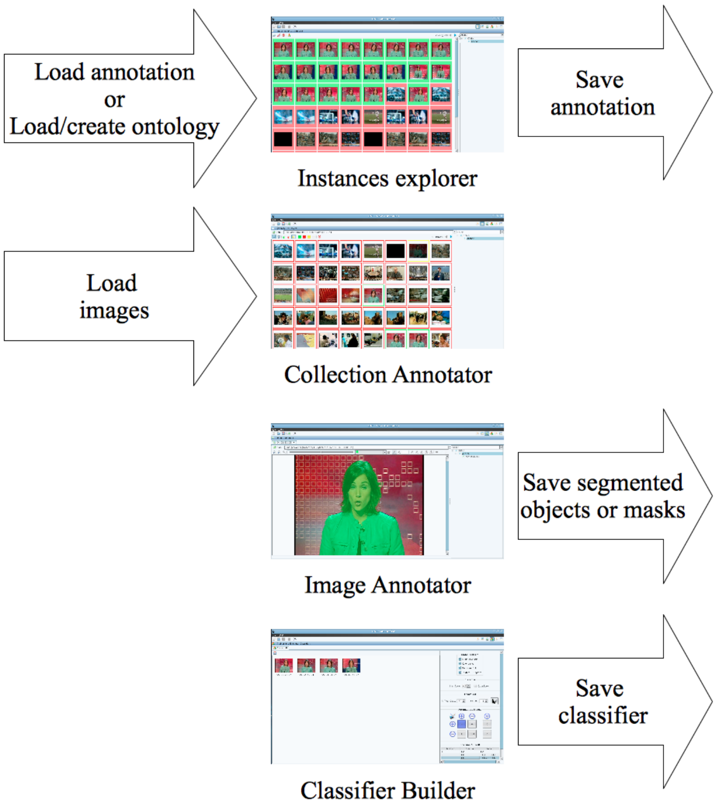


Fig. 1 The four perspectives in GAT

at the global scale from this perspective (presented in Section 4), or generate a local annotation of the images by selecting their thumbnails and double clicking on them (explained in Section 5). This second action will change to the *Image Annotator* perspective, where each selected image is loaded in a different tab. Each tab allows the local annotation of the image with a diversity of tools included in GAT.

The annotated instances can always be reviewed by returning to the *Instances Explorer*, that contains a *disk* icon to save the annotation to a local file. This perspective is also the entry point to the *Classification* perspective, where the annotated images are used to train an image classifier. GAT offers the necessary tools to set up a cross validation experiment and analyze the results both numerically and visually. From this perspective, the user can also export the trained classifier for its external exploitation.

4 Image Annotation at a Global Scale

The annotation of images can be performed at two basic spatial scales: *global* or *local*. In the global case the area of support is the full image, while local annotations mark a subset of the image pixels that depict a semantic object. Global annotations are especially suited for scene classification, but in several cases they are also used to label images that contain local entities (objects, people, etc.). Their lower requirements in terms of user interaction makes them attractive even for local analysis, at the expense of an indetermination about the exact location of the referred instance.

GAT offers two different perspectives that deal with image annotation at a global scale. A first one, which is completely manual, and a second one that allows training and evaluating an image classifier capable of generating automatic annotations.

4.1 Manual Annotation

GAT provides a dedicated *Collection* perspective to allow a quick annotation of images at a global scale. This perspective explores the content of a folder in the file system and shows the thumbnails of the included images. In most cases, viewing the thumbnails is enough for users to decide about the label but, if necessary, a double click on any of them will display the full image on a dedicated *Image* tab.

A broad range of machine learning techniques require that annotations should not only consider which samples correspond to the modeled class but also which of them do not correspond to this class. A classic example are binary classifiers, that use two types of labels: positive and negative. In some situations a third type of label, the neutral one, is also used to merely state the existence of the observation. These neutral images are usually discarded for training or experimentation [24] as its inclusion may harm the overall performance. These three types of labels are supported in GAT only in the case of global annotations, as local annotations imply by default a positive label for the selected segment.

The assignment of global labels starts by clicking on one of the six icons located on the perspective's toolbar. Their color intuitively indicates what label they are related to: green (positive), red (negative) or yellow (neutral). These icons provide two different types of selection tools: individual or all. The first group activates the associated label so that every new click will assign the label to the image. The second group assigns the selected labels to all currently non-annotated images. For example, this functionality becomes very practical in those cases where only a few of the displayed images belong to the class. In this situation, an initial green labelling of the few images belonging to the class can be quickly completed by a massive red selection of the rest of the images.

