Multimodal Indexing of Presentation Videos

Michele Merler

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2013

ABSTRACT

Multimodal Indexing of Presentation Videos

Michele Merler

This thesis presents four novel methods to help users efficiently and effectively retrieve information from unstructured and unsourced multimedia sources, in particular the increasing amount and variety of presentation videos such as those in e-learning, conference recordings, corporate talks, and student presentations.

We demonstrate a system to summarize, index and cross-reference such videos, and measure the quality of the produced indexes as perceived by the end users. We introduce four major semantic indexing cues: text, speaker faces, graphics, and mosaics, going beyond standard tag based searches and simple video playbacks. This work aims at recognizing visual content "in the wild", where the system cannot rely on any additional information besides the video itself.

For text, within a scene text detection and recognition framework, we present a novel locally optimal adaptive binarization algorithm, implemented with integral histograms. It determines of an optimal threshold that maximizes the between-classes variance within a subwindow, with computational complexity independent from the size of the window itself. We obtain character recognition rates of 74%, as validated against ground truth of 8 presentation videos spanning over 1 hour and 45 minutes, which almost doubles the baseline performance of an open source OCR engine.

For speaker faces, we detect, track, match, and finally select a humanly preferred face icon per speaker, based on three quality measures: resolution, amount of skin, and pose. We register a 87% accordance (51 out of 58 speakers) between the face indexes automatically generated from three unstructured presentation videos of approximately 45 minutes each, and human preferences recorded through Mechanical Turk experiments.

For diagrams, we locate graphics inside frames showing a projected slide, cluster them according to an on-line algorithm based on a combination of visual and temporal information, and select and color-correct their representatives to match human preferences recorded through Mechanical

Turk experiments. We register 71% accuracy (57 out of 81 unique diagrams properly identified, selected and color-corrected) on three hours of videos containing five different presentations.

For mosaics, we combine two existing suturing measures, to extend video images into in-the-world coordinate system. A set of frames to be registered into a mosaic are sampled according to the PTZ camera movement, which is computed through least square estimation starting from the luminance constancy assumption. A local features based stitching algorithm is then applied to estimate the homography among a set of video frames and median blending is used to render pixels in overlapping regions of the mosaic.

For two of these indexes, namely faces and diagrams, we present two novel MTurk-derived user data collections to determine viewer preferences, and show that they are matched in selection by our methods.

The net result work of this thesis allows users to search, inside a video collection as well as within a single video clip, for a segment of presentation by professor X on topic Y, containing graph Z.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost I would like to thank my advisor John Kender, who is not only an incredible researcher but also a splendid person. He has always had my best interest at heart from the very first day of my PhD journey, when he showed me how to read a research paper, to the multiple times when he has allowed and helped me to gain new research perspectives with internships outside Columbia, from the advice on how to teach a class, to the supervision of all the work of this thesis. I am extremely grateful for the guidance, wisdom and patience that he has showed me over the years. His breadth of knowledge not only in the in multimedia and high level vision fields, but also in a wide range of scientific and non-scientific topics is truly remarkable, and I have always felt grateful for the opportunity to interact with him.

My deepest thanks goes also to the rest of my dissertation committee, not only for their comments and suggestions on this work, starting from the candidacy and proposal, but also for all they have taught me over the years. Shih-Fu Chang for the guidance, collaboration and encouragement in the Aladdin project, since its inception. Peter Belhumer for allowing me, in his computational photography class, to have fun working on visual Captchas. Paul Natsev for all the research and life advice, for encouraging me to write my first journal article, and for calling me back to work with him so many times! Rong Yan for being a great mentor, guiding me to my first big conference publication and then beyond with invaluable help and advice on research, career and personal goals.

To Hui, the love of my life and my biggest inspiration, I can only say *xie xie*.

I want to thank my mom Gabriella and dad Fernando who have always been the lever in my life, supporting me unconditionally especially since I decided to come to Columbia, even if it meant not having their son living close to them any more. A huge hug to thank my fantastic family who always make me feel their love and support from across the ocean. My sister Giorgia, who always has my back, and my wonderful grandparents Mario and Pia, aunts Graci, Adriana and Germana, uncles Franco and Renato and cousins Chiara, Anna and Andrea.

At Columbia I have had the privilege to meet and interact with many remarkable people. I wish to thank them all starting from my labmates John Zhang, with whom I have had great fun going to conferences, and Mitch Morris. Sharing the office with Bert Huang and Blake Shaw, and lately

Anna Choromanska and Adrian Weller, has been a pleasure. From all of them I have learned the "awesomeness" of machine learning! I want to thank Yu-Gang Jiang and Yadong Mu for the great help and collaboration on the Trecvid MED tasks.

I wish to thank all the students of Fall 2009 COMS 3101-2, for their interest in Matlab and for making the first class I taught such a rewarding experience. A special thanks goes to my wonderful TAs: Gaurav Agarwal, Daniel Miau and Rohit Sethi. I would also like to thank the master students who have helped me greatly in my projects over the years: Pryiank Singh, Xueying Lu, Ran Wang and Sohpia Li.

A huge thanks to Daisy Nguyen, Shlomo Hershkop, Paul Blaer and all the CRF staff who have given me all the support I needed, when I needed it!

I also wish to thank all the researchers at IBM TJ Watson with whom I have worked over the years and have opened to the "universitarian" me a different perspecive on resaerch. They have inspired me and ignited my will to start there the next chapter of my life: Rong Yan, Paul Natsev, John Smith, Lexing Xie, Matt Hill, Gang Hua, Liangliang Cao, Noel Codella, Jelena Tesic, Bao Nguyen, Leiguang Gong, Rogerio Feiris, Ching-Yun Lin and my fellow interns Wei Jiang, Yi Zhang, Mingyuan Zhou and Hua Ouyang.

I will always be grateful to Serge Belongie. This thesis would not exist, hadn't he lighted my passion for computer vision research when I was visiting UCSD and inspired me to pursue a PhD.

My first impact with New York would not have been so fantastic without a group of wonderful friends that I have met at I-House. Innanzi tutto menzione d'onore per la mitica triade italiana: Piero e l'ipocondria, Lupo e il mitico N., Eva e l'Italian Cultural Hour. I want to thank Caroline, Michelle and Tan for their friendship and the wonderful conversations that started from the I-House dinner tables and ended all over NYC, even Williamsburg!

I would like to thank also all the friends who have come over the years to visit me, bringing me a little slice of home: Guglie, Tarolli, Chiara, Bobo, Ale, Da, Mattia, Vale.

The list of people that I should thank is really endless. I am tremendously grateful to all the wonderful people I have met since coming to New York and who have helped me view things differently or even for the first time, both in research and life: thank you all!

to re-search

Chapter 1

Introduction

1.1 Motivation and Domain Description

The exponential diffusion of unstructured multimedia content on sites such as Youtube (48 hours of video is uploaded every minute[1]) and Flikr[2] has lately fostered a rapid growth of interest in the multimedia and vision community toward a new line of systems to recognize people and activities "in the wild" [Huang *et al.*, 2007; Liu *et al.*, 2009], that is, in less structured, unconstrained and more realistic domains. Due to the lack of structure and to the low quality of the data, algorithms and paradigms designed for professional content often cannot directly be applied to the aforementioned domains, thus presenting a new challenge.

The work of this thesis is motivated by the need of an analysis paradigm for a particular instance of such "wild videos": unstructured presentation videos. Videos in this category have also been wide spreading on the web, besides the already existing large archives of universities and private companies. For example the portal Videolectures.net[3] has reported 700 events, 10K authors, over 12K lectures, and 15K videos of approximately 50 minutes each, while TalkMiner[4] has already harvested from the web more than 24K video lectures and talks.

The focus of this thesis lies in finding effective and efficient methods to index and such videos

[1] http://www.youtube.com/t/press_statistics

[2] www.flickr.com

[3] http://www.videolectures.net/

[4] http://talkminer.com/

based on four major cues: text, speaker faces, graphics and mosaics. Our system is intended to work in an unsourced framework, that is, without relying on any additional information besides the video itself. While many new recording systems are explicitly designed to synchronously capture all contents of presentations (including audio, video, and presentation material) [Zhang *et al.*, 2008], there already exist many large scale archives of raw videos, such as University lectures recordings, for which no other information, including electronic copy of the slides, is available.

The "wild" presentation videos that we analyze present a challenge for standard processing techniques, starting from the low quality. Unlike broadcast news, sports and music videos, TV series episodes or movies, they were not captured by professionals or through an ad-hoc capture systems, but with camcorders, personal video cameras or even smartphones. Besides the recording process, resolution constitutes an issue also because of the compression coding applied when these videos are uploaded to the web, due to bandwidth and storage constraints. Furthermore, they were not edited in any way. Therefore they completely lack a structure, which is usually exploited by content based video indexing and retrieval systems.

Figure 1.1 showcases some of the challenges involving the analysis of the content in this domain.

From the text and graphics recognition point of view, they present a challenge in that the camera is rarely steady and its movement is unconstrained, the projected slides are often truncated out from the field of view or occluded by the speakers. For what concerns face matching and recognition, the system needs to take advantage of the segments in which the recording person focused his attention on the speaker, zooming in on his or her face, as the regular full view recordings present faces with a resolution too limited to properly extract rich descriptors.

The result work of this thesis allows users to search, inside a video collection as well as within a single video clip, for a segment of presentation by professor X on topic Y, containing graph Z.

1.2 Thesis Contributions

This thesis introduces four novel semantic cues to index and cross-reference presentation videos: text, speaker faces, graphics, and mosaics. These semantic indexing tools go beyond standard tag based searches and simple video playbacks.

Most of the existing methods to summarize presentation videos rely on the availability of elec-

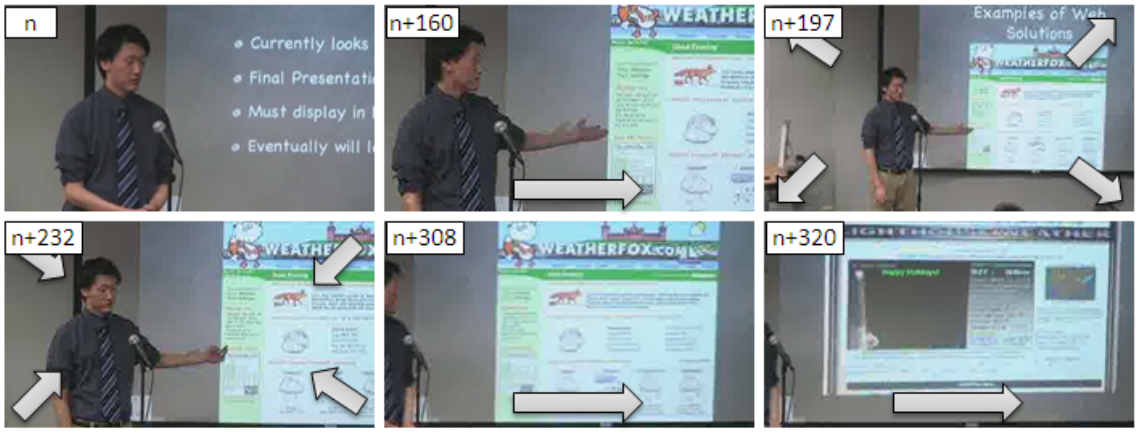


Figure 1.1: Examples of some of the challenges introduced by the presentation videos domain. The frames are sampled from an 11 seconds sequence from a video of students presentations, compressed as 432x240 MPEG-1 at 30fps. Note the rapidly changing camera movement, represented by gray arrows (right pan and zoom-in/zoom-out), which affects the resolution of the slide content as well as the speaker face. Slides are truncated in almost all the frames and the compression artefacts are evident at frame n+160.

tronic copies of the slides, which is not always realistic. Our work, on the other hand, focuses on the particularly challenging "unsourced" domain, in which no other source of information is available besides the video itself. Hence we propose a fully automatic method for summarizing and indexing unstructured presentation videos based on the four semantic cues.

The contributions of this thesis can be summarized as follows:

1. Within the the text indexing module, we introduce a novel binarization algorithm, Local Adaptive Otsu (LAO), to explicitly deal with the low quality of the video and the detected scene text regions. Using the LAO algorithm we obtain character recognition rates of 74%, as validated against ground truth of 8 presentation videos spanning over 1 hour and 45 minutes, almost doubling the baseline performance of the Tesseract[5] open source OCR engine.

2. We introduce a combination of a keyframe sampling method, which is proportional to estimated PTZ camera motion, and of a local-features based image stitching algorithm, in order to build video shots mosaics.

[5]http://code.google.com/p/tesseract-ocr/

3. The visual indexing algorithms proposed in this thesis were explicitly designed to match human preferences. When building the speaker faces and diagrams indexes, we adopt a user centric perspective, using the results of two Amazon Mechanical Turk surveys to gauge user preferences in terms of the visual appearance of the index icons. The results suggest that for a face index, users prefer to see a three-quarter, head and shoulder view of a person. For a diagram index, users prefer a white-balanced, color corrected image that covers as much as possible the whole area of the diagram.

4. For speaker faces, we detect, track, match, and finally select a humanly preferred face icon per speaker. The tracking algorithm integrates a generic object/face tracker as a noisy prediction in a simplified version of a Kalman filter named *K-Track*, which uses object/face detections as noisy observations. *K-Track* is used to mitigate the drifting effect, which typically affects appearance based tracking algorithms. We registered up to 5.7% relative improvement in tracking precision with respect to a state of the art multiple instance learning tracker on 3 unstructured presentation videos with a total of more than a quarter million frames.

5. We introduce the use of three quality measures, namely resolution, amount of skin, and pose, in order to simultaneously perform two selection tasks needed within the face indexing framework. The first selection process is necessary for tracks matching, in order to avoid the computational burden of comparing every pair of faces in each track. The second selection is needed for choosing a unique speaker face icon to be used in the final index. We register a 87% accordance (51 out of 58 speakers) between the face indexes automatically generated from three unstructured presentation videos of approximately 45 minutes each, and human preferences recorded through Mechanical Turk experiments.

6. We introduce, to the best of our knowledge, the first video index based on diagrams. We employ the average amount of color shift in a frame, either toward a high or low color temperature, to detect frames showing a projected slide. We cluster detected diagram regions according to an on-line algorithm based on a combination of visual and temporal information, and select and color-correct a representative per unique diagram. We register 71% accuracy (57 out of 81 unique diagrams properly identified, selected and color-corrected) on approximately three hours of videos containing five presentations.

1.3 Organization

The remainder of the thesis is organized as follows. In Chapter 2 we provide a survey of related work in automatic summarization and visualization of presentation videos, focusing in particular on text and visual (faces and diagrams) based indexing.

We then proceed to presents the contributions of the thesis, articulated as parts of a system that generates the semantic indexes, as outlined in Figure 1.2. In Chapter 3 we introduce the text indexing component, focusing on the contribution of the new Local Adaptive Otsu binarization algorithm in a standard scene text detection and recognition pipeline. This work has been introduced in [Merler and Kender, 2009]. Chapter 3.5 describes the generation of video shots mosaics. Chapter 4 presents the building blocks of the speaker face index, which rely on a steady-state Kalman Filter for tracking (Chapter 4.2.2), and on quality measures for matching human preferences on the appearance of the index icons (Chapter 4.3). Part of this work has been discussed in [Merler and Kender, 2011]. In Chapter 5 the diagram index construction process, which is also designed to match human preferences, is described in its parts: slide detection, graphics regions detection and clustering, followed by icon selection and color-correction. Finally we draw some conclusions and indicate directions of future work in Chapter 6

Some mathematical details of the derivation of the simplified Kalman filter used to track faces in the speaker face indexing module are provided in Appendix A. Given the extensive use of Amazon Mechanical Turk surveys and experiments to collect human preferences, we provide details and lessons learned in Appendix B.

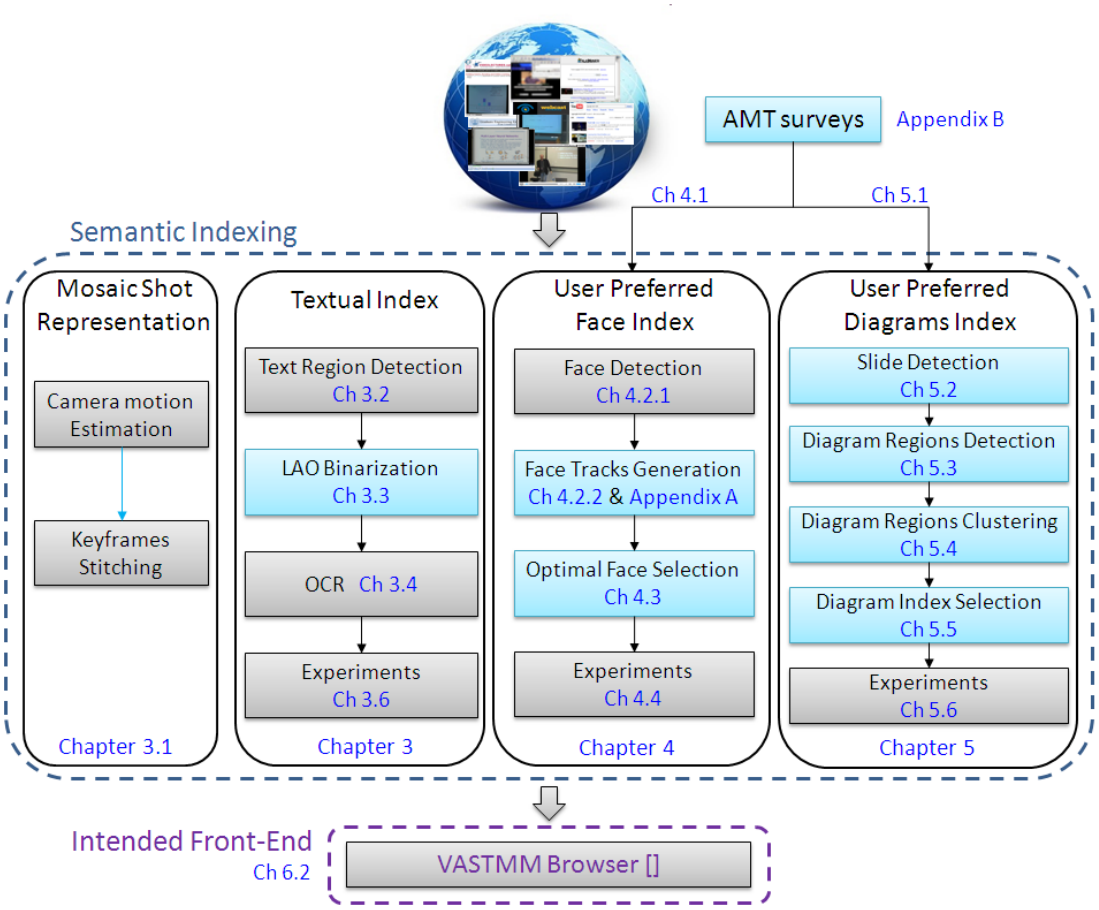


Figure 1.2: Overview of the presentation videos indexing system proposed in this thesis. Each video is processed to generate multi-modal semantic indexes, originated to satisfy user preferences collected thriugh Amazon Mechanical Turk surveys (Appendix B and Chapters 4.1 and 5.2): text and mosaics, discussed in Chapter 3, speaker faces, introduced in Chapter 4, and diagrams, outline in Chapter 5. In future work we plan to integrate the resulting indexes in the VAST-MM browser, as explained in Chapter 6.2. The original contributions of this thesis are highlighted in cyan.

Chapter 2

Related Work

The most common methods employed to index videos are based on keyframes and user-assigned tags. However, presentation videos offer richer semantic cues that can be exploited to perform indexing. In the following we review the state of the art in terms of presentation videos indexing based on slides, text, faces and mosaics.

2.1 Automatic Summarization and Visualization of Presentation Videos

Many systems proposed in the literature to summarize presentation videos use the slides as a reference to segment the videos into semantic shots, following studies which assess the reliability of slides as a summarization tool [He *et al.*, 2000]. Such systems generally perform synchronization by matching the content of the video stream to electronic copies of the slides.

[Fan *et al.*, 2006] propose a template matching approach, where the slides are treated as objects to be found in the videos. Alignment is performed by matching SIFT keypoints extracted from both the image of the slides and frames from the video, also estimating the projective transformations in the video. In a similar fashion, [Gigonzac *et al.*, 2007] detect the slide area within video frames with a color segmentation scheme, model the slide transitions with a Hidden Markov Model, and recover the slides sequence in the video with the Viterbi algorithm. A probabilistic model of slides transitions is also employed by [Liu *et al.*, 2002]. [Chen and Heng, 2003] instead synchronize electronic copy of the slides and speech transcripts obtained through a commercial software.

[Wang *et al.*, 2008] use the textual content of the slides to perform the matching. Text regions

are first automatically located within extracted keyframes, then enhanced through a superresolution technique before being binarized and finally fed to a commercial OCR engine. Using automatically extracted text from the video for the sole scope of synchronization with already available electronic copies of the slides offers limited performances compared to template matching approaches. Instead, in the (not uncommon) case where the slides are not available, we propose to use extracted text to automatically summarize and index the video, without any external reference.

Besides synchronizing slides and video or audio stream, other systems have been developed to index and summarize presentation videos, with particular attention to the structure of the final summaries and how they are exposed to users. Also, a general trend consists in integrating information from multiple information channels. [Amir *et al.*, 2004] introduced a system to almost fully automatically generate video proceedings of conference presentation videos recorded with an ad-hoc system with 4 cameras, a projector recording high quality images of the slides, and an audio recorder. The videos are indexed by the speech transcripts and made available for retrieval and browsing in a web page, together with other metadata information obtained from the conference website. The results of a user study show that, when only a representative keyframe per shot is presented and the audio track is played up to 1.7x faster, no noticeable difference in users understanding of the material is registered.

[Haubold and Kender, 2005; Haubold and Kender, 2007] propose a system to segment, summarize and browse presentation videos. Segmentation is performed both on audio (through the results of an automatic speech recognizer) and video (based on histograms of visual dissimilarity patterns) cues, which are integrated into the final result. A list of words extracted from external sources (electronic copies of slides or course website, if available) and the audio stream is automatically aligned with the video. The interface, named VASTMM browser, presents a temporal timeline of keyframes, audio and video activity, and words, which granularity can be adjusted by the user. User studies accessing the performances in terms of retrieval of useful information from the videos were conducted with 176 students on 32 presentations. Also [He *et al.*, 2000] conducted a user study to evaluate four different types of presentation video summarization: slides only, slides with text transcripts of the presenter's speech, slides with text in which keypoints have been highlighted, and audio/video summary. The evaluation is twofold. On one hand the effectiveness of the summarizations is analyzed based on how well users perform on tests concerning the subjects of the presentations when

using one or another summarization as a reference. Furthermore, a survey collects the impressions of the users in terms of efficiency, enjoyability, coherency and so forth. The results were favorable to the audio/video summarization.

[Friedland and Rojas, 2008] investigated the enhancement through contextualized content of presentation video summaries. They propose a lecture recording system in which the silhouette of the presenter, segmented from the background in the video stream, is superimposed to chalkboard or slides content, which was electronically recorded. This reproduction overcomes the split of attention effect [Friedland and Rojas, 2008], being able to convey the useful information included in gestures and facial expressions of the presenter and the text content of the board (or slides) in the same visual window.

[Anderson *et al.*, 2004] conducted an empirical study the results of which proved the existence of a relationship linking digital ink to spoken words in terms of co-expression, and to the text in the slides in terms of stressing of certain semantic concepts already present in the slides.

Most of the existing systems provide good performances in terms of video segmentation and indexing. However, most of them still present a heavy dependence from additional sources of information besides the videos, or an ad hoc hardware recording equipment. In many cases, a ”human in the loop” is still needed. In this framework, we propose unsourced, fully automatic methods to bridge the gap between existing systems and archived presentation videos which do not have additional references and were not recorded professionally.

2.2 Presentation Video Indexing and Retrieval

Presentation videos offer rich semantic cues that can be exploited to perform indexing, starting from the text projected in the video. [Vinciarelli and Odobez, Oct 2006] introduced a system to automatically index and retrieve presentation videos based on the semantic content of their slides. Each slide is obtained during the presentation through a frame grabber and then processed before automatically extracting its text with an OCR engine. Standard information retrieval techniques are then used to index words and documents and to perform queries based on terms. Also [Misra and Sural, 2006] extract and recognize scene text from videos to the end of indexing them based on text keywords. Tang and Kender [Tang and Kender, 2005] use automatically recognized handwritten

text to index and retrieve presentation videos. Candidate text regions are recognized and segmented in an integrated framework, using a neural network trained on different stroke representing features and dynamic programming to match segmented strokes to candidate vocabulary words, taken from a set of available course documents (textbook, online syllabus, electronic copies of the slides).

Recently researchers at FXPAL have introduced Talkminer [Adcock *et al.*, 2010], which is the first large scale fully automatic presentation video indexing system. The key component of the system is the full screen slide detector, which is based on frame differencing (color histogram based) augmented with spatial cues, speaker appearance modeling (Viola Jones face detection) and a text detector. An SVM is trained online to distinguish slide vs. speaker or background frames. Once a slide-frame is found, standard OCR is applied to it to generate an index of keywords, so that a user can perform a text search on the lecture videos corpus.

Besides text, many multimodal systems for video indexing have been proposed. One example in the presentation videos domain is the method proposed by [Martin *et al.*, 2007], who segment the videos into semantically meaningful shots according to two criteria: slides transitions and speaker detection and tracking. The presenter's gestures are also recognized.Content indexing is based on text recognition from the detected slides, which is further used to improve the language model of an automatic speech recognizer which analyzes the audio track of the video.

[Liew and Kan, 2008] propose a method to retrieve synthetic images in slides given textual queries. Features used are textual (directly from PPT + OCR), image (images size, number of colors, image type) and presentation (slide number, image location within slide, # images in slides). Slide segmentation is employed to to obtain slide images, and a two level hierarchy of image types (from [Wang and Kan, 2006]) is used to classify each image.

Also the use of camera motion cues to index videos has been investigated. In the system proposed by [Hirakawa *et al.*, 2002], queries can be performed based on pan and tilt motions, object motion within a shot, background color matching and moving object color matching. Each shot in the database to be queried is represented as a mosaic built from its frames, on which the moving object is superimposed. The camera motion parameters used to build the mosaic are also stored, and used to estimate the trajectory of a moving object in the scene. The background is then represented as a color histogram, while the object is represented with a chain code corresponding to its moving pattern and its dominant hue color. The system provides a visual interface where to input graphical

queries such as drawn camera or object motion pattern, and background and object color. [Aner-Wolf and Kender, 2004] also use mosaic shot representation to index and cross-reference shots from sit-coms. Following this line of work, we propose to apply a mosaic based shot representation to presentation videos.

The literature suggests that the interaction of different information channels (slides text, white-board text, audio, video) is beneficial in terms of indexing and retrieval. Therefore, we propose to build semantic indexes based on text, graphics, speaker face and mosaics, which could be integrated into the unified framework such as the VASTMM browser [Haubold and Kender, 2007].

2.3 Automatic Text Detection and Recognition

Most of the existing video text detection and recognition systems are composed of 3 modules: a text regions detector, an enhancement module, and an OCR engine to perform recognition. The enhancement block is needed because directly applying an OCR engine to the color or grayscale text regions extracted from video frames usually yields not reliable results, because of the low quality and low resolution of such regions. A common solution in literature consists then in somehow enhancing and binarizing the text before feeding it to the OCR engine [Wang *et al.*, 2008; Jung *et al.*, 2004]. Most of the proposed methods focus on recognizing artificial text which is superimposed to the video in a post processing step[Lienhart and Wernicke, 2002; Lyu *et al.*, 2005]. The text of the slides we focus on belongs instead to the category of scene text, which is embedded in the scene and captured with the rest of the data. In this domain, [Chen *et al.*, 2002] propose an affine rectification of detected scene text regions in order to improve text recognition performance. [Chen and Yuille, 2004] introduced a method to detect and recognize text from signs embedded in natural scenes. They train a text region detector by combining features based on intensity mean and standard deviation, xy derivatives, histograms and edge linking into a cascaded classifier using Adaboost. The text detector is used in combination with an adaptive binarization algorithm and a commercial OCR system to automatically extract the text in the scene. Recently, [Wang *et al.*, 2011] have introduced a framework where both text detection and recognition are based on learned appearance based models of characters and words, thus assimilating the problem to a standard object recognition one.

In the specific domain of presentation videos, [Chong-wah Ngo and Huang, 2002; Wang *et al.*, 2003; Wang *et al.*, 2008] use a fixed camera and model the presenter as foreground, while the projected slide and the lecture hall are considered as background. Text regions are detected and slide changes are determined by finding local peaks of energy change in text regions and background regions in temporal windows. In [Wang *et al.*, 2003; Wang *et al.*, 2008; Wang *et al.*, 2007] the recognized text (in particular the title text) is used to perform a match and synchronize the projected slides and their electronic copies. Finally the electronic copies are superimposed to enhance the video, by registration, with the homography estimated from points in the matching text.

Binarization plays a major role in the enhancement module, and can be implemented both in color [Badekas *et al.*, 2007; Saidane and Garcia, 2008] and grayscale domain. Most of the methods applied to video focus on the grayscale channel for efficiency reasons. Recently, [Shafait *et al.*, 2008] have proposed a fast version of the Sauvola's binarization algorithm which, thanks to the use of integral images, has a computational time independent from the local window size. The Adaptive Local Otsu binarization we propose in Chapter 3.2 follows this direction.

For what concerns text recognition, Tesseract[1] [Smith, 2007] is the de-facto state of the art open source Optical Character Recognition Engine. Originally developed by HP from 1985 to 1996, in 2005 its code was released as open source and is currently developed by Google. The system takes a binary image as input, applies a connected component analysis, a least square fit of parallel lines and candidate character regions are found using A* search on the segmentation graph. Classification of characters is based on two classifiers: a static classifier trained on features based on polygonal approximation of character contours and a font-adaptive classifier trained on the most confident outputs of the static classifier. Linguistic analysis consists in finding the best match among frequent, dictionary, numeric, lower and upper case datasets and the choice of the classifier. We propose to use Tesseract as part of our slides text recognition pipeline.

2.4 Camera Motion and Mosaics

Image based approaches to build mosaics are generally divided into direct [Baker and Matthews, 2002; Irani *et al.*, 1996] and feature based [Brown and Lowe, 2003; Szeliski, 2006]. The latter gen-

[1] http://code.google.com/p/tesseract-ocr/

erally allow greater flexibility in terms of selecting the images (or frames) to be used to build them, and are not limited by temoporal constraints. Brown and Lowe [Brown and Lowe, 2003] for example created a system to perform fully automated panorama stitching. Starting from an unordered set of images, they extract multi-scale oriented patches (MOPS [Brown *et al.*, 20 25 June 2005]) or SIFT features (which ensure invariability to scale, rotation and affine distortions) and match them across images using a k-d tree structure. Once a subset of candidate image matches is found, RANSAC is used to estimate the homography between image pairs and a probabilistic scheme based on the number of inliers and outliers in the matching regions is employed to verify actual image matches and group images belonging to the same panorama. Then bundle adjustment is used to estimate the camera parameters and register the images together. A sum of squared projection errors over all keypoints in all images is optimized to solve for all of the camera parameters contemporarily, therefore avoiding the accumulated errors introduced by concatenations of pair wise homographies. Finally, two bands blending is used to render the panorama.

While highly accurate, this framework is computational quite expensive. In order to apply it to the video domain, some efficiency enhancements are required. To this end, [Steedly *et al.*, 2005] introduced a selection of keyframes and frames pairs to be matched, which assumes that temporally near frames have a consistent overlap. Regular frames are then matched only to the frames included in the range between closest keyframes. They also use a match compression algorithm, where spatially near matching features are averaged into a single one. Hence the number of points used in the final bundle adjustment is reduced.

[Bevilacqua and Azzari, 2007] devised a method to build video mosaics in real time. Instead of SIFT they employ the KLT tracker, supported by an initial phase correlation estimate, to match features across frames. Frame to mosaic homographies are computed besides the consecutive frame pairs ones, in order to reduce global registration errors. The technique employed allows to register all the n frames by estimating $O(n)$ homographies, rather then the $O(n^2)$ ones usually required to perform bundle adjustment. The authors then use a color intensity mapping function based on cumulative histograms to perform photometric registration.

We also propose to use efficient ways of extracting mosaics from video shots implementing some heuristic approximation of standard feature based algorithms, based on the limited changes of camera parameters available in this domain.

A critical but difficult to evaluate aspect in mosaic generation is the estimation of the quality of the mosaic. One option presented by [Boutellier *et al.*, 2008] consists in quantitatively evaluate the performance of image mosaics which were automatically generated from a ground truth image. The authors simulate the generation of input images for a stitching algorithm by applying a series of distortions (motion blur, vignetting, radial distortion and exposure time) to subwindows of the image. They then compare the mosaic generated by any stitching algorithm to the original image, using a method which takes into account the distortions visible from the human vision system. [Paalanen *et al.*, 2007] use instead a sequence of real images taken of a scene where markers are embedded to the end of determining the correct registration among images.

Instead of using synthetic or labor consuming quantitative methods to assess mosaics quality, in the future we plan to resort to user studies to verify the usefulness and likability of the mosaics we build as video shot representations and enhance by highlighting semantic content (e.g. slides, graphics).

2.5 Speaker Face Based Video Indexing

Previous work on automatic indexing of videos based on faces has mainly focused on controlled scenarios found in professional content[Everingham *et al.*, 2009; Krüger and Zhou, 2002; Li and Wang, 2008; Yang *et al.*, 2005], or surveillance footage[Feris *et al.*, 2007]. For example, [Yang *et al.*, 2005] employ a multiple instance learning framework to label faces in news videos. [Everingham *et al.*, 2009] combine aligned transcripts with visual features to automatically name characters in TV video. This approach follows in a significant line of work has been devoted to automatically detecting and recognizing characters in movies and TV series [Arandjelovic and Zisserman, 2005; Everingham *et al.*, 2006; Sivic *et al.*, 2009]. Those approaches rely on the structure and editing of the content to extract shots, within which faces are located and tracked. Each face in a track is then represented with local descriptors extracted on facial salient points located on the corner of the eyes, mouth and nose. Finally either clustering or matching techniques are employed to group, retrieve or recognize people in tracks.

Not much attention has been given instead to unstructured videos shot in uncontrolled environments. In this domain, [Haubold and Kender, 2007] have introduced a presentation videos browsing

interface where shots can be queried via a visual index of presenters faces. To the best of our knowlegde, that is the only work trying to assess the quality of speakers video indexes for unstructured presentation videos. They conducted user studies comparing head vs. head and shoulders person representation indexes. However, such indexes were created manually and not automatically.

The determination of the "best" faces for recognition purposes (especially in terms of pose) has been studied both from a psychological [Burke *et al.*, 2007] and computational point of view [Li *et al.*, 2007; Liu *et al.*, 2006]. Psychological studies have been conducted to analyze the viewpoint generalization ability of the human vision system, that is, the ability to recognize a face in different poses, given a single pose example as reference. The results suggest that the generalization ability is maximized when the reference pose is a three-quarters view of a face, not only for face recognition [Burke *et al.*, 2007; Longmore *et al.*, 2008], but also for face detection [Burton and Bindemann, 2009]. This confirms that humans are able to infer fuller 3D information about the head of a person when seeing a 3/4 view of their face. Liu et al. [Liu *et al.*, 2006] analyzed the same problem from the computer vision point of view and verified that a 32° pose provides the best generalization performance for face matching algorithms.

Some works build visual summaries of scenes based on "canonical views" extracted from photo collections[Simon *et al.*, 2007].

The selection of faces, based on quality measures, to be presented to human users has been studied in the context of surveillance video logs construction and biometrics [Del Bimbo *et al.*, 2009; Fourney and Laganiere, 2007; Nasrollahi and Moeslund, 2009; Nasrollahi and Moeslund, 2008]. A face log is a collection of time stamped image samples representing faces collected from a surveillance video. For example [Fourney and Laganiere, 2007] present a series of quality measures (based on pose, illumination, sharpness, resolution and skin) that allow them to reduce face tracks into face logs containing as low as 5% of the original tracks frames, while still retaining the full information about the people in the video.

[Chen *et al.*, 2007] also used similar quality measures (skin ratio, symmetry, etc.) to train a neural network to validate detected faces in surveillance sequences. Typically, those systems focus on discriminating between faces and not-faces, not *among* faces. The final ranking is given by the amount of "faceness" of each candidate, with little correlation between the individual quality measures and the actual quality of the selected face with respect to other faces in the track.

Object tracking, and face tracking in particular, is one of the most studied problems of computer vision. Many different methods have been proposed, from global template-based trackers, shape-based methods, probabilistic models using mean-shift or particle filtering to flow-based trackers [Everingham *et al.*, 2009].

In face detection, the Viola Jones [Viola and Jones, 2002] detector, which uses a cascaded classifier of boosted Haar-like features, is considered the baseline state of the art for non-commercial systems, giving reliable performances both on terms on accuracy and efficiency. It belongs to the family of frequency based face detection algorithms. Other methods make use of skin color filters, such as the one introduced by [Gomez and Morales, 2002]. Frequency and color based methods have also been used in combination [Ramanan *et al.*, 2007].

In the popular tracking-by-detection framework[Breitenstein *et al.*, 2010; Godec *et al.*, 2010], the tracker is built to fill the gaps between successive detections according to some optimization criterion. Particle filtering is commonly used to perform such task, which has been extended also to multiple objects [Duffner and Odobez, 2011].

Recently, general purpose appearance based trackers[Saffari *et al.*, 2010; Kalal *et al.*, 2010] have been gaining attention. They train a classifier on the appearance of the object and convert the tracking problem into a classification one, where the goal is to discriminate the target object from its surrounding background. Some of these methods train the model on the first instance of the object, and use it in subsequent frames. While this approach is relatively robust to occlusions, it does not perform well when an object undergoes appearance or viewpoint changes, since it is not adaptive.

Systems have been proposed to employ online adaptive learning of the appearance model of the object, updating the model in each frame using the current object region as a positive instance and the surrounding regions as negative examples. However, such systems often suffer from drift, which consists in a gradual adaptation of the tracker to non-object, background regions. [Babenko *et al.*, 2009] use a multiple instance learning framework to alleviate the problem. [Kim *et al.*, 2008] incorporate additional visual constraints (in terms of pose and alignment) in an appearance model. While such solutions produce better tracking performances, there is still room for improvement, especially for videos where the tracked object does not occupy a significantly large region of the frame.

One possible solution consists in combining template matching and generic object trackers. For

example [Santner *et al.*, 2010] propose the PROST framework to combine three different trackers: an adaptive appearance one that uses on-line random forests, a mean-shift one based on optical flow, and a correlation template matching one.

In the domain of face tracking in professional content such as movies and TV series, [Everingham *et al.*, 2009] use a combination of the Viola Jones face detector [Viola and Jones, 2002] with KLT tracker [Shi and Tomasi, 1994] to extend face tracks within shots. When two faces are detected in two different frames, KLT features falling inside the bounding boxes of the detected faces are tracked in both temporal directions and a face track is established based on how many of those features match. If the number of feature tracks which intersect the bounding boxes of both faces is greater than half of the the number of feature tracks which pass through either but not both faces, the face pair is considered to belong to the same face track.

2.6 Diagram/Graphics Spotting and Recognition

Diagram and graphics detection and recognition is an area of research of particular interest for the document analysis community. There are two typical approaches to find graphics in a document. The first consists in segmentation from the textual content, while the second method is called spotting, in which there is a reference symbol or graphical element and detection basically relies on matching, as recognition and localization are integrated in the same framework.

The closest existing works to our proposed slides graphics detection and matching method, are applied to logos and trademarks. [Wang and Chen, 2009] segment logos in documents by iteratively expanding seeds regions obtained after binarization. Such expansion is performed until the region fails to retain geometrical statistics (position, size, aspect ration) learned from a training set and used as prior knowledge. [Zhu and Doermann, 2007] trained a multiscale Fisher classifier based on geometrical and edge density features to detect logos image documents. [Patricio and Gómez-Allende, 2000] segment documents images by learning discriminative components of the magnitude of the Fourier transform of the grayscale histogram. They train a multilayer perceptron on top of such features and perform binary classification (text vs. graphics/images) on sliding windows over the document image. [Chowdhury *et al.*, 2003] propose to refine the segmentation of documents by taking into consideration an additional category to be distinguished from regular

text and graphics: mathematical formulas. They do so by exploiting the layout characteristics of formulas, which present a large amount of small font subscripts and superscripts, and generally are separated from the regular text by larger spacing. [Morris and Kender, 2009] introduce yet another segmentation category, as they use sort-merge selection and fusion of frequency based features to classify individual frames from presentation videos as displaying or not programming code in projected slides.

Once a diagram region has been localized, an appropriate descriptor must be employed to represent it. Many color, shape and texture descriptors have been proposed in the literature, a survey of which is beyond the scope of this thesis. In the following we will review those employed in the context of presentation diagrams classification and matching.

Diagram classification has been recently explored, although no consensus has been achieved on a unified taxonomy of diagram categories, nor any principled method has been investigated to generate them. In fact, most systems employ ad-hoc categories. For example, NPIC [Wang and Kan, 2006] proposes a web images taxonomy, which includes a Figure tree with different graphs subcategories. [Djordjevic and Ghani, 2010] extract graphical entries related to the corporate enterprise domain (Accenture) from MS Office documents (using some heuristics on size, distance between objects and arrows to groups graphical entities in a single unified graph) and define a taxonomy of graphical elements (Fact Box, Graph, Process flow, Table, Architecture Diagram, Logo, Photograph, Work Team, Organizational Chart, Plan). Features used to describe the graphical objects are: textual (both from slide text and OCR), **structural** (size of components, type of components, relative size and coordinates), visual (color layout, edge histogram, texture). [Prasad *et al.*, 2007] use standard local descriptors (HOG, SIFT and Distance Map histograms) to represent graphs, and adopt the pyramid match kernel to match them. This representation is used to train a classifier for five ad-hoc diagram categories: bar, lines, pie, scatter, surface. [Savva *et al.*, 2011] represent a graph using two complementary descriptors: the first is a histogram of normalized 6x6 patches extracxted uniformly in the image and associated with a codebook of size 200 according to the bag of visual words principle, the second is based on text regions detection, with a histogram containing the distribution of position, size and relative orientations of text regions over a uniform spatial grid in the image. A multiclass SVM is trained based on this representation to classify diagram images into ten categories: AreaGraph, BarGraph, LineGraph, Map, ParetoChart, PieChart, RadarPlot, Scatter-

Graph, Table.

We utilize texture and color descriptors to represent diagrams and match them across presentations. To the best of our knowledge, no other system has elevated graphics in slides to the semantic level, where to perform either presentation video indexing or matching across presentations.

Chapter 3

Text Indexing

Most of the methods proposed to summarize presentation videos with shots segmentations based on slides rely on the availability of electronic copies of the slides themselves, which is not always realistic. The solution we propose, on the other hand, works without any slide template to be compared, hence the definition ”unsourced”. We propose to distinguish between different slides based on their textual content (see Figure 3.1), by automatically recognizing the text in the video stream using an Optical Character Recognition (OCR) Engine. Once a semantic shot has been identified, that is, a shot containing text from a unique slide, we proceed to build a mosaic of it using a standard local feature based registration algorithm.

Directly applying an OCR engine to the text regions extracted from video yields not reliable results, because of the low quality and low resolution of such regions. A common solution consists

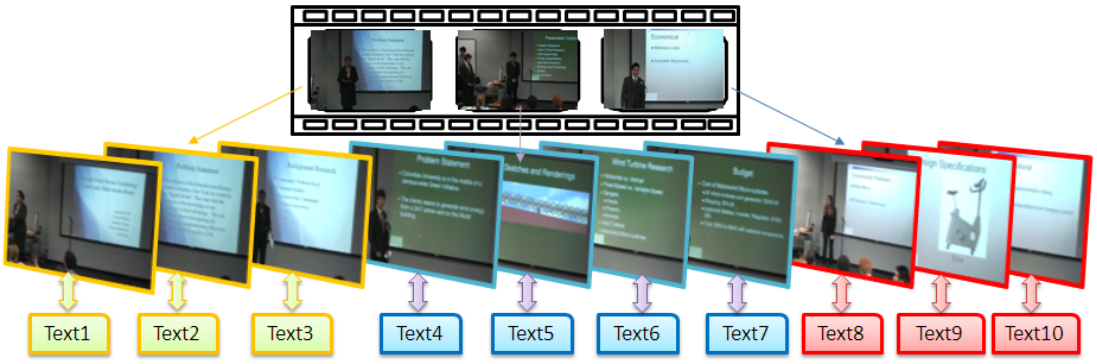


Figure 3.1: Semantic shot segmentation based on unique slides recognized text.

in somehow enhancing and binarizing the text before feeding it to the OCR engine [Wang *et al.*, 2008; Jung *et al.*, 2004].

We propose a new Local Adaptive version (LAO) of the Otsu binarization algorithm [Otsu, 1979], which is implemented with integral histograms. The method is robust to illumination changes and background variations within the text areas, while still efficient thanks to the use of integral histograms. In particular, it allows the determination of an optimal threshold that maximizes the between-classes variance within a subwindow, with computational complexity independent from the size of the window itself.

The presentation video text-based indexing pipeline we propos consists in four modules: candidate text regions detection, binarization, recognition and index construction. Figure 3.2 presents an example of the full pipeline applied to a frame. In the following we will describe each component in detail.

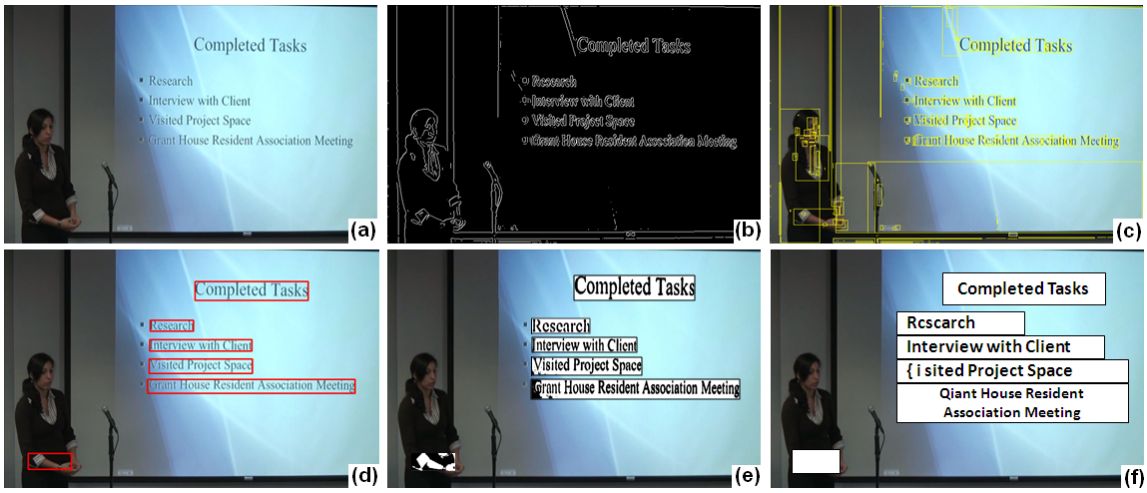


Figure 3.2: Text recognition pipeline. (a) Original frame. (b) LoG edge detection. (c) Edge connected components. (d) Results of region pruning based on geometric and edge density based constraints. (e) LAO binarization results. (f) Output of the Tesseract OCR engine. The final result, after text post-processing, is the following correctly recognized text: Completed Tasks, Research, Interview, Client, Project Space, House Resident, Association Meeting.

3.1 Text Regions Detection

Usually presentation slides do not present text overlaid to particularly challenging backgrounds, therefore we apply a simple and fast text detection approach. Initially a Laplacian of Gaussian operator is applied to a frame in order to extract edges. Subsequently connected components are located within the edge map and textural and geometrical properties of the regions enclosing such connected components are extracted. The inspected properties are: coordinates of the center, area, width, height, width/height ratio, density of edges, vertical and horizontal alignment. Empirically validated thresholds are applied to each feature in order to prune non-text regions. The candidate text regions R within a frame F must satisfy:

$$F_{Area}/1000 \leq R_{area} \leq F_{Area}/10 \tag{3.1}$$

$$2 \leq R_{width} \leq F_{width}/3 \tag{3.2}$$

$$6 \leq R_{height} \leq F_{height}/5 \tag{3.3}$$

$$E_{density} \geq 0.2 \tag{3.4}$$

Then, partially overlapping and edge map similar regions R^i and R^j are merged if:

$$0.5 < \frac{R^i_{height}}{R^j_{height}} < 1.5 \tag{3.5}$$

$$0.5 < \frac{R^i_{EdgeDensity}}{R^j_{EdgeDensity}} < 1.5 \tag{3.6}$$

$$HO\left(R^i, R^j\right) \geq min\left(R^i_{width}, R^j_{width}\right) + 10 \tag{3.7}$$

$$VO\left(R^i, R^j\right) \geq min\left(R^i_{height}, R^j_{height}\right)/2 \tag{3.8}$$

where HO and VO represent the horizontal and vertical overlap, respectively.

The candidate text regions are then passed to the recognition block to be finally confirmed, in the case one or more characters are recognized, or discarded when no character is recognized.

3.2 Local Adaptive Otsu (LAO) Binarization Algorithm

Binarization techniques are usually split into two categories: global and local. The Otsu [Otsu, 1979] global thresholding method assumes a bimodal distribution within the gray scale histogram

of an image, and aims at automatically selecting an optimal threshold T to minimize the within-class variance of the two modes, or equivalently to maximize their between-class variance. Given an image with pixel values ranging in an interval of intensity levels $[0, L-1]$, the optimal T is computed as the one maximizing the between-class variance $\sigma^2_{between}(T)$, computed as

$$\sigma^2_{between}(T) = n_B(T)\, n_F(T)\, (\mu_B(T) - \mu_F(T))^2 \tag{3.9}$$

where $\mu_B(T)$ and $\mu_B(T)$ are the means of the background (below the threshold T) and foreground (above T) pixels clusters, while $n_B(T)$ and $n_F(T)$ represent the number of pixels belonging to each cluster.

Despite being parameter free, the classical Otsu method presents a main limitation in its globality. Computing an optimal threshold for the whole image makes it sensitive to shading and local noise, as shown in Figure 3.4. In order to overcome such limitation, local methods have been introduced. Those methods work by sliding a $W \times W$ window and select a threshold for the pixel where it is centered based on the statistics of its neighbors. One of the most popular among such algorithms is Sauvola's, in which the threshold $t(x, y)$ is computed as

$$t(x, y) = \mu(x, y, W) \left[1 + k \left(\frac{\sigma(x, y, W)}{R} - 1\right)\right] \tag{3.10}$$

where R is the maximum value of the standard deviation within the window, and k is a parameter which takes positive values in the range $[0.2, 0.5]$. Despite the improvement offered with respect to global solutions, this algorithm is limited by the dependence of $t(x, y)$ from two parameters: the window size W, which also determines efficiency, and the value k. The computational complexity has been recently made independent from W thanks to the introduction of integral images [Shafait *et al.*, 2008]. However, the method is still chained to an ad hoc selection of k. Hence, the choice of $t(x, y)$ is not related to any optimization process, and remains quite arbitrary.

We propose to eliminate the dependency of $t(x, y)$ from k by computing it as the threshold that optimizes the between-class variance within the window. In other words, our solution consists in a localized version of the Otsu algorithm, or an optimal version of Sauvola's one, which can combine the strengths of the two methods: locality and optimality. The Local Adaptive Otsu algorithm (LAO) slides a window of size W across the image and computes the threshold with the optimal Otsu criterion in each position. At each position one can choose whether to apply the threshold directly to the whole window (used in the rest of this manuscript) or simply to the pixel at which the

window is centered (as in Sauvola's approach). In order to limit the computational complexity derived from the application of the window, we use the integral histogram [Porikli, 2005], a structure consisting in L integral images, one per bin. Once paid the initial cost of building the integral histogram for the whole image, such structure allows to compute the values of $n_B(T, W)$, $n_F(T, W)$, $\mu_B(T, W)$ and $\mu_F(T, W)$ for Equation 3.9 in any subwindow with a constant number of operations, independently from the window size. In Figure 3.4 is reported the experimental gain in time achieved by introducing the integral histogram over a baseline implementation of the algorithm.

3.3 Text Recognition

Word recognition is implemented by the Tesseract [Smith, 2007] OCR engine. We trained Tesseract with 15 character sets, using the most commmon fonts for PowerPoint presentations [Mackiewicz, August 2007], with a text reflecting the frequencies of English letters[1]. Each font was represented in its regular, italic and bold version during training, with characters of height equal to 30pt. A post-processing method is applied to the output of the OCR engine to discard text containing non-alphanumeric symbols. Porter stemming [Porter, 1980] is also applied to each word, which is then passed through a list of English stop words[2].

3.4 Index Construction

Once the text for a frame has been recognized, it is stored to be compared to the text extracted from neighboring frames for indexing. The comparison is performed according to the edit distance between the strings of extracted text, normalized by the length of the strings themselves.

$$d(s1, s2) = \frac{ED(s1, s2)}{|s1|\,|s2|} \tag{3.11}$$

If such distance is lower than a predefined threshold τ, the frames are considered as belonging to the same slide and grouped together. The longest string and the frame from which it was extracted is kept as reference for the slide. Retrieval of a certain concept can be performed, as exemplified

[1] http://en.wikipedia.org/wiki/Letter_frequencies

[2] http://www.textfixer.com/resources/common-english-words.txt

in Figure 3.6, by looking for a query word among the strings chosen to be representative of their slides.

3.5 Camera Motion Estimation and Video Mosaic Construction

Given a semantic shot, we have implemented a simple video mosaicking algorithm.

The first step in the construction of the video shots mosaics consists in a pan, tilt and zoom camera movement estimator. We evenly split each frame into 45 regions, 9 horizontally and 5 vertically, compute the average grayscale value of each region and perform a least square estimation of the camera parameters based on the luminance constancy assumption, as proposed by Kender et al. [Kender, 2000]. The preprocessing step is quite fast, yet accurate enough to disambiguate static shots (i.e., shots without movement) from shots that can be actively used to build a mosaic. Temporal sampling proportional to the amount of camera movement is employed to select a group of keyframes, which will be the input to the mosaicking algorithm. In the future, we also plan to use quality measures to select the best keyframes to build the mosaic (one possibility could be computing the amount of blur as in [Boutellier and Silvn, 2006] to retain only the sharpest frames). The temporally central frame of each sequence of keyframes is used to provide the reference coordinate system.

We build mosaics following the feature based approach of Brown and Lowe [Brown and Lowe, 2003], but in a simpler framework. We extract SIFT features, match them and estimate homography transforms between keyframes using RANSAC. Our code is built on top of the SIFT and RANSAC implementation of Robin Hess[3]. Finally, blending is simply performed by keeping the median value among the corresponding pixels of keyframes overlapping at a given location in the mosaic. The various steps of the mosics construction are presented in Figure 3.3.

[3] http://web.engr.oregonstate.edu/∼hess/index.html

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

Figure 3.3: Mosaic example. (a) SIFT features extracted from two frames in the set. (b) Local features matching. (c) Remaining matches after RANSAC based homography constraints enforcement. (d) Final mosaic obtained by registration of the set of selected keyframes.

3.6 Experiments

We analysed videos containing 8 student presentations, for a total of 1 hour and 45 minutes of video. Each presentation has on average 13 slides, for a total of 2276 words and 13804 characters. In this Section we present the performances of our system in terms of localization of text regions, binarization quality and semantic concepts recognition (in particular, the text extracted from the slides). All the experiments were carried on a Pentium 4 2.33 GHz machine.

3.6.1 Text Detection

Text localization performances were tested on a set of 500 frames randomly selected (100 of which did not contain text). For each frame, Precision and Recall are defined as the intersection between the ground truth text area TA_{GT} and the text area estimated by the system TA_E, divided respectively by TA_E and TA_{GT}.

$$Precision = \frac{TA_{GT} \cap TA_E}{TA_E}, \quad Recall = \frac{TA_{GT} \cap TA_E}{TA_{GT}} \tag{3.12}$$

Table 3.1 presents how the average Precision and Recall rates obtained by the system change with the application of the character and word recognition step. The $simple$ performances refer to the regions found by the simple text detector and successively passed to the word recognition block. The $refined$ precision and recall rates are calculated after the recognition block has either rejected or confirmed such regions as text (containing at least one recognized character). Following intuition, at the $refined$ stage Precision increases and Recall diminishes, as some candidate text regions (including some relevant ones) are rejected by the recognition step.

$Precision_{simple}$	$Recall_{simple}$	$Precision_{refined}$	$Recall_{refined}$
0.71213	0.85914	0.88584	0.68046

Table 3.1: Text Precision and Recall localization rates.

3.6.2 Binarization

We now present a quantitative comparison of the three binarization algorithms mentioned in Section 3.2. The evaluation was performed on a subset of 54 regions localized by the text detection block

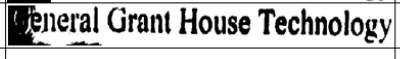
Algorithm	Precision	Recall	F1	t(sec)		
Otsu	0.8611	0.8555	0.8583	0.539		General Grant House Technology
Sauvola (k = 0.5)	0.9003	0.8759	0.8879	0.626		Nm} Grant House Technology
LAO	0.8831	0.9278	0.9049	2.126		[yoga! Grant House Technology
LAO + Int. Hist.	**0.8831**	**0.9278**	**0.9049**	**1.29**		[(cncra1l Grant House Technology
						[(cncra1l Grant House Technology

Figure 3.4: Binarization performance comparison. Example of the compared binarization methods. Original image (a) and its versions binarized with (b) original Otsu, (c) Sauvola and (d) adaptive Otsu. Under each image is reported the text recognized by the system. In this case adaptive Otsu outperformes the other methods in dealing with the shaded area around the word *General*. In fact, it manages to identify 4 of its characters, against none of the original Otsu and 1 of Sauvola.

N_{gtc}	N_{corc}	$Prec_c$	Rec_c	TCED	N_{gtw}	N_{corw}	$Prec_w$	Rec_w
13804	7376	0.5343	0.7446	6428	2276	1126	0.4947	0.6651

Table 3.2: Character and Word Recognition rates. Number of ground truth (N_{gtc}) and correctly recognized characters (N_{corc}) characters. Total Characters Edit Distance (TCED). Number of ground truth (N_{gtw}) and correctly recognized (N_{corw}) words. $Prec_c$, Rec_c, $Prec_w$, Rec_w refer respectively to Precision and Recall measures at character and word level.

and manually segmented with the aid of a heuristic visualization tool, in order to generate ground truth, for a total of over 2 million pixels. Precision and Recall metrics are defined similarly to what was described earlier, by re-defining TA_{GT} to be the set of ground truth pixels representing a character (foreground of the region) and TA_E as the set of pixels labeled as foreground by the algorithm. F1 measures the combination of precision and recall. From the results in Figure 3.4, we notice that the performances of the different algorithms are comparable. The original Otsu algorithm is the fastest but has the lowest Recall and Precision, because it looks at every region globally and works well only up to a limited detail. The others provide higher precision, thanks to their focus on locality. It must be noted that our algorithm's results are the best in terms of F1, and are comparable to the precision of Sauvola's method, which is known to be one of the best performing binarizations methods, but without the need for a predetermined threshold. We show two versions of the Local Adaptive Otsu algorithm (LAO): one with and one without the use of the integral histogram. The $Time$ results demonstrate the utility of integral histogram, which allows us

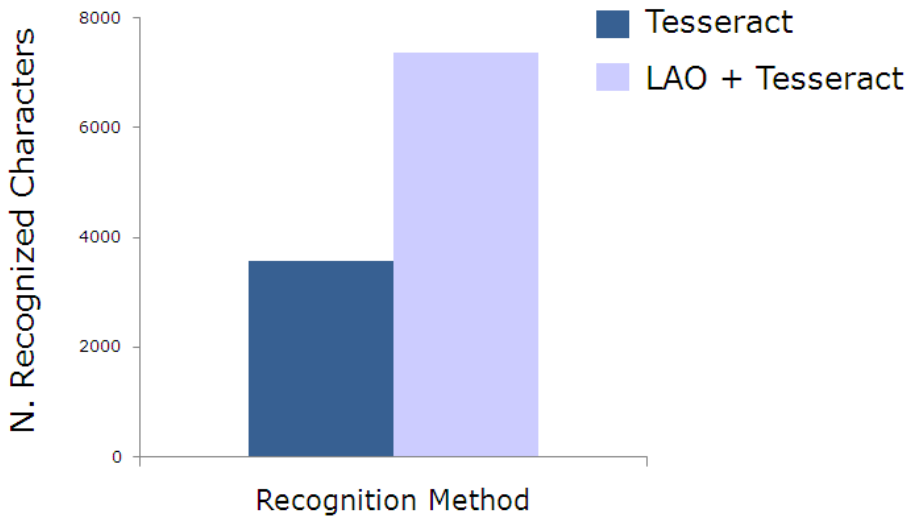


Figure 3.5: Character recognition comparison between Tesseract alone and Tesseract after the application of the Local Adaptive Otsu (LAO) binarizatiion.

to compute the optimal thresholds in a time which is independent from the window size. For all the local adaptive algorithms used in the experiments, we used a window of size 25x25.

3.6.3 Text Recognition

As explained in Section 3.3 the Tesseract OCR engine was used as a tool to recognize characters and words. We analyzed the performance of the system both at the word and character level. We defined the following metrics:

$$Precision_c = \frac{N_{corc}}{N_{gtc}}, \quad Recall_c = \frac{N_{corc}}{N_{rc}} \tag{3.13}$$

with $N_{corc} = N_{gtc} - ED\,(s_g, s_r)$, N_{corc} is the number of correctly recognized characters, that is, the number N_{gtc} of ground truth characters minus the edit distance $ED\,(s_g, s_r)$ between the ground truth text s_g and the text output from the system s_r. N_{rc} is the number of recognized characters. Substituting the subscript c with w we obtain the same type of statistics at the word level, instead of character level ($Precision_w$ and $Recall_w$). N_{corw} is simply defined as the number of ground truth words correctly recognized by the system.

We compared the character and word recognition rates with and without our binarization preprocessing step. From Figure 3.5 can be appreciated how our binarization process allows to basically

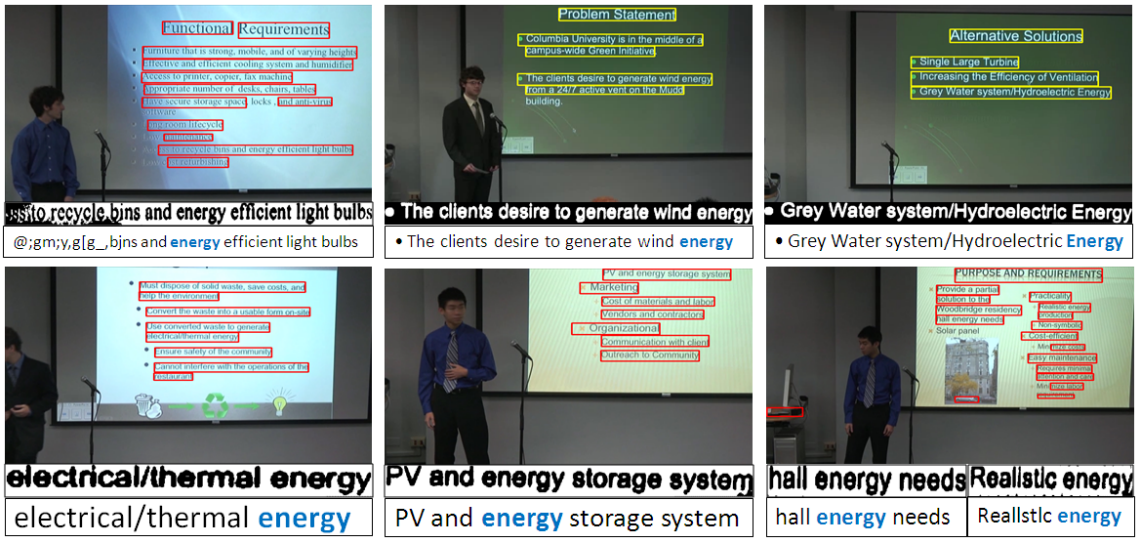


Figure 3.6: Image text recognition. The word ***Energy*** is localized in different slides across 4 different presentations(top left, top center and right, bottom left, bottom center and right) and also within the same (top center and right, bottom center and right). In each frame are highlighted the localized text regions. Under every image the binarized version of the text region containing the word ?Energy? (correctly recognized by the system) is presented.

double the number of characters correctly recognized by Tesseract, which internally uses by default the Otsu algorithm instead.

Finally, Table 3.2 reports the best performances obtained with the parameters set to the values reported in the previous Sections. It is interesting to notice that the word recognition rates are lower than their character equivalents. In fact, even if all its characters but one are correctly matched, a word is considered wrongly recognized. This suggests the use of ranking measures, such as the edit distance, which take into account also partial word matches in order to improve the quality of indexing and retrieval of semantic segments extracted from such videos. A system would then be more robust to single or limited character recognition errors.

Chapter 4

Face Indexing

Face detection and recognition in digital multimedia collections has been an active area of research for the past three decades. Recently the use of faces to index videos has seen a growth in interest in the multimedia community. In fact, various systems have been proposed to automatically index professional videos such as movies, TV shows and news based on characters appearing in them [Arandjelovic and Zisserman, 2005; Everingham *et al.*, 2006; Everingham *et al.*, 2009; Yang *et al.*, 2005; Sivic *et al.*, 2009]. However, one aspect which has not been fully explored so far by the multimedia community is the quality of the generated indexes in terms of end users experience.

The exponential diffusion of unstructured multimedia content on sites such as Youtube[1] and Flikr[2] has lately fostered a rapid growth of interest in the multimedia and vision community toward a new line of systems to recognize people and activities "in the wild", that is, in less structured, unconstrained and more realistic domains. Due to the lack of structure and to the low quality of the data, algorithms and paradigms designed for professional content often cannot directly be applied to the aforementioned domains, thus presenting a new challenge.

Our work lies at the convergence of these two trends, as it aims at indexing unstructured presentation videos based on speaker appearances, and uses quality measures to select representative face images. We propose a system to select the best faces in unstructured presentation videos with respect to two criteria: the first is to optimize matching accuracy between pairs of face tracks, the

[1] www.youtube.com

[2] www.flickr.com

second one is for indexing purposes.

Our system extracts candidate faces from where to start tracking (face tracks "seeds"), with a combination of the Viola Jones face detector [Viola and Jones, 2002] and pixel based skin color filtering. It then integrates an online multiple instance learning tracker [Babenko *et al.*, 2009] with Viola Jones face detections within a simplified steady state Kalman filter framework, in order to mitigate the drifting effect which typically affects appearance based tracking algorithms.

Subsequently, the system selects samples within tracks based on quality measures related to ***resolution, skin color area and pose***, and uses them both for matching and indexing. Finally, it selects one head and shoulder image per track to be its representative index. The last step was suggested by a user study which partially confirmed the results of a consistent line of psychological literature, which asserts that the human vision system prefers to see a 3/4 view of a face in a visual index, because it helps to better generalize to the other possible poses of the head [Burke *et al.*, 2007; Burton and Bindemann, 2009; Longmore *et al.*, 2008]. Providing a head and shoulder view introduces useful contextual information.

To the best of our knowledge, the only work trying to assess the quality of speakers video indexes for unstructured presentation videos is the one by [Haubold and Kender, 2007], who have conducted user studies comparing head vs. head and shoulders person representation indexes. However, such indexes were created manually and not automatically.

4.1 Human Preference Assessment for Visual Face Indexes

The final goal of this processing unit of our system is to generate a visual index of speakers that satisfies human users. To that end, we conducted Amazon Mechanical Turk[3] experiment to evaluate the preferences of people in terms of how the face of a speaker should be presented in the visual index, with respect to two criteria: head pose and context. The two criteria were selected based on previous results in related contexts in psychology [Burke *et al.*, 2007; Burton and Bindemann, 2009; Longmore *et al.*, 2008] and multimedia [Haubold and Kender, 2007], which suggested a human preference for a head and shoulders, 3/4 view of a face.

The experimental setup was the following. Each Human "Intelligence Task" (HIT) presented

[3]https://www.mturk.com

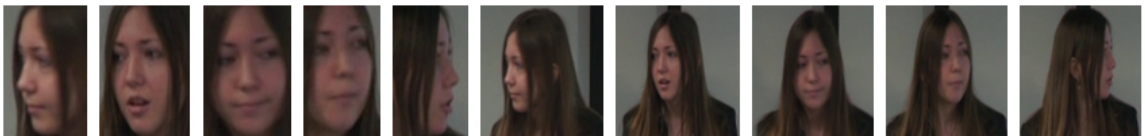


Figure 4.1: Face Quality Selection views example: 5 poses, from left profile to right profile, and two view types, face only or head and shoulders.

the user with two type of face views of a speaker from our presentation videos: one type showing only the face, the other with a head and shoulder view. For each type, 5 poses at equal intervals from -90 to +90 degrees were shown. An example of the choices is shown in Figure 4.1

In order to avoid preferences based on a specific speaker or a specific ordering in which the views were shown, we conducted experiments with 15 different speakers and 3 random views orderings. Furthermore, we requested 35 different workers to complete each combination of speaker and views order, therefore amounting a total of 1575 unique HITs.

All HITs were completed within two and a half hours, with an average time per assignment of 19 seconds. 69 unique workers participated to the experiment. Out of the possible 1575 votes, 11 were invalid, leaving a total of 1564 valid entries. The results, reported in Figure 4.2 showed a strong preference for a head and shoulder rather than face only view (76% versus 24%). The single most selected pose was the frontal one (45%). However, the combination of the left and right 3/4 poses amounted to 47% of the votes. The detail of the HIT setup, as well as the distributions of votes per speaker, motivations, etc. are reported in Appendix B.1.

These results, together with the evidence from the literature, motivated us to choose a head and shoulder, 3/4 pose as the canonical view for the speaker face index generated by our system.

4.2 Face Tracks Generation

4.2.1 Face Detection

Face tracks are sequences of consecutive frames in which a face is tracked. Since the videos we investigate are unedited and unstructured, we cannot rely on standard shot detection algorithms to segment the video into shots. We therefore implemented a simple loose shot boundaries detector, which works by splitting each frame into 9x5 regions, and then thresholding mean gray scale differ-

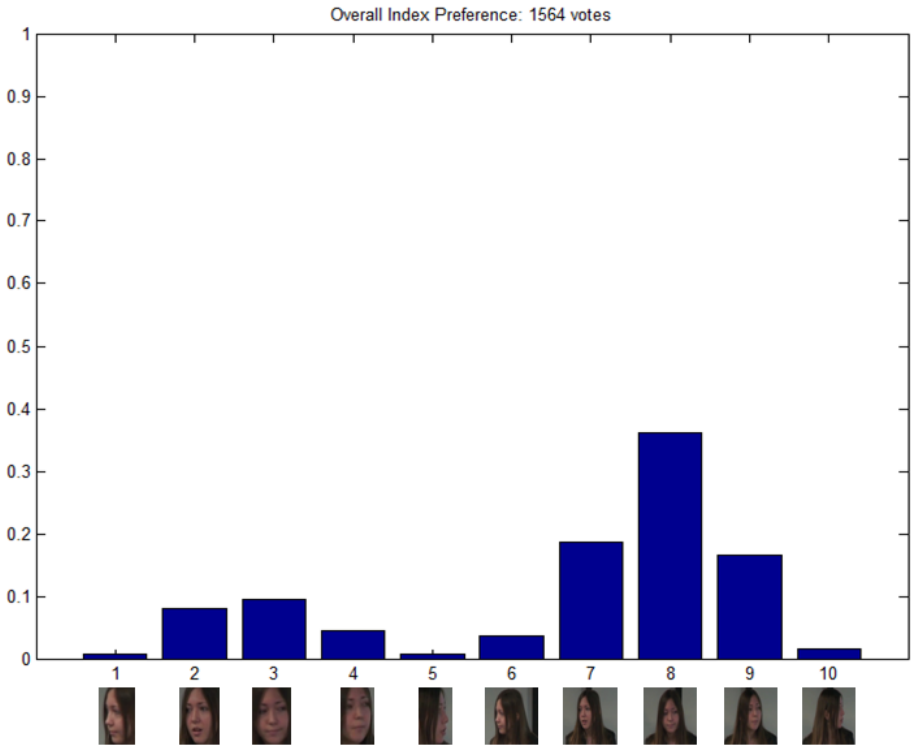


Figure 4.2: Face Quality Selection overall results.

ences between corresponding regions in frames separated by a step of 3 frames. While simple, this loose shot boundaries detection algorithm is 98% accurate and prevents inconvenient behaviors of the tracker.

In order to find "seed" faces (where to initialize the tracker), we use the Viola Jones face detector. To alleviate the significant amount of false detections originating from the noisy videos, we applied the skin color filter introduced by [Gomez and Morales, 2002] to each pixel in the candidate face regions. The resulting skin model in the RGB colorspace is the following:

$$Pixel = skin \iff \begin{cases} R/G > 1.185 & and \\ \frac{R*B}{(R+G+B)^2} > 0.107 & and \\ \frac{R*G}{(R+G+B)^2} > 0.112 \end{cases} \tag{4.1}$$

We then empirically evaluated that restricting a face track seed to require that more than 20% of the pixels in a candidate region had skin tone, resulted in doubling face track detection precision performances with respect to the default Viola Jones face detector, while maintaining the same level

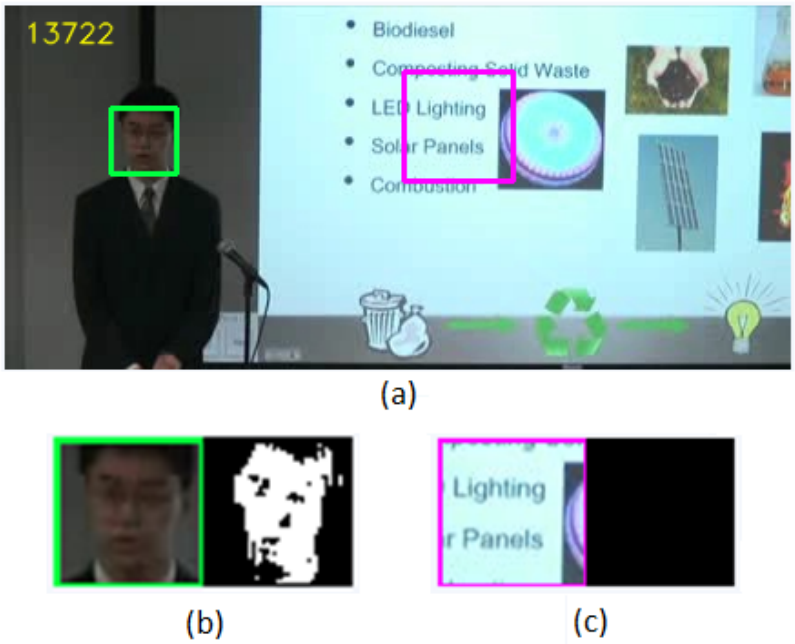


Figure 4.3: Example of benefit of the skin filter on top of the Viola Jones face detector.

of recall (see details in Figure 4.6). We refer to this skin area filter as $skinRatio$. Increasing the level of precision is of fundamental importance in this stage, since it influences the number of instances of tracker that will be generated and consequently the number of tracks that will be further processed for matching and indexing purposes. Figure 4.3 illustrates an example of the benefit of using the filter over relying on the Viola Jones detector by itself.

4.2.2 Steady-State Kalman Filter Tracking Approach

Once the seed has been established for a track, we track the face in both temporal directions, until the track exits the frame borders or one of the detected shot boundaries is encountered.

We propose a system that integrates an online multiple instance learning tracker as a noisy prediction in a simplified version of a Kalman filter, named $K - Filter$, which uses Viola Jones detections as noisy observations. $K - Filter$ is used in order to mitigate the drifting effect, which typically affects appearance based tracking algorithms.

We consider a framework in which each face is tracked individually and independently from the others. While we are aware of works which solve multiple object tracking in a single optimization

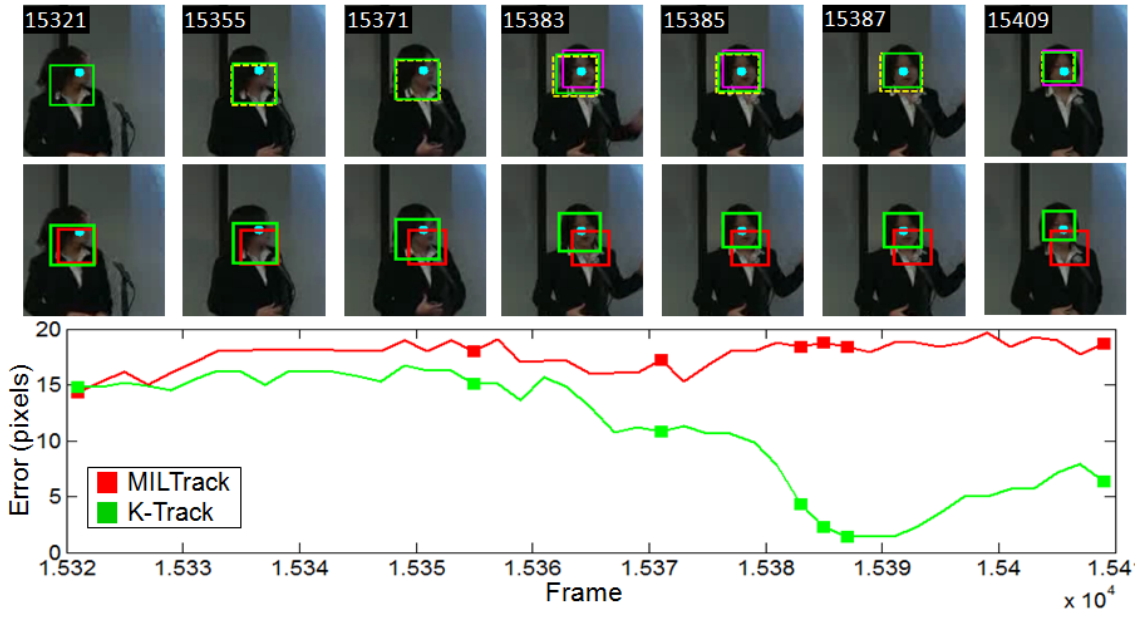


Figure 4.4: Example of the benefit of the proposed Kalman filter framework in dealing with the drifting effect. The first row shows the position of the prediction $\tilde{\boldsymbol{x}}_t$ (dashed yellow), the observation $\boldsymbol{x}_t^O$ (magenta) and the final output $\hat{\boldsymbol{x}}_t$ (green) of *K-Track* for the frames in the sequence. The second row highlights the difference in behavior between the MILTrack tracker (red) and *K-Track*. The noisy observation $\boldsymbol{x}_t^O$ allows *K-Track* not to keep on drifting and therefore reduce the error (bottom graph, L2 distance between centers of the system region and the ground truth face(cyan dot)).

framework such as [Breitenstein *et al.*, 2010], the nature of the videos we investigate (mainly presentation videos showing one single person at a time) suggest no need for multi-target optimization frameworks.

The algorithm we propose is not a traditional track-by-detection one. We correct the drift effect of a general purpose tracker with class-specific detection, in order to perform long sustained tracking. Our detector corrects the tracker; in standard track-by-detection the tracker corrects (the gaps of) the detector. Therefore the prospective of the optimization framework is reversed with respect to traditional tracking-by-detection systems.

Figure 4.4 shows an example sequence from video highlighting the benefit of using the proposed *K-Track* framework to augment the performances of a state of the art appearance based tracker, and alleviate drift. In the example, as in most of the remaining of the paper, the MILTrack system

proposed by [Babenko *et al.*, 2009] is used as the predictor and Viola Jones face detections are regarded as observations in the *K-Track* framework. However, we stress that our framework is general and can use any generic object tracker or detector. Both *K-Track* (second row, in green) and MILTrack (in red) are initially drifted away from the true face center (in cyan). However, the observations of the Viola Jones detector help *K-Track* to quickly recover to the proper position, thus reducing the distance from the face center within a few frames (green curve in bottom graph). On the other hand MILTrack alone, not relying on such additional information, is unable to recover and its error remains approximately constant (red curve).

It seems intuitive that enhancing a general purpose tracker with a class specific detector should improve tracking performances. However, it is not clear how to achieve such a combination. In this Section we present our solution in the form of a steady state Kalman filter framework, which provides a principled and (under certain assumptions) optimal solution to the combination problem.

The full mathematical derivation of the filter is described in detail in Appendix A.

We regard the general object tracking problem in the framework of a Kalman filter which has reached the equilibrium stage of its prediction/observation recurrence, following the framework proposed by [Friedland, 1973] and revised by [Ramachandra, 2000]. The approximation of a Kalman filter to steady-state form has been successfully adopted in various domains, including recent works in neural interfaces [Malik *et al.*, 2011] and channel estimation [Liyanage and Sasase, 2009], providing more efficient yet accurate estimations. In the object tracking domain we explore, the convergence patterns of the Kalman gain for five *Standard* sequences (see details in Figure 4.7) show a quick convergence rate to a constant value for both position and velocity gains on both vertical and horizontal directions, thus suggesting that substituting the regular Kalman filter with its steady-state counterpart does not significantly alter tracking performances, while providing an easier and more efficient working framework.

A generic appearance-based tracker represents the prediction of the process, and a generic object detector represents the measurement. We will now see how these assumptions affect the Kalman filter equations.

The model proposed by [Friedland, 1973] assumes that the object moves freely in space with a constant velocity which is perturbed by a zero mean random acceleration. The position of the object is assumed to be measured by a detector at uniform sampling intervals of time, and all measurements

are noisy. The problem is to obtain the optimum estimates of position and velocity of the object at each interval. Each component of the object position is considered to be independently measured by the detector in the Cartesian coordinate system with constant accuracy, and the observation errors have zero mean and are uncorrelated.

We associate the *a priori* estimate $\tilde{\boldsymbol{x}}_t$ with the object tracker. This intuitively makes sense, since the tracker represents a prediction based on the past state of the object. In this framework, the *a priori* estimator tries to predict at each step t the position of the tracked object and also its velocity.

We associate the measurement $\boldsymbol{x}_t^O$ of the filter with an object detector. This intuitively also makes sense, since the detector, just like a measurement, is based solely on an observation at the current step, and does not have any relationship with past states. In this case $m = n$.

In the described framework, $\boldsymbol{x} = (x, \dot{x}, y, \dot{y})^T \in \mathbb{R}^4$, where $\dot{x}$ and $\dot{y}$ represent the velocity of the object in the x and y direction, respectively. We consider the x and y coordinates to be independent, therefore the following analysis will be done on a single dimension, with $\boldsymbol{x} = (x, \dot{x})^T \in \mathbb{R}^2$. The accuracy of position and velocity estimates at each moment t depends not only upon the sensor accuracy, but also upon the perturbing acceleration a, which is a random constant between successive observations. This random constant is assumed to have zero mean and to be uncorrelated with the acceleration at other intervals, therefore the only relevant statistic that we need to estimate about it is its constant variance σ_a^2. The motion of the tracked object in a time interval T is then described by the following Equations

$$x_t = x_{t-1} + \dot{x}_{t-1}T + 0.5a_{t-1}T^2 \tag{4.2}$$

$$\dot{x}_t = \dot{x}_{t-1} + a_{t-1}T \tag{4.3}$$

since we consider unit time intervals (T=1), we can write

$$\boldsymbol{x}_t = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} a_{t-1} = A\boldsymbol{x}_{t-1} + Ga_{t-1} \tag{4.4}$$

Equation 4.4 represent the mapping of a standard Kalman Equation to this specific framework. The measurement noise $\boldsymbol{w} = Ga$ is assumed to be white and have normal probability distribution

$$p(\boldsymbol{w}) \sim N(0, Q) = N\left(0, GG^T\sigma_a^2\right) \tag{4.5}$$

The measurement $\boldsymbol{x}_t^O$ of the state of a single coordinate is represented as

$$x_t^O = H\boldsymbol{x}_t + v_t \tag{4.6}$$

where $H = [1, 0]$, since at each observation only the position of the tracked object is measured, not its velocity. Hence $m = 1$ and $x_t^O \in \mathbb{R}^1$. The random variable v represents measurement noise. Like $\boldsymbol{w}$, v is assumed to be white, independent from $\boldsymbol{w}$, and to have normal probability distribution $p(v) \sim N(0, R)$. $R = \sigma_o^2$ represents the variance of the observation error, and is a scalar. The *a priori* and *a posteriori* covariance matrices $\tilde{P}_t$ and $\hat{P}_t$ are then [2x2] symmetric matrices, and in the steady state $\tilde{P}_t = \tilde{P}$ and $\hat{P}_t = \hat{P}$. The Kalman filter time and measurement update Equations become

$$\tilde{\boldsymbol{x}}_t = A\hat{\boldsymbol{x}}_{t-1} = MILTrack(\hat{\boldsymbol{x}}_{t-1}) \tag{4.7}$$

$$\tilde{P} = A\hat{P}A^T + G\sigma_a^2 G^T \tag{4.8}$$

and

$$K = \tilde{P}H^T \left(H\tilde{P}H^T + R\right)^{-1} \tag{4.9}$$

$$\hat{\boldsymbol{x}}_t = \tilde{\boldsymbol{x}}_t + K\left(x_t^O - H\tilde{\boldsymbol{x}}_t\right) \tag{4.10}$$

$$\hat{P} = (I - KH)\,\tilde{P} \tag{4.11}$$

where now $K = (K_1, K_2)^T$ has dimension [2x1]. Note that in our model according to Equation 4.7 , we are assuming that the MIL tracker is simply predicting following a linear model in position and velocity. This approximation is shown in the following Equation

$$\tilde{\boldsymbol{x}}_t = \begin{bmatrix} \tilde{x}_t \\ \tilde{x}_t - \hat{x}_{t-1} \end{bmatrix} = MILTrack\left(\hat{\boldsymbol{x}}_{t-1}\right) \approx A\hat{\boldsymbol{x}}_{t-1} \tag{4.12}$$

Combining Equations 4.8 and 4.11 we can solve and obtain the following notations of $\hat{P}$, $\tilde{P}$ and K (details are provided in Appendix A)

$$\tilde{P} = \begin{bmatrix} \frac{\sigma_o^2 d(d+1)^2}{r^2} & \frac{\sigma_o\sigma_a(d+1)^2}{2r} \\ \frac{\sigma_o\sigma_a(d+1)^2}{2r} & \frac{\sigma_a^2(d+1)}{2} \end{bmatrix}, \quad \hat{P} = \begin{bmatrix} \frac{\sigma_o^2 d(d-1)^2}{r^2} & \frac{\sigma_o\sigma_a(d-1)^2}{2r} \\ \frac{\sigma_o\sigma_a(d-1)^2}{2r} & \frac{\sigma_a^2(d-1)}{2} \end{bmatrix}$$

$$K = \begin{bmatrix} \frac{d(d-1)^2}{r^2} \\ \frac{2(d-1)^2}{r^2} \end{bmatrix} \tag{4.13}$$

with $r = \frac{4\sigma_o}{\sigma_a}$ and $d = \sqrt{1+2r}$. We have now reached a closed form solution for K in terms of the constant error variances σ_a, on the prediction of the object's acceleration, and σ_o, on the position measured by the detector. We can therefore predict the position and velocity of the object at time t with Equation 4.10, which we rewrite separately for position and velocity (along one dimension):

$$\hat{x}_t = K_1 x_t^O + (1 - K_1)\tilde{x}_t \tag{4.14}$$

$$\hat{\dot{x}}_t = \tilde{\dot{x}}_t + K_2(x_t^O - \tilde{x}_t) \tag{4.15}$$

where K_1 and K_2 are the elements of the matrix K, representing the filter gain with respect to position and velocity. Note that while K_1 is a scalar, the unit of K_2 is one over time.Therefore the unit metrics are preserved in Equation 4.15, since $K_2(x_t^O - \tilde{x}_t)$ is a velocity.

The Kalman filter framework provides a disciplined way of combining prediction and measurement for our problem, and in a way that is in fact optimal under fairly general assumptions.

We applied a class specific framework to initialize and bound in time the object tracker. However, we stress that the proposed *K-Track* framework is not restricted to work solely on faces, but can operate on *any* object as long as there exists a detector to build upon, as confirmed by the results on the standard *Liquor* sequence in Chapter 4.4.2.

Once the seed has been established for a face track, we start tracking the face in both temporal directions, until the track exits the frame borders or one of the detected shots boundaries is encountered, as explained in Algorithm 1.

The base of our tracking method is the online multiple instance learning tracker (MILTrack) recently introduced by [Babenko *et al.*, 2009] At each frame t, the tracker extracts two sets of image patches around the tracked face location from the previous frame $\mathbf{x}_{t-1} = (x, y, w)$: X^r and $X^{r|\beta}$. Patches in X^r are taken in any direction such that their Euclidean distance from $\mathbf{x}_{t-1}$ is smaller than a radius r, and are inserted into one positive bag. Multiple negative bags for the MILTrack appearance model are filled with patches $X^{r|\beta}$ from an annular region of radius rn such that $r \leq rn \leq \beta$. The motion model of the tracker assumes that any position within a radius s from the location of the previous frame is equally likely.

Then the estimated position of the tracker $\mathbf{x}_t$ is chosen such that it maximizes the log-likelihood of bags in the appearance model. While better than other adaptive appearance tracking systems, MILTrack is still affected by the drifting effect, as shown in Figure 4.4.Therefore we integrate

Algorithm 1 K-Track

1. **Input**:
   (a) Face track seed $\mathbf{S}_t = (x, y, w)$ at frame t
   (b) Frame Borders **FB** $= (0, 0, imageW, imageH)$
   (c) Object Tracker $Tr(\mathbf{x})$, Object Detector $De(\mathbf{x})$
   (d) Weight parameter K (estimated from Eq. 4.13)
2. **Initialize:**
   $\hat{\boldsymbol{x}}_t, \tilde{\boldsymbol{x}}_t$ and Estimated Position $EP(\hat{\boldsymbol{x}}_t) \leftarrow \mathbf{S}_t$
3. **Forward**:
   While $(EP(\hat{\boldsymbol{x}}_t) \subset$ **FB**$)$ **and** $(t \neq shotBound)$
   (a) **if** ($\exists$ seed $\mathbf{S}_t$)
   (b) $\quad \hat{\boldsymbol{x}}_t \leftarrow \mathbf{S}_t$
   (c) **else**
   (d) $\quad$ compute prediction $\tilde{\boldsymbol{x}}_t \leftarrow Tr(\hat{\boldsymbol{x}}_{t-1})$
   (e) $\quad$ **if** ($\exists$ observation from $De : \boldsymbol{x}_t^O \cap \tilde{\boldsymbol{x}}_t$)
   (f) $\quad\quad \hat{\boldsymbol{x}}_t \leftarrow \tilde{\boldsymbol{x}}_t + K\left(\boldsymbol{x}_t^O - H\tilde{\boldsymbol{x}}_t\right)$ (Eq. 4.10)
   (g) $EP(\hat{\boldsymbol{x}}_{t+1}) \leftarrow \hat{\boldsymbol{x}}_t + \hat{\dot{\boldsymbol{x}}}_t$
   (h) $tEnd = t \leftarrow t + 1$
4. **Backward**: **Repeat** Step 3 beginning at $\mathbf{S}_t$, but decrementing t at each iteration to find $tStart$

Result: track $T_m = [\mathbf{x}_{tStart}, ..., \mathbf{x}_{tEnd}]$

the tracker into the proposed *K-Track* framework, expressed in Algorithm 1, which we previously showed to be a simplified version of the Kalman filter. We begin at a track seed, and initialize MILTrack with it. At each frame t, if we are still within the loose shot boundaries determined as described above and the predicted position of the tracked region (computed using its estimated velocity at frame t) is not outside the frame, we proceed with the tracking step.

In case the Viola Jones detector finds a region $\boldsymbol{x}_t^O$ overlapping the output of the MILTrack $\tilde{\boldsymbol{x}}_t = (x, y, w)$, we then consider $\tilde{\boldsymbol{x}}_t$ to be the noisy prediction part of our filter, while the noisy observation $\mathbf{x}_t^O$ is provided by the Viola Jones detection. We then update the general K-tracker position according to Equation 4.10 (Step 3.(f)). It must be noted that the tracking process is re-initialized in case a face track seed $\mathbf{S}_t$ is encountered while tracking, since the confidence of being

correctly on target in a track seed is extremely high.

The tracking process is re-initialized in case another face track seed $\mathbf{S}_t$ is encountered while tracking, since the confidence of being correctly on target in a track seed is extremely high.

4.3 Optimal Face Selection

We propose to process face tracks to select the most useful faces for both track matching and indexing purposes. We perform such selection based on the following three quality measures, which are specifically designed to extract good candidates to create an index to be presented to humans, but resulted to be useful also for face track matching.

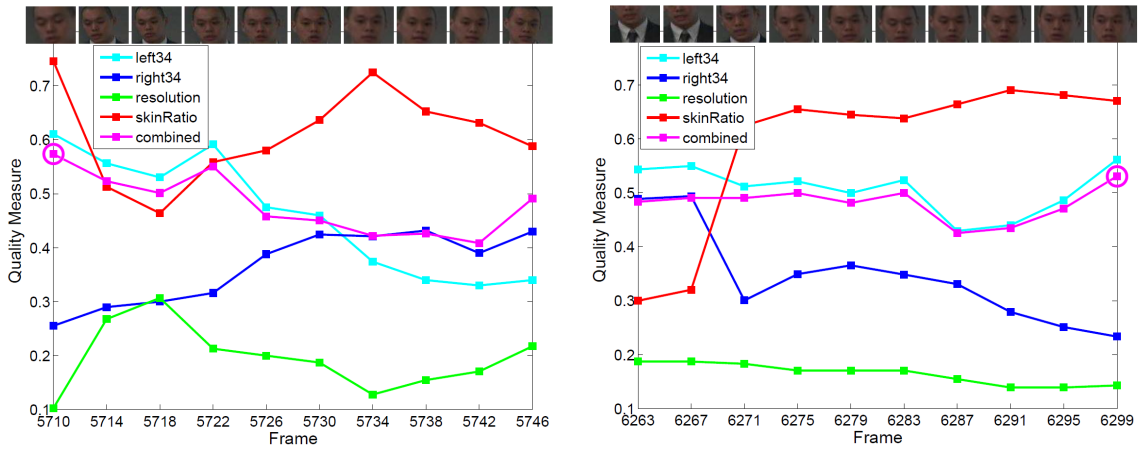
Pose. We select faces presenting a 3/4 view, following the literature [Burke *et al.*, 2007; Liu *et al.*, 2006] and the results of an informal user study in which we asked people which pose representation of a face they preferred to be shown to them as part of a visual face index. They confirmed the conclusions of psychological and computational tests, suggesting that humans prefer (or are able to infer most of the 3D information about the head of a person) seeing a 3/4 view of a face. In order to do so, we trained a left 3/4 and right 3/4 pose detector using 1200 images from the FaceTracer[4] dataset. Each classifier is an SVM with RBF kernel based on edge histogram extracted in 5x5 uniformly split regions in an image.

SkinRatio. We select faces with a high fraction of the image occupied by skin pixels, using the filter introduced in Equation 4.1. This measure is useful to exclude samples where the tracker drifted away from the face of a speaker.

Resolution. We select faces that are large. The low resolution of the videos in our dataset (in particular video 3, 432x240) demands that the index must contain face images with as much close-up as possible, so that compression artifacts be mitigated and details in the face be clearer.

Figure 4.5(a) and (b) show examples of how the quality measures affect the selection of a representative face for a sequence. Part(a) shows a face rotating from a left 3/4 to a right 3/4 view. The *pose* classifiers (in light and dark blue) follow the transition smoothly. The sequence starts with a close-in on the face, which cuts part of the forehead and of the chin, and then expands to a larger, zoomed out view (see the green resolution line). The combination is obtained as a weighted sum of

[4] http://www.cs.columbia.edu/CAVE/projects/face_search



(a) (b)

Figure 4.5: Quality measures example sequences from Video 2. (a) Detail of the performance of the left/right three quarter view classifier. (b) Detail of the benefit of the skinRatio classifier. Note how in both sequences the combination of the quality measures ensures that a close-up, 3/4 centered view of the face is selected from the sequence (circled in the magenta combined plot, frame 5710 for sequence (a), frame 6299 for sequence (b)).

the quality measures according to Equation 4.16, and leads to the selection of frame 5710.

The benefit of the *skinRatio* quality measures is exemplified in the details of Figure 4.5(b). Since the tracker is drifted at the beginning of the shown sequence, the scores of the pose classifiers are erroneously high. However, the skinRatio score for drifted regions is low and increases as the face of the person becomes more central to the region of interest (as more of his skin becomes visible). Therefore the skinRatio filter drives the combined sore to select a much better face (6299) in the sequence.

To determine whether two faces match, a suitable representation to describe each face must be adopted. Our approach for face selection within a track is independent from the matching descriptor choice across tracks. In order to perform across tracks matches, we chose to represent each face with the Local Binary Pattern (LBP) descriptor [Ahonen *et al.*, 2004]. We split each face into a 7x7 grid and concatenate LBP histograms computed from all the regions into a 2891 dimensional feature vector **v**. Finally, we use the square root of the Euclidean distance between feature vectors as a metric to evaluate the similarity between two feature vectors **v1** and **v2**.

In order to select faces for the final speakers visual index, we took into account another result

of our informal user study: people preferred to be shown a head-and-shoulder representation of a speaker, rather than simply the face by itself. Therefore, while the quality analysis was conducted on the face region, when producing the visual indexes reported in Figure 4.16 we enlarged the face bounding box by 20% horizontally and vertically, and further duplicated its height to include a head and shoulder view.

4.4 Experiments

We ran experiments on 3 different MPEG videos containing student presentations. Each video is approximately 45 minutes long, for a total of more than 2 hours of footage and one quarter of a million frames. The videos were recorded by non-professionals and are unedited. They also present challenges in that the camera is rarely steady, there are no clean cuts to identify shots, resolution is low, and they lack structure. Table 4.1 presents the information about number of speakers and tracks of the inspected videos.

4.4.1 Face Detection

We tested on our corpus of videos the tracks seeds detection method. Figure 4.6 illustrates the performances of our skin filter, in comparison to the raw Viola Jones [Viola and Jones, 2002] face detector (applied with the default parameters of the OpenCV implementation). Each curve is obtained by calculating the values of precision and recall, with respect to ground truth face tracks, as the parameter $skinRatio$ of our filter varies:

$$DetPrec = \frac{\#PDTracks}{\#TotDetTracks}, \quad DetRecall = \frac{\#PDTracks}{\#TotTracks}$$

where $\#PDTracks$ represents the number of positively detected tracks, that is, the number of tracks for which at least one of the faces belonging to it has been detected by the system. $\#TotTracks$ indicates the total number of (manually verified) tracks present in the video, while $\#TotDetTracks$ represents the total number of candidate tracks detected by the algorithm. The benefit of the proposed filter appears clearly in terms of recall. In fact, filtering doubles precision performances with respect to the default Viola Jones face detector from 0.407 to 0.8 in video 2, while keeping the same level of recall (0.935). Increasing the level of precision is of fundamental

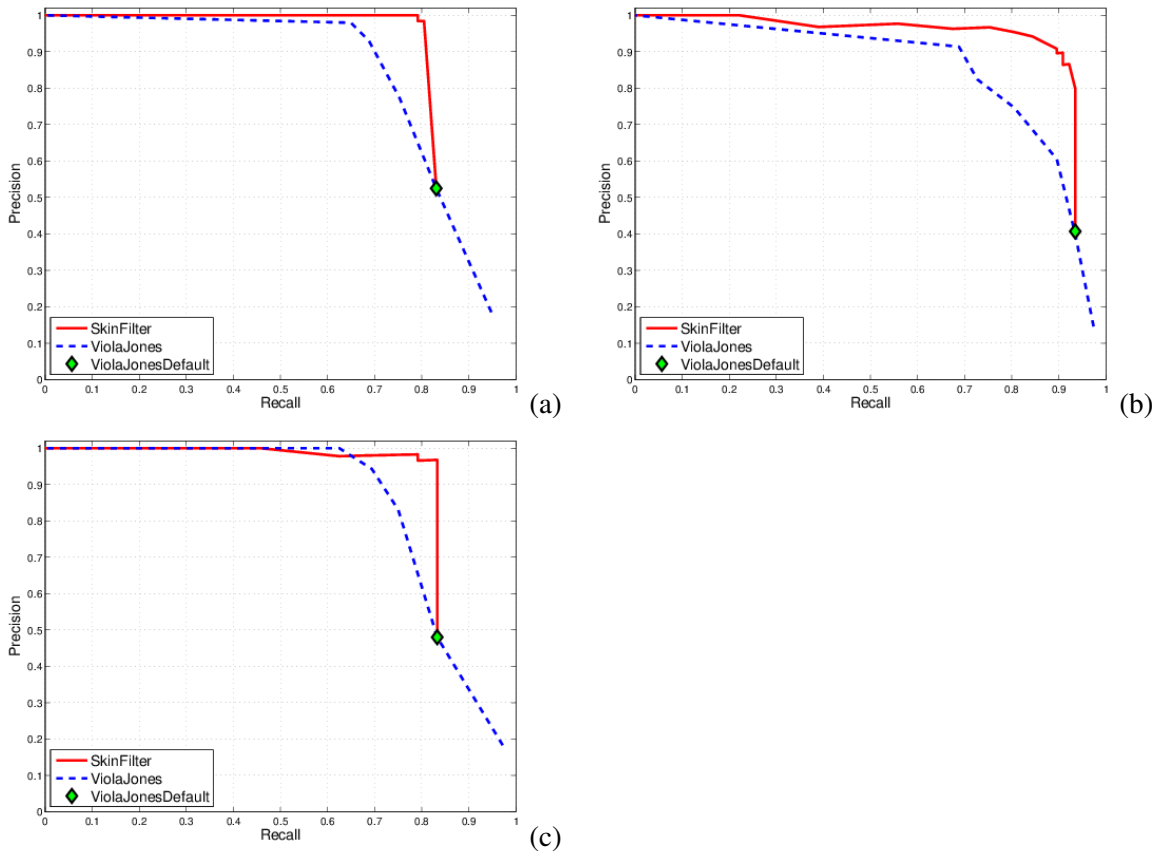


Figure 4.6: Precision-recall curves representing performance of the tracks seeds detection method, as function of the $skinRatio$ filter parameter. Results for videos 1 (a), 2 (b) and 3 (c) show that applying the skin color filter improves the performance of the Viola Jones face detector (dashed blue line). The green diamond represents the performance of the default OpenCV implementation of the Viola Jones detector.

importance in this stage, since it influences the number of instances of tracker that will be generated and consequently the number of tracks that will be further processed for matching and indexing purposes. If the tracks seeds detector produces a large quantity of false candidate seeds, the rest of the processing pipeline will be heavily affected in a negative way.

Given the reported results, we set the parameter $skinRatio = 0.2$ for track seeds detection.

Video	# Frames	# Speakers	# GTracks	GTATL
1	85K	19	77	1718
2	103K	19	77	2249
3	65K	20	72	1367
Total	253K	58	226	1787

Table 4.1: Experiments videos ground truth information: number of frames, number of speakers, number of ground truth tracks and Average Track Length (ATL, in frames).

4.4.2 Tracking

In order to test the performances of the proposed tracking framework, we employed a second set besides the *Presentations* one. We denominate it *Standard*, and consists in five publicly available short video sequences which have been extensively used to evaluate state of the art tracking algorithms.

In all experiments we use MILTrack as our predictor $Tr(\mathbf{x})$. We used the MILTrack algorithm in its default implementation, with $r = 5$ and $\beta = 50$, which is publicly available[5]. In face sequences we adopt the Viola Jones face detector (applied with the default parameters of the OpenCV implementation[6]) to be $De(\mathbf{x})$. For the *Liquor* sequence, we detect the object by matching SIFT features from a reference image (downloaded from the web) in each frame, and estimating its bounding box by computing the affine transformation between the matching points. Such approach (shown in Figure 4.11(a)) has been employed before for object detection, for example by [Merler *et al.*, 2007].

In all experiments we compare against two baselines. The first baseline is formed by the state of the art general purpose trackers MILTrack and PROST, which provide top performances on the *Standard* sequences. This comparison has the goal to quantify the boost in performance obtained by exploiting class specific information (namely the detections) to improve generic appearance based trackers. This is particularly relevant as MILTrack is used as the predictor in our framework. The comparison in our experiments confirm and quantify the (considerable) performance improvement, which is intuitively expected.

The second baseline we compare against is the track-by-detection algorithm proposed by [Ever-

[5]http://vision.ucsd.edu/∼bbabenko/data/MilTracker-V1.0.zip

[6]http://opencv.willowgarage.com

ingham *et al.*, 2009] which, like ours, is class-specific. In the experiments, particularly for the long, unstructured *Presentations* videos, the limitation of track-by-detection systems emerges in scenarios where the gaps to fill between consecutive detections are too extended in time. Our algorithm provides instead better performances by exploiting its adaptive, self-updating predictor part. We will now describe in detail the experimental results on each set separately.

Standard sequences.

The five sequences we used for this set of experiments are *David Indoor*[Lin *et al.*, 2004] (462 frames), *Occluded Face*[Adam *et al.*, 2006] (886 frames), *Occluded Face 2*[Babenko *et al.*, 2009] (812 frames), *Girl*[Birchfield, 1998] (502 frames) and *Liquor*[Santner *et al.*, 2010] (1741 frames). Each sequence presents a number of challenges for tracking algorithms in terms of illumination changes, 3D motion, occlusions, moving camera and scale.

All video frames in the five sequences are grayscale and were resized to 320x240 pixels. For all the sequences,we used the ground truth object center and bounding box every 5 frames which are publicly available[7][8], the estimation process operated only on the (x, y) coordinates of the center of the region of interest, and size of the object bounding box was fixed.

Since the ground truth of the inspected standard sequences is offered only as fixed bounding boxes, we had to keep the scale fixed in order to produce the results reported in Table 4.2. However, our algorithm is easily extendable to include scale estimations using the same Kalman framework (both MILTrack and VJ offer multiple scale outputs). In fact, we tested this approach(named *K-TrackProp* and *K-TrackPropBin*, respectively) obtaining comparable results to the other methods in terms of center location errors (Table 4.2). As reported in Table 4.3, proportional scaling is significantly worse than its single scale counterparts in terms of overlap with the ground truth regions. This is due to the nature of the ground truth annotations, which are provided all at a single fixed scale, even when the size object shrinks or grows given different distances from the camera.

According to the model described in Section 4.2.2, the coordinates are considered to be independent from each other, and variances σ_o and σ_a of the measurement and prediction errors respectively were independently estimated for x and y. For each of the four face tracking sequences, we com-

[7] http://vision.ucsd.edu/∼bbabenko/project_miltrack.shtml

[8] http://gpu4vision.icg.tugraz.at/index.php?content=subsites/prost/prost.php

puted K from the σ_o and σ_a estimated on the other three sequences, while for the *Liquor* sequence we used the estimates from all four other clips. According to the integration of the tracker (predictor) in the Kalman framework, σ_a refers to the acceleration of the ground truth object across frames, while σ_o refers to the variance of the detection error. From the sequences we estimated values of σ_a ranging from 1 to 2.48 pixels/frame^2 in the x direction and from 0.36 to 1.7 pixels/frame^2 in the y direction, since people tend to move more horizontally than vertically. The range of σ_o is wider, from 7.72 to 26.72 pixels along x and from 4 to 7 pixels along y. These estimations were then used to compute the steady-state values for the Kalman gain K.

From the graphs in Figure 4.7 emerges that when adopting the regular Kalman framework, as introduced in Section A.1, the steady-state and therefore a constant value for the K is quickly reached in all the sequences, thus justifying the use of this simplified assumption from the start in our framework. The solid lines were obtained by applying a regular Kalman filter, using the ground truth Q and R values for each sequence. On the other hand, for each sequence, the dashed lines represent the values of K estimated by adopting the steady-state assumption and the ground truth values of σ_o and σ_a computed from other, *independent* sequences, therefore we did not exploit any a priori information about each specific clip, but about object tracking behavior in general. The substantial equivalence of the constant gains estimated with the two methods confirms the benefit of adopting our framework, since it allows to estimate valid values from K **without any assumption** about the measurement and prediction noise for the specific sequence to be analysed.

For these *Standard* clips we simply initialized the tracker at the beginning of the sequence and let it progress in an online fashion, without relying on the shot boundaries and track seeds detection framework described in Chapter 4.2.2.

In the following, our algorithm is denoted as *K-Track* and *K-TrackBin*. Those indicate two different strategies that we adopted in the case of multiple detections in the same frame to re-evaluate the value of $K1$. In fact, as mentioned in the previous Sections, the values of K were estimated based on the assumption of single measurements, since in the investigated videos there is only one object present for most of the time. In case of multiple detections, *K-Track* selects the detection closest to the prediction $\tilde{\boldsymbol{x}}_t$ to be the observation $\boldsymbol{x}_t^O$ to be combined according to Equation 4.14. We consider that the probability that detections not related to the object of interest (either false detections or detections associated with other objects of the same class) could mislead the overall

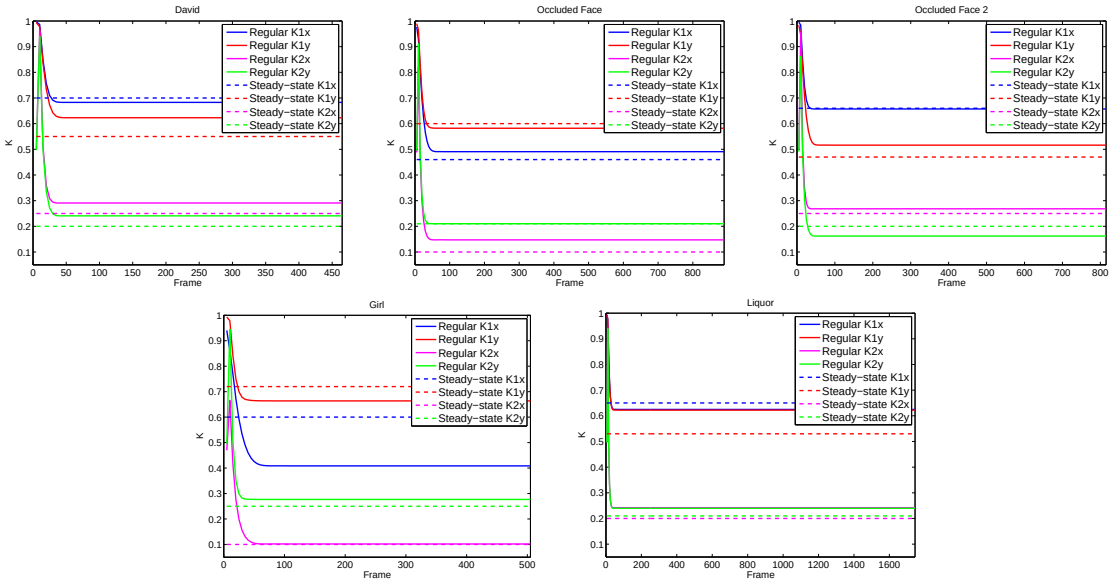


Figure 4.7: Kalman gain estimate on the *Standard* sequences. In all cases, the estimate of a regular Kalman filter which uses the ground truth Q and R values evaluated from each sequence (solid line) quickly converges to a steady-state value in both x and y directions for the position as well as the velocity components. The steady state estimates reached after convergence are equivalent to the gains directly estimated from the other sequences (dashed lines), therefore justifying the simplifying assumption of a steady-state Kalman framework.

tracker is (to a first approximation) inversely proportional to the distance between the "correct" detection $\boldsymbol{x}_t^O$ and second closest (to the predictor output) detection $\boldsymbol{x}_t^{O2}$. In fact, $\boldsymbol{x}_t^{O2}$ is basically a distractor from the point of view of the tracker, and the closer it is to the output of the predictor (and the proper detection candidate), the higher the chances that it could be chosen by the tracker and could lead it away from its proper target. Taking into account that $K1$ fundamentally measures the trust of the overall tracker in its detector, we compute the value of $K1$ to be proportional to the distance between $\boldsymbol{x}_t^O$ and the second closest detection $\boldsymbol{x}_t^{O2}$. The further the second and potentially confusing detection $\boldsymbol{x}_t^{O2}$, the higher the confidence assigned to the measurement $\boldsymbol{x}_t^O$, according to the following formula, where W and H are the frame width and height: $K1 = \frac{2\left|x_t^O - x_t^{O2}\right|}{WH}$

On the other hand, *K-TrackBin* simply considers multiple detections as an enormous increase in the measurement error variance, so therefore sets $K1$ to zero, trusting only the prediction $\tilde{\boldsymbol{x}}_t$ according to Equation 4.14.

Video	MILTrack	PROST	Everingham	K-Track	K-TrackBin	K-TrackProp	K-TrackPropBin
David	22.98	15.25	5.95	***4.66***	4.98	**4.66**	***4.16***
Occluded Face	27.23	***6.98***	10.95	**10.08**	10.18	10.64	10.12
Occluded Face 2	20.19	17.18	14.18	11.38	11.17	***8.66***	**9.81**
Girl	31.99	18.99	***9.86***	20.05	21.69	19.22	**15.81**
Liquor	153.31	***21.6***	67.37	**43.37**			

Table 4.2: Tracking performances in terms of average Euclidean distance between ground truth and tracked box centers. Lower values represent better performances. Best performance is highlighted in bold and italic, second best performance in bold. In the *Liquor* sequence, since the detector is engineered to find only one or zeros occurrences of the object and the bounding box is determined through an affine transform, the performances of the K-Track methods coincide.

In Table 4.2 are reported the tracking results in terms of Euclidean Distance in pixels between ground truth and predicted face center, averaged over all frames in the sequence. The performances for our algorithm are the result of an average over 5 runs. It can be seen that *K-Track* offers the best or second best performances on most sequences.

In Figure 4.8 we report the pixel errors on each frame of the four face video sequences. The plots compare *K-Track* with MILTrack, PROST and Everingham. Again, in most sequences our approach outperforms the others. It is interesting to observe the improvement with respect to MILTrack, since it is integrated as the predictor in our framework.

In the *Girl* sequence, we notice that there are two intervals (frames 200 to 250 and 430 to 470) in which our approach drives significantly away from the ground truth. The details of such intervals are reported in Figures 4.9 and 4.10. In the first case the head of the person is rotating, therefore preventing $De(\mathbf{x})$ from correctly finding a proper face. In the second case, another person enters the scene and multiple face detections are reported. The detail of Figure 3 reports how *K-Track* is able to deal with the problem more gracefully than *K-TrackBin*. It is interesting to notice that in both cases our approach is able to quickly recover the correct position once a proper face is detected.

Of particular interest is also the comparison with Everingham's class specific tracking mechanism, which is well known to perform close to perfection on short shots in professional videos. The length and challenging conditions of the investigated clips made it impossible to generate a single track lasting for the whole clip using their method. When two detections were separated in

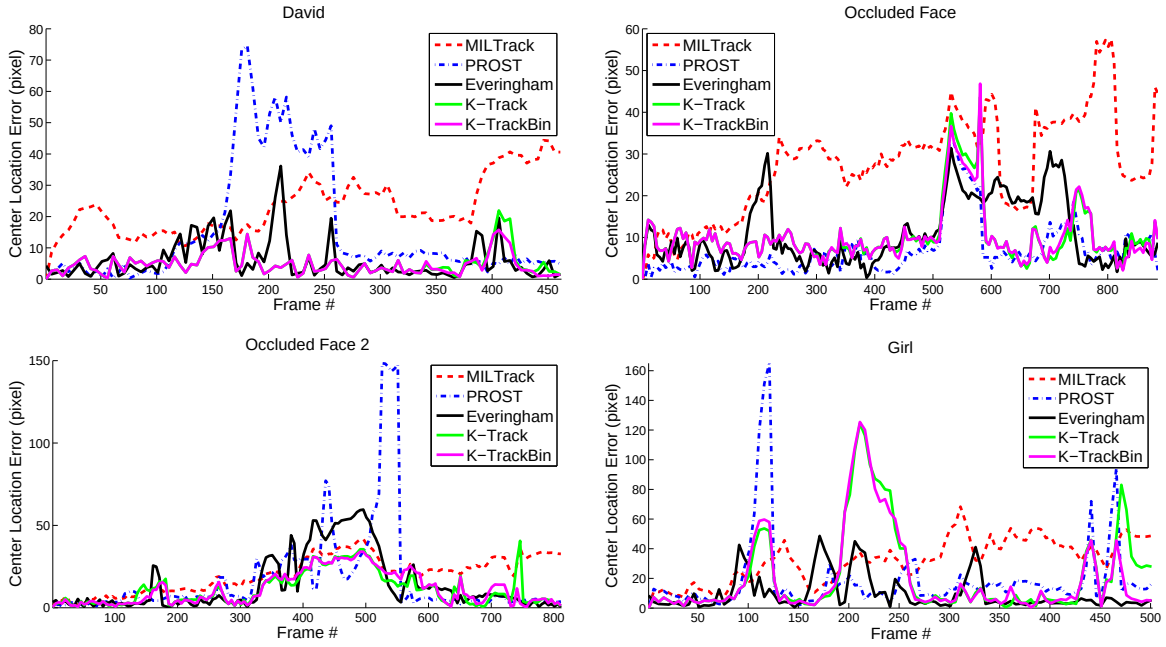


Figure 4.8: Tracking Precision as function of the Euclidean distance between the center of ground truth and tracked regions. Comparison of *K-Track* and *K-TrackBin*with MILTrack, PROST and Everingham et al.

time by a long interval, not enough KLT tracked features where found within both detections, and that part of the track was missed. We had to fill those gaps by interpolating the object coordinates. This problem was particularly evident in the *Liquor* sequence, where the object is smaller and the number of KLT features to track is limited. Furthermore, being based on optical flow, this method suffers occlusions by other moving objects (i.e. the book in the sequence *Occluded Face*).

The experiments on the *Liquor* sequence confirm the generality of the proposed framework, as it can be extended to integrate *any* object tracker and detector. The particular in Figure 4.11 (b) shows a large improvement with respect to MILTrack and Everingham's method, and the results are comparable to the state of the art PROST algorithm for such sequence.

Recently [Santner *et al.*, 2010] noted that the mean center location error may be a limited performance measure due to scale changes and suggest instead to use the following score from the PASCAL challenge[9], which measures the area overlap between system predicted region SR and

[9]http://pascallin.ecs.soton.ac.uk/challenges/VOC/

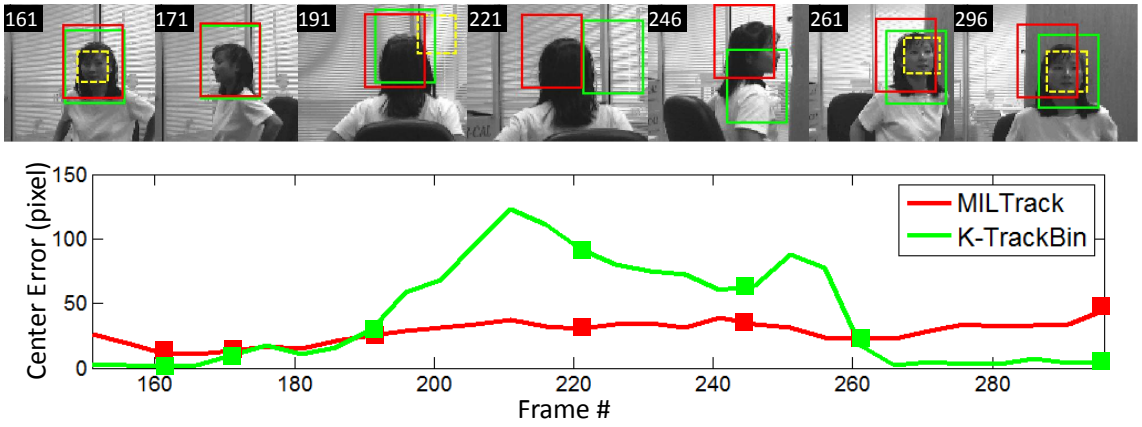


Figure 4.9: Detail of interval 150-300 in the sequence *Girl*. When the person turns her head, $K - TrackBin$ (in green) does not have any face detection to rely on.

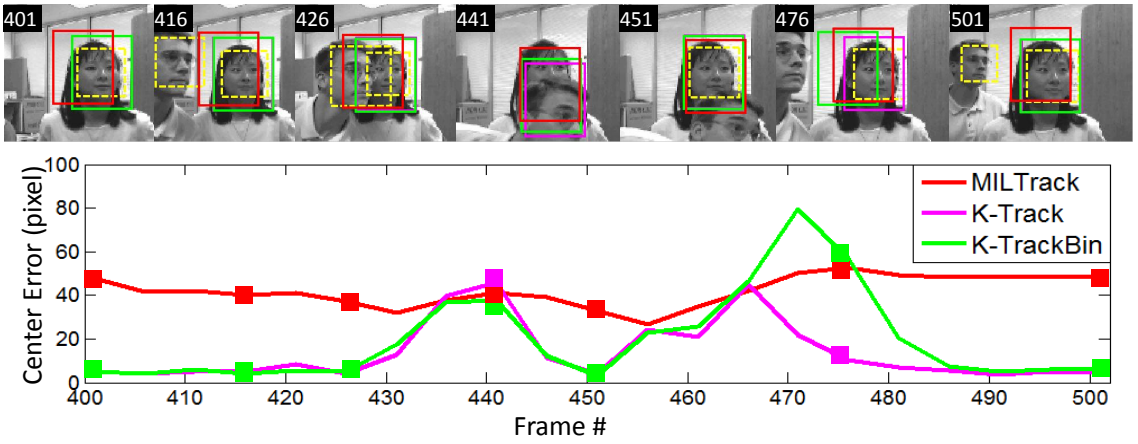
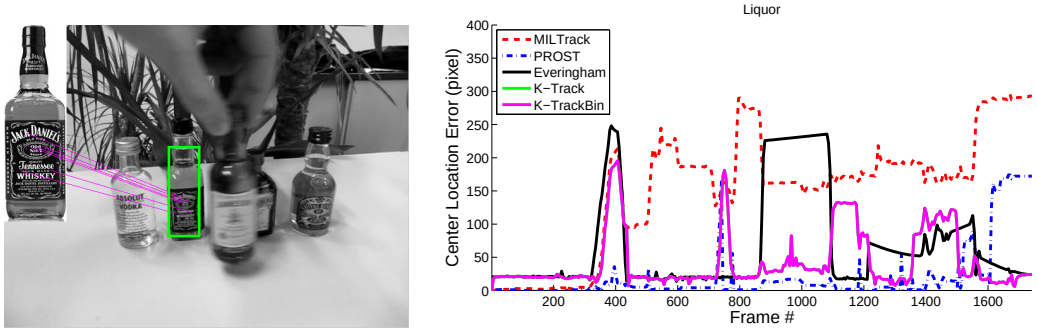


Figure 4.10: Detail of interval 400-502 of sequence *Girl*. When there are multiple detections (yellow dashed lines), *K-Track* (in green) uses the closest one to the prediction and recomputes K based on the position of the closest distractor. *K-TrackBin* (in magenta) simply sets K to zero. Between frames 460 and 480 the latter strategy leads momentarily the tracker to follow the wrong person. As soon as the distractor face disappears, *K-Track* recovers the correct position faster than MILTrack (in red) (error graph detail after frame 476)

ground truth region GT

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

We compare the performances of our algorithm using the above measure with respect to MIL-



(a) (b)

Figure 4.11: Analysis on the standard *liquor* sequence. (a) Example of the liquor detector, with highlights of the SIFT matches between the reference web image (top left) and the video bottle, and of the estimated object bounding box. (b) Tracking Precision as function of the Euclidean distance between the center of ground truth and tracked regions. Comparison of *K-Track* and *K-TrackBin*with MILTrack, PROST and Everingham et al. Note that since there is always only one single detection (or none) in each frame, K-Track and K-TrackBin coincide.

Track, PROST and Everingham in Table 4.3. From the Table results that *K-Track* performs comparably or better than the other methods, with the proportional way (K-Track) to deal with multiple detections offering better results than the binary strategy (K-TrackBin).

Presentation videos.

For each of the three MPEG-1 videos in this set, we manually labeled the ground truth center coordinates for all the face tracks in the videos. Table 4.1 presents the information about number of speakers, number of ground truth tracks and tracks length of the inspected videos.

For these videos we adopted the shot boundary and track seeds detection framework described in Chapter 4.2.1. Once the seed has been established for a face track, we start tracking the face in both temporal directions, until the track exits the frame borders or one of the detected shots boundaries is encountered as explained in Algorithm 1.

Since there are multiple speakers in each video and cases similar to the ones outlined in Figure 3 are quite common, for this dataset we track each face individually and adopt the *K-Track* strategy in its *K-TrackPropBin* form. We evaluate face tracking following the frameworks introduced by [Yin *et al.*, 2007]. We analyzed performances at the face regions level for all the frames, comparing

Video	MILTrack	PROST	Everingham	K-Track	K-TrackBin	K-TrackProp	K-TrackPropBin
David	0.56	0.71	0.85	***0.88***	**0.87**	0.53	0.53
Occluded Face	0.65	***0.88***	0.81	**0.83**	0.82	0.56	0.56
Occluded Face 2	0.63	0.75	0.76	**0.79**	**0.78**	0.69	0.69
Girl	0.50	0.72	***0.83***	**0.73**	0.71	0.37	0.39
Liquor	0.22	***0.77***	0.49	**0.56**			

Table 4.3: Tracking performances in terms of area overlap between ground truth and tracked box. Higher values represent better performances. Best performance is highlighted in bold and italic, second best performance in bold. In the *Liquor* sequence, since the detector is engineered to find only one or zeros occurrences of the object and the bounding box is determined through an affine transform, the performances of the K-Track methods coincide.

our algorithm (using K estimated from the *Standard* sequences) against MILTrack and the Everingham's. We define a match between a system region SR_j and a ground truth face region GT_i when SR_j contains the center of a GT_i. We can then calculate precision, recall and F1 measures as follows:

$$TrackPrec = \frac{\#(SR \cap GT)}{\#SR}, \quad TrackRecall = \frac{\#(GT \cap SR)}{\#GT}$$
$$TrackF1 = 2 * \frac{TrackPrec * TrackRecall}{TrackPrec + TrackRecall}$$

note that there is a difference between $SR \cap GT$ and $GT \cap SR$: $SR \cap GT$ expresses the total number of system regions matching at least one ground truth region, while $GT \cap SR$ refers to the number of ground truth regions having at least one match with a system region. In Table 4.4 are reported precision, recall and F1 performances for each of the three videos we investigated. For videos 1 and 2, the F1 performance of *K-Track* is significantly superior to the one of the original MILTrack algorithm. In video 3 they are essentially equal.

The method of Everingham et al. suffers from the lack of editing in the videos, which results in an extensive length of the shots. Such length, and the reduced size of the faces in the videos, make it hard for their system to track a large number of KLT features from one detected face to another. As a result, the tracks produced by their system are quite short and localized around temporally near face detections.

Therefore the recall rate is quite low. On the other hand, the precision rate and the $radius$ are much better than other methods, since they are heavily influenced by the detections. Looking at

Video	Measure	MILTrack	Everingham	K-Track
1	TrackRecall	0.891	0.238	**0.899**
	TrackPrec	0.608	**0.878**	0.640
	TrackF1	0.723	0.374	**0.747**
	$radius$	6.76	**4.698**	7.09
2	TrackRecall	0.912	0.231	**0.919**
	TrackPrec	0.535	**0.877**	0.591
	TrackF1	0.674	0.366	**0.719**
	$radius$	6.74	**3.91**	6.14
3	TrackRecall	**0.921**	0.292	0.920
	TrackPrec	0.580	**0.970**	0.574
	TrackF1	**0.712**	0.449	0.707
	$radius$	6.29	**3.31**	6.04

Table 4.4: Tracking performances in terms of Precision, Recall, F1 and average Euclidean distance($radius$) between ground truth region and system region on Presentation videos. Comparison between MILTrack, Everingham et al. and *K-TrackPropBin*. Best performances are highlighted in bold.

the F1 statistic, our proposed framework clearly outperforms their method. Such poor performance rates for a method which behaves almost perfectly on professional content is due to the lack of editing, the low quality and the unstructured nature of the investigated videos, and shows how new methods such as our proposed framework are needed to deal with this "wild" content.

As pointed out by [Yin *et al.*, 2007], the above definition of match between ground truth and system face region tend to advantage larger regions returned by the system, which are not necessarily accurate. In order to provide a more complete analysis of the tracking performances, we provide another definition of a match between ground truth region and system region: a match is determined if the Eucidean distance between the centers of the two regions is smaller than a threshold $radius$. In Figure 4.12 is reported the variation of tracking recall as the value of $radius$ is incremented. Note the limited but consistent improvement in performance with respect to MILTrack in all videos, in particular for smaller values of $radius$.

As stated in Section 4.2.2, we use the Kalman framework to determine a principled and optimal

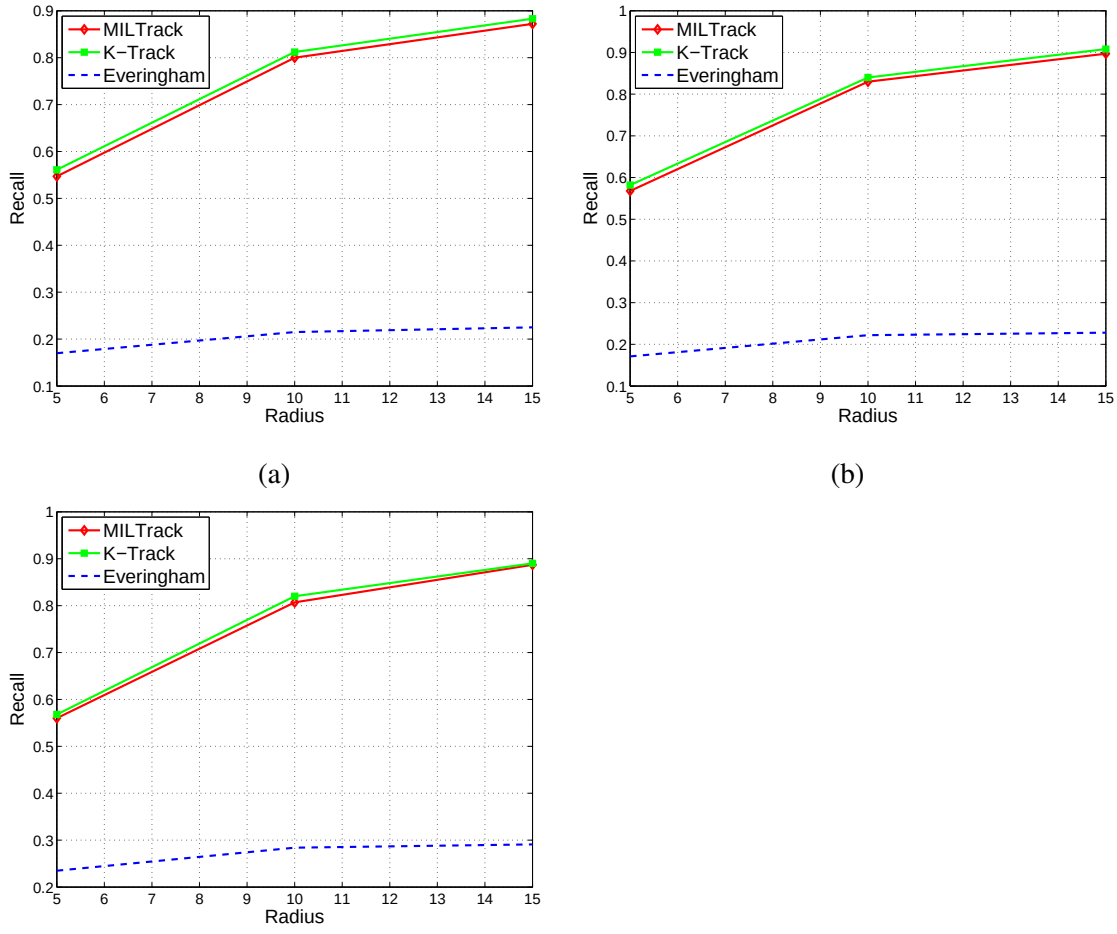


Figure 4.12: Tracking Recall as function of the $radius$ on Presentation videos 1 (a), 2 (b) and 3 (c). The benefit of *K-Track* is particularly evident for small values of $radius$. A limited but consistent improvement is registered when using *K-Track* (green line) with respect to the MILTrack algorithm (red line) in all videos. Everingham's method is unable to produce long tracks due to the reduced size of the faces in the videos, and therefore its recall rate is much lower than other methods.

(under certain assumptions) way to combine the predictor and the detector at each instant. One simple alternative could be to pick arbitrary values for K to combine the two components. The tracking performances on the *Presentation* videos reported in Figure 4.13 confirm the benefit of adopting the proposed Kalman framework to pick "optimal" values instead. In the Figure, tracking precision, recall, F1 and $radius$ are presented as functions of the combination parameter $K1$ (fixed at the same value for x and y). We remark that the selected values The values of $K1$ selected by

$K-Track$ are highlighted with diamond markers and were estimated from the *Standard* sequences, therefore no parameter tuning was performed with respect to the investigated *Presentation* videos. Our Kalman framework choice of $K1$ produces the best or close to best results in almost all cases for all performance measures. We registered suboptimal results for $radius$, particularly in Video 1. Looking at the range of values for $radius$ we realize however that the difference in performances between different choices is minimal (subpixel).

4.4.3 Face Tracks Matching

In order to perform matching between tracks, a standard approach used among others by [Everingham *et al.*, 2009] consists in computing the min-min distance between tracks T1 and T2, that is, compute the distance between each possible pair of elements $t1 \in T1$ and $t2 \in T2$.

We tested two methods to selection subsets of elements to match in each track: the first one involves a simple temporal sampling of the faces in the track, while the second consists in an unsupervised method based on image quality measures. For temporal sampling, we simply selected n uniformly distributed example faces per track, and we tried values of $n = 1, 3, 10, 100$.

We evaluated track matching accuracy for each video, using the extracted tracks with best tracking precision in each video. Matching is performed in a Nearest Neighbor classification framework: given a reference track T_r, we compute the distance between the reference and all the tracks T_i in the video which do not temporally overlap with it (we consider that if two tracks overlap temporally, they must belong to different individuals). We retain the track T_i which has the smallest distance $d(r, i)$ to T_r to be a candidate match, and if $d(r, i)$ is smaller than a matching threshold, we consider that T_r and T_i are a match. We compared four different modalities to compute the distance between two tracks.

The first modality (min-min) is to compute the distance between all pairs of images $d(\mathbf{v}_r^m, \mathbf{v}_i^n)$, where $\mathbf{v}_r^m \in T_r$ and $\mathbf{v}_i^n \in T_i$, and keep the minimum distance to represent the distance between the tracks. This method presents two limitations in our unconstrained domain. The first one is efficiency, given the large amount of redundancy present in each track due to the temporal domain of a video, using all the faces in a track is an unmotivated cost. The second is accuracy, since the face tracks generation module returns a set of tracks which can be considered noisy, containing false positives from the seeds detection stage, and drifts in tracking caused cutting of face regions.

The second modality involves using a temporal sampling of n faces in each track, computing the distances between these reduced sets and retain the minimum one.

The third modality employs K-means clustering in the LBP feature space, as suggested by [Mau *et al.*, 2010]. The $K = n$ cluster centers are used to compute the distances between tracks, and the minimum distance is kept. We tested with the same values of n as for the temporal sampling approach.

The last modality consists in computing the distances only between the n top track faces which are selected based on the response to our quality measure filters (see Section 4.3).

For temporal, K-means, and selection matching we retain $n = 100$ samples from each track (or the whole track in case its length is smaller than 100). Experiments with $n = 1, 3, 10$ provided worse matching accuracy results.

Figure 4.14 reports track matching accuracy as a threshold on the maximum distance between tracks to consider them a match varies (expressed in percentage of the range of values of the track distances). It must be noted that random guessing would provide a 0.5 matching accuracy, but given the high imbalance between the number of matching and non-matching track pairs, in the Figure we report a baseline which predicts all pairs to be non-matches. For all videos, all selection-based matching methods present a global maximum in the accuracy with respect to the chosen threshold. The optimal threshold seems to be consistently located between 60% and 70% of the range of distances between tracks. Such observation could be generalized to fix a threshold for processing new videos. From the results shown in Figure 4.14, not only the computational cost of computing the distances between tracks is reduced when using filtering techniques to reduce the number of image pairs to match, but matching accuracy increases in two out of three videos. This is due to the reduction or removal of noisy, drifted, or partially cut images from the tracks which is accomplished through sampling. It is also interesting to notice that the best filter results in such videos is the skin color-based one. In fact, proper face matching requires a full face occupying most of the image, which is best guaranteed by the skin color filter.

Figure 4.14(d) shows the computational gain of using the proposed face selection method within tracks before matching. Notwithstanding the overhead introduced by feature extraction and face selection before matching, the proposed approach achieves a higher level of track matching accuracy while needing approximately 6% the running time of min-min matching. This is due to the greatly

reduced computational complexity of matching each pair of tracks: $O(k^2)$ (with $k = 100$ in our experiments) versus $O(n^2)$ for min-min, where n is the number of frames in a track. According to Table 4.1 the relationship between k and n is on average of 1 to 6.7, which is quite significant when squared.

We tested a series of possible combinations of the three quality measures according to the following formula (each quality measure is normalized between 0 and 1):

$$Q = w1 \cdot pose + w2 \cdot resolution + w3 \cdot skinRatio \tag{4.16}$$

with $pose$ being either $left34$ or $right34$. We empirically found that the best combinations of $\mathbf{w} = (w1, w2, w3)^T$ were $(0.0, 0.3, 0.7)^T$, $(0.3, 0.1, 0.6)^T$ and $(0.0, 0.7, 0.3)^T$ for video 1, 2 and 3 respectively, