Università degli Studi di Catania
Dipartimento di Ingegneria Elettrica, Elettronica
e Informatica (DIEEI)

DOTTORATO DI RICERCA IN INGEGNERIA
INFORMATICA E DELLE TELECOMUNICAZIONI
XXVI Ciclo

LARGE SCALE GROUD TRUTH GENERATION FOR PERFORMANCE EVALUATION OF COMPUTER VISION METHODS

Roberto Di Salvo

Coordinatori
Prof. Orazio Mirabella
Prof. Vincenza Carchiolo

Tutor
Prof. Daniela Giordano

ACKNOWLEDGEMENT

First of all I would like to thank my supervisor, Prof. Daniela Giordano, for the support she gave me during this period of intensive study. Under her guidance and her advices it was possible to get this achievement.

I sincerely thank my great friend and mentor Concetto. Thanks to his infinite patience, his untiring passion and his constant efforts I managed to get this great and well-deserved success. I hope I can continue to count on him as a guide not only in the work but also in daily life.

Much gratitude also goes to my friends and collaborators Carmelo, Isaak and Simone who have always supported me in realizing this thesis.

Finally, I want to extend a special thanks to my family and especially to my beloved wife Francesca. Thank you my love for believing in me, for always being there and for supporting me in every moment of this long and challenging path. Thank you infinitely my love... because being with you is REALLY the best you could wish for.

Contents

PREFACE

In computer vision, the creation of labeled datasets (ground truth) is of crucial importance both for performance evaluation and for algorithms' training. In particular, large scale visual datasets providing a wide coverage of different object classes and scenarios are nowadays needed because, beyond performance evaluation, would allow the development of "visual data-driven" approaches, which in the long run could be independent from human annotations.

Unfortunately, the ground truth generation task is tedious and error-prone and computer vision scientists usually dedicate a large part of their time to generate annotations tailored to specific application domains , thus a new trend in computer vision is to find methods for acquiring large scale ground truth data more efficiently and to combine effectively the existing datasets.

Motivations

In the last decade, the advancements in camera technology and the reduction of costs have led to a widespread increase in the number of applications for automatic video analysis, such as video surveillance [1, 2] and real-life study of animal species behavior [3, 16]. For all of these purposes, the scientific community has put a lot of effort in the development of algorithms for object detection [4], tracking [5] and recognition [6]. Of course, one of the most important stages in the development of such algorithms is the performance evaluation. Because of the varying nature of the targeted environments, it is very difficult – if not impossible – to devise an algorithm able to perform very well on all possible scene conditions (i.e. indoor/outdoor, different objects, different motion patterns, scene lighting, background activity, etc). For this reason, it is often necessary to establish the suitability of an algorithm to a specific application context by comparing its results to what are expected to be the correct results.

Therefore, large scale annotated datasets, covering as much scenarios and objects as possible, are needed in order to train and evaluate the existing approaches. The main limitation to achieve this goal is the daunting amount of time and human effort needed to generate high quality ground truth data, in fact it has been estimated that labeling an image may take from two to thirty minutes, depending on the operation, and it is, obviously, even worse in the case of videos. Indeed, in the context of object detection, segmentation, tracking and recognition, ground truths typically consist of a list of the objects which appear in every single frame of a video, specifying for each of them information such as the bounding box, the contour, the recognition class and the associations to other appearances of the same object in the previous or following frames. The manual generation of such information by a user is therefore a time-consuming, tedious and labor-

intensive task, since it requires the operator to be focused on drawing accurate contours and handling tracking information between objects.

Objectives

Despite the existence of several tools able to support users in the labeling task, in recent years much research has been devoted to find methods for automatic or semi-automatic ground truth generation.

This thesis is specifically focused on the development of methods for automatic ground truth generation and the problem is faced from two main perspectives: 1) to resort to human computer interaction techniques to ease the labeling process and 2) to exploit the crowd of users to generate reliable annotations. More in detail, we present a set of tools for collecting ground truth data by combining specific techniques i) to guide and speed up the annotation phase meeting the users' needs and ii) to build up a large scale database of labeled visual data to be used for performance evaluation of computer vision algorithms.

The integration of effective methods for quality control of the collected data and for the combination of multiple users' annotations is also needed to overcome the limitations of the existing approaches which are not user-centric and often not enough usable, since they are designed for a single, specific task.

To reach the objective of generating more diversified and larger annotated datasets, a collaborative method, exploiting a large population of skilled users [7, 8], has been proposed to improve functionalities and performances of the state-of-the-art methods [12].

Nevertheless, the usage of collaborative methods for image and video annotation, even when supporting tools are available, still requires a lot of

concentration, especially when the quality of the visual data is low or in the presence of crowded scenes, making the identification of objects of interest very difficult. In all these cases, both the identification and the subsequent annotation of objects are tedious and time-consuming tasks.

Besides semi-automatic and web-based collaborative solutions, crowdsourcing to non-experts has recently been adopted with the objective of supporting the generation of large-scale annotation datasets. Generally, most of the existing solutions rely on self-motivated persons. In order to provide further motivation, several crowdsourcing methods award the users with money for producing specific labeled data. On the other hand, if no money is available, the only way to follow is to engage and motivate users by exploiting their amusement, for example by playing a game.

To this end in this thesis we have developed an online game [66], named *Flash the Fish*, for generating large-scale object detection ground truth for video segments. The idea behind the game is to engage users to play a game by simply clicking on objects throughout the game levels. By using this mechanism no a priori knowledge is required for users who must only "take photos" of objects (in our case fish) providing an increasing dataset of annotations which can be used for detection and tracking performance evaluation.

However, crowdsourcing approaches lack mainly in mechanisms for assessing the reliability of annotators and for combining multiple users' annotations. Also, the usage of this solution necessitates resorting to specific post-processing techniques that require a considerable effort in developing algorithms able to process the raw data derived from the crowdsourcing.

Finally, since crowdsourcing is stirring large interest among the scientific community, we also propose some methods, which exploits the above mentioned on line game, for gathering video annotations, and demonstrate that starting from

big noisy annotations and by using simple image segmentation techniques and statistical approaches, it is possible to generate reliable ground truths for testing computer vision algorithms.

Thesis overview

According to what we have already discussed concerning the motivations and objectives of the proposed work, we first review the state-of-the-art on the existing annotation frameworks, followed by a detailed description of the requirements for the proposed systems. Then a comprehensive discussion of the suggested semi-automatic tools for ground truth generation is given, followed by the description of the adopted crowdsourcing techniques to obtain reliable annotations. After presenting the performance evaluation of the proposed approaches comparing the results of each method, a discussion of the results and the conclusions complete the thesis.

The outline of the thesis is as follows: *Chapter 1* contains the review of the state-of-the-art. Starting from a brief description of the existing ground truth applications, a review of the semi-automatic techniques, label propagation methods and crowdsourcing approaches is given. *Chapter 2* presents two developed tools, namely, GTTool [7] and PERLa [8] which are, respectively, a stand-alone tool and a web-based collaborative tool for annotating videos. *Chapter 3* describes a crowdsourcing approach based on the "Flash the Fish" game [66] together with data-driven approaches [88] to generate annotations starting from big noisy data as provided by the game's players. *Chapter 4* shows the performance evaluation of the proposed techniques by comparing the results obtained from

each methods. Finally, *Chapter 5* concludes the thesis and provides a discussion on the obtained results drawing the lines for future developments.

ABSTRACT

In this thesis a set of novel video annotation methods for performance evaluation of object detection, tracking and recognition applications is proposed.

Large scale labeled datasets are of key importance for the development of automatic video analysis tools as they, from one hand, allow multi-class classifiers training and, from the other hand, support the algorithms' evaluation phase. This is widely recognized by the multimedia and computer vision communities, as witnessed by the growing number of available datasets; however, the research still lacks in usable and effective annotation tools, since a lot of human effort is necessary to generate high quality ground truth data. However, it is not feasible to collect large video ground truths, covering as much scenarios and object categories as possible, by exploiting only the effort of isolated research groups.

For these reasons in this thesis we first present a semi-automatic stand-alone tool for gathering ground truth data with the aim of improving the user experience by providing edit shortcuts such as hotkeys and drag-and-drop, and by integrating computer vision algorithms to make the whole process automatic with

a little intervention by the end users. In this context we also present a collaborative web-based platform for video ground truthing which integrates the stand-alone tools and provides an easy and intuitive user interface that allows plain video annotation and instant sharing/integration of the generated ground truths, in order not to only alleviate a large part of the effort and time needed, but also to increase the quality of the generated annotations.

These tools are specifically thought to help users in collecting annotations thanks to the introduction of simple interfaces, which considerably improve and facilitate their work, also by integrating novel methods for quality control, but still remain a burdensome task with regard to the attention and time needed to obtain good records.

To motivate the users and relieve them from the tiresome task of making manual annotations, we devised strategies to automatically create annotation by processing data from the crowd. To this end we initially develop an approach based on an online game to collect big noisy data. By exploiting the information, we then propose data-driven approaches, mainly based on image segmentation and statistical methods, which allow us to obtain reliable video annotations by using low quality and noisy data gathered quickly and easily from the game. Also we demonstrate that the quality of the obtained annotations increases as more users play with the game making it an effective and valid application for the collection of consistent ground truth data.

CHAPTER 1

Literature overview

1.1 Introduction

Ground truth generation is a fundamental task in the design and testing of computer vision algorithms, thus in the last decade the multimedia and, more in general, the computer vision community have developed a disparate number of annotation frameworks and tools to help researchers in collecting datasets, which are then used in the tasks of image segmentation, object detection and tracking, face recognition and classification. The majority of the existing approaches for ground truth generation are "ad-hoc" tools created by isolated research groups, and as such, they are designed to fulfill specific requirements. The main limitations of these methods can be summarized as:

- the very simplistic quality control on annotations, since no means to evaluate the reliability of the user's work are considered;
- the lack of an efficient way to integrate annotations of the same object made by different users and, above all, a design conceived specifically for still images and not videos;
- the inability at generating large scale ground truth datasets because these methods exploit the effort of a limited number of people and do not support the sharing and propagation of labeled data.

Starting from these concerns, in the following sections an analysis of the existing approaches for ground truth generation is given.

1.2 Tools for ground truth generation

With the rising popularity and success of affordable and massive data sets of labeled video in computer vision, the community has put considerable effort for designing efficient visual annotation tools.

There exist, in the literature, a few attempts, such as Caltech 101 and 256 [9, 10] and the Berkeley Segmentation dataset [11], produced by some vision groups that have collected consistent annotated datasets, which, however, are too task-oriented and cannot be generalized since only segmentation algorithms can be evaluated, taking into account standard error measure for assessing their performance.

To reach the objective of creating more diverse and larger annotated datasets, several methods, exploiting large population of expert and motivated users, have been proposed [12, 13]. Generally, these methods rely on specific tools and

algorithm mainly conceived to meet the need of self-motivated investigators, which have extensive knowledge in computer vision research field. Since these techniques are based on quite complex architectures and interfaces, other approaches, which involve non-experts users, have been proposed [1, 14]. These approaches include crowdsourcing techniques which, however, lack mechanisms both for assessing the reliability of annotators and for combining multiple users' annotations. Moreover, most of the above approaches are not user-centric, being tailored to a specific task and not adapting to users' needs [15].

In addition, most of the common stand-alone approaches which include, for instance, ViPER-GT [17], GTVT [2], ODViS [18], are useful for specific application contexts, such as image labeling and object tracking, but lack any data sharing capabilities and they cannot be used for generating large scale ground truth datasets.

One of the most used application for this purpose is ViPER-GT [17], which is a baseline application for gathering ground truth data stripped of any intelligent method for assisting the annotation task. It relies only on its simplicity and its standard file format (XML), which contain all video metadata information inserted by the user. The tool provides a user interface with a spreadsheet representation of objects' data, timeline panels to navigate the video and view objects' life span, and metadata propagation features across multiple frames.

Although ViPER is widely used, it does not support automatic or semi-automatic processing, which might be implemented by adding a basic object detection/tracking algorithm to give hints to the user about likely object locations or tracking associations (although, of course, user supervision is still required to guarantee the correctness of the results). Furthermore it is a stand-alone application which does not provide any practical means to integrate different datasets and share them with other researchers.

In [19], the authors propose a ground-truth generation tool which employs very simple object detection and tracking algorithms to retrieve only object's bounding boxes and associate them across frames, yet allowing the user to add, delete or resize the bounding boxes and edit the associations. The major drawback of this approach consists of the absence of any annotations' quality control and in the possibility of using the framework only for specific video-surveillance applications.

The GTVT tool, described in [2], aims at improving the user experience with respect to ViPER. Although the tool shows a user-friendly interface for interacting with the framework, it is strictly designed for users who have some background of computer vision and does not allow to select ground truth for object that were not detected by the video-surveillance application under consideration. Also, no tracking information is managed for the labeled object in the image sequence.

On the other hand, ODVis [18] provides an interactive framework capable of supporting researchers to integrate video-surveillance modules, observe the behavior of these modules as they track and recognize events, interactively adjust tracking results and define ground truth data, as well as analyze the performance of tracking systems using a number of different error metrics.

In spite of the efforts to give users an effective method for template-based tracking approaches evaluation, many other ODVis functionalities should be extended in order to generalize the application context by allowing the system to support more video formats (e.g MPEG, FLV), to import/export the generated output file format and to integrate camera calibration information to relate the position of tracked structure across multiple views. Also, ODVis do not support any mechanism for performance evaluation of object detection algorithms, since it focuses only on tracking and event detection purposes.

Finally, in [20] two methods, namely *Minimean* and *Minimax*, are proposed to generate consensus ground truth for real images, which can be used for comparing edge detection techniques. The main drawbacks of this method are 1) the usage of the system specifically targeted to still images and not videos and, 2) the system seems to be suitable only for task-oriented purposes (e.g. edge detection) and cannot be generalized for recognition, classification and segmentation tasks. To overcome all these limitations, in recent years several semi-automatic tools, which integrate reliable object detection and tracking algorithms, have been proposed for supporting video annotation with the aim of minimizing users' involvement in gathering ground truth data, and ensuring high-quality labeled objects.

In the next section a review of the most renowned methods designed for these purposes is given.

1.3 Semi-automatic methods

Semi-automatic approaches concern all those methods which integrate specific image processing techniques (such as object detection and tracking modules) that might help user in gathering annotations in a simpler and more immediate fashion. As we previously discussed, the scientific community has put a lot of effort in designing efficient and affordable platforms to minimize the cognitive load of the user, still providing trustworthy ground truth data collection.

In line with this goal, much research has been carried out to build specialized interfaces tailored for video annotation. On the other hand, the rapid growth of the Internet have also favored, in the last years, the expansion of web-based collaborative tools, which take advantage of the efforts of large groups of people

in collecting labeled datasets. In [12] the authors introduced an early description of LabelMe, a database and an online annotation tool that allows the sharing of images and annotations providing functionalities such as drawing polygons, querying images, and browsing the database. Since the proposed tool was designed only for annotating images and not videos, in [21] the web-based platform was upgraded to be able to provide high-quality video labels with arbitrary polygonal paths using homography, preserving linear interpolation, and generating complex event annotations between interacting objects. However, LabelMe lacks intelligent mechanisms for quality control and integration of user annotations. In fact, quality control is achieved by a simple approach that counts the number of annotation landmarks, and it does not exploit the full potential of its collaborative nature (being a web-based platform) since annotations of multiple users of the same object instance are not combined. In fact, the LabelMe dataset, though being one of the largest datasets available, is notably inaccurate. As for previous tools, LabelMe was designed specifically for still images and although the video based version has been proposed in [21], it has not reached the same success of the image based version.

Similarly, in [22] the FlowBoost tool is presented. The method starts from a sparse labeling of the video, and alternates the training of an appearance-based detector with a convex, multi-target, time-based regularization. The latter re-labels the full training video in a manner that is both consistent with the response of the current detector, and in accordance with physical constraints on target motions. Although the proposed tool is able to cope with geometrical poses of greater complexity, only the location in the image plane, without variations in scale or orientation can be considered.

Agarwala et al. in [23] propose an original approach that uses a tracker as a more reliable, automatic labeling scheme compared to linear interpolation. The

peculiarity of this approach lies in the combination of computer vision techniques together with user interaction allowing the system to effectively track any sequence of images reducing user involvements and providing better reliability than "pure" computer vision systems. Although innovative and useful, the proposed method is not able to guarantee any quality control for the labeled objects. Also, there are many problems in computer vision and graphics in which fully-automated techniques produce imperfect results: in this case the user intervention becomes essential, significantly increasing his workload.

Again, Buchanan and Fitzgibbon in [24] discuss efficient data structures that enable interactive tracking for video annotation, while Fisher [25] simply discusses the labeling of human activities in videos where the sequences regard a public space surveillance task, and are ground truth labeled frame-by-frame by considering the bounding boxes and some semantic description of the activity in each frame.

In [26], a large benchmark video database of annotated television programs (TRECVid) was suggested, to support Video Retrieval Evaluation. What is important to highlight is that the proposed architecture allows a comparison of the collected data with others in an open, metric-based environment. Thus they provide shared data, common evaluation and often also offer collaboration and sharing of resources.

Finally, in [27], the authors present an approach for automatically collecting training data for human actions in Hollywood movies, showing that this data can be used to train a classifier for action recognition. The proposed method scales easily to a large number of action classes, providing a convenient semi-automatic tool for generating action samples with manual annotation. Since their methodology is strictly related to human actions, much more efforts should be

made to extend the application field and to improve the robustness of the proposed classifier to noisy training labels.

While all the above tools are effective at building large datasets, an important issue to deal with concerns the cost to obtain high quality labels, sensibly reducing, at the same time, the workload of the user. In fact, in order to scale up to the next generation of datasets, a system that should be able to guarantee high quality annotation is needed, without exhausting the scientific community's funding and tiring our workers.

To address this last topic, in the next sections, a detailed review of the label propagation methods together with some of the most renowned approaches based on crowdsourcing, is given.