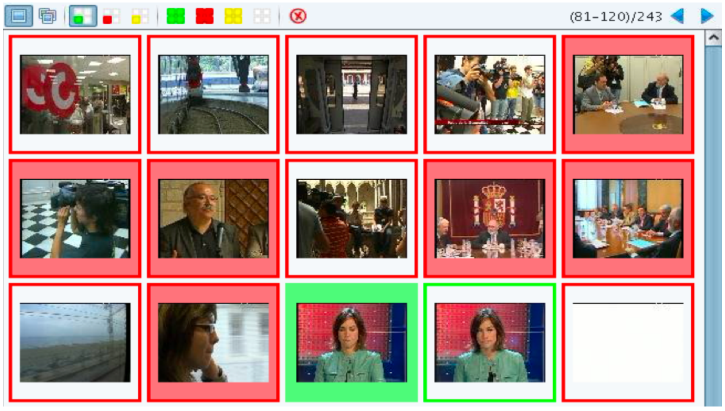


Fig. 2 Selected (blank panel) vs annotated (filled green/red panel) thumbnails

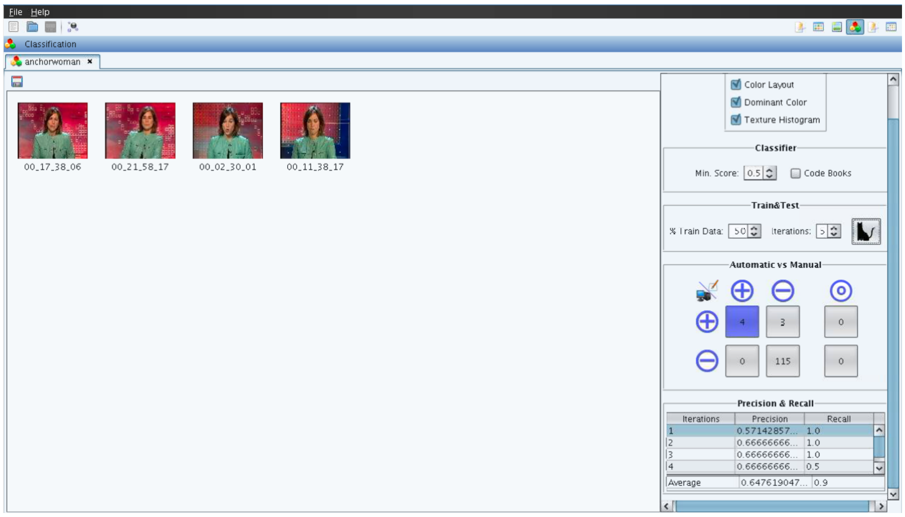
Figure 2 shows how selected and validated thumbnails are distinguished. When a thumbnail is selected, a frame of the associated label's color is painted around the panel containing the thumbnail. When the assigned labels are validated with a right-click, the thumbnail panel is painted with the color of the label.

The creation of new instances is also represented in the interface in the *Semantic Panel*, located on the right-side of the interface. This panel includes a tree whose root corresponds to the name of the ontology and its children the semantic classes available for annotation. Whenever a new instance is added to the annotation, a new node is added to this tree. This *Semantic Panel* is also present in the *Instances Explorer* and *Image Annotator* perspectives. In all cases, the panel provides an overview of the instances contained in the main visual panel and allows their review and deletion.

4.2 Automatic Annotation

In addition to the tools for manual annotation, GAT includes a perspective that exploits the generated annotation in the framework of an image classification system. This perspective provides an intuitive environment to evaluate an image classifier trained with the annotated content. In the current implementation, GAT relies on an external tool that classifies images based on their MPEG-7 visual descriptors [4] and using an SVM classifier with an RBF kernel [7].

The *Classification* perspective is accessible by clicking a dedicated icon on the toolbar of the *Instances explorer* perspective. This action switches perspectives and creates a new tab associated to the selected class, as shown in Figure 3. The tabs in the *Classification* perspective are organized in two large areas: a central panel that shows image thumbnails, and a vertical panel on the right to control the parameters for the classification and the evaluation.



☑ Color Layout
☑ Dominant Color
☑ Texture Histogram

Classifier

Min. Score: 0.5 □ Code Books

Train&Test

% Train Data: 50 Iterations: 5

Automatic vs Manual

	⊕	⊖	⊙
⊕	4	3	0
⊖	0	115	0

Precision & Recall

Iterations	Precision	Recall
1	0.57142857...	1.0
2	0.66666666...	1.0
3	0.66666666...	1.0
4	0.66666666...	0.5
Average	0.647619047...	0.9

Fig. 3 Classification perspective

The control panel allows the selection of different parameters related to an image classification engine. In particular, it allows choosing among a catalogue of visual descriptors, setting a minimum confidence score for detection and deciding if a codebook must be used to quantize the visual features. A second type of controls refer to the evaluation process itself. The adopted approach follows a cross-validation scheme with a random partition between training and test data. The user can select the amount of folds to run as well as the proportion of annotated images assigned to the training and test sets.

A left-click on the cat-shaped icon launches the evaluation process. In each iteration of the cross-validation process, the dataset is randomly partitioned and the training data is used to learn the visual model for the class. Once built, the images from the test partition are classified one by one as belonging to the class or not. The label predicted by the classifier is compared with the annotated ground truth, so that the every test image is counted as a true or false classification.

The graphical interface allows a rapid assessment of the results. Firstly, the panel on the right includes a table that displays the precision and recall obtained on each iteration of the cross-validation. The last row of the iterations table averages the precision and recalls obtained in each cross-validation fold. A click on any of the rows of the table will select one cross-validation fold and will display its associated data in the main panel of thumbnails. The images shown there depend on the active button from another grid panel which represents the confusion matrix. The diagonal of the matrix corresponds to the correct predictions, while the rest of cells in this grid corresponds to errors from the classifier. Given the single-class nature of the perspective, the size of the square grid is 2x2, each of its cells associated to a *true/false positive/negative* prediction. There exists, though, an additional column that corresponds to the neutral labels.

The *Classification* perspective also allows exporting a model of the selected class to any location in the file system. This way, if the user is satisfied with the presented results, a version of the classifier can be saved for its external exploitation. In that case, a new model is built considering all annotated images as belonging to the training dataset.

5 Interactive segmentation

In addition to the annotations at the global scale, GAT also provides tools that help in the accurate annotation at a local scale. These tools are available through a double-click on the thumbnails at the *Collection Annotator* or *Instances explorer* perspectives. This action will activate the *Image Annotator* perspective, where every selected image is d in a dedicated new tab with different methods available to annotate the image. All local annotations are assigned to the positive label, so in this mode the color code used for global annotations does not apply. The color of the markers used for local selection can be configured by the user to avoid visual confusion between the instance selection and the background. By default, though, it is set to green for coherence with the global labels as well as for its high perceptual brightness for the human visual system, which improves the contrast of the markers.

Local-scale solutions can be divided in two groups depending on the sought precision. A first family of techniques provides *rough* descriptions of the objects [21] [23], giving approximate information about their location and shape, normally, using geometric figures (e.g. bounding boxes, polygons, etc.) . A second option for local annotations is the precise *segmentation* of those pixels that represent the object, by defining the exact area of support associated to the object [19]. GAT provides tools for both options, but this paper focuses on interactive segmentation strategies, given that a successful solution of this more complex case can be easily transformed into a rough annotation.

The remainder of the section is devoted to present and assess the interactive-segmentation techniques presented in this paper. First, Section 5.1 explains the hierarchical structure on which the segmentation is based. Then, Section 5.2 describes the three different interaction methods used. Finally, Section 5.3 evaluates and compares these three methods.

5.1 Binary Partition Trees

Systems offering precise local annotations can be classified into region-based or contour-based approaches. Region-based annotations let the user select among a set of segments from an automatically generated partition of the image, while contour-based solutions aim at generating a curve that adjusts to the pixels located at the border between object and background. GAT provides three methodologies based on the first family to interactively generate a segmentation of the instance. In all of them, the success of the interaction is tightly dependent on the goodness of the segmentation. GAT does not include a segmentation engine, but several state of the art techniques offer nowadays enough precision to be used into the proposed interactive framework [13,18].

One basic limitation of considering a single image partition for its semantic analysis comes from the diversity of scales where the semantics can be present. Not only in many cases semantics are represented at the global or local scale, but in several situations semantic entities contain other semantic entities. For example, composite visual objects such as *people* have clearly separate visual parts with their own semantics, such as *head* and *body*, and each of these parts could be further decomposed semantically as *face* and *hair* for the *head* or *trunk* and *legs* for the *body*. If each of these semantic entities is to be represented by a segment in the image, it is not enough to consider a single partition at a single scale, multiple scales must be considered.

The multi-scale analysis is supported in GAT by generating the object segmentations on a *hierarchical partition*. This type of image representation defines a set of segments based on an initial partition at a fine spatial scale. The segments in this partition are iteratively merged with other neighboring segments to define new segments at larger scales. The creation of such a hierarchy ends when all regions have been merged into a single one that represents the complete image. GAT uses a specific case where the number of region merged at each iteration is limited to two, a which leads to a structure named *Binary Partition Tree (BPT)* [22]. Figure 4 shows the hierarchical decomposition of an image into the regions defined by a BPT. In this work, we have used the trees known as *Ultrametric Contour Maps* (UCM) [3], which have proven state-of-the-art performance in contour detection and segmentation.

5.2 Interaction Methods

The user interaction is combined with the BPT in three different ways, defining three interaction methods for semi-supervised object segmentation, namely *bounding boxes*, *scribbles (brush strokes)*, and *BPT navigation*; which are described in the following sections.

5.2.1 Bounding boxes

The simplest considered mode in terms of user interaction is the drawing of a bounding box around the object of interest. This selection mode requires the system to ideally adjust this box to the actual object contours. The selected regions are shown to the user as transparent in an overlaid mask, as shown in Figure 5. This interaction mode is very intuitive for users, who are very familiar with drawing rectangles.

Three different strategies have been considered for solving the adjustment of the bounding box to the regions defined by the BPT:

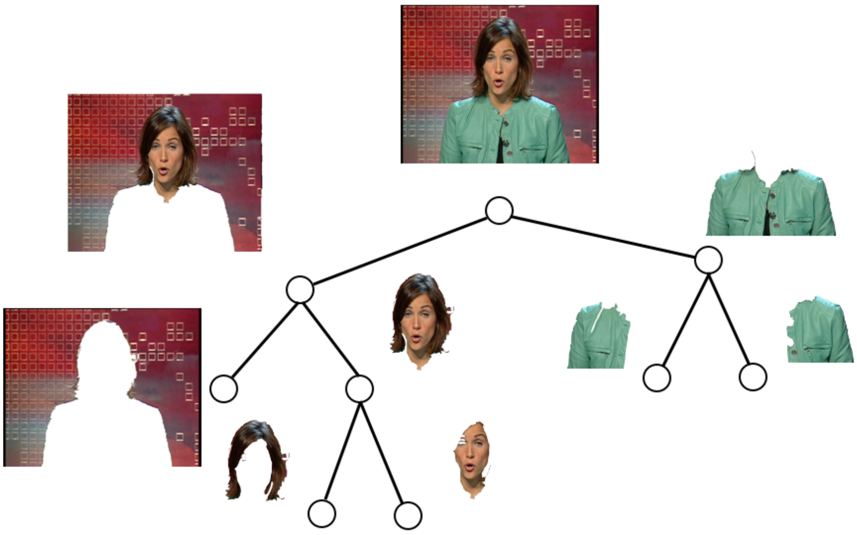


Fig. 4 Binary Partition Tree

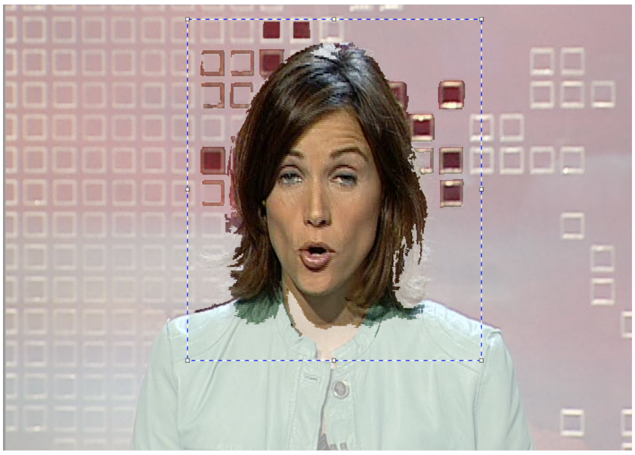


Fig. 5 Rectangle marker and selected regions

- **Strategy 1** (inside regions): Select all those BPT leaves completely included in the bounding box.
- **Strategy 2** (region areas): Select all those BPT leaves completely included in the bounding box and sort them by area. Then, explore the list starting from the largest area and adding regions until the whole selection overlaps with the four sides of the bounding box reduced P pixels on each side.
- **Strategy 3** (subtree depths): Select all those sub-BPTs completely included in the bounding box and sort them by height. Then, explore the list starting from the largest area and selecting regions until the whole selection

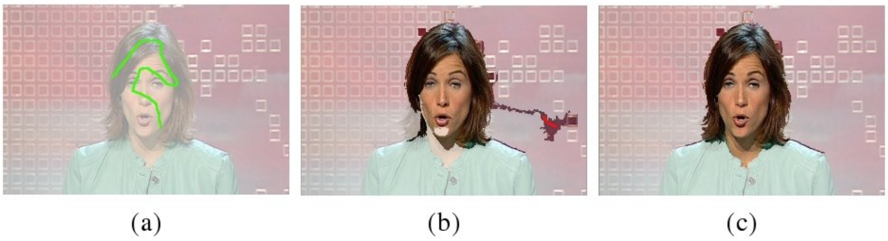


Fig. 6 Segmentation with object (in green) and background (in red) scribbles

overlaps with the four sides of the bounding box reduced P pixels on each side.

In all cases, it is also considered to force a single connected component. In the first case, the largest candidate is considered, in the two later cases the first region of the ranked list is taken as an anchor; the rest of candidates are kept only if connected to the anchor or to another of its connected components. In total, six different strategies are considered.

5.2.2 Scribbles

A second application of BPTs to interactive segmentation is the propagation of labels through its structure. In this case, the user interaction requires drawing *scribbles (brush strokes)* on the image, specifying if these markers are on the object or on the background. Analogously to the bounding box case, three different strategies for label propagation have been considered:

- **Strategy 1** (no propagation): Only the BPT leaves intersecting the foreground scribbles are labelled as object. Background scribbles are ignored.
- **Strategy 2** (object propagation): Object labels are iteratively propagated to the parent node in the BPT if the subtree defined by the considered node's sibling contains at least one object label, but no background label.
- **Strategy 3** (no-background propagation): Object labels are iteratively propagated to the parent node in the BPT if the subtree defined by the considered node's sibling does not contain any background label.

The selection can be refined by combining object and foreground scribbles, as the example in Figure 6. A first step (a) draws an object scribble (green) over the anchorwoman's face. Step (b) shows how the label propagation has erroneously selected some regions belonging to the background, so a background (red) scribble is drawn over them to finally obtain a better segmentation in step (c).

5.2.3 BPT Navigation

The third method for interactive object segmentation directly navigates through the BPT-tree structure in order to select the nodes representing the object or the background. The visual difference between selecting object or background is that, while the object selection corresponds to showing new regions, the background selection (or object deselection) is represented by covering the region with the semi-transparent overlaid mask.

The selection starts by placing the cursor on the area of support of the object. With this action, the user is implicitly selecting one branch from the BPT, as every pixel in the image corresponds to one, and only one, branch in the BPT. After this first interaction, the interface highlights the region associated to the BPT leaf so that the user can evaluate if the proposed region correctly depicts the object. If this is not the case, the selected BPT node can be modified by rotating the mouse wheel, moving upwards or downwards in the branch at every wheel rotation. Every new move will expand or contract the selection depending on the direction of the rotation. The navigation path is defined between the BPT root, where the whole image is selected, and a BPT leaf, where a region at the initial partition is shown. A left-click will consolidate the current selection and allow processing regions from other BPT branches. Notice that this scheme allows the local annotation of non-connected components.

The mouse icon changes whenever any region is currently toggled but not consolidated. This allows the user knowing whether the shown regions are temporal or have been consolidated.

The selection mode for object or background is switched with a left click on a region which is not temporary toggled. This will change the state of the alpha mask over the clicked regions and update the the mouse icon according to the state of the region below.

Every time the user moves the cursor out of the current selection, the system must propose a new BPT node in the new active branch. This selection is automatically made by the interface depending on the size of the previously selected BPT. The algorithm will look for the BPT node in the new branch whose size is smaller but closest to the previous state. This solution is adopted based on the assumption that, in general, the user is more likely to look for new objects at the same spatial scale of the previous selection.

5.3 Evaluation

A quantitative comparison of these strategies was performed on the 50 images and ground truth masks of the GrabCut dataset [20]. This collection defines instances of different object classes on different backgrounds, all of them contained in natural photo images. The degree of difficulty of the segmentation varies, so that in some cases the object is clearly different from the background, in others its segmentation is highly challenging.

The metric used to compare a given segmentation to the provided ground truth masks is the Jaccard index, ignoring those pixels marked as mixed area in the ground truth. Formally, being O and GT the set of pixels in the mask of the object and the ground truth, respectively, and M the set of pixels in the mixed area; the Jaccard index is defined as:

$$J(O, GT) = \frac{|O \cap GT|}{|\,(O \cap \overline{M}) \cup GT|}$$

where $\overline{\cdot}$ refers to set complimentary and $|\cdot|$ to set cardinality.

In order to evaluate the impact of the quality of the image segmentation provided by the BPT, the upper-bound Jaccard index that can be obtained from the regions in the tree was computed. This upper-bound models the case in which a perfect algorithm for interactive segmentation was adopted, so that the markers and user interaction would provide the best possible selection of regions in the BPT. If we consider only regions completely included in the bounding box, the mean upper-bound J is 0.967. This results shows that the tree could deliver very good results, so it is not the limiting factor in our experiments. The upper-bound J obtained when the algorithm can select any region in the image is 0.970, which proves that the degree of leaks of the trees from the object to outside of the bounding box is minimal.

Below we provide the evaluation results for the three proposed interaction methods with the described dataset and accuracy measure.

5.3.1 Bounding boxes

The six variations considered for mapping bounding boxes where tested with the bounding boxes provided by [12] with $P = 20$, that is, considering that a region *touches* each side of the bounding box if its distance to the border is less than 20 pixels.

Table 1 shows the mean Jaccard index for the six strategies proposed. Each row refers to a type of strategy and each column shows whether the object was forced to be connected or not (see Section 5.2.1).

Strategy	Non-connected	Connected
(1) Inside regions	0.713	0.716
(2) Region areas	0.799	0.809
(3) Sub-tree depths	0.785	0.813

Table 1 Mean Jaccard indices for the six strategies of bounding box fitting

The first conclusion that can be extracted from these results is that forcing the object to be connected is an improvement on any of the strategies, which is coherent with the fact that all the objects in the database are connected. Second, we can conclude that the strategies that take advantage of the hierarchy, either via the area of its regions (Strategy 2) or the depth of its subtrees (Strategy 3), are a clear improvement over selecting all the regions completely included in the bounding box (Strategy 1). Regarding the comparison between Strategies 2 and 3, there is not a significant difference between them.

5.3.2 Scribbles

The proposed strategy for label propagation after scribble markers was tested with the GrabCut dataset [20] and the scribbles (brush strokes) published by [10]. However, in the provided dataset there is a missing scribble for image 124084 from the GrabCut collection. The scribbles for this image was taken from another work [18].

The experiments considered three different strategies, as described in Section 5.2.2. Results are shown in Table 2.

Strategy	Scribbles
(1) No propagation	0.311
(2) Object propagation	0.436
(3) No-background propagation	0.755

Table 2 Mean Jaccard indices for the scribbles propagation

These results clearly show the gain of expanding the labels of the initially selected BPT leaves through the hierarchical structure, as the very low 0.311 average Jaccard Index (Strateg 1) is raised to 0.755 with no further interaction from the user (Strategy 3). The experiments indicate that the BPT structure provides enough consistency to expand labels on those sub-BPTs with no labelled leaf beneath (Strategy 3), with a clear gain with the more conservative option of only expanding on those sub-BPTs that already contain a positive label among its leaves (Strategy 2).

The best 0.755 accuracy obtained with scribbles is still below the best 0.813 value achieved with bounding boxes. This result, combined with a lower user interaction required to draw a bounding box with a mouse, points to a superior performance of the bounding box marker. However, other input devices, such as touch screens, might offer a better framework for the scribble-based mode.

5.3.3 User tests

The presented tools for interactive segmentation were tested with real users to segment the same 50 images from the GrabCut [20] dataset that were used to evaluate the mapping of bounding boxes and scribbles on the BPT. The user experiments focused only in combining an initial selection with a bounding box with a posterior refinement through BPT navigation. This set up was adopted given the superior performance of the bounding box mode compared to the scribbles one. Among the two strategies with superior results in the experiments of Section 5.3.1, the option based on the sub-tree depths was selected (Strategy 3) given its slightly better behaviour for connected components.

Each image in the dataset was segmented by 8 different people, from a group of 14 different participants. The amount of objects segmented by participant went from 20 to 30, in blocks of 10. This partition was introduced

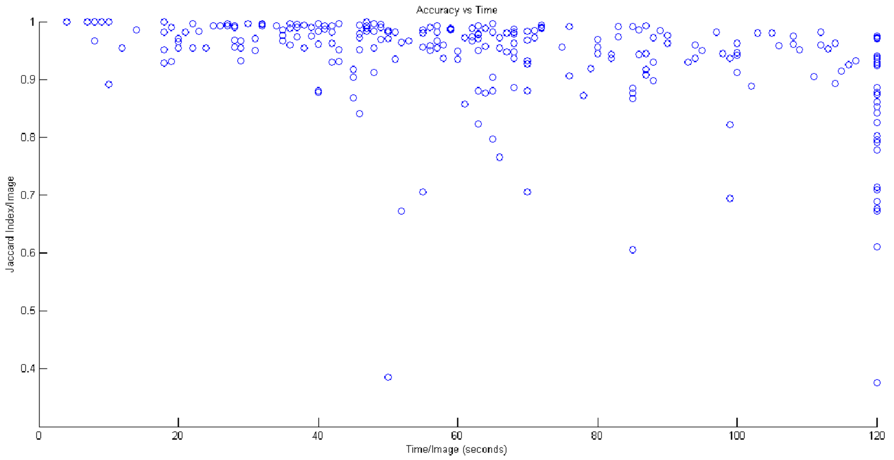


Fig. 7 Accuracy vs Time of each user segmentation

to reduce the level of user stress.. Each participant was given a maximum of two minutes to segment every object, as in [13]. The timing was manually controlled by an experiment supervisor, who also read out loud a brief description of the object that had to be segmented. Each experiment began with a brief tutorial about how to use the segmentation tool and a mock up test with one of the images in the dataset which would not belong to the block to be annotated.

The average Jaccard index obtained in the user experiments was 0.914. As expected, this value is higher than the best configuration using only a bounding box (0.813), but still not reaching the upper bound that a perfect selection would reach (0.970). The average time invested in segmenting an object was 46.9 seconds.

Figure 7 shows the a set of points, each of them representing an individual segmentation. The graph shows the diverse complexity of the segmentations in the dataset. The lower accuracy values are related to images where the underlying BPT already merged foreground and background pixels in the same BPT leaf, so users could not obtain better values. A few of these low accuracy values are also associated to a lack of attention of the users, who were not aware that their selection was incomplete. The column of points at the right side of the graph represent all those cases where the timer expired.

Figure 8 normalized the Jaccard indexes and invested time by the average values associated to the image and the user involved in the segmentation. This normalization tries to compensate the two types of diversities that affect each measure: the one associated to the user skills and the one linked to the segmentation difficulty of each image. Results shows a larger deviation in terms of time than in terms of accuracy, so most experiments resulted in a similar

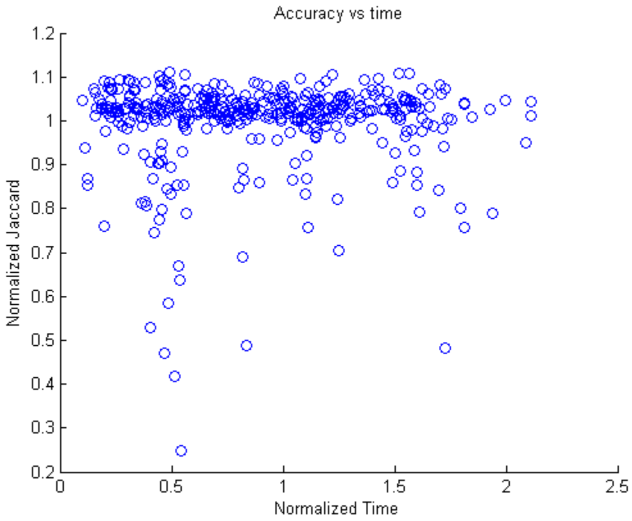


Fig. 8 Normalized Accuracy vs Normalized Time of each user segmentation, limited to 120 seconds

Jaccard index (a high one), independently if the invested time was below or over the average.

It was observed that in some cases the users preferred to toggle by clicking on many tiny regions instead of navigating through the BPT. Some users also missed an *undo* function to allow correction. Finally, another common problem was the accidental click on the right button, which validated the selection. In these cases, the segmentation had to be started from scratch.

6 Conclusions

This paper has presented a framework capable of annotating still images both at a global scale as well as at a high precision level. The tool integrates the two options to provide a unified solution for those researchers who need to create ground truth datasets. The annotation at a global scale is complemented with an additional perspective that allows the evaluation of an external classifier. Additionally, the presented tool implements different methods for the interactive segmentation of images to annotate local objects. The tests indicate that the BPT structure is a promising tool to assist users in expanding their interaction to select the complete object, both adjusting the bounding box or expanding a scribble to the object. The presented experiments show that, on average, users will spend almost 47 seconds to generate a high quality segmentation of the objects. This performance is possible thanks to the combination of a hierarchical segmentation of the image with an interactive environment.

The GAT annotation tool has been funded by two industrial companies who agreed to open the source code of this tool under a free software license to facilitate its promotion, reuse and further extension among the scientific community. The source code is available on a public website[1], where video-demos of the software can be watched and the tool itself downloaded and launched. Regarding data formats, GAT is based on MPEG-7/XML to code the ontologies, annotations and BPTs. Examples of all types of file formats are provided with the software package.

GAT is currently being used in a teaching environment for a practical exercise on image classification, where students complete the whole annotation, training and evaluation cycle with an intuitive and graphical environment. Moreover, it has been used to annotate datasets of hundreds of images at the local scale, which have been exploited in experiments about object detection and segmentation.

Future work will concentrate on an automation of the evaluation of the system following the guidelines suggested in [13]. These advances will imply recording every user interaction from GAT to study the temporal evolution of the segmentation. In addition, efforts will be focused on improving the client-server architecture to be able to easily collect crowd-sourced annotations as well as more extensive interactive segmentation experiments.

References

1. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)
2. von Ahn, L., Dabbish, L.: Labeling images with a computer game. In: SIGCHI conference on Human factors in computing systems, pp. 319–326 (2004). DOI 10.1145/985692.985733
3. Arbeláez, P., Maire, M., Fowlkes, C.C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(5), 898–916 (2011). DOI 10.1109/TPAMI.2010.161
4. B.S. Manjunath, P.S., T. Sikora, E. (eds.): Introduction to MPEG-7: Multimedia Content Description Interface. Wiley, Chichester, West Sussex, UK (2002)
5. Carcel, E., Martos, M., Giro-i Nieto, X., Marques, F.: Rich internet applications for semi-automatic annotation of semantic shots in keyframes. In: MUSCLE Intl. Workshop. Pisa (2011)
6. Cardoso, J.: The semantic web vision: Where are we? Intelligent Systems, IEEE **22**(5), 84 –88 (2007). DOI 10.1109/MIS.2007.4338499
7. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2**, 27:1–27:27 (2011). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
8. Dasiopoulou, S., Giannakidou, E., Litos, G., Malasioti, P., Kompatsiaris, Y.: A survey of semantic image and video annotation tools. In: G. Paliouras, C. Spyropoulos, G. Tsatsaronis (eds.) Knowledge-Driven Multimedia Information Extraction and Ontology Evolution, *Lecture Notes in Computer Science*, vol. 6050, pp. 196–239. Springer Berlin / Heidelberg (2011)
9. Fellbaum, C.: Wordnet. In: R. Poli, M. Healy, A. Kameas (eds.) Theory and Applications of Ontology: Computer Applications, pp. 231–243. Springer Netherlands (2010)
10. Gulshan, V., Rother, C., Criminisi, A., Blake, A., Zisserman, A.: Geodesic star convexity for interactive image segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2010)
11. Hanbury, A.: A survey of methods for image annotation. Journal of Visual Languages and Computing **19**(5), 617 – 627 (2008). DOI 10.1016/j.jvlc.2008.01.002

[1] `http://upseek.upc.edu/gat/`

12. Lempitsky, V.S., Kohli, P., Rother, C., Sharp, T.: Image segmentation with a bounding box prior. In: IEEE International Conference on Computer Vision, pp. 277–284 (2009). DOI 10.1109/ICCV.2009.5459262
13. McGuinness, K., O'Connor, N.E.: A comparative evaluation of interactive segmentation algorithms. Pattern Recognition **43**(2), 434 – 444 (2010). DOI 10.1016/j.patcog.2009.03.008
14. Mezaris, V., Kompatsiaris, I., Strintzis, M.G.: Region-based image retrieval using an object ontology and relevance feedback. EURASIP J. Appl. Signal Process. **2004**, 886–901 (2004). DOI 10.1155/S1110865704401188
15. Naphade, M., Smith, J., Tesic, J., Chang, S.F., Hsu, W., Kennedy, L., Hauptmann, A., Curtis, J.: Large-scale concept ontology for multimedia. Multimedia, IEEE **13**(3), 86 –91 (2006). DOI 10.1109/MMUL.2006.63
16. Giro-i Nieto, X., Camps, N., Marques, F.: Gat, a graphical annotation tool for semantic regions. Multimedia Tools and Applications **46**(2), 155–174 (2010). DOI 10.1007/s11042-009-0389-2
17. Giro-i Nieto, X., Ventura, C., Pont-Tuset, J., Cortes, S., Marques, F.: System architecture of a web service for content-based image retrieval. In: ACM Intl. Conference on Image and Video Retrieval, CIVR '10, pp. 358–365 (2010). DOI 10.1145/1816041.1816093
18. Noma, A., Graciano, A.B., Cesar, R.M., Consularo, L.A., Bloch, I.: Interactive image segmentation by matching attributed relational graphs. Pattern Recognition **45**(3), 1159 – 1179 (2012). DOI 10.1016/j.patcog.2011.08.017
19. Petridis, K., Anastasopoulos, D., Saathoff, C., Kompatsiaris, Y., Staab, S.: Montomat-annotizer: Image annotation, linking ontologies and multimedia low-level features. In: Intl. Conf. on Knowledge Based, Intelligent Information and Engineering Systems (2006)
20. Rother, C., Kolmogorov, V., Blake, A.: "grabcut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. **23**(3), 309–314 (2004). DOI 10.1145/1015706.1015720
21. Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T.: Labelme: A database and web-based tool for image annotation. Int. J. Comput. Vision **77**(1-3), 157–173 (2008). DOI 10.1007/s11263-007-0090-8
22. Salembier, P., Garrido, L.: Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. IEEE Transactions on Image Processing **9**(4), 561 –576 (2000). DOI 10.1109/83.841934
23. Steggink, J., Snoek, C.: Adding semantics to image-region annotations with the name-it-game. Multimedia Systems **17**, 367–378 (2011). DOI 10.1007/s00530-010-0220-y
24. Volkmer, T., Smith, J.R., Natsev, A.P.: A web-based system for collaborative annotation of large image and video collections: an evaluation and user study. In: ACM Intl. Conference on Multimedia, pp. 892–901 (2005). DOI 10.1145/1101149.1101341