An Extensible Tool for the Annotation of Videos Using Segmentation and Tracking

Marc Ritter and Maximilian Eibl

Chemnitz University of Technology
Chair Media Computer Science
09107 Chemnitz, Germany
{marc.ritter,eibl}@informatik.tu-chemnitz.de

Abstract. Due to massive amount of data, the description of audiovisual media by metadata nowadays can benefit by the support of (semi-)automatic methods during the annotation process. The presented tool enables the user to mark, interactively segment and track preselected objects. An integrated shot detection splits the video into disjoint parts, for instance to circumvent the semi-automated tracking of objects across shot boundaries. Arbitrary application dependent custom image processing chains can be created in conjunction with the research framework AMOPA. Created data is exported in compliance to MPEG7-DAVP.

Keywords: Annotation tool, Image and video processing, Workflow, Object segmentation and tracking.

1 Introduction

In the past years, the application of audiovisual media has grown rapidly among all types of media archives, multimedia centers, knowledge management and e-learning systems, producing a vast and steadily increasing amount of data. Making this information overload searchable becomes more and more challenging concerning time and capacity. The main goal of our works is to decrease the user's workload coming with the annotation process. This goal is successively being achieved by automating the single steps of the workflow.

Machine learning and statistical pattern recognition is engaged in the fields of audio, video and image processing in order to develop classifiers capable of detecting (arbitrary) objects. Unfortunately, the inherent and inevitable training process of a classifier usually demands a huge set of annotated training data, containing examples of the targeted objects. These intellectual annotations are time-consuming and repetitive, and they require a lot of human interaction and attention. Easy, fast and reliable annotation processes may speed up the development cycles of algorithms as well as strengthen a hypothesis of scientific methods when applied on larger data sets.

1.1 Related Work

During the last decades, a lot of different tools have been developed to ease the annotation of images and videos. Successfully applied on the *TREC Video Retrieval*

A. Marcus (Ed.): Design, User Experience, and Usability, Pt I, HCII 2011, LNCS 6769, pp. 295–304, 2011.

Evaluation campaign [1] the ViPER toolkit [2] enables the user to mark single objects in videos, whilst creating descriptors by assistance of a schema editor. In *Caliph* the context of an image can be depicted using a directed semantic graph. Its nodes describe objects and the edges describe the relation between them. Unfortunately, the position of objects cannot be denoted. Its co-partner *Emir* retrieves similar images by means of MPEG-7 low-level descriptors [3]. *Vezzani et al.* train and detect objects on videos while also relying on low-level descriptors [4]. Descriptors of user-selected regions within images are extracted within the *M-OntoMat-Annotizer* to determine objects and their properties.

To minimize the error rate of demanding intellectual annotation the *Multimedia Analysis and Retrieval System* (*MARVEL)* substitutes the graphical user annotation interface against methods from the area of statistical analysis. These methods sort images automatically into categories of a complex taxonomy [6]. A similar approach is proposed by the *VideoAnnEx Annotation Tool* by providing predefined dictionaries for the annotation of key objects, events and static scenes [7].

Structured knowledge can be modeled by extensible XML dictionaries containing categories that can be used for tagging frames or shots [8]. To facilitate frame based annotation, algorithms for semi-automated segmentation and tracking of objects by active contours have proven valuable [9]. *Goldman et al.* initially annotate the shape of objects to interactively track and manipulate its shape within a video [10].

Originally developed to annotate spoken dialogues, *ANVIL* [11] now supports the annotation of diverse video coding schemes. It also integrates XML based export of generated metadata for post processing by toolboxes like SPSS. *Kipp* exploits these coding schemes by adding a spatiotemporal function to mark objects beyond several frames. Multiple sources of video and various sensor data can be processed by the Mac tool *VCode and VData* [12] allowing marking of objects at a timeline. Finally, the consistency of annotations might be checked using integrated verification functions.

A different approach is followed by the tool *FilmEd* [13], which grants multiple annotations from various people simultaneously when connected via a heterogeneous network. Even mobile devices start to be equipped for the annotation process but with very obsolete functions [14].

1.2 Motivation

A lot of efforts have been done to facilitate the cost-intense process of *frame by frame* annotation. On the one hand both object segmentation and tracking could be automated. Unfortunately this option is rarely supported. On the other hand multiple images of a sequence might be tagged with a common semantic concept or item. Albeit the tag itself may be restricted due to the accuracy and correctness of the results of machine preprocessing.

Fan et al. try to close the semantic gap between low-level features and high-level concepts by identifying salient objects within images and matching them with respect to base-level atomic image concepts [16]. A simplified approach is worth striving for

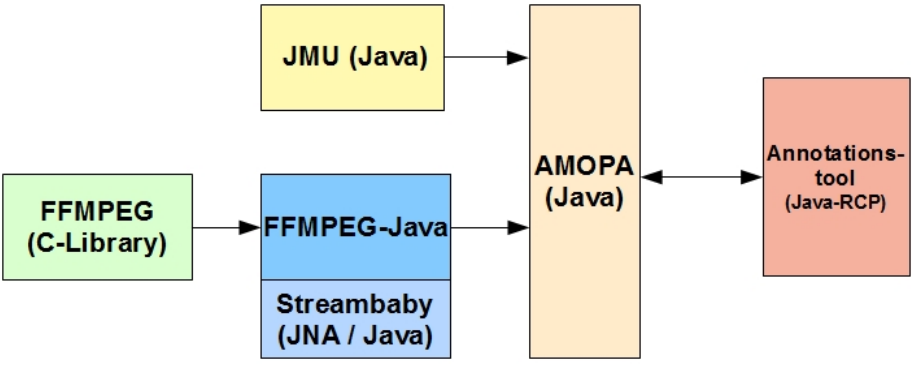

Fig. 1. Architecture of the annotation tool (right). Data is exchanged with AMOPA (middle), that aggregates the functionality of JMU and Streambaby. (from [15])

in any video annotation process. Furthermore, tools potentially lack of the choice of data export to (post) process data individually, depending on the objective of the application.

The here presented annotation tool combines the previously mentioned advantages, while providing the export of annotations to MPEG7-DAVP and the opportunity to model and create flexible and extensible application-dependent processing chains by interconnecting the annotation tool to the research framework AMOPA [17]. Thus, a possibility to integrate fast multi-threaded processing chains is supplied beside methods for rapid structural decomposition of videos as well as marking, segmentation and tracking of objects.

In contrast to images, videos usually contain a lot of motion and various camera perspectives of objects. Hence, we can enhance the quantity of annotations when capturing a number of different views of an object. This raises the amount of available training data leading to a higher quality for further processing, especially for instance in machine learning. Inherently the effort of annotation may decrease.

2 System Description

The annotation tool consists of two components. The first one is the framework *Automated MOving Picture Annotator* (AMOPA) – a detailed description is provided by [18]. AMOPA integrates a huge set of concepts and process chains that are frequently used in image, video and audio processing. Access to video data is granted by the open source C-library FFMPEG, whereas this functionality is encapsulated and passed on to Java via *Streambaby* (cf. fig. 1). Recent extensions of the process concept of *Java Media Utility* (JMU) even allow the implementation of non-linear processing chains. By this, concepts of almost arbitrary topology can be realized. Relaying single image processing steps in the chain is achieved by a graphical editor or via XML. Every step is started as a single thread to fully utilize the capacity of multi-core CPU environments.

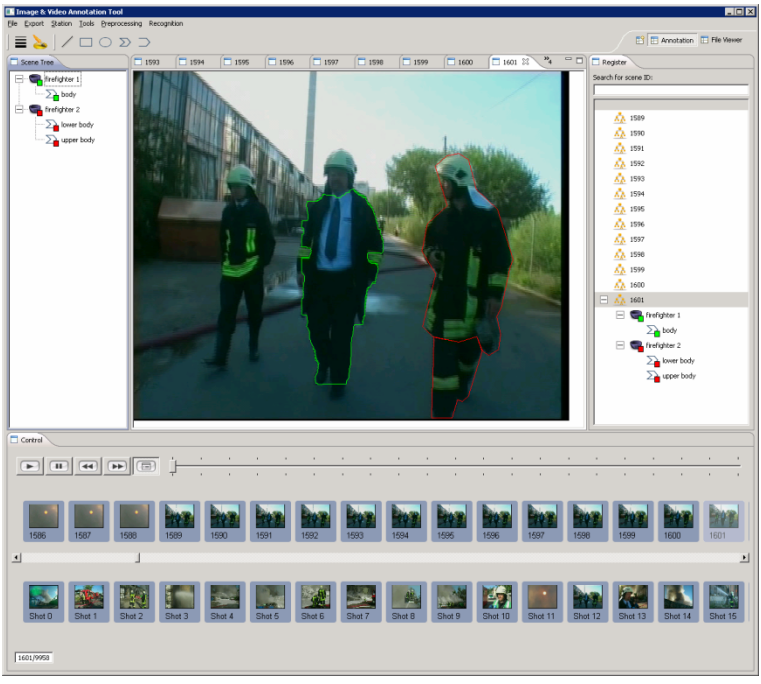

Fig. 2. Process of the annotation: a person is semi-automatically segmented (middle). The segmentation is show by a green polygon. The other person (right) is intellectually annotated by two red polygons indicating upper and lower body.

The second component basically comprises the annotation tool, which is developed in *Java RCP* and runs in Windows and Linux systems. The combination of *JFace* and *SWT* allows fast rendering of graphical content.

Fig. 2 shows the graphical user interface during the process of annotating. A classical menu bar provides functions to load, save, export data and to preprocess images and recognize objects. Preprocessing includes methods for shot boundary detection and interactive object segmentation. The recognition stage currently contains a method to track selected objects. The icon bar symbolizes options to create different figures like lines, rectangles, circles, and open and closed polygons as well as properties to set color and line width. The current annotation frame is shown in the center.

Object hierarchies are visualized by trees. The leftmost (scene) folder reflects a tree displaying the annotated content of the current frame. A more global content register containing all annotated frames of the video in form of trees is displayed on the right. Navigating through videos is realized by the control buttons below the annotation frame.

At the bottom of the screen two rows of images are presented. The upper row grants access to all frames of the current video. After the completion of the shot boundary detection, representative key frames of the corresponding shots are

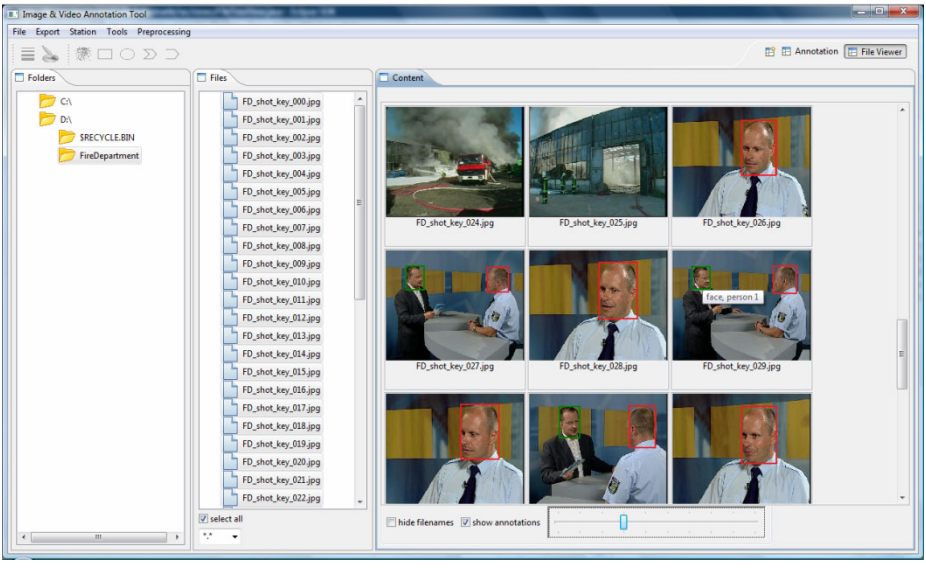

Fig. 3. The file viewer perspective of the annotation tool showing the annotated data as overlays on the image data

displayed in the lower row. Thus, the tool enables the user to scene-wise navigation as well as shot-wise navigation. Frames are transferred from the control to the annotation view via double clicking.

The latest feature of the annotation tool is a file viewer perspective that enables one to load and show images from disk in a scalable grid orienting on the well-known capabilities to view collections of thumbnails or photos within tools like *IrfanView* and *Microsoft Photo Gallery 2011*. Fig. 3 shows disk-stored representative key frames extracted from a fire department movie after the application of the shot detection. Two persons that occur in a dialogue scene have been consistently intellectually annotated with different colors (green and red) for each face over multiple images.

Microsoft Photo Gallery 2011 appends XML based annotations like the name and position of a face directly to image. In contrast our perspective loads annotated data from a separate file from the same folder and file name and overlays each image with its annotations. If the mouse is hovered over an annotation its description is shown in a toolbox tip.

As work in progress we currently prepare this perspective for (re)editing of overlays/annotations in the grid of the file viewer on the fly. Hence, simple annotation in multiple images can be done at once. On the other hand we target to implement the opportunity to rapidly evaluate the results of object classifiers on images or videos. Therefore, results must be stored within an compatible format to be visualized in the

file viewer. An implemented logic function would allow to easily accept, reject or even score detections to grant further evaluations.

3 Applied Algorithms

In the following further insights into the available algorithms of the annotation tool are provided. Each method is implemented in a separate image processing chain within the framework AMOPA and can be controlled by an interface of shared objects.

3.1 Navigating in Videos and Data Export

Jumping to a specific frame within a video can turn out to be very difficult to implement, since the sequential streaming of the video has to be interrupted. Methods to solve this problem highly depend on the underlying video codec. This especially concerns the here applied version of FFMPEG when decoding MPEG video.

In contrast to decoding and object instantiation which is needed to finally access the data of a video frame, simple sequential decoding without object creation is multiple times faster. In simple benchmarks our system created between 70 and 100 accessible frames per seconds and decoded more than 1,200 frames per seconds. Therefore, currently we elude the problem of jumping by simply decoding the video data from the start to the requested time stamp.

When processing long videos, the performance of this method can be very poor because of longer reloading periods. Speeding up the process can eventually be achieved when indexing the *IFrames* of the video by assistance of an external tool like *Avidemux*[1]. Indexed access would allow us to jump directly onto the extracted byte position of the requested *IFrame* without any time loss.

Annotation data can be exported optionally to MPEG-7-DAVP [26], plain text or a customizable XML format.

3.2 Shot Boundary Detection

The detection of shot boundaries is based on [19], but actually uses only a small sample of the features proposed there to detect hard cuts (cf. [17]). Our test system (Dual Quad Core CPU, 3 GHz) achieved a processing speed of factor 2.2 times faster than real-time at 65 per cent workload in average, independent of the resolution of the video. Therefore, the video is first resized to half PAL resolution and then divided into disjoint blocks of *48 x 48* pixels.

Individual motion vectors are calculated for each block between two successive frames. The error between the original block and its motion compensation is computed by the minimum absolute distance of the blocks pixels. The ratio between the cumulative sum of the errors from all blocks and an additive function, smoothing the complete error sum of previous frames, triggers the detection of a transition.

[1] http://avidemux.org/

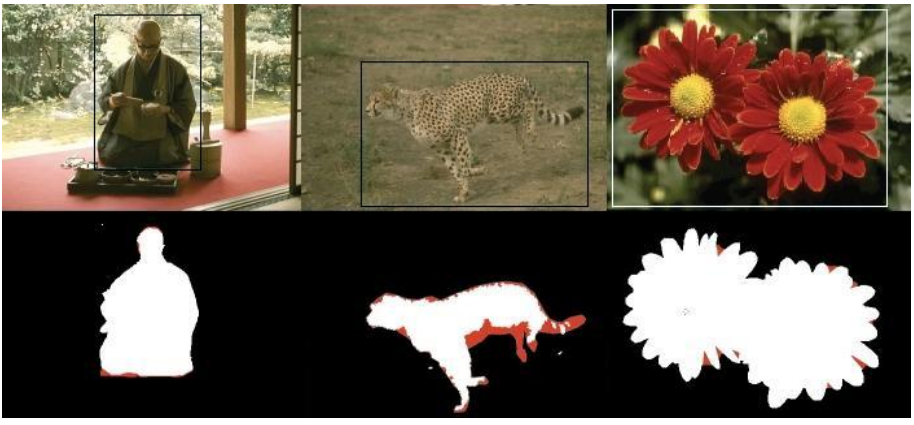

Fig. 4. Evaluation of the results of the *GrabCut* algorithm. Original images with selected regions in bounding boxes are displayed in the upper row. Grey color visualizes the differences to the manually created ground truth shown below. (modified from [24])

Table 1. Precision of the *GrabCut* method (information is given in pixels)

Image from Fig. 3	Overall area	GrabCut-Segmentation	Intellectual Segmentation	False detections	Error rate
Buddhist	151.526	24.826	24.501	575	2,35 %
Leopard	150.416	19.973	24.510	4.669	19,05 %
Flower	152.044	67.627	68.259	632	0,93 %

In contrast to the evaluation of [20], we were able to enhance the precision of the method by using MPEG-7 edge histograms within an environment of five adjacent frames of a shot candidate. Each candidate is tested for dissimilarity. Regarding the content of local television stations (> 100 hours of video), this method achieves detection rates of about 99 per cent with an empirical false-positive rate of max. 1.5 per cent.

3.3 Object Segmentation

Determining the exact contours of objects may be crucial for the process of feature extraction and subsequent processing. The intellectual annotation of such contours can be performed by free form or polygon tools, but is often time-consuming. To calculate the contours of an object, we propose the user to surround the object with a bounding box, before starting the semi-automated segmentation. Therefore, we use an implementation of [21], where two mixtures of Gaussians (MOG) are separately modeled for fore- and background. Pixels within the bounding box are assigned to the foreground, others to the background MOG. For each model the color reduction method of [22], which associates the pixels of the selected region to the most likely model is run. Then, its color distribution is computed. Afterwards, a graph is built by

Fig. 5. Tracking of the selected object (inner bounding box) within the initial image (left). Result of the automatic object tracking 24 frames later (right). Displayed imagery from TRECVID 2009 [1]. (modified from [24])

using the pixel distribution of both MOGs as source and sink to calculate the minimum cut [23]. This procedure is iterated until the set of pixels within both models remains constant. Results are visualized by fig. 4 and table 1. Acceptable results are achieved, if the object does not contain too much dissimilar colors. Problems also arise at strong textures, shadows as well as inhomogeneous background preventing a sharp distinction of the object.

3.4 Object Tracking

The annotation tool can track selected objects for a user defined number of frames. Fig. 5 outlines the process of tracking. First, an object is encircled by a bounding box, visualized by the blue inner rectangle on the left. The light yellow-green outer rectangle symbolizes the automatically calculated search window used for tracking the object within the next frame. Referring to [25], we apply a simple block-matching method with n step search by computing the minimum absolute distance from all pixels of the selected region and potential positions within the search window.

Optionally, the curved light blue line along the body of the walking person has been created by automated segmentation. In the current implementation this segmentation is optional and can be applied within every frame but does not affect the results of the tracking. Future work may aim to investigate methods to make this approach more reliable and even invariant to object deformation and changes. Usually, this can be achieved by the attempt to extract and track standardized object features from the bounding box or automated segmentation.

4 Future Work

We have described the current stage of development of an annotation tool that allows the intellectual annotation of semi-automated segmented and automatically tracked objects. Future implementations will concern functions to create ontologies and thesauri to make the tagging of objects more reliable and consistent. Furthermore, an object browser will support global searching for annotated objects within a video.

Another important step will be the integration of algorithms for the composition of shots to identify scenarios like dialogs and news broadcasting as well as to further enhance the exploited techniques for segmentation and tracking (cf. [27]). Methods for automated text and speaker recognition, and speech detection have already been developed within AMOPA. They are ready to be integrated into the annotation tool to enrich the set of functionality for annotation and analysis.

Acknowledgments. This work was partially accomplished within the project *sachsMedia – Cooperative Producing, Storage and Retrieval* (Project 03IP608), funded by the program Entrepreneurial Regions of the Federal Ministry of Education and Research, Germany.

References

1. Smeaton, A.F., Over, P., Kraaij, W.: Evaluation Campaigns and Trecvid. In: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval MIR 2006, New York, NY, USA, pp. 321–330 (2006)
2. Doermann, D., Mihalcik, D.: Tools and Techniques for Video Performance Evaluation. In: Proc. 15th International Conference on Pattern Recognition, vol. 4, pp. 167–170 (2000)
3. Lux, M.: Caliph & Emir: MPEG-7 Photo Annotation and Retrieval. In: Proceedings of the 17th ACM International Conference on Multimedia, Beijing, China, pp. 925–926 (2009)
4. Vezzani, R., Grana, C., Bulgarelli, D., Cucchiara, R.: A Semi-Automatic Video Annotation Tool with MPEG-7 Content Collections. In: Proceedings of the 8th IEEE International Symposium on Multimedia, San Diego, CA, USA (2006)
5. Petridis, K., Anastasopoulos, D., Saathoff, C., Timmermann, N., Kompatsiaris, Y., Staab, S.: M-OntoMat-Annotizer: Image Annotation. In: Linking Ontologies and Multimedia Low-Level Features. Engineered Applications of Semantic Web Session at the 10th Int'l. Conf. on Knowledge-Based & Intelligent Information & Engineering Systems, U.K. (2006)
6. Columbia University. IBM T. J. Watson Research Center: MARVEL: Multimedia Analysis and Retrieval System, `http://domino.research.ibm.com/comm/research_people.nsf/pages/jsmith.projects.html`
7. Naphade, M.R., Lin, C.-Y., Smith, J.R., Tseng, B., Basu, S.: Learning to Annotate Video Databases. In: Proc. SPIE, Storage and Retrieval for Media Databases vol. 4676, pp. 264–275 (2002)
8. Kounoudes, A., Tsapatsoulis, N., Theodosiou, Z., Milis, M.: A multi-level Video Annotation Tool based on XML-dictionaries. In: Proc. of the 10th WSEAS International Conference on Mathematical Methods, Computational Techniques and Intelligent Systems, Corfu, Greece (2008)

9. Luo, H., Eleftheriadis, A.: Designing an Interactive Tool for Video Object Segmentation and Annotation. In: Proc. of the 7th ACM International Conference on Multimedia, Orlando, FL, USA, pp. 265–269 (1999)
10. Goldman, D.B., Gonterman, C., Curless, B., Salesin, D., Seitz, S.M.: Video Object Annotation, Navigation, and Composition. In: Proc. of the 21st Annual ACM Symposium on User Interface Software and Technology, Monterey, CA, USA (2008)
11. Kipp, M.: Spatiotemporal Coding in ANVIL. In: Proc. of the 6th International Conference on Language Resources and Evaluation, LREC (2008)
12. Hagedorn, J., Hailpern, J., Karahalios, K.G.: VCode and VData: Illustrating a new Framework for Supporting the Video Annotation Workflow. In: AVI 2008, Napoli, Italy, pp. 317–321 (2008)
13. Schroeter, R., Hunter, J., Kosovic, D.: FilmEd: Collaborative Video Indexing, Annotation and Discussion Tools Over Broadband Networks. In: Proc. of the 10th IEEE International Conference on Multimedia Modeling, Los Alamitos, California, USA, pp. 346–353 (2004)
14. Concejero, P., Munuera, J., Lorenz, M.: The MESH Mobile Video Annotation Tool. In: Proc. of the 5th ACM Nordic Conference on Human-computer Interaction: Building Bridges, NordiCHI 2008, Schweden (2008)
15. Ritter, M., Eibl, M.: Ein erweiterbares Tool zur Annotation von Videos. In: Proc. 12th International Symposium on Information Science 2010, Hildesheim, Germany (2010) (in press)
16. Fan, J., Gao, Y., Hangzai, L., Jain, R.: Mining Multilevel Image Semantics via Hierarchical Classification. IEEE Transactions on Multimedia 10(2), 167–187 (2008)
17. Ritter, M., Eibl, M.: Visualizing steps for shot detection. In: LWA 2009: Lernen - Wissen - Adaption, Workshop Proceedings, Darmstadt, Germany, pp. 98–100 (2009)
18. Ritter, M.: Visualisierung von Prozessketten zur Shot Detection. In: Workshop Audiovisuelle Medien: WAM 2009, Technical Reports of Computer Science (CSR-09-04), Chemnitz University of Technology, Germany, pp. 135–150 (2009)
19. Liu, Z., Gibbon, D., Zavesky, E., Shahraray, B., Haffner, P.: AT&T RESEARCH AT TRECVID 2006 Workshop Contribution. AT&T Labs-Research (2006)
20. Zwicklbauer, S.: Evaluierung und Implementierung von Shot-Boundary-Detection-Algorithmen zur automatischen Video-Anno¬tation. Bachelor thesis, Universität Passau, pp. 48–52 (2010)
21. Talbot, J.F., Xu, X.: Implementing GrabCut. Brigham Young University, Provo, UT, USA (2006), http://www.justintalbot.com/course-work/
22. Orchard, M., Bouman, C.: Color Quantization of Images. IEEE Transactions on Signal Processing 39(12), 2677–2690 (1991)
23. Boykov, Y., Kolmogorov, V.: An Experimental Comparison of Min-cut/Max-flow Algorithms for Energy Minimization in Vision. IEEE Trans. on Pattern Analysis and Machine Intelligence 26(9), 1124–1137 (2004)
24. Höhlig, S.: Analyse und Implementierung eines Verfahrens zur interaktiven semi-automatischen Objektmarkierung und -verfolgung. Bachelor thesis, Chemnitz University of Technology (2010)
25. Beck, P.: Implementierung eines flexiblen Algorithmus zum Tracking von Objekten in Java. Seminar paper, TU Darmstadt, Germany (1999)
26. Bailer, W., Schallauer, P., Neuschmied, H.: Description of the MPEG-7 Detailed Audiovisual Profile (DAVP). Technical Report, Joanneum Research, Graz, Austria (2007)
27. Price, B.L., Morse, B.S., Cohen, S.: LIVEcut: Learning-based Interactive Video Segmentation by Evaluation of Multiple Propagated Cues. In: Proc. International Conference on Computer Vision (ICCV 2009), Kyoto, Japan (2009)