1.4 Label propagation methods

The fundamental reason to investigate label propagation methods is the need of reducing the manual effort in the labeling process. Generally, the process of inferring images' labels from community-contributed images and noisy tags is performed by considering the available small dataset of labeled data as a reference and attempting to classify the large dataset of unlabeled images through a set of specific semantic methods to extend the dataset of labeled data.

To this aim, in recent years, multimedia researchers have engaged in machine learning techniques to improve image annotation performance, paying special attention to techniques to access, organize and extend the massive image labeled datasets available. However, the effectiveness of these machine learning techniques heavily relies on the availability of a sufficiently large set of labeled samples, which typically comes from a large group of heterogeneous users in an

interactive, manual process. Instead, this data is often available on small quantities, while unlabeled data may be abundant, hence much research has been directed towards the exploitation of unlabeled data together with labeled data with the aim of inferring new annotations from the set of labeled visual content.

To this purpose, many semi-supervised learning or active learning approaches have been proposed [28, 29].

Also, many traditional methods, such as the support vector machine and *k*-nearest neighbors (kNN) method, have been applied to infer images' labels from the user-shared images and associated tags, starting from the assumption that closed data points tend to have similar class labels. By exploiting this simple concept, in [30] the author dealt with the semantic labeling problem by using an SVM-based scheme to transfer the massive 2D image labels from ImageNet [31] to point clouds. Also they proposed a graphical model to integrate both the intra-image and inter-image spatial context in and among reference images to fuse individual superpixel labels onto 3D points.

To annotate the images more accurately, in [32] the authors propose an innovative, *sparse graph-based semi-supervised learning* method which uses the labeled and unlabeled data at the same time, proving that the method is more effective than purely supervised learning when the training data is limited [33, 34]. Graph based semi-supervised learning methods [32] for image annotation are also used in [28, 35], but they show a common disadvantage, that is they all have certain parameters which require manual tuning. Similarly, one of the most typical approach involves the adoption of *Gaussian Random Field* (GRF) and harmonic function methods as well as the local and global consistency [36], which, as in the previous cases, needs some specific parameters to be manually adjusted. On the contrary, in [37], the same issue is addressed under a different point of view. In fact, the proposed work is concerned with the question on how to make a better

use of the provided labels already in the early stages of popular semi-supervised learning techniques. Here, the main contribution consists of employing a metric learning approach to improve the construction of a graph which exploits the neighborhood structure to identify highly connected nodes leading to a consistent improvement in performance. In fact, graph-based methods are very popular in semi-supervised learning due to their well-founded theoretical background, intuitive interpretation of local neighborhood structure and strong performance on a wide range of challenging learning problems. Also, the authors proposed methods of querying more informative labels based on domain knowledge following an approach that is complimentary to traditional active learning settings.

Again, in [38] a comparison between local linear neighborhoods and global linear neighborhoods approaches is given, showing how the global structures among the data points, ensure more robust and reliable similarity graphs for label propagation.

Finally, a probabilistic collaborative multi-label propagation formulation for large-scale image annotation, which is founded on Kullback-Leibler divergence based label similarity measurement and scalable l_1-graph construction, is proposed in [39]. By using a locality sensitive hashing-based scheme [40, 41, 42] the construction process of the l_1-graph becomes scalable and faster, with respect to the candidate selection of similar neighbors for one image.

Since the available datasets of labeled data are usually obtainable from the Web, the most widely used procedure [43, 44] is to query an image by first search for similar images and then mine representative and common descriptions from the surrounding tags of these similar images as the annotation for the query image. In line with this theory, Torralba et al. in [45] claimed that with enough number of samples, the simple nearest neighbor classifier can achieve reasonable performance for several object and scene detection tasks, when compared with

the more refined state-of-the-art systems. However, the assignment of only one noun to each image and the use of *32-by-32* pixels image lacks to reflect the complex content of real-world scenarios.

To overcome these limitations, the linear neighborhood propagation method [46] has been considered in which the sample reconstruction method is used to construct a graph. It has been proved that in most cases, linear neighborhood propagation is more effective and robust than the traditional semi-supervised methods on similarity graphs [47], but still cannot handle the links among semantically unrelated samples. As a consequence, visual similarity does not guarantee semantic similarity, which in general is conflicting with the behavior of many generative-based image annotation tools. To this aim, in [48] a solution to the mismatch problem between semantic and visual space has been proposed. Starting from the assumption that each unlabeled image corresponds to one topic implied by the training image, the method adaptively models a local multi-label classification indicator function which captures the keyword contextual correlations and also exploits the discrimination between visual similar concepts. Though innovative and reliable, unfortunately the procedure of finding the neighborhood of an image for large datasets, is a very time-consuming task. Also no information is gathered by exploiting the Web which could enhance the estimation of semantic and visual similarity.

1.5 Crowdsourcing methods

C*rowdsourcing methods* are defined as a set of practices aiming at obtaining needed services, ideas, or content by soliciting contributions from a large group of people (also from the Web community), rather than from traditional employees or

suppliers. This process is often used to subdivide tedious work by combining the efforts of numerous self-motivated volunteers or part-time workers, where each contributor of their own initiative adds a small portion to the final result.

The use of this kind of methodologies has the objective to allow the collection of raw but, at the same time, meaningful data which can be used to address the tedious task of collecting ground truth in order to enable the build up of large scale annotation datasets.

Since these methodologies typically rely on users' motivation and quality control for creating reliable image and video annotation, the scientific community has turned a lot of attention on crowdsourcing to human resources (non-experts) [49, 50] demonstrating the utility of this approach with respect to the annotation task. Nevertheless, two main aspects have to be taken into account when crowdsourcing: workers' motivation and quality control. The easiest and most natural way to motivate people is paying them for their work. This strategy is applied by Amazon's Mechanical Turk service [14], which has revolutionized static data annotation in vision, and enabled almost all large-scale image data sets collected since then to be labeled [51, 12, 52], and CrowdFlower [53].

A valid alternative for workers motivation is personal amusement: this is the case of the ESP and Peekaboom games [54, 55] which exploit players' agreement (randomly pairing two players and let them guess each other's labels) to collect ground truth data. While fun, these games aim at producing high level labels which describe the contents of the image, providing no means to acquire lower level data (e.g. object contours). Moreover, these games do not offer any means of quality control and the annotation integration mechanisms adopted are rather primitive.

In [56], a Bayesian generative probabilistic model for the annotation process is proposed. The major contribution of this work lies in the improvement of the

current state-of-the-art methods for crowdsourcing by introducing a more comprehensive ad accurate model of the human annotation process. In particular, the authors gained insight into this procedure by learning a richer representation that distinguishes amongst the different sources of annotator error. Despite the possibility to infer not only the class of an image, but also parameters such as image difficulty and annotator competence and bias, the proposed work considers only binary labels of image from many different annotators, lacking in an effective mechanism to integrate and also validate annotations deriving from different categories of users.

In fact, besides workers motivation, another concern of crowdsourcing solutions is the quality control over annotators, which has been tackled with different strategies that can be summarized [57] as: Task Redundancy (ask multiple users to annotate the same data), User Reputation and Ground Truth seeding (i.e. coupling ground truth with test data). Although these solutions are able to build large scale datasets, they might be very expensive and contain low quality annotation since workers (even if paid) are not as motivated as researchers.

To this end, Vondrik et al. in [50] outlined a protocol for generating high-quality annotations by classifying the workers into three types and describing methods for identifying each: *good* workers who are honest and skilled; *bad* workers who are honest but unskilled and *ugly* workers who are dishonest and cheaters. Thus, the aim of this protocol becomes to eliminate bad and ugly workers, which unfortunately cause a decrease of the throughput. To overcome this limitation, the authors also propose a mechanism based on workers compensation with respect to the workload needed for labeling more or less objects also considering the related difficulty. Even in this case, a policy for rejecting bad annotation is used to inform workers about the possibility to discard

poor work and this is a strategy to motivate workers in advance to produce high quality labels.

Similarly to this last work, but consistently saving a lot of money to provide reliable annotations, in [58] the authors explored the use of Amazon's Mechanical Turk system [14], to provide a cheap and fast evaluation of non-expert annotations for natural language tasks. As a result, they showed high agreement between Mechanical Turk non-expert annotations and existing gold standard labels provided by expert labelers, founding that from many tasks, only a small number of non-expert annotations per item are necessary to equal the performance of an expert annotator (e.g. 4 non-expert labels per item in order to emulate expert-level quality for a recognition task).

However, it is not so simple to convince users to be involved in the annotation task, especially if no money is available. For this reason in *Chapter 3*, a set of crowdsourcing approaches based on personal amusement will be discussed.

CHAPTER 2

Semi-automatic ground truth generation

2.1 Introduction

In order to support users in dealing with the onerous task of ground truth generation, several tools have been developed to provide them a set of simple and intuitive graphic interfaces to detect and draw object features such as contours, bounding boxes, membership class, tracking information and, more in general, object metadata. As we already said, one of the most common application designed for these purpose is ViPER [17] which provides a standard XML file containing object metadata information inserted by the user. Starting from this

assumption, the application described in the following sections, called GTTool [7], aims at:

- providing an easy-to-use interface, specific for generating ground truths for object detection, segmentation, tracking and classification;
- improving the user experience with respect to ViPER, by showing two panels, each containing a frame at a different time, thus allowing the user to compare a frame's annotations with those from a previous or following frame, and providing quick methods to specify object associations and perform attribute propagation (e.g. hotkeys, drag-and-drop);
- integrating automatic tools for object segmentation and tracking, effectively reducing the number of objects/frames to be manually analyzed;
- error handling by allowing the users to correct and edit their annotation whenever they want;
- supporting ViPER XML format, for ground truth importation.

To evaluate the performances of our approach in *Chapter 4* we show a comparison between GTTool and ViPER concerning the generation of ground truth for a video file. To be more precise, our evaluation approach is based on the assessment of time required to label the video with each tool and on the accuracy analysis of the generated contours, compared with those obtained from a higher resolution version of the videos.

Below the details of the proposed application together with the peculiarities of the user interface are given.

2.2 GTTool

The proposed tool relies on a modular architecture (Fig. 2.1) which allows users to define the ground truth by using an easy graphical user interface (GUI). The developed application integrates a number of computer vision techniques, with the purpose of enhancing the ground-truth generation process in terms of both accuracy and human effort. In particular, Active Contour Models (ACM) are integrated to automatically extract objects' contours; object detection algorithms and state-of-the-art edge detection techniques are employed in order to suggest to the user the most interesting shapes in the frame. Moreover, by using a two-window GUI layout, the application enables the user to generate tracking ground truth through straightforward drag-and-drop and context-menu operations. The user can also open previous ground-truth XML files in order to add new objects or edit the existing ones and save the performed improvements to the same or a new file.

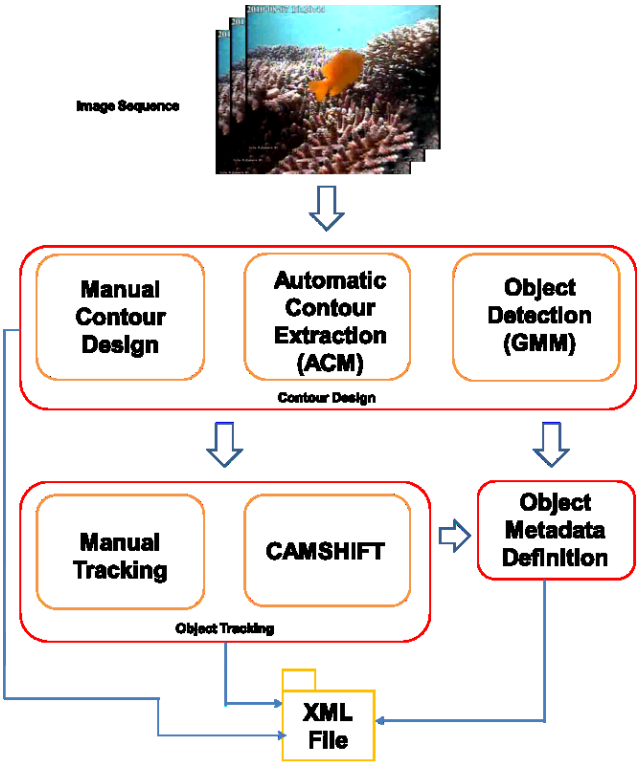

Figure 2.1 – Ground truth generation flowchart

2.2.1 Automatic contour extraction

In order to make ground truth generation faster, automatic contour extraction techniques have been integrated. In particular, when the object's boundaries can be clearly identified (i.e. the object's border colors differ substantially from the background in its vicinity), the application is able to automatically extract the object's contour by using one of the following methods:

- Snakes [59];
- GrabCut [60];

- Snakes with Canny contour enhancement.

To accomplish this, the user has to draw just a bounding box containing the whole object and choose one of the available techniques for automatic contour extraction from the corresponding panel (Fig. 2.2).

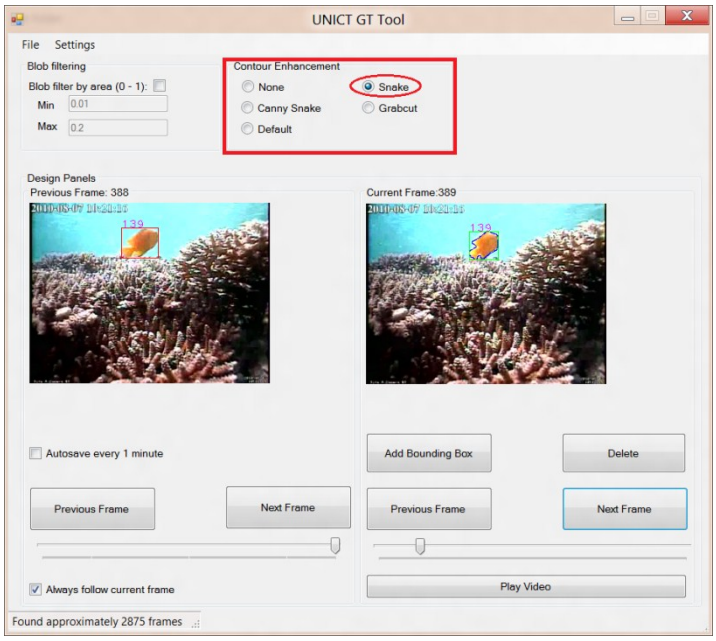

Figure 2.2 – GUI for automatic contour extraction

2.2.2 Manual contour extraction

As in nearly every common ground-truth generation application, the developed tool allows the user to draw ground truths manually by using the pencil tool or the polygon tool to trace the contour of an object of interest.

Though slow and tedious to the user, the usage of these tools is often necessary, because the automatic contour extraction methods may fail to segment correctly the objects of interest.

2.2.3 Automatic object detection and tracking

While automatic contour extraction allows the user to extract object contours in an automatic and easy way, object detection aims at identifying possible interesting objects, and to do so the Gaussian Mixture Model algorithm (GMM) [61] is employed. At each new frame, the GMM algorithm detects moving objects and allows the user to automatically add the detected objects' contour to the generated ground truth by using the object's context menu (Fig. 2.3). Because the GMM algorithm needs to be initialized with an adequate number of frames, this method performs progressively better in later stages of long video sequences.

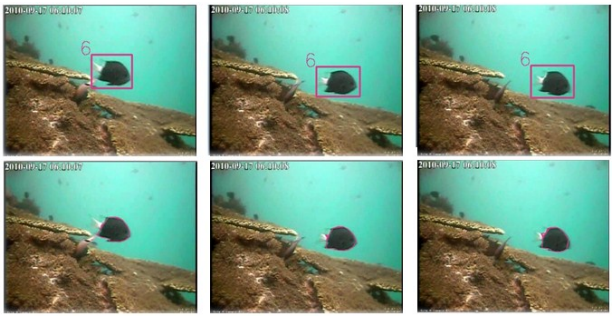

Figure 2.3 - Automatic Object Tracking and detection: In the top row the output of the tracker is shown, while in the bottom row the output of the automatic detection module is shown.

In conjunction with the GMM algorithm, CAMSHIFT [62] is used to generate automatic object tracking ground-truth data. The algorithm takes as input the objects identified in the previous frames and suggests associations with the objects localized (either manually or automatically) in the current frame (Fig. 2.3). As in the case of automatic object detection, the user is always given the choice to accept or refuse the suggested associations.

2.2.4 Manual object tracking

As aforementioned, the two-window GUI layout makes the task of creating tracking ground truth easier to the user. The right window always shows the current frame, while in the left window the user can select one of the previously labeled image. By using the right window's objects' context menus, the user can specify the associations to the objects in the left window (Fig. 2.4).

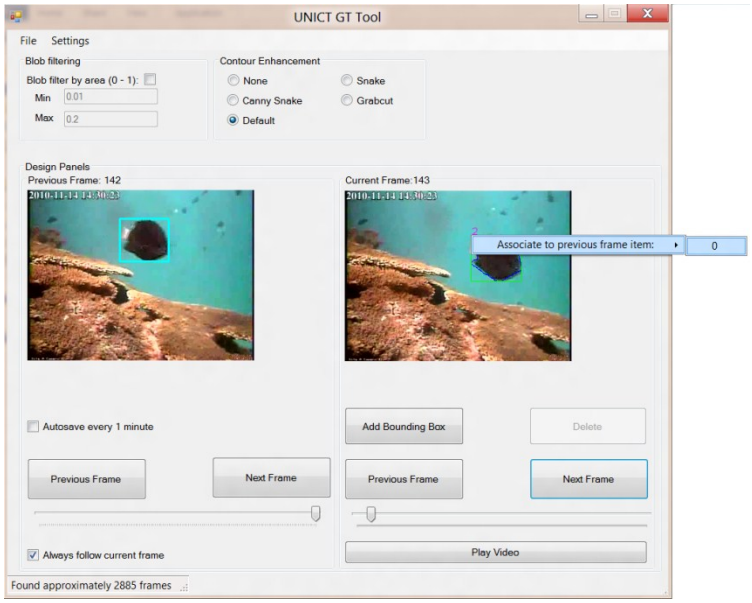

Figure 2.4 – Manual object tracking

2.2.5 Metadata definition

Besides object segmentation and tracking, it is possible to add arbitrary metadata, such as for classification purposes, by defining labels and assigning values to each object. When used in conjunction with tracking, this metadata is

automatically propagated across all instances of an object, thus requiring the user to specify them only once.

The set of annotations added to a video can be exported to an XML file, for example to simply store it or to share it with others. An example of the XML format we use is shown in Fig. 2.5. In order to make the adoption of GTTool easier to ViPER users, the application allows to import and convert ViPER files to GTTool's schema, so no loss of previous work occurs when switching from the former to the latter.

```
<?xml version="1.0" ?>
<video name="test_video">
    <frame id="0">
        <object trackingId="0" instanceId="0">
            <bounding_box>
                <point x="170" y="189" />
                <point x="170" y="230" />
                <point x="206" y="230" />
                <point x="206" y="189" />
                <point x="170" y="189" />
            </bbox>
            <contour>
                <point x="17" y="17" />
                ...
            </contour>
            <metadata>
                <label name="class">pedestrian</label>
                <label name="action">walk</label>
                ...
            </metadata>
        </object>
        ...
```

Figure 2.5 – Example of GTTool's output XML file

2.3 PERLa

Starting from the previous work, we subsequently propose a web-based collaborative approach for video annotation, which is based on the same architecture of GTTool, but it has been mainly conceived to integrate effective method for quality control of the collected data and for the combination of multiple users' annotations.

This tool, currently, is being used to collect large scale ground truth on underwater video footage gathered for the Fish4Knowledge project[1] which aims at developing automatic video and image analysis methods to support marine biology research.

The strengths and limitations of the existing video and image annotation approaches have been largely discussed in the previous sections, in the following subsections we first describe the proposed framework, highlighting functionalities and improvements with respect to the state-of-the-art, then the collected content on the aforementioned underwater environment is presented. The performance analysis is instead given in *Chapter 4* where a detailed analysis concerning the accuracy of the generated ground truths, the efficiency of the platform in collecting annotations, its learnability and user satisfaction is given.

2.3.1 General description of the web annotation tool

The proposed tool[2] is a web-based collaborative environment which allows users to share their own annotations with others accelerating high quality video ground truth generation process by increasing/integrating the number of annotations in a sort of inherent user supervision.

[1] www.fish4knowledge.eu
[2] http://f4k.ing.unict.it/perla.dev

Given an input video stream, our platform extracts video frames and provides a set of utilities to annotate each video frame and to follow objects across frames. It is a rich internet application, based on a standard client-server architecture: the client is implemented in Silverlight while the server's logic and the communication with the database is developed in C#.

In the next subsections a more detailed description of the proposed application's GUI is given.

2.3.2 The workspace

Immediately after login, the user is presented with a private workspace where it is possible to review past activities (Fig. 2.6).

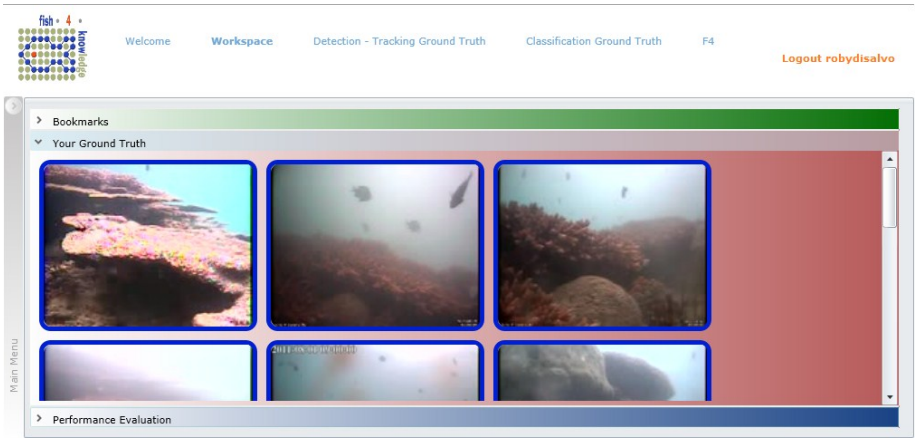

Figure 2.6 – User workspace in PERLa

The workspace serves also as a shortcut to the labeling operations already performed and is divided in the following sections:

- ***Bookmarks***: the use of bookmarks is necessary to reference videos in case of large video collections as the one previously discussed, which contains about half million videos;
- ***Ground Truths***: in this section (Fig. 2.7), the user can manage the ground truths that she owns. In particular, by using the context menu's option a user can create a new ground truth, modify a previously generated one, or derive a new ground truth from an existing one. Moreover, the context menu deals with the collaborative aspect of the platform by allowing the users to make available their ground truths to the other users of the platform. In this way, the users can benefit from the existing annotations and apply the appropriate changes instead of having to generate a new ground truth from scratch.

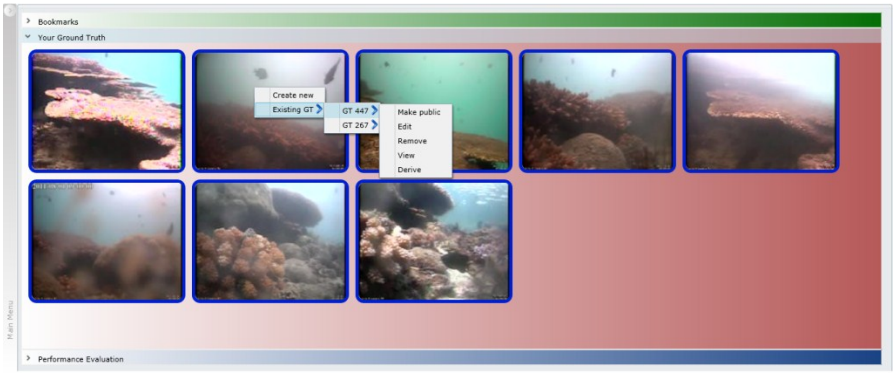

Figure 2.7 – The ground truth management part of the application. In this section, the videos for which the user created at least one ground truth are included

2.3.3 Video selection

By clicking on the "Detection—Tracking Ground Truth", the user is presented with the video selection screen (Fig. 2.8) where it is possible to browse all the available videos, filter them according to specific criteria, bookmark them and start the annotation application. The search engine allows users to limit the number of the shown videos by defining criteria regarding the videos' resolution, acquisition time, enabling the user to select videos with specific features (e.g. day or night) and the exact date of the acquisition.

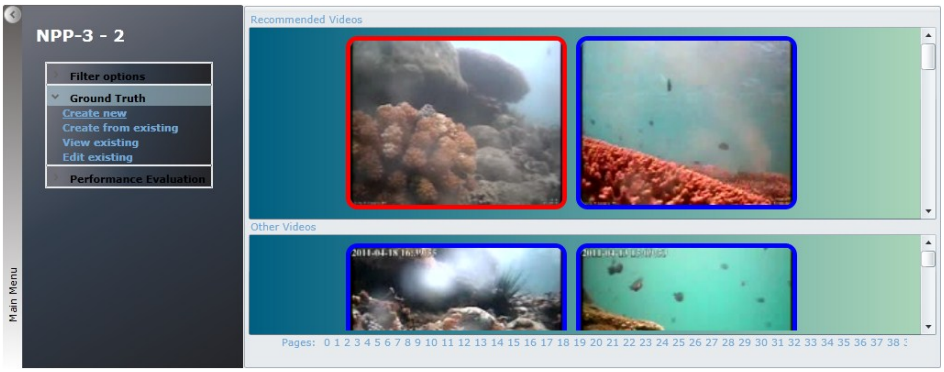

Figure 2.8 – The video selection window

2.3.4 Ground truth generation

Once the user identifies the videos she wants to create ground truth for, she can initiate the labeling process by launching the annotation application. This part of the platform permits to create annotations by using multiple windows. Each drawing window (Fig. 2.9, top left) shows one image and, by using the available toolbox (Fig. 2.9, bottom), annotations can be drawn on it.

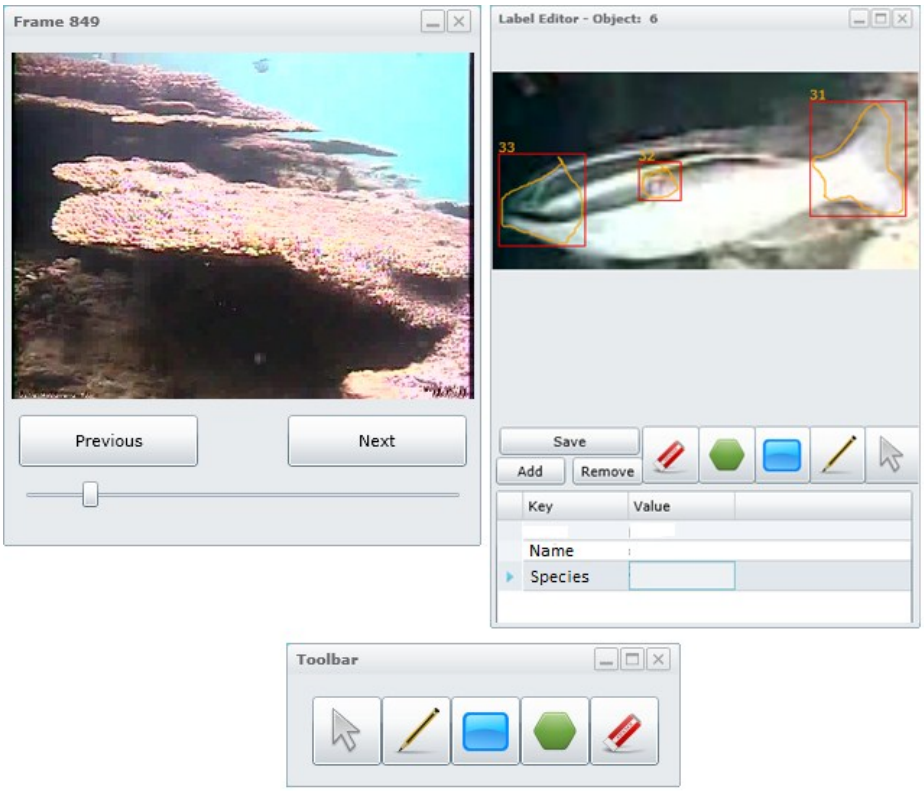

Figure 2.9 – *Top left*: A drawing window that shows an image to be annotated. The Next and Previous button and the slider located at the bottom of the windows allow the user to navigate through the video. *Top right*: A labeling window aiming at supporting the user to annotate sub-parts and for adding textual metadata to the designed object. *Bottom*: The toolbar. From left to right the bounding box selection, pencil, rectangle, polygon and eraser tools

2.3.5 Object detection ground truth and contour drawing

The proposed application offers the basic tools (polygon and pencil) to support users in the task of manually drawing object contours. However, manual annotation is discouraging in lengthy image sequences where the numbers are overwhelming. For example, one of the most populated videos in our repository, contained about 18,000 fishes on a 10 min, low resolution, 5 fps video clip. Under

these conditions any means assisting users in drawing object contours as efficiently as possible seems necessary. To this end, the proposed tool implements three automatic contour extraction methods, *Grabcut* [60], *Snakes* [59] and *Canny* [63]. These algorithms were chosen because not only they are well established and tested methods for contour extraction, but also they offer the best ratios in terms of resources and quality of the results. The automatic contour extraction can be applied by drawing the bounding box containing the whole interesting object, right clicking on it and selecting from the "Contour Extraction" sub menu one of the available methods (Fig. 2.10). This is a trial-and-error process that does not always yield the desired result, because the success of the automatic image contour extraction algorithms depends on the type of image used on (image color patterns, contrast etc.).

In case of automatic contour extraction failure, the user can resort to the manual drawing tools.

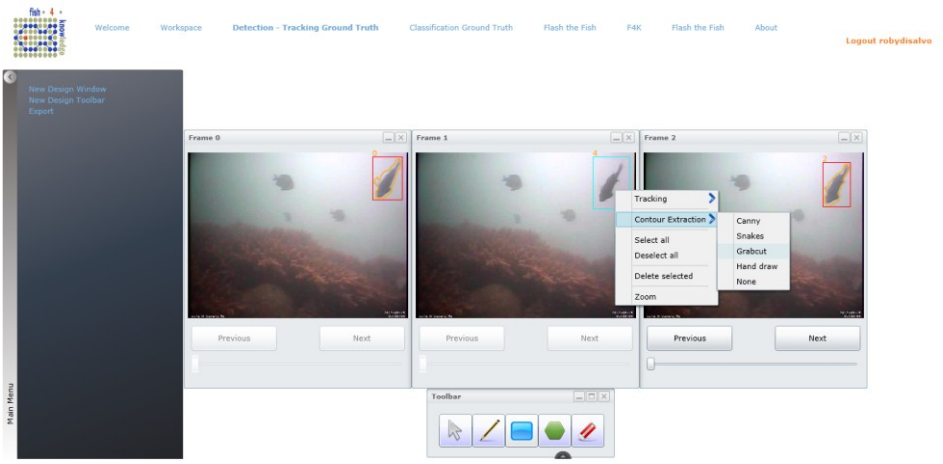

Figure 2.10 – Semi-automatic contour extraction applied on the center drawing window's image

After an object is drawn, the user can further annotate subparts of it (Fig. 2.9, Top left) by employing the same tools described above. Furthermore, from the same window, the user is able to add textual metadata to the object (that are included in the exported XML file) that can be useful in other image processing contexts (object recognition, image segmentation, information retrieval etc.).

2.3.6 Object tracking ground truth

In the proposed tool, the ground truth generation for object tracking exploits the capabilities of multiple windows applications in order to implement an easy-to-use and intuitive way to follow objects across consecutive frames. In particular, to be able to annotate multiple instances of the same object in consecutive frames, the user must arrange side-by-side multiple drawing windows. When the user places two windows with their boarders in direct contact, they become, what we call, a "drawing chain". While chained, the Next and Previous buttons and the sliders of all the drawing windows are disabled except from the last one's (the rightmost), which serves as a control to navigate through the image sequence. Moreover, all the chained windows maintain all the drawing functionalities as if they were unchained. When an adequate, for the user's needs, chain is formed the user must draw an object and bring up the context menu by right clicking on it, then select the voice "Tracking" and select an object from the previous frames she wants to assign the clicked object to (Fig. 2.11).

When used in high resolution desktop setups, the application can create multiple long chains (as shown in Fig. 2.11) of successive frames in the same annotation instance (about 3 chains of 6 windows using a 1920×1080 resolution, more on a multi-monitor setup).

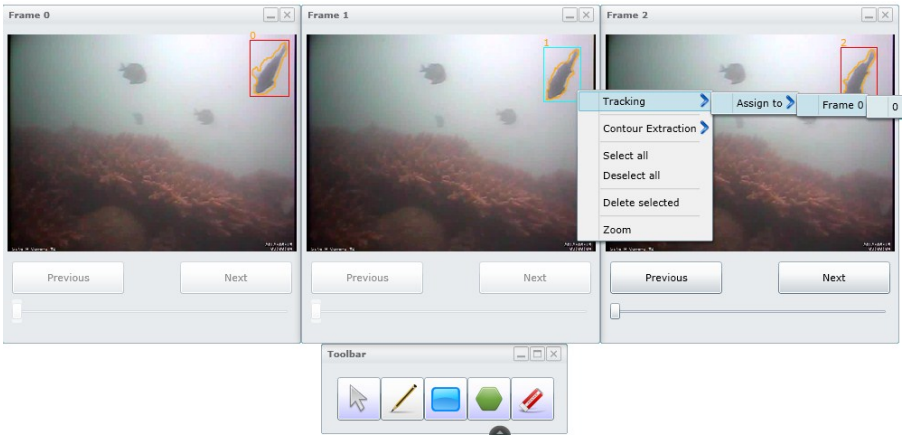

Figure 2.11 – A three-window chain for tracking ground truth generation

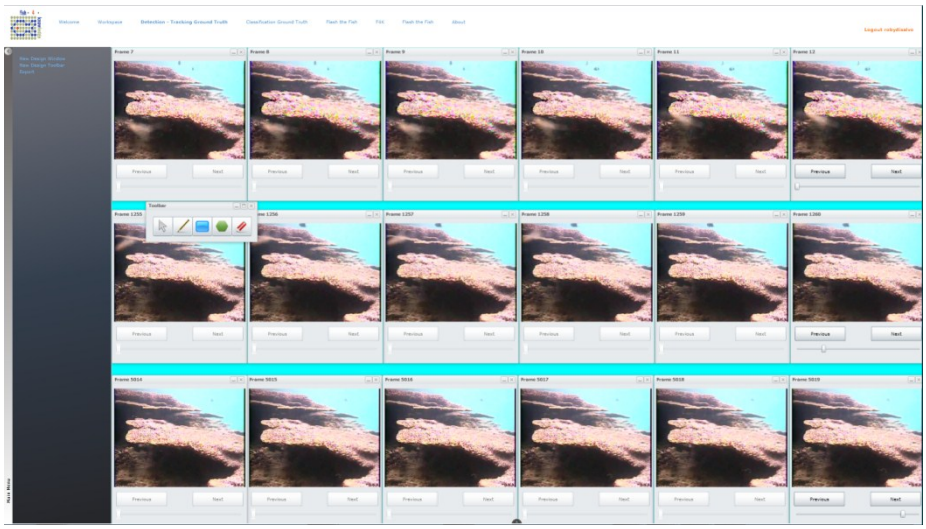

Figure 2.12 – Three independent six-window chains

2.3.7 Combining users annotations

The collaborative nature of the proposed tool implies that there may exist multiple annotations of the same object. Such multiple annotations are combined

in order to produce a much more accurate object representation since we can safely assume that combined opinions are more objective than single ones [64, 65].

The underlying idea is that for each video clip we can have more ground truths, annotated by the same or different users, which are integrated by adopting a voting policy in order to generate the one herein called "best ground truth".

The "best ground truth" (BGT) building process (see Fig. 2.13) involves two basic steps: i) add new annotated objects to the BGT, ii) integrating objects' contours.

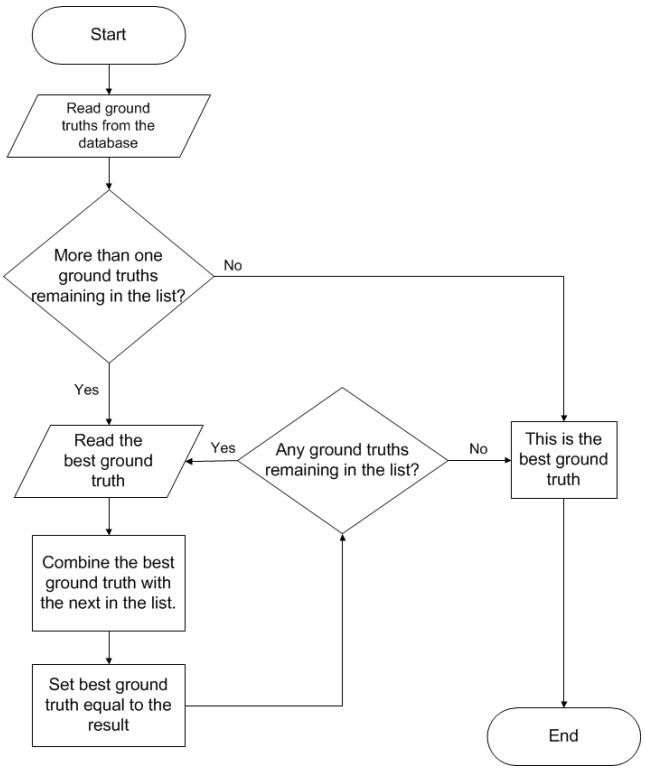

Figure 2.13 – Flowchart of the "best ground truth" building process

Let us suppose that the *BGT* has been already built for a given video *V* and a user annotates *V* again. For each new annotated object *A*, two cases may occur:

- ***New object instance***. The object *A* has been never annotated and it is added directly to the *BGT*. This exploratory strategy avoids limiting the number of objects on each ground truth; however, to prevent noisy ground truths, each object instance in the *BGT* comes with a number describing the number of annotators that have labeled it over the total number of annotators, thus allowing us to filter out the object instances which have received few annotations;
- ***Existing object instance***, i.e. there is already an instance (referred in the following as GT) of object *A* in the *BGT*. In this case we assess a matching score between object *A* and object GT and if this score is greater than a given threshold (in our case 0.75) the contours of *A* will be combined with the ones of GT. The matching score is computed as weighted mean of the two following measures:
  - *Overlap Score*. Given the object *A* and the corresponding object GT of the best ground truth *BGT*, the overlap score, O_{score}, is given by:

$$O_{score} = \frac{area(A \cap GT)}{area(A \cup GT)} \qquad (1)$$

  - *Euclidean Distance Score*. Pairwise Euclidean distance between *A* points (X, Y), with $(X_i, Y_i) \in A$, and *GT* points (x, y), with $(x_{i'}, y_{i'}) \in GT$, computed as:

$$E_{score} = 1 - \frac{\sum_i^n \sqrt{(X_i - x_{i'})^2 + (Y_i - y_{i'})^2}}{\max\left(\sum_i^n \sqrt{(X_i - x_{i'})^2 + (Y_i - y_{i'})^2}\right)} \quad (2)$$

Usually, a resampling procedures is applied, in order to equal the number of points in the two contours.

The objects' contours combination is based on the assumption that the "best ground truth" contours are more accurate than the new ones since they result from the combination of multiple annotators. In detail, once a new object is considered for being part of the "best ground truth" (see above) its contours C_A are combined with the contours C_{GT} of the corresponding "best ground truth" object to form the new object contours C_{NGT}, where each point is computed as:

$$C_{NGT}(i,j) = \frac{1}{2^{N-1}} \sum_{n=1}^{N} \left(\omega_A \times C_A(i,j) + C_{GT}(i,j)\right) \quad (3)$$

where $\omega_A \in [T, 1]$ (where T is the threshold described above, and is set to 0.75) is the matching score between A and GT computed as above described and N is the number of different annotations for that given object. Figure 2.14 shows the result of a combination of four annotations (one belongs to the already existing best ground truth) on the same object, whereas Fig. 2.15 shows how object contours evolve as the number of annotators increases.

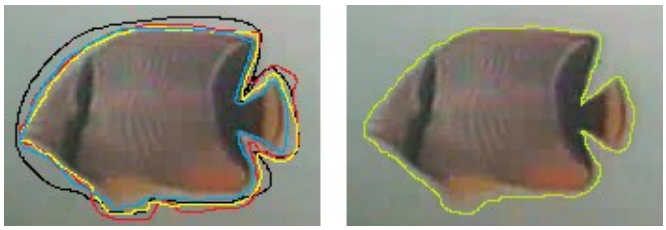

Figure 2.14 - Combination of annotations for building a "best ground truth" object. *On the left* there are four annotations: three (black, yellow, red) from different users and one (blue) belonging to the already existing best ground truth. *On the right*, the resulting contour to be assigned as the new best ground truth

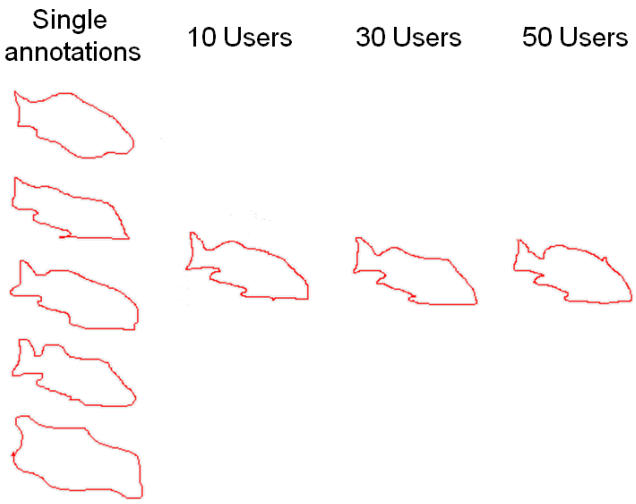

Figure 2.15 – Object contours quality improvements as the number of annotators gets bigger

Finally, a quality score is assigned to the user (U_{qs}) that represents her ability in ground truth creation, equal to:

$$U_{qs} = \frac{1}{N}\sum_{i}^{N_{GT}} q_i n_i \qquad (4)$$

where N is the total number of objects that the user has drawn, N_{GT} is the number of the created ground truths, q_i is the quality of the i_{th} ground truth and n_i is the number of objects belonging to that ground truth.

2.3.8 Data content

The proposed tool has been conceived for ground truth data collection within the Fish4Knowledge project, whose video repository holds more than half a million videos at different resolutions and frame rates. Those videos are acquired by eight high definition underwater cameras that are set to work 24/7.

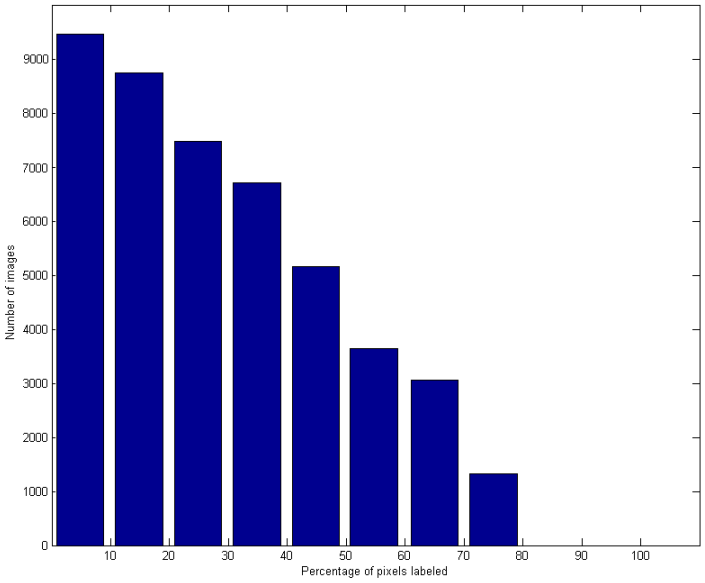

Figure 2.16 – Histogram of the number of images with respect to the pixel coverage

At the moment, our database contains 55 annotated videos with 55.332 annotations (about 2,900 different objects) in 24.136 video frames, collected by several users with our web-based tool, which is online since July 01, 2012.

Figure 2.16 shows the histogram of the total number of annotated images with respect to the percentage of labeled pixels. In particular, 10.034 frames have less than 10% of pixels labeled and no image has more than 60% of pixels labeled.

The histogram of the number of images per the number of objects in these images (see Fig. 12), instead, shows that there exists a high number of images with only one annotation (a little more than 11.000).

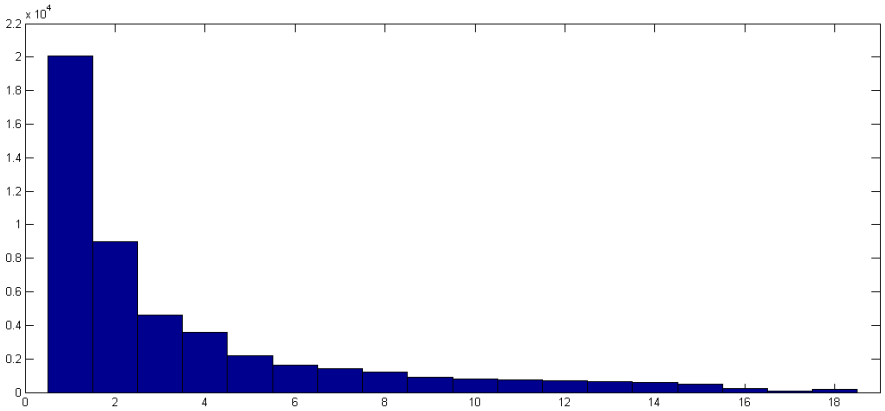

Figure 2.17 – Histogram of the number of images with respect to the number of objects present

Currently, the tool's database is constantly growing up, since more and more new users are working on the annotation of new image sequences. At the current rate, we estimate that about 350 10-min videos annotated by the end of 2013, resulting in about 500.000 annotations of about 25.000 different objects.

CHAPTER 3

Crowdsourcing approaches

3.1 Introduction

As discussed in *Chapter 1*, all the tools for collecting ground truth proposed by the scientific community do not meet some important requirements:

- to generate cheap and mid to high quality annotation in the least amount of time possible;
- to involve workers through individual gimmicks that will enhance their motivation;
- to provide a significant quality control with respect to the produced work.

In line with the growing number of users which use the Web and considering the need of relieving the operators from the onerous task of gathering annotation for testing computer vision algorithms, in the following subsections a set of crowdsourcing methods are proposed.

We first start by exploiting the peculiarities of the "Flash the Fish" online game [66], where the user is shown videos from underwater environment and has to take photos of fish by clicking on them. The collected "clicks" are then used to drive some computer vision techniques for automatic image segmentation based on Gaussian Mixture Model (GMM), exploiting visual features around the gathered points.

More specifically, the main contributions of the proposed work are:

- to show that reliable video annotations can be derived by using low quality and noisy data gathered quickly and easily by playing with the game;
- the integration of crowdsourcing and user quality control methods to provide a valid support for creating reliable image and video annotations;
- to show that the quality of such annotations increases as more users play with the game, making it an effective and valid crowdsourcing application for the collection of ground truth data.

In the next paragraphs, a description of the *Flash the Fish* game focusing on its advantages over classical video annotation applications is given. Then the ability of the game to generate accurate annotations starting from big-noisy data is described. Finally, for assessing the performances of the proposed approaches, we refer to *Chapter 4* in which a comparison of the collected data with the hand-drawn ground truth, is discussed.

3.2 Online Game for Ground Truth Generation

Flash the Fish exploits the amusement strategy to generate large scale object detection ground truth.

Playing with the game is simple: the user is presented a segment of an underwater video and she has to take photos of the fish, by clicking on them (Fig. 3.1) gaining as many points as possible. The user needs to gather a certain score to advance to the next game levels. Each "photo" contributes in estimating the presence or absence of fish at the corresponding point in the video.

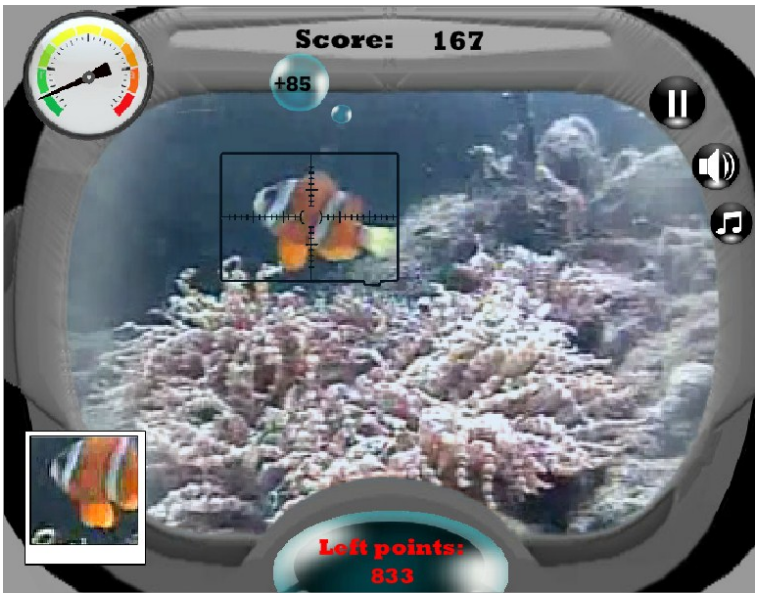

Figure 3.1 - The game's interface. *On the top left*, the manometer shows the oxygen remaining (time) before the level ends. *On top*, the acquired points and on the top right three button controls to pause the game, mute the music and the sound effects, respectively, can be seen. *On bottom left*, the last taken photo is shown and on bottom the points needed to advance to the next level are shown. Finally, *the central area* shows the video and the camera's shutter, which is center on the mouse's pointer.

The game consists of *7* different levels of progressively increasing difficulty. Every time a game session starts, a list with *7* random selected video segments, taken from our repository that contains more than *600.000 10-minute* videos, is generated. The first level serves the role of assessing the skills of the player (see next section) and has an initial frame rate of *5 FPS* and the time available is *35 seconds*. At each successive level the frame rate of the video segment is increased by one, while the time available is reduced by *2 seconds*, to a maximum of *11 FPS* and a minimum of *23 seconds* at the seventh and last level. The game can be found at http://f4k.dieei.unict.it/fish_game/.

In order to make the game more appealing, we adopted a scoring system that rewards users according to the quality of their annotations. In other words, the more precise the user is, the more points she earns and climbs up the final classification. Of course, in order to be able to assign scores, it is necessary that each video segment comes with a reference ground truth. If, for the specific video, there exists a hand-made ground truth, it will be used. Otherwise, if the video is not a new one (i.e. several players have already played it, meaning that several annotations exist), the reference ground truth is given by the combination of all the existing annotations (see paragraph 3.2.2). If, instead, the video is a new one (i.e. no one has played a session with this video yet) then the detection algorithm's [67] output is used as reference ground truth.

A reference ground truth is also used to compare the annotations provided by the users against it. For each object in the reference ground truth a 2D Gaussian Distribution is placed, centered on the object's bounding box center. If a player clicks on this point, she gains the maximum score bonus she can get, while the bonus awarded is reduced as the clicked point gets more distant from the center.

In order to make sense of the data produced by this game, we had to deal with two important issues:

- assess the quality of the users (see paragraph 3.2.1);
- combine the different annotations to derive a single "best" representation of the objects (see paragraph 3.2.2).

3.2.1 Assess the quality of the users

The contribution of each user playing the game cannot be equal. In fact, there exist casual players that dedicate a little time playing, achieving, usually, low scores and on the other extreme, hardcore players can be found. Assessing user quality is of key importance for generating a ground truth based on the weighted contribution of users. The weight is the quality score itself, meaning that the higher a player's score is, the more influential her annotations will be in determining the final ground truth.

To estimate user quality we resort to the ground truth seeding technique, i.e. the first level of the game always contains a video for which a hand-drawn ground truth (G_{GT}) already exists. When the first level of the game ends, the acquired data (GT_u) of the user u is compared to the G_{GT} . Each submitted ground truth starts with a quality score (S_{GT}) of *1* and the number of False Positives (FP_u, a location where the user clicked but fish does not exist), False Negatives (FN_u, a location where the user did not click but fish does exist) and True Positives (TP_u, a location where the user clicked and fish does exist) are determined.

While a TP_u does not decrease the quality of the ground truth and a FP_u decreases it always, a FN_u is more complicated because it can occur for two reasons:

1) the user did not click on it at all, because she was not fast enough;
2) because, at the same time, she was clicking on another object.

In the former case, if the user was not fast enough to click, S_{GT} is decremented by N_{ft}/N_d, where N_d and N_{ft} are the objects contained, respectively, in the G_{GT} and in the frame ft. If the user was clicking other objects at the time that FN_u occurred, is determined by seeking for objects in frame f_t. If at least one such object exists, and it was shot by the user, no action is taken. Conversely, the user's quality score is decremented as before.

Summarizing the score of each submitted ground truth is given by:

$$S_{GT} = 1 - \frac{1}{N_d} \sum_{N_d} N_{false} \qquad (1)$$

where

$$N_{false} = \begin{cases} 0, \text{if } Click \text{ is a } TP_u \text{ or } (FN_u \text{ and } \exists\ TP_u\ \in Frame) \\ 1, \text{if } Click \text{ is a } FP_u \text{ or } (FN_u \text{ and } \nexists\ TP_u\ \in Frame) \end{cases}$$

If this is the first ground truth created by the user, her quality score is equal to S_{GT} . If, instead, previous assessments already exist, the quality score of the user is determined by:

$$S_u = \frac{1}{N_{Tot}} \sum_{i=1}^{U_{GT}} S_{GT_i} \times N_{GO_i} \qquad (2)$$

where N_{Tot} is the number of objects in all the ground truths of the user, U_{GT} is the set of her ground truths, S_{GTi} is the quality of i^{th} ground truth, given by (1), and N_{GOi} is the number of objects in it.

3.2.2 Build the ground truth objects

Once the users obtain a quality score, their annotations can be integrated in order to build the best ground truth representations. In order to identify the locations that users clicked the most, we apply iteratively an unsupervised clustering algorithm. In particular, initially, a *K-means* analysis is applied with a predefined number of clusters (set to 10 or to the number of fish in the existing ground truth, if it contains more). The clustering result is further refined by iterating through each point (clicked by the user) and determining whether it fits well in the assigned cluster or not, by calculating the Euclidean distance from the cluster's centroid.

Figure 3.2 - Clustering applied on the acquired data: Red dots are the locations clicked by the users. Yellow circles represent the result of the first clustering iteration, while the blue circles are the final result of the clustering method. The radius of each circle is equal to the sum of the quality scores of the users that made an annotation that belongs to that cluster, given by equation (3).

If such distance is over a threshold *T*, it means that the point does not fit well into that cluster and it is removed from it. Afterwards, the Euclidean distance of the removed point from the centroid of the other clusters is calculated. If a more suitable cluster (distance less than *T*) is found the point is marked as confirmed and it will be included in the next iteration. On the contrary, if no appropriate cluster exists, the point in question is excluded from successive iterations.

At each iteration, every cluster c is assigned a value that represents its significance, or radius, and is given by:

$$r_c = \frac{1}{N}\sum_{p}^{P_c} Q_{u,p} \qquad (3)$$

where *N* is the total number of points in the current frame, *p* represents the points in that cluster and $Q_{u,p}$ is the quality of the user that created that point.

The algorithm stops when all the clusters have a value of $r_c > Tr$ (Tr empirically set to 0.4) or the initial maximum number of clusters is equal to zero. In case these conditions are not satisfied, the maximum cluster number is decreased by one and the algorithm proceeds with the next iteration.

The resulting clusters can be represented as heat maps, showing how the users' clicks are distributed over the scene.

When the algorithm execution ends, the obtained clusters are the objects of the "best" ground truth. In detail, each object is represented by the bounding box of the corresponding cluster.

Algorithm 1 shows the clustering algorithm, Fig. 3.2 shows an example output of the method described, where the 10 initial small clusters (in yellow) are

reduced to 2 bigger ones (in blue) and Fig. 3.3 shows the heatmaps produced in a 4-frame sequence.

```
MaxClusters = max(10;Count(ObjectsInGT));
P = ClickedLocations;
C = Clustering(MaxClusters; P);
while (MaxClusters > 0) do
        ClustersOK = True;
        foreach k in C do
              r_c = radius(k);
              if r_c < T_r then
                    ClustersOK = False;
                    break;
              end
        end
        if ClustersOK == True then
        Output: C
        end
        foreach p in P do
              Cp = Centroid of cluster containing p;
              if distance(p,C_p) > T then
                    Found = False;
                    foreach k in C do
                          Ck = Centroid of cluster k;
                          if distance(p,C_k) < T then
                                Found = True;
                                break;
                          end
                    end
                    if Found == False then
                          remove p from P;
                    end
              end
        end
        MaxClusters = MaxClusters - 1;
        C = Clustering(MaxClusters,P);
  end
```

Algorithm 1: The clustering Algorithm

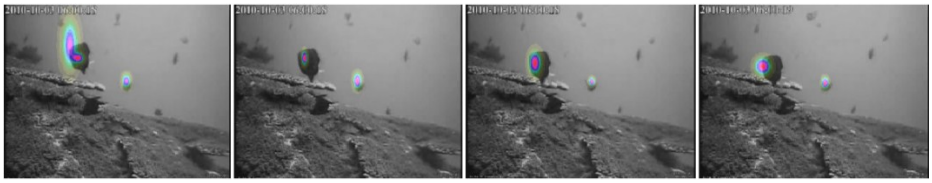

Figure 3.3 - Heatmaps of two fish detected in an 4-frame sequence

3.3 Automatic image segmentation using crowdsourced data

In this section we present the approach employed to label images by using the click points collected with the online game described in the previous section. In detail, the users' clicks drive the image segmentation approaches, namely the *Region Growing* and *Grab Cut*, which allow us to derive meaningful annotations (Figure 3.4 show the basic schema of the proposed approach). By comparing the results obtained with this approach against hand-labeled ground truth data, in *Chapter 4* we demonstrate that the proposed method constitutes a valid alternative to the existing video annotation approaches and allow a reliable and fast collection of large scale ground truth data for performance evaluation in computer vision, relieving the users from the noisy work of annotating images and videos manually.

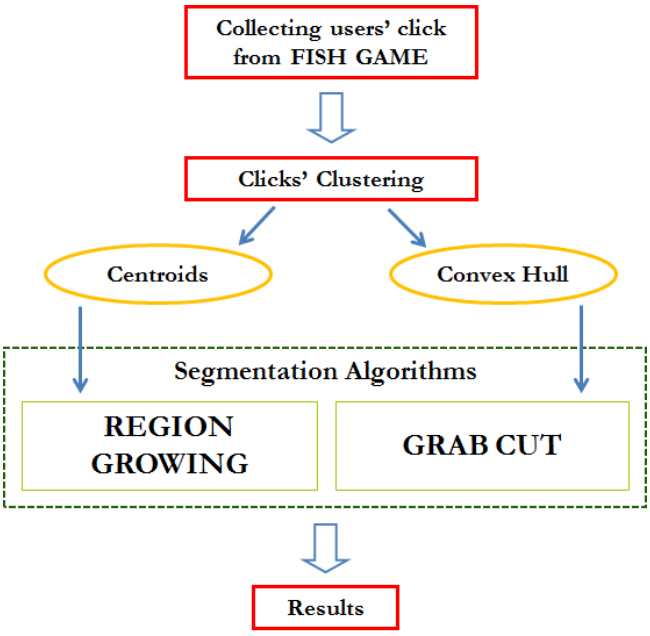

Figure 3.4 – Basic schema of the proposed approach

More specifically, the user who plays the game has to take photos of objects through the game's levels. In order to assess the accuracy of our approach, we adopted two classic techniques for image segmentation which require an initial labeling that may be either a point within the object (seed) or some region (or line) outside the object to be segmented. Needless to say, this initial labeling is the single most important parameter that influences the performance of image segmentation algorithms: if the initial labels are not positioned accurately either the result will contain undesirable information (a segment that contains the object and part of its surroundings) or it will omit desirable information (a partial result).

Starting from the raw users' clicks taken while playing Flash the Fish, unsupervised K-Means cluster analysis [68] is performed in order to extract the locations of the most clicked areas. Since the game's purpose relies on the belief

that the most clicked areas represent actual objects, the resulting clusters will be devoid of the influence of noisy clicks, because clusters with low numbers of clicks are discarded.

So, the output of the game are the clusters with their associated points which can be also represented with heatmaps showing where the majority of the clicks are located.

Then we resort to image segmentation approaches to generate annotations on moving objects. In detail, we used two approaches: the classic *region growing* that works by identifying the differences between objects in the image according to their color characteristics, and the *Grabcut* that performs image segmentation by means of a probabilistic approach.

Figure 3.5 shows the clicks' handling phase which include the clusters obtained from the points gained by playing the game and the corresponding heatmaps generated by considering the sum of the 3D Gaussian distribution of each point in a cluster.

Figure 3.5 – Clicks' handling phase. *On the left*: the obtained clicks from the game. *On the center*: clustering results. *On the right*: heatmaps of the most clicked areas.

3.3.1 Region Growing based approach

Region growing [69] is a fairly common technique for image segmentation, which groups together the pixels or sub-regions in gradually larger regions

according to a given criterion. The approach starts from a set of key points, also known as seeds, from which the regions grow. Afterwards, all the surrounding pixels that have similar properties to those of the starting seed are added to the region until a specific ending condition is satisfied.

Two issues must be addressed when dealing with the region growing approaches:

- the initial seed's position;
- the pixels' similarity policy.

In our case, the initial seeds' position is determined by selecting the local maxima of the heatmaps as coming out from the previously calculated clusters.

Starting from these points, the region is iteratively grown by comparing all neighbouring pixels to the region by using the difference between a pixel's intensity value and the region's mean. Then, the pixel with the smallest measured difference in this way is allocated to the respective region. This process stops when the intensity difference between the region's mean and the new pixel becomes larger than a certain threshold.

In Figures 3.6 and 3.7 two examples of segmentation by using the region growing technique described above are shown. As we can see, starting from a single seed derived from the clustered users' clicks, good results are obtained in terms of object contour.

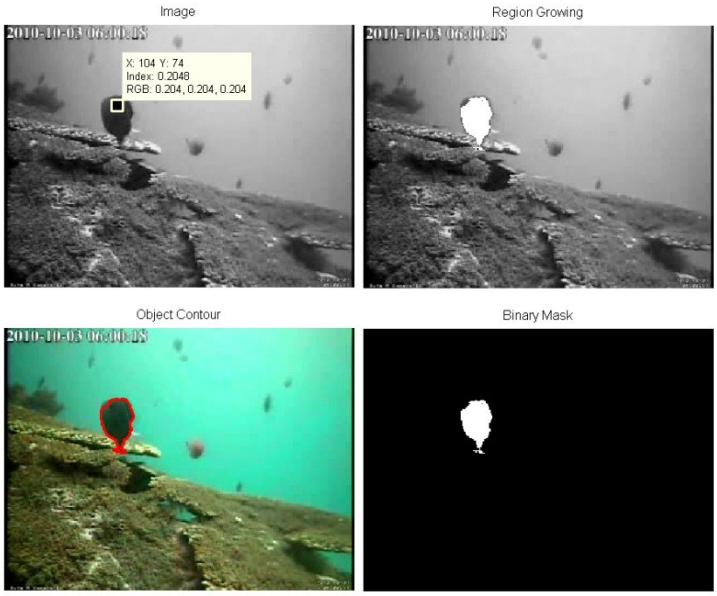

Figure 3.6 - *Region Growing*. On the *top left* the seed derived from Flash the Fish game heatmaps. The remaining images show the result of the applied algorithm in terms of object contour and the corresponding binary mask.

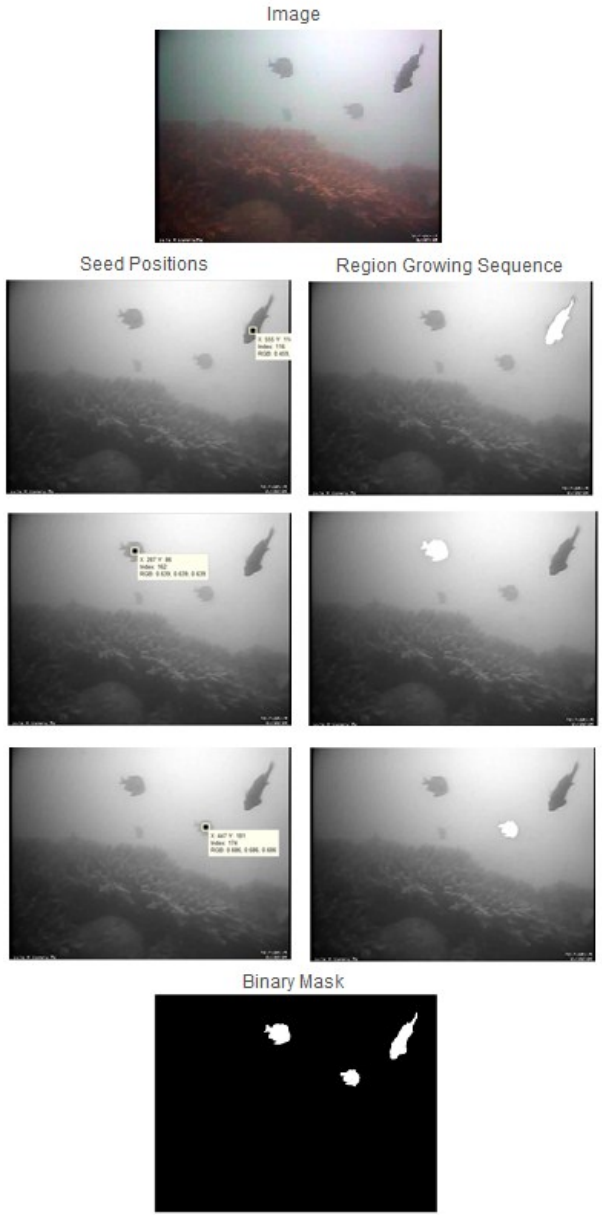

Figure 3.7 – *Top*: source image. *Left column*: the seeds positioning by using the proposed approach. *Right column*: Region Growing. *Bottom*: labeled objects.

The main drawbacks of the region growing segmentation algorithm are encountered when:

- the background has a similar texture and color to the object of interest;
- the seed is not positioned accurately inside the object.

Fig. 3.8 shows what might happen in the above cases: the area of the region grows beyond the object boundaries including parts of the image that are not logically connected to object or the object is not completely segmented, and misses desirable information.

In the next section the use of Grab Cut approach will be discuss which aim at overcoming the limitation of the just discussed method.

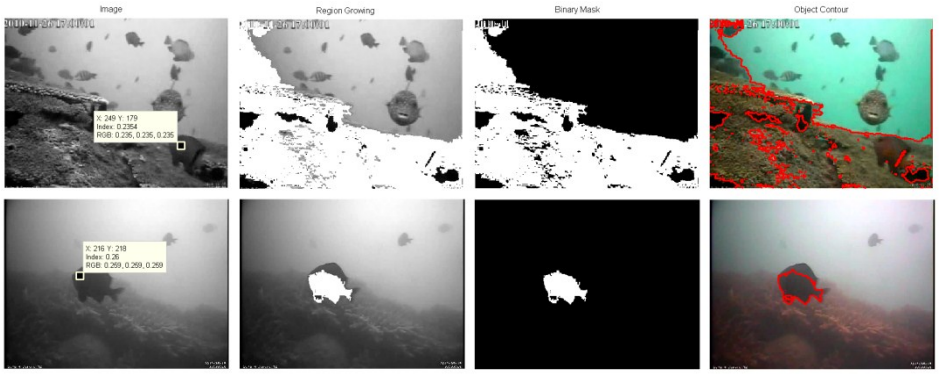

Figure 3.8 - *Region Growing drawbacks*. *Top*: Background and object have similar colors and texture. *Bottom*: Inaccurate positioning of the seed.

3.3.2 Grabcut Approach

Grabcut [70] is a dynamic image segmentation algorithm that applies graph cuts [71] iteratively: each successive iteration aims at minimizing further the energy of the result of the previous ones. In contrast to the region growing algorithm,

Grabcut operates in a different manner: instead of using a single pixel of the image as seed for determining the part of it that belongs to the desired segment or not, Grabcut uses an area where the object should be located.

Describing the exact theory behind Grabcut is not in the scope of this dissertation, but the reader can find more information in [70]. What this work addresses, instead, is the definition of the initial labeling for Grabcut to start the segmentation process.

In detail, by using the game data, the initial labeling is derived by processing the players' clicks, in order to define a region large enough to contain the whole object, but also small enough in order not to include unnecessary information. In our case, this region is computed as the convex hull containing all the points belonging to the same cluster.

A labeling mask is then created, where white points (foreground) are all the points inside the convex hull, and the black ones (background) are the points outside. This mask constitutes the initial labeling for Grabcut.

The application of the Grabcut algorithm is shown in Fig. 3.9. The same figure also shows that Grabcut performs well even when both the background and the object have similar colour and texture characteristics.

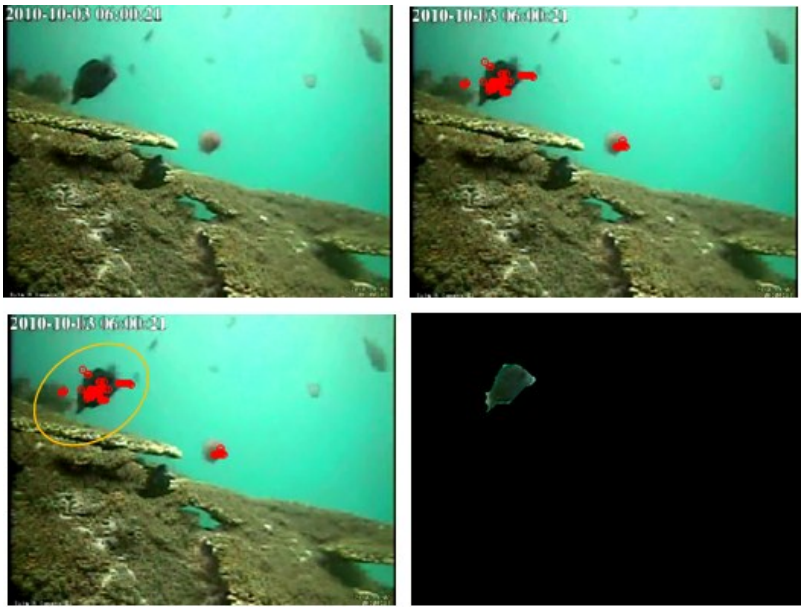

Figure 3.9 - Example of the application of the *Grabcut segmentation algorithm*. On the *top left* the original image can be seen, while in the *top right* the points that belong to an identified cluster are shown in red. The convex hull of these points is calcu lated (*bottom left*, in yellow) and Grabcut is applied on it (*bottom right*).

3.4 Data-driven segmentation by using a statistical approach

In this section we describe a statistical method resting on the same basis of the previous approaches, since, also in this case, we exploit the clicks gained by the users who played with *Flash the Fish* game, but we resort to Expectation Maximization for Gaussian Mixture Model (EMGMM) with the aim of defining the belonging of the acquired points to the object model rather than to the background one.

To be more precise, starting from the collected clicks we first apply a shift-back of the points obtained for the current frame in order to consider the delay with which the user clicks after seeing the object of interest. Then a uniform random point distribution on the source image is generated for constructing the background model, which does not include the users' clicks. For each point belonging to the two images, a *features vector* is created containing information about *position*, *color* and *texture*. As concerns the texture features, the Histogram of Oriented Gradient (HOG) descriptors [72, 73] are used. The collected feature vectors are then assembled to build a matrix for which the Expectation Maximization for Gaussian Mixture Model is calculated to obtain the model related both to the object and the background. Finally, for each pixel of the source image, the features vector is also constructed and the probability that each feature vector belongs either to the object model or the background model is computed. Then, according to the log-likelihood ratio, each pixel is classified as background if the probability value is greater than a given threshold, as foreground otherwise. In Figure 3.10 the basic schema of the proposed method is shown.

Experimental results (see *Chapter 4* for the details) show that the proposed system, when compared against the hand labeled ground truth dataset, is able to generate reliable annotations starting from big-noisy data, providing a valid alternative to the existing ground truth generation methods.

3.4.1 Features selection process from big noisy data

The users' clicks acquired while playing the game, are first processed to make sense of them. In fact, we initially shift-back the raw clicks to consider the delay with which the user clicks after seeing the object of interest. Typically, the delay ranges between 1 and 4 ms and considering that the video have been acquired at 5 fps, the shift has to be on 1 or at most 2 frames backward. In this way, more

accurate clicks are obtained for each frame in the video, resulting in a more accurate set of points which depict approximately the area where the user identified the object during the game session.

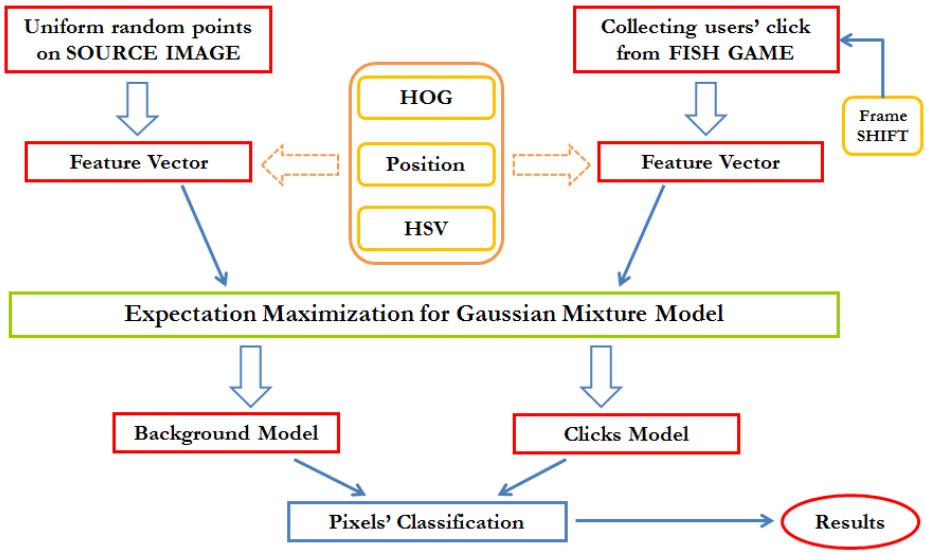

Figure 3.10 – Basic schema for ground trothing from big noisy data

Starting from the collected clicks, a number at least 5 time larger of uniform random points are then positioned on the source image, leaving out the clicks which we have already used for the object identification. Figure 3.11 show a set of images reporting clicks' handling phase.

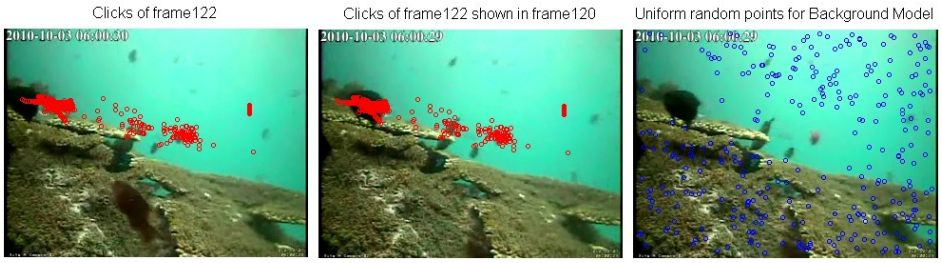

Figure 3.11 – *Clicks' handling phase*. On the *left image* the raw clicks derived from the game. In the *center image* the clicks are positioned according to the shift-back procedure (in this case *2 frame ahead* considering a *delay of 4 ms*). On the *right image* the same number of points distributed uniformly in the background.

Afterwards, the model for the objects and the background can be built. To do this, for each point a *feature vector* is created containing information about *position*, *color* and *texture features*. As concern the position, x and y coordinates are extracted from each point, while for color features the values of H, S and V *channels* are taken into account. The V channel could not be considered because it does not contain any specific information about color, but only with respect to the brightness.

To provide a consistent description of texture characteristic of the considered points, we refer to the *Histogram of Oriented Gradient* theory [73]. Generally the HOG descriptor is used to determine local object appearance and shape within an image by computing the distribution of intensity gradients or edge directions. In our case, the same concept is applied not to the entire image, but to a given windows which strictly surround each point located in the source images.

Anyway, the technique counts occurrences of gradient orientation in localized portions of an image and the features are computed on a dense grid of uniformly spaced cells by using overlapping local contrast normalization for improved accuracy. The implementation of these descriptors can be achieved by dividing the portion of image surrounding the available points (in our case a windows of *9x9*

pixels around each click is used) into small connected regions, called cells, and for each cell compiling a histogram of gradient directions for the HS channel of pixels within the cell.

The main advantage of using the HOG descriptor relies on the invariance to geometric and photometric transformations, since it operates on localized cells.

Once all the features are presented, we can construct the *feature vector* $\overrightarrow{v_c}$ for each click in the form of:

$$\overrightarrow{v_c} = [x_c, y_c, h_c, s_{c,} b_c, \vec{H}_c\,]$$

where x_c and y_c are the values of width and height of the click respectively, h_c, s_c and b_c (optional) are the HSV values of the pixel corresponding to the click and $\vec{H}_c$ is a vector of 27 elements (9 for each HSV channel) which includes the texture features of the pixel. The same procedure is computed for the uniform random points related to the background (in this case we use $\overrightarrow{v_b}$ to refer the feature vectors of background points).

Then a matrix of feature vectors is created for all the clicks of the source image as follow:

$$V_c = [\overrightarrow{v_{c_1}}, \overrightarrow{v_{c_2}}, \ldots, \overrightarrow{v_{c_n}}]$$

where V_c is an n-by-f matrix with n rows which define the total number of clicks and f columns which are the total number of extracted features. Even in this case, a matrix V_b is created for all points belonging to the background.

In the next section a detailed description on how to use the Expectation Maximization of Gaussian Mixture on V_c and V_b matrices is given.

3.4.2 EMGMM

The *Expectation Maximization of Gaussian Mixture Model* (EMGMM) [74, 75, 76] is a numerical algorithm for the maximization of functions of several variables suitable for the estimation of the parameters of a mixture of Gaussians from a set of data points.

The basic issue to deal with when using the EM approach, is to compute the density estimation for mixture of Gaussian functions: given a set of N points in D dimensions ($x1, \ldots, x_N$) ϵ R^D ($D=1$ in our case), and a family F of probability density functions on R^D, find the probability density $f(x)$ ϵ F that is most likely to have generated the given points.

One way to define the family F is to give each of its members the same mathematical form, and to distinguish different members by different values of a set of parameters θ. For instance, the functions in F could be *mixtures of Gaussian functions*:

$$f(x;\theta) = \sum_{k=1}^{K} p_k g(x;\mu_k,\sigma_k) \qquad (4)$$

where

$$g(x;\mu_k,\sigma_k) = \frac{1}{\sqrt{(2\pi)^k|\Sigma_k|}} e^{\left\{-\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)\right\}} \qquad (5)$$

is a *D-dimensional* Gaussian function and $\theta = (\theta_1, \ldots, \theta_K) = ((p_1, \mu_1, \sigma_1), \ldots, (p_K, \mu_K, \sigma_K))$ is a $K(D + 2)$-dimensional vector containing the mixing probabilities p_k as

well as the means μ_k and standard deviations σ_k of the *K Gaussian functions* in the mixture.

Thus, considering the selected features, we assume that each object is modeled by a Gaussian distribution [77] characterized by μ_k (the mean vector of the object *k*) and Σ_k (its covariance matrix). Hence, the final learned object model set can be defined as:

$$O_L = [O_{L1}(\overrightarrow{\mu_1}, \Sigma_1), \dots, O_{Lk}(\overrightarrow{\mu_k}, \Sigma_k)]$$

3.4.3 Pixels' Classification

The system starts by computing the model both for object and background which are then used to obtain a set of probability maps. Each map is associated to a known object and contains the probability for every pixel of the source image to be classified as the current object.

To be more precise, we compute the Gaussian distribution, not only for the points which represent the users' clicks and the surrounding background, but also for each pixel of the image. Then we use the models acquired from the learning to calculate the probability the a pixel belongs to the object model rather than the background model.

The appearance probability of a pixel *j* characterized by the features v_j of belonging to the object model *Obj* is given, under Gaussian assumption [78] by the probability density function:

$$P_A(j|Obj) = \frac{1}{\sqrt{(2\pi)^k|\Sigma_o|}} e^{\left\{-\frac{1}{2}(\overrightarrow{x_j}-\overrightarrow{\mu_o})^T {\Sigma_o}^{-1}(\overrightarrow{x_j}-\overrightarrow{\mu_o})\right\}} \qquad (6)$$

where $\overrightarrow{\mu_o}$ is the mean vector of the object model *Obj,* Σ_o its covariance matrix, and *k* the number of characteristics. Note that *k* is also the dimensionality of $\overrightarrow{x_j}$ and $\overrightarrow{\mu_o}$ vectors. This dimension is the number of features used to represent each object and its value depends on the features selection process (see paragraph 3.4.1 for the details).

Similarly, we calculate the probability that a pixel *j* of the source image belongs to the background model as follow:

$$P_A(j|Bkg) = \frac{1}{\sqrt{(2\pi)^k|\Sigma_b|}} e^{\left\{-\frac{1}{2}(\overrightarrow{x_j}-\overrightarrow{\mu_b})^T \Sigma_b{}^{-1}(\overrightarrow{x_j}-\overrightarrow{\mu_b})\right\}} \quad (7)$$

where $\overrightarrow{\mu_b}$ is the mean vector of the background model *Bkg* and Σ_b its covariance matrix. Even in this case *k* represents the dimensionality of $\overrightarrow{x_j}$ and $\overrightarrow{\mu_b}$ vectors.

Once the probability maps of each model is constructed, we classify each pixel according to the *log-likelihood* ratio as follow:

$$M(x,y) = \begin{cases} 0 \; if - ln\frac{P_b}{P_o} > Th \\ 1 \qquad otherwise \end{cases}$$

The matrix *M(x,y)* represents a binary mask which provide the segmented object corresponding to the users' clicks.

Figure 3.12 shows how, starting from the source image and applying our method, we obtain very good annotation of the current objects.

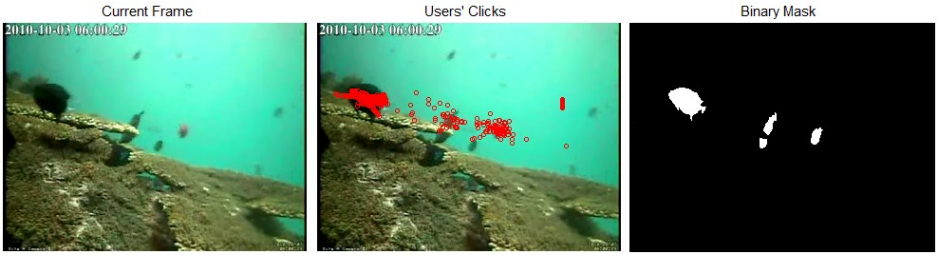

Figure 3.12 – Final segmentation results. On *the left*, the source image as presented to the user during the game session. In the *center image*, the users clicks obtained for the current frame. On the *right image*, the binary mask with the segmented objects.

To validate the obtained result, in *Chapter 4* we present a comparison against the hand labeled ground truth dataset to assess the validity of the proposed approach as an alternative to existing methods.

CHAPTER 4

Performance evaluation

4.1 Introduction

This chapter is fully devoted to describe the methodologies adopted to evaluate the performance of the approaches proposed in *Chapters 2* and *3*.

The performance of the methods for semi-automatic ground truth generation is assessed in terms of i) the accuracy of the generated ground truth by also referring to the VIPER-GT tool for a comparison of the obtained annotations, ii) the efficiency of the proposed platforms, as measured by time needed to generate annotations and iii) the learnability and user satisfaction.

For the crowdsourcing approaches instead we evaluate the ability of *the Flash the Fish* game to generate accurate annotations by assessing its performance and

comparing it to hand-drown ground truth data. Similarly, for the data-driven approaches we show how, starting from big noisy and raw data gained from the game, reliable annotation can be achieved with respect to the existing tools for hand labeled ground truth generation.

4.2 GTTool

In order to assess the performance of the proposed tool in terms of time and accuracy, we asked 20 users, aged between 20 and 40 years old and with a minimal IT experience, to annotate fish in 100 consecutive frames of 10 different videos taken from underwater cameras (resulting in 20000 annotated frames), with both GTTool [7] and ViPER [17]. Each user was provided with a PC with the same hardware characteristics, in a closed laboratory. The users were also encouraged to speak aloud during the interaction with the systems, and were observed by the experimenter avoiding both to affect their choices and to provide specific indication in difficult situations. Only in the case of specific user's requests for support or due to system failures, the intervention of the observer was admitted. The experiment was conducted along three specific phases:

1) brief oral interview aimed at reassuring the user, explain the purpose of the investigation, explain the peculiarities of the tools and understand how to perform the annotations;
2) annotation phase: the users were asked not only to draw the boundaries of the objects, but also to create tracking ground truth by using the tools offered by the two applications. This task was constantly performed under the supervision of the observer which took notes the user's actions;

3) administration of a final questionnaire: the users were asked to complete a questionnaire in order to gain both their profiles, rate the usability of the systems along various dimensions, and report the problems that were found during the use of the tools.

The achieved results in terms of efficiency and accuracy are shown in Table 4.1. In particular, the accuracy of the segmented objects was computed by evaluating the overlap ratio with ground-truth data drawn by experts on higher-resolution versions of the same videos.

Method	GTTool	ViPER
Total drawn objects	16347	13315
Manually drawn objects	3114	13315
Automatically drawn object (GMM)	8101	-
Automatically drawn object (ACM)	5132	-
Average time per object	4.8 seconds	13.7 seconds
Accuracy	91%	76%
Learnability	804	3.2
Satisfaction	7	5.1

Table 4.1 – Comparison between the proposed tool and ViPER

As can be seen from the results, the time required to analyze manually the videos with GTTool is about one third of the time needed to perform the labeling task by using ViPER. This was mainly due to the markedly smaller number of objects which had to be drawn manually by the users (about 3 objects out of 4 are automatically segmented by our tool).

We also asked users to fill in the usability questionnaire, in order to get their feedback on how they felt using the two tools. In particular, we asked the participants to grade both tools in terms of learnability and satisfaction.

Learnability represents the ease of learning the usage of the tools, while satisfaction represents the subjective feelings of the users about their experience with each tool; both values range from 1(worst) to 10 (best). The results show that their experience with GTTool was more satisfactory than with ViPER, mainly, according to most comments, because of the two-window layout (which avoids having to go back and forth through the video to check one's previous annotations) and of the integrated algorithms (which drastically reduced the number of frames and objects which had to be manually analyzed).

4.3 PERLa

The first evaluation of system's performance was carried out in terms of time needed to perform annotations and accuracy of collected annotations. In particular, we asked 50 computer science undergraduate students to annotate fish in 200 consecutive frames of 5 different videos (320 × 240, 5 fps, 10 min long), provided with high quality ground truths, taken from the application's repository, with the proposed tool, the GTTool [7]and ViPER-GT [17]. The users were given a time period of two weeks to complete the task. The time spent on the proposed tool was measured automatically. For the GTTool and ViPER-GT the students were asked to accurately take note of the time spent during the whole process. The achieved results in terms of efficiency and accuracy are shown in Table 4.2.

Method	PERLa	GTTool	ViPER
Total drawn objects	34131	43124	31409
Manually drawn objects	16832	14563	31409
Automatically drawn object	17299	28561	-
Average time per object	7.4 seconds	4.2 seconds	11.2 seconds
Contour accuracy	89% (95%)	90%	79%
Learnability	9.1	8.2	3.4
Satisfaction	7.3	7.3	4.3

Table 4.2 – Comparison between the proposed tool, the GTTool and ViPER-GT. The number in parenthesis is the accuracy obtained by using the contour integration module (see Fig. 4.1)

The accuracy of the contours was compared against the gold standard ground truths available for those five videos by calculating the average of the PASCAL score and the Euclidean distance.

The results shown that the time required to annotate the videos on average, was lower for the GTTool. The reason behind this is that the GTTool employs automatic object detection (in addition to automatic contour extraction methods, which are the same as the proposed tool) to support users' annotations, thus resulting in a major number of automatically drawn objects, as shown in Table 4.2. For the same reason the accuracy of the annotations drawn with GTTool was also slightly better than the one achieved with the proposed tool. ViPER-GT ranked last in this comparison because of its complete manual nature. It is important, though, to notice that these results refer to a one-vs-one user setting and do not include possible advantages that can be exploited by the proposed tool's multi-user nature.

So, in order to compare the effort needed to generate high quality ground truth by using the aforementioned tools, the time needed to annotate a video containing a hundred fish objects, was measured. In single user applications, such as ViPER-GT and GTTool, which do not offer any annotation integration

method, the time necessary to create a ground truth, increases exponentially with respect to its quality. Considering though, that the proposed tool is devised to permit multiple users to collaborate, integrating their annotations gives a significant boost to the quality/effort ratio. In fact, in Fig. 4.1, is shown the time needed in order to achieve different quality scores. In the single-user case, as it was aforementioned, the best performer is the GTTool, needing about 61 min, in the best case, in order to get a ground truth quality of 0.8. When the annotation integration module was used, the same quality was achieved in about 30 min (in the 50 users setting).

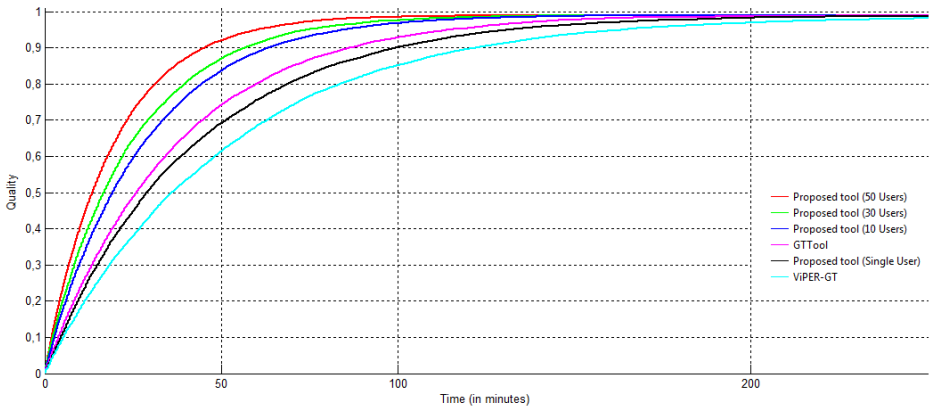

Figure 4.1 - The time (in minutes) needed and the obtained quality, for annotating a video containing 100 objects. For the single user cases the graphic represents the time needed by the best performing user. For the proposed tool, when annotation integration takes place, it represents the average time needed for each single user to achieve the corresponding quality score.

Upon annotation completion, a usability questionnaire was compiled by the users, in order to obtain some feedback about the user experience. In particular, the students had to grade the used tools in terms both of learnability and satisfaction. Learnability expresses how easy it is to learn to use the tools, while

satisfaction represents the general feelings of the users about their time spent with each tool; both values range from 1(worst) to 10 (best).

As shown in Table 4.2, the totality of users voted the proposed tool as the easiest to learn, achieving a score of 9.1/10, with the GTTool coming near second (8.2/10). ViPER-GT ranked third with a very low score (3.4/10) mainly because of its complex interface and the time needed to achieve a satisfactory level of knowledge on its usage.

When user satisfaction is concerned, both the GTTool and the proposed application achieved a score of 7.3 out of 10. This tie was due to two main reasons, based on the users' comments:

1) the GTTool's object detection and tracking algorithms, alleviated a large part of the work,
2) the proposed web tool was easier to use, better organized and more appealing to the eye.

The worst performer was, again, ViPER-GT because of the total lack of automated tools and its steep learning curve.

4.4 *Flash the Fish*

In order to test the accuracy and the efficiency of the proposed approach, we compared the annotations generated from the game against a hand-made accurate dataset. In particular, we used 7 hand-labeled videos (with frame rate and duration values as described in the 3.2.2 paragraph, for a total of 1568 frames) that contained 4140 annotated objects.

Clustering methods work better when applied to as bigger datasets as possible and for this reason we organized a Facebook event to gather users. To motivate

the players we also offered a prize for the winner. For the event's duration (4 days), 80 users participated and played 1273 game sessions that resulted in 264316 annotations (Table 4.3).

Level	Annotated objects	Acquired clicks	Number of users	Precision	Recall	F_1
1	722	71105	80	0.71	0.96	0.70
2	1847	70406	80	0.69	0.64	0.66
3	593	58528	69	0.70	0.64	0.67
4	251	47137	52	0.74	0.71	0.72
5	446	16276	46	0.57	0.51	0.54
6	104	522	19	0.31	0.21	0.25
7	177	342	18	0.26	0.09	0.13
Total	4140	264316	80	0.66	0.60	0.63

Table 4.3 - The datasets we used for performance evaluation and the results obtained. The precision and recall values refer to the case where the annotations of all the users that played the corresponding level were used and the values in the last row are the weighted averages with respect to the number of ground truth objects.

For determining whether an object (BB, the bounding box calculated by the algorithm described in paragraph 3.2.2) is either a hit or a miss we calculate the overlap, or PASCAL score, between it and the objects (GT) in the same frame in the hand-drawn ground truth given by:

$$O_{score} = \frac{area(BB \cap GT)}{area(BB \cup GT)}$$

If there exists at least one object where such value is greater than a threshold, empirically set equal to 0.7, then the cluster is considered a true positive; conversely, it is considered a false positive. A false negative is an object in the GT that has no corresponding BB.

The performance of the application was evaluated in terms of precision, recall and F1 measure, given by:

$$Precision = \frac{TP}{TP + FP'} \qquad (1)$$

$$Recall = \frac{TP}{TP + FN} \qquad (2)$$

and

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall'} \qquad (3)$$

respectively. The obtained results, together with the dataset used for testing the application, are shown in Table 4.3.

All the players reached level two, but with the progressively increasing difficulty, a little less than 25% (19 out of 80) of them reached the last one. The absolutely best performance was achieved at the fourth level, where the precision, recall and F1 values were 0.74, 0.71 and 0.72 respectively. This was due to the fact that it has the highest clicks/object ratio, which was about 187 at that level.

It should be noted that the turning point of the game is the fifth level where more than half players who played it could not advance through it (27 out of 46). As a consequence, it severely hampered the performance of the clustering method at the last two levels, where an inadequate number of annotations was collected (a little less than 2 annotations per object at the last level). In particular, the recall value at that level was too low, 0.09, because of the high number of false negatives (i.e. many objects in the *GT* did not have any clicks at all). Furthermore, we also

analyzed how the user quality influenced the method's performance. First we excluded all the annotations made by the users who had a quality score lower than 0.7. Surprisingly, this choice influenced minimally the precision, whereas it affected the recall. This result is explained by the fact that the true positives TP_u of the low-quality users were able to raise the r_c value over the threshold T_r (see paragraph 3.2.2), thus, resulting in a lower number of false negatives. Conversely, the false positives FP_u produced by the same users were not enough to create false positive clusters because this creation depends also on the user quality. More quantitatively, for level 1 the precision kept almost stable (0.69), whereas the recall dropped to 0.37. This also explains why the performance decreased drastically at the higher levels (Levels 6 and 7) when the number of players decreased. On the contrary, when we excluded the users with quality higher than 0.7 both precision and recall dropped to, respectively, 0.44 and 0.25.

The user quality allows us, therefore, to keep balanced precision and recall, whereas, the number of users serves to support the explorative nature of the game, i.e. the more users play, the higher is the probability (which also depends on the users' quality) to detect correctly objects.

4.4.1 Combining Flash the Fish annotations with image segmentation algorithms

In this section we describe the method used to evaluate the performance of the image segmentation approaches discussed in paragraph 3.3 (i.e. region growing and grabcut) by comparing the obtained results against hand-drawn ground truth. This ground truth contained 4140 objects and it was generated with PerLa [8]. Also in this case, we exploit the same outlines of the Facebook event organized for testing *Flash the Fish* game (4 days duration, more than 80 users, about 1300 game sessions and more than 260000 clicks). Quantitative performance analysis

was carried out on a pixel basis by comparing the results obtained by the segmentation algorithms against the hand-labeled ground truth and by computing the metrics (1), (2) and (3) stated in paragraph 4.4.

In this case a T_P (True Positive) was defined as a pixel that was contained both in the ground truth and in the segmentation result, while a F_P (False Positive) was defined as a pixel contained in the resulting segment but not in the ground truth. Finally, a F_N (False Negative) was a pixel that was contained in the ground truth but not in the resulting segmentation.

The obtained results, together with the dataset used for testing the application, are shown in Table 4.4.

These results take into account only the annotations that had a PASCAL score of at least 0.7 with respect to an object in the ground truth. The PASCAL score is given by:

$$P_{score} = \frac{area(Annotation \cap GT)}{area(Annotation \cup GT)} \qquad (4)$$

where *Annotation* is the resulting annotation and *GT* is the corresponding ground truth object.

Level	Annotated objects	Acquired clicks	Region Growing			Grabcut		
			Precision	Recall	F_1	Precision	Recall	F_1
1	722	71105	0.19	0.30	0.23	0.78	0.81	0.79
2	1847	70406	0.31	0.41	0.35	0.79	0.74	0.76
3	593	58528	0.44	0.47	0.45	0.63	0.68	0.65
4	251	47137	0.43	0.40	0.41	0.67	0.64	0.65
5	446	16276	0.51	0.41	.045	0.39	0.42	0.40
6	104	522	0.63	0.61	0.62	0.29	0.28	0.28
7	177	342	0.65	0.67	0.66	0.22	0.24	0.23
T	4140	264316	0.45	0.47	0.45	0.54	0.54	0.54

Table 4.4 - Performance evaluation of the segmentation algorithms.

From the same table, it is also possible to notice how the performance of the segmentation algorithms shows a different behaviour. In particular, region growing performed better when the number of clicks available was low. In fact, in the lower levels (levels 1 and 2) the region growing based approach achieved, on average, 25% in precision and 36% in recall. The precision score was so low because of the large number of inaccurate clicks which, therefore, resulted in clusters whose centroids were outside the object's boundaries. On the contrary, in the highest levels (i.e. 6 and 7), where only few motivated and reliable users were able to get score, both precision and recall achieved, on average, 64%. In this case, even if the number of clicks was considerably lower than those of the first levels, they were extremely accurate as obtained by the best performing users.

Grabcut, instead, achieved better performance in the lower levels. In the first 4 levels, the precision and recall values were, on average, 71%. These scores reflect the much better capacity of Grabcut to handle complex backgrounds and its ability in the choice of the initial labeling.

Finally, the performance of Grabcut in the last two levels was very low because an accurate initial labeling could not be determined due to the lower number of clicks.

4.4.2 Generating annotations by using Gaussian Mixture Models

As in the previous case, the performance is assessed by exploiting the raw data gathered from the *Flash the Fish* game sessions and by comparing the method's outputs with the 4140 hand-labeled object annotations. The metrics used to evaluate the performance of the proposed technique are (1), (2), (3) in paragraph 4.4 and (4) in paragraph 4.5, but the obtained results show some basic differences with respect to the previous work.

In Table 4.5 the results obtained by considering only the annotations that had a PASCAL score of at least 0.8 with respect to an object in the ground truth are shown.

As we can see, as the levels of the game increase, the values of precision, recall and $F_{measure}$ slowly decrease achieving on average the value of 67%. This behaviour is mainly due to the fact that, during the game session much more clicks are gained in the first levels, since almost all players were able to exceed from a level to the successive one, significantly increasing the number of clicks for the appearing objects. This results in a more accurate design of the object model, which can be more easily distinguished with respect to the background. On the contrary, since only few players were able to reach the final levels (i.e. 6 and 7), not much clicks can be used to drive object model, thus it becomes too hard to discriminate if a pixel belongs either to object model or to the background model, resulting in a less accurate object segmentation.

Level	Annotated object	Acquired clicks	Precision	Recall	$F_{measure}$
1	722	71105	0.81	0.79	0.80
2	1847	70406	0.77	0.75	0.76
3	593	58528	0.71	0.71	0.71
4	251	47137	0.69	0.67	0.68
5	446	16276	0.64	0.62	0.63
6	104	522	0.58	0.60	0.59
7	177	342	0.51	0.55	0.53
Total	4140	264316	0.67	0.67	0.67

Table 4.5 – Performance evaluation of data-driven segmentation by using the statistical approach

Indeed, the major advantage of using this method is the total independence with respect to the user's quality. In fact, unlike the previous approach, no users' skills have to be considered for achieving the final levels and provide more accurate clicks, because with this approach what is important is, instead, the number of available clicks for creating a more accurate object model.

CHAPTER 5

Discussion and Conclusion

This research has investigated several approaches for large-scale ground truth generation which is the basis to allow an efficient evaluation of the performances of computer vision methods.

Precisely, we addressed the problem of automatic ground truth generation by considering two main points of view: 1) to overcome the limitation of the existing approaches by developing easy and intuitive tools to facilitate the labeling process and 2) to construct a large scale database of visual annotations by exploiting the crowd of users.

To this aim, in this thesis we first described the GTTool method, which represents an application that improves the user's experience during the extraction of the object contours by means of a simple graphic interface and the use of automatic techniques for the detection of object across frame sequences. A

modular architecture has been developed in order to enhance ground truth generation in term of both accuracy and human efforts. Several techniques for automatic contour extraction (Active Contour Models and the Gaussian Mixture Model motion detection algorithms) and object tracking (CAMSHIFT) have been integrated, while still allowing the user to define ground truth data manually if the automatic methods fail to identify and track correctly the objects of interest. XML support allows to both save the inserted ground truth to file (to share it with others or to be modified at later time) and to import VIPER-GT files, thus supporting the migration process to GTTool. The experimental results show that the proposed solution outperformed the VIPER-GT approach in every test we run, reducing the time needed to label an entire video by a factor of 3.

Despite the advantages introduced by the GTTool, the integration of crowdsourcing and collaborative capabilities in order to permit to different users to cooperate in the ground truth generation process, is still needed. This can be achieved by providing a web interface that implements the same functionalities of GTTool, adding multi-user capabilities and video library management.

To this end, in this work a web-based video annotation tool (PERLa) is also presented for dealing with all the aspects of the ground truth generation process at different vision levels.

Although the tool is online since July 2012 and only few users have had access to it (for testing reasons), more than 55000 annotations have been drawn, but this number is expected to grow exponentially in the following months. Besides, the experimental results have shown that the proposed tool allows users to speed-up the generation of high quality ground truth due to the distribution of the workload to multiple users. Moreover, the contour integration module performed as expected, increasing the quality of the produces ground truth.

Currently we are working on integrating a performance evaluation module which will enable scientists to test and compare their algorithms using the generated ground truth. This module will be further extended with social interaction capabilities, in order to enable scientists to share code, datasets and ideas.

In spite of the great support provided by PERLa for collecting ground truth data through the use of a web-based collaborative platform, much more investigations have been conducted to further relieve users from the onerous task of gathering annotation, providing at the same time, an effective method for integrating quality control and increasing the users' motivation. Actually, in PERLa a first attempt to generate high quality ground truth was carried out by exploiting the combination of the annotation of multiple users and the voting approach to generate what we defined as the *Best Ground Truth*. Thus, the more user use annotate objects and share their annotation with other users, the better will be the resulting quality of the generated ground truth.

As concern instead the problem of motivating the users to contribute in gathering annotation, some approaches which rely on crowdsourcing methods were taken into account. To this purpose we presented *Flash the Fish*, a simple online game that aim at generating video annotation for object detection algorithms. The acquired clicks were then fed to a clustering module which refined the results, producing good quality ground truth and we are confident that the quality will increase as more and more users will play the game.

While, in its current form, the game generates ground truth for object detection, as the number of annotation increases, it should be interesting to assess whether it is possible to derive the exact object shapes from the heatmaps. While a preliminary analysis demonstrated that this can be possible, a very large dataset should be considered. To accomplish that, we thought to integrate advanced

voting methods [86] and use the clicked points to drive some object segmentation approaches. Moreover, bonus levels that permit the creation of annotations for testing object tracking and object classification methods are already implemented and are currently under testing.

One interesting observation that we made during the revision of the datasets is that many clicks should fit better in successive frames from the one that they were acquired. This happens due to the fact that the time that passes from the moment the eye catches the visual stimulus (fish moving) to the moment of the reaction (mouse movement and click) is not negligible and it should be taken into account.

For this reason we developed a module that analyses the reflexes of each user independently by controlling how well the clicks fit with the "best" ground truth, and we introduced a delay.

Given the effectiveness of this game, we aim at creating an open platform, where researchers can upload their videos, to be used in the game, and the generated annotations will be publicly available.

To accomplish the purpose of using the clicked point as seeds for developing object segmentation approaches, in this thesis we also have proposed two different methods which exploit the data gathered by playing the *Flash the Fish* online game for supporting video annotation.

In the first case we exploit the *Region Growing* and *Grabcut* segmentation techniques, which also use unsupervised clustering to compute the centroids used as the starting seeds and the convex hull surrounding the obtained clusters respectively, to drive the segmentation phase. In this case the obtained results ($F_{measure}$ of about 80% in the best case) when compared against a hand labeled ground truth dataset, showed that the proposed approach is able to generate reliable annotations providing a valid alternative to the existing ground truth

generation methods. However, a drawback found by using this approach is related to the fact that the quality of annotations are strongly connected to the user quality. To avoid this issue we proposed a statistical approach which, first of all, takes into account the delay with which the user clicks after seeing the object of interest by shifting-back the points retrieved for the current frame thus obtaining a more accurate positioning of the users' clicks. Then, similarly to the previous approach, we calculate the $F_{measure}$ metric which provides, in the best case (e.g. when the largest number of click is available) a value of about 80%. Also in this case we are able to supply very good annotation with respect to the hand-drawn ground truth providing an effective method which proved to be completely independent from the user's quality.

As future developments we plan to add automatic video analysis tools for object detection, tracking, recognition and image segmentation that may save annotation time. We also aim to map the currently available XML format, used for GTTool and PERLa, into a web ontology, in order to give users the possibility to insert semantic metadata for each annotated object, which could not only support interoperability with other semantic web applications (e.g. multimedia retrieval, like in [79, 80, 81]), but also enable users to generate ground truth for higher level tasks (e.g. object recognition etc.).

Machine learning methods [82, 83, 84] will be applied on these textual annotations in order to exploit the advantages offered by integrating annotations to multiple types and levels of information. These semantic data will be available to the end users via SPARQL Endpoints. The integration of more extensive collaborative capabilities, e.g. simultaneously editing the same ground-truth or systemically distributing the ground truth generation among different users, would undoubtedly accelerate even more the whole process. Moreover, multiple

annotations of the same object by different users could be integrated by using Adaboost [85] in order to enhance the quality of the produced ground truth.

Finally, we are going to further investigate new and more efficient methods for treating the raw data derived by the game, by integrating effective algorithms for segmenting objects in videos. Also, the data gathered by the game can also be used in order to derive other types of annotations. For example, object-tracking ground truth by grouping the user's clicks in the frame sequences and by applying object tracking methods, like [87] in order to find the trajectories' limits.

REFERENCES

[1] Ahn L.V. (2006), *Games with a purpose*. Computer 39(6), pp. 92–94.

[2] Ambardekar A., Nicolescu M., Dascalu S. (2009), *Ground truth verification tool (GTVT) for video surveillance systems*. In: Proceedings of the 2009 second international conferences on advances in computer-human interactions, ACHI '09, pp. 354-359.

[3] Barbour B., Ricanek Jr K. (2012), *An interactive tool for extremely dense landmarking of faces*. In: Proceedings of the 1st international workshop on visual interfaces for ground truth collection in computer vision applications, VIGTA '12. ACM, New York, pp. 13:1–13:5.

[4] Barnich O., Van Droogenbroeck M. (2011), *ViBe: a universal background subtraction algorithm for video sequences*. IEEE Trans Image Process 20(6), pp. 1709–1724.

[5] Bassel G.W., Glaab E., Marquez J., Holdsworth M.J., Bacardit J. (2011), *Functional network construction in Arabidopsis using rule-based machine learning on large-scale data sets*. Plant Cell 23(9), pp. 3101–3116.

[6] Bertini M., Del Bimbo A., Torniai C. (2005,) *Automatic video annotation using ontologies extended with visual information*. In: Proceedings of the 13th annual

ACM international conference on multimedia, MULTIMEDIA '05, pp. 395–398.

[7] Kavasidis I., Palazzo S., Di Salvo R., Giordano D., Spampinato C. (2012), *A semi-automatic tool for detection and tracking ground truth generation in videos*. In: VIGTA '12: proceedings of the 1st international workshop on visual interfaces for ground truth collection in computer vision applications. ACM, New York, pp. 1–5.

[8] Kavasidis I., Palazzo S, Di Salvo R., Giordano D., Spampinato C., *An innovative web-based collaborative platform for video annotation*. Multimedia Tools and Applications, pp. 1-20, 2013.

[9] Fei-Fei L., Fergus R., Perona P. (2007), *Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories*. Comput Vis Image Underst 106(1), pp. 59–70.

[10] Griffin G., Holub A., Perona P. (2007), *Caltech-256 object category dataset*. California Institute of Technology, Tech. Rep. 7694.

[11] Martin D., Fowlkes C., Tal D., Malik J. (2001), *A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics*. In: Proc. 8th int'l conf. computer vision, vol. 2, pp. 416–423.

[12] Russell B.C., Torralba A., Murphy K.P., Freeman W.T. (2008), *Labelme: a database and web-based tool for image annotation*. Int J Comput Vis 77(1–3), pp. 157–173.

[13] Stork D.G. (1999), *Character and document research in the open mind initiative*. In: Proceedings of the fifth international conference on document analysis and recognition, ICDAR '99.

[14] Rashtchian C., Young P., Hodosh M., Hockenmaier J. (2010), *Collecting image annotations using Amazon's Mechanical Turk*. In: Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's Mechanical Turk, CSLDAMT '10, pp. 139–147.

[15] Spampinato C., Boom B., He J. (eds) (2012), VIGTA '12: proceedings of the 1st international workshop on visual interfaces for ground truth collection in computer vision applications. ACM, New York.

[16] Spampinato C., Palazzo S., Boom B., van Ossenbruggen J., Kavasidis I., Di Salvo R., Lin F., Giordano D., Hardman L., Fisher R. (2012), *Understanding fish behavior during typhoon events in real-life underwater environments*. Multimed. Tools Appl. doi:10.1007/s11042-012-1101-5.

[17] Doerman D., Mihalcik D. (2000), *Tools and techniques for video performance evaluation*. In: Proceedings of 15th international conference on pattern recognition, vol. 4, pp. 167–170.

[18] Jaynes C., Webb S., Steele R., Xiong Q. (2002), *An open development environment for evaluation of video surveillance systems*. In: PETS02, pp. 32–39.

[19] D'Orazio T., Leo M., Mosca N., Spagnolo P., Mazzeo P.L., *A semi-automatic system for ground truth generation of soccer video sequences*, in Advanced Video and Signal Based Surveillance, 2009. AVSS '09. Sixth IEEE International Conference on, Genova, 2009, pp. 559-564.

[20] Fernández-García N.L., Carmona-Poyato A., Medina-Carnicer R., Madrid-Cuevas F.J., *Automatic generation of consensus ground truth for the comparison of edge detection techniques*, Image and Vision Computing, Vol. 26, Issue 4, 1 April 2008, pp. 496-511.

[21] Yuen J., Russell B.C., Liu C., Torralba A. (2009), *Labelme video: building a video database with human annotations*. In: ICCV'09, pp. 1451–1458.

[22] Ali, K., Hasler, D., Fleuret, F., *FlowBoost - Appearance learning from sparsely annotated video*, Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, pp. 1433-1440, 20-25 June 2011 doi: 10.1109/CVPR.2011.599540.

[23] Agarwala A., Hertzmann A., Salesin D., Seitz S. (2004), *Keyframe-based tracking for rotoscoping and animation*. In: ACM Transactions on Graphics (TOG), ACM, vol. 23, pp. 584-591.

[24] Buchanan A., Fitzgibbon A. (2006), *Interactive feature tracking using kd trees and dynamic programming*. In: CVPR 06, Citeseer, vol. 1, pp. 626-633.

[25] Fisher R.B. (2004), *The pets04 surveillance ground-truth data sets*. In: Proc. 6th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, pp. 1-5.

[26] Smeaton A., Over P., Kraaij W. (2006), *Evaluation campaigns and TRECVid*. In: Proceedings of the 8th ACM international workshop on Multimedia information retrieval, ACM, pp. 321-330.

[27] Laptev I., Marszalek M., Schmid C., Rozenfeld B. (2008), *Learning realistic human actions from movies*. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE, pp. 1-8.

[28] He, J., Li, M., Zhang, H.J., Tong, H., Zhang, C. 2004, *Manifold-ranking based image retrieval*. In Proceedings of the 12th Annual ACM International Conference on Multimedia.

[29] Goh, K.S., Chang, E. Y., Lai, W.-C. 2004, *Multimodal concept-dependent active learning for image retrieval*. In Proceedings of the 12th Annual ACM International Conference on Multimedia, pp. 564–571.

[30] Wang Y., Ji R., Chang S-F., *Label Propagation from ImageNet to 3D Point Clouds*, Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pp. 3135-3142, 23-28 June 2013, doi: 10.1109/CVPR.2013.403.

[31] Deng J., Dong W., Socher R., Li L-J., Li K., Fei-Fei L., *ImageNet: A Large-Scale Hierarchical Image Database*. CVPR, 2009.

[32] Tang J., Hong R., Yan S., Chua T-S., Qi G-J., Jain R., 2011, *Image annotation by kNN-sparse graph-based label propagation over noisily tagged web images*. ACM Trans. Intell. Syst. Technol. 2, Article 14 (February 2011), 15 pp. doi:10.1145/1899412.1899418.

[33] Chapelle O., Zien A., Scholkopf B., 2006, *Semi-Supervised Learning*. MIT Press.

[34] Zhu X., 2005, *Semi-Supervised Learning with Graphs*. Ph.D. dissertation, Carnegie Mellon University.

[35] Wang C., Jing F., Zhang L., Zhang H-J., 2006a, *Image annotation refinement using random walk with restarts*. In Proceedings of the 14th ACM International Conference on Multimedia.

[36] Zhu X., Ghahramani Z., Lafferty J., 2003, *Semi-supervised learning using gaussian fields and harmonic function*. In Proceedings of the 20th International Conference on Machine Learning.

[37] Ebert S., Fritz M., Schiele B., *Pick your Neighborhood - Improving Labels and Neighborhood Structure for Label Propagation*. DAGM-Symposium, vol. 6835 of Lecture Notes in Computer Science, pp. 152-162. Springer, (2011).

[38] Tian Z., Kuang R., *Global Linear Neighborhoods for Efficient Label Propagation*, SDM, pp. 863-872, SIAM/Omnipress (2012).

[39] Chen X., Mu Y., Yan S., Chua T-S., 2010, *Efficient large-scale image annotation by probabilistic collaborative multi-label propagation*. In Proceedings of the international conference on Multimedia (MM '10). ACM, New York, NY, USA, pp. 35-44. doi:10.1145/1873951.1873959.

[40] Indyk P. and Motwani R., *Approximate nearest neighbors: Towards removing the curse of dimensionality*. In Proceedings of the Symposium on Theory Computing, 1998.

[41] Andoni A. and Indyk P., *Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions*. Commun. ACM, 51(1), pp. 117–122, February 2008.

[42] Mu Y., Shen J., Yan S., *Weakly-supervised hashing in kernel space*. In CVPR, 2010.

[43] Wang X-J., Zhang L., Jing F., Ma W-Y., 2006b, *Annosearch: Image auto-annotation by search*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[44] Wang X-J., Zhang L., Li X., Ma W-Y., 2008, *Annotating images by mining image search results*. IEEE Trans. Patt. Anal. Mach. Intell. 30, vol. 11, pp. 1919–1932.

[45] Torralba A., Fergus R., Freeman W., 2008, *80 million tiny images: A large data set for nonparametric object and scene recognition*. IEEE Trans. Patt. Anal. Mach. Intell. 30, vol. 11.

[46] Wang F. Zhang C., 2008, *Label propagation through linear neighborhoods*. IEEE Trans. Knowl. Data Engin. 20, vol. 1, pp. 55–67.

[47] Tang J., Hua X-S., Song Y., Qi G.J., Wu X., 2008, *Video annotation based on kernel linear neighborhood propagation*. IEEE Trans. Multimedia 10, vol. 4.

[48] Wang M., Zhou X., Chua T-S., 2008, *Automatic image annotation via local multi-label classification*. In Proceedings of the 2008 international conference on Content-based image and video retrieval (CIVR '08). ACM, New York, NY, USA, pp. 17-26. doi:10.1145/1386352.1386359.

[49] Sorokin A., Forsyth D., *Utility data annotation with amazon mechanical turk*. In Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on, pp. 1-8, 2008.

[50] Vondrick C., Ramanan D., Patterson D., *Efficiently scaling up video annotation with crowdsourced marketplaces*. In Computer Vision-ECCV 2010, pp. 610-623. Springer, 2010.

[51] Deng J., Dong W., Socher R., Li L., Li K., Fei-Fei L., (2009), *ImageNet: a large-scale hierarchical image database*. In: Proc. CVPR, pp. 710-719.

[52] Kumar N., Berg A.C., Belhumeur P.N., Nayar S.K., (2009), *Attribute and Simile Classifiers for Face Verification*. In: ICCV.

[53] Biewald L., (2012), *Massive multiplayer human computation for fun, money, and survival*. In: Proceedings of the 11th international conference on current trends in web engineering, ser. ICWE'11, pp. 171–176.

[54] von Ahn L., Dabbish L., (2004), *Labeling images with a computer game*. In: Proceedings of the SIGCHI conference on human factors in computing systems, ser. CHI '04. ACM, New York, pp. 319–326.

[55] von Ahn L., Liu R., Blum M., (2006), *Peekaboom: a game for locating objects in images*. In: Proceedings of the SIGCHI conference on Human Factors in computing systems, ACM, pp. 55-64.

[56] Welinder P., Branson S., Belongie S., Perona P., (2010), *The multidimensional wisdom of crowds*. In: Neural Information Processing Systems Conference (NIPS), vol. 6, p. 8.

[57] Quinn A.J., Bederson B.B., (2011), *Human computation: a survey and taxonomy of a growing field*. In: Proceedings of the 2011 annual conference on human factors in computing systems, CHI '11, pp. 1403–1412.

[58] Snow R., O'Connor B., Jurafsky D., Ng A.Y., *Cheap and Fast - But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks*. In EMNLP, 2008.

[59] Kass M., Witkin A., Terzopoulos D., *Snakes: Active contour models*, International Journal of Computer Vision, vol. 1, no. 4, pp. 321-331, Jan. 1988.

[60] Rother C., Kolmogorov V., Blake A., *"GrabCut": interactive foreground extraction using iterated graph cuts*, ACM Trans. Graph., vol. 23, no. 3, pp. 309-314, 2004.

[61] Stauffer C., Grimson W. E. L., *Adaptive background mixture models for real-time tracking*, Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol. 2, pp. 246-252, 1999.

[62] Bradski G. R., *Computer Vision Face Tracking For Use in a Perceptual User Interface*, Intel Technology Journal, pp. 1-15, 1998.

[63] Canny J., (1986), *A computational approach to edge detection*. IEEE Trans Pattern Anal Mach Intell. 8(6), pp. 679–698.

[64] Brabham D., (2008), *Crowdsourcing as a model for problem solving an introduction and cases*. Convergence 14(1), pp. 75–90.

[65] Howe J., (2006), *The rise of crowdsourcing*. Wired Magazine 14(6), pp. 1–4.

[66] Kavasidis I., Spampinato C., Giordano D., *Generation of ground truth for object detection while playing an online game: Productive gaming or recreational working?* 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2013.

[67] Barnich O. and Van Droogenbroeck M., *Vibe: a powerful random technique to estimate the background in video sequences*, in Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on. IEEE, 2009, pp. 945–948.

[68] Dubes R. and Jain A.K., *Clustering techniques: The user's dilemma*. Pattern Recognition, 8(4), pp. 247-260, 1976.

[69] Adams R. and Bischof L., *Seeded region growing*, Pattern Analysis and Machine Intelligence, IEEE Transactions on, 16(6), pp. 641-647, 1994.

[70] Rother C., Kolmogorov V., Blake A., *Grabcut: Interactive foreground extraction using iterated graph cuts*. In ACM Transactions on Graphics (TOG), vol. 23, pp. 309-314, ACM, 2004.

[71] Kohli P. and Torr P.H.S., *Effiently solving dynamic markov random fields using graph cuts*. In Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, vol. 2, pp. 922-929, 2005.

[72] Dalal N., Triggs B., *Histograms of oriented gradients for human detection*, Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol.1, pp. 886-893, 25-25 June 2005 doi: 10.1109/CVPR.2005.177.

[73] Ludwig O., Delgado D., Goncalves V., Nunes U., *Trainable Classifier-Fusion Schemes: An Application To Pedestrian Detection*, In: 12th International IEEE

Conference On Intelligent Transportation Systems, 2009, St. Louis, 2009 vol. 1, pp. 432-437.

[74] Hartley H., *Maximum likelihood estimation from incomplete data*. Biometrics, 14, pp. 174–194, 1958.

[75] Dempster A. P., Laird N. M., Rubin D. B., *Maximum likelihood from incomplete data via the EM algorithm*. Journal of the Royal Statistical Society B, 39(1), pp. 1–22, 1977.

[76] Neal R. M. and Hinton G. E., *A new view of the EM algorithm that justifies incremental, sparse and other variants*. In M. I. Jordan, editor, Learning in Graphical Models, pp. 355–368. Kluwer Academic Publishers, 1998.

[77] Bosch A., Muñoz X., Freixenet J., *Segmentation and description of natural outdoor scenes*, Image and Vision Computing, Vol. 25, Issue 5, 1 May 2007, pp. 727-740.

[78] Barnard K., Forsyth D., *Learning the semantics of words and pictures*, in: International Conference on Computer Vision, vol. 2, 2001, pp. 408–415.

[79] Giordano D., Kavasidis I., Pino C., Spampinato C., (2011), *A semantic-based and adaptive architecture for automatic multimedia retrieval composition*. In: 2011 9th international workshop on content-based multimedia indexing (CBMI), pp. 181–186.

[80] Mai H.T. and Kim M.H., (2013), *Utilizing similarity relationships among existing data for high accuracy processing of content-based image retrieval*. Multimed Tools Appl. doi:10.1007/s11042-013-1360-9.

[81] Rotter P., (2013), *Relevance feedback based on n-tuplewise comparison and the ELECTRE methodology and an application in content-based image retrieval*. Multimed Tools Appl. doi:10.1007/s11042-013-1384-1.

[82] Marques O., Barman N., (2003), *Semi-automatic semantic annotation of images using machine learning techniques*. The Semantic Web-ISWC 2003, pp. 550–565.

[83] Mihalcea R., Corley C., Strapparava C., (2006), *Corpus-based and knowledge-based measures of text semantic similarity*. In: Proceedings of the national conference on artificial intelligence, vol 21, no 1. AAAI Press, Menlo Park, MIT Press, Cambridge, p. 775, 1999.

[84] Wache H., Voegele T., Visser U., Stuckenschmidt H., Schuster G., Neumann H., Hübner S., (2001), *Ontology-based integration of information-a survey of existing approaches*. In: IJCAI-01 workshop: ontologies and information sharing, 2001, Citeseer, pp. 108–117.

[85] Freund Y., Schapire R.E., (1995), *A decision-theoretic generalization of on-line learning and an application to boosting*. In: Computational learning theory. Springer, pp. 23–37.

[86] Li X., Aldridge B., Rees J., Fisher R.B., *Estimating the ground truth from multiple individual segmentations with application to skin lesion segmentation*, in Proc. Medical Image Understanding and Analysis Conference, UK, 2010, pp. 101–106.

[87] Spampinato C., Palazzo S., Giordano D., Kavasidis I., Lin F-P., Lin Y-T, *Covariance based fish tracking in real-life underwater environment*. In VISAPP (2), pp. 409-414, 2012.

[88] Di Salvo R., Giordano D., Kavasidis I., 2013, *A crowdsourcing approach to support video annotation*. In Proceedings of the International Workshop on Video and Image Ground Truth in Computer Vision Applications (VIGTA '13). ACM, New York, NY, USA, , Article 8 , p. 6, doi=10.1145/2501105.2501113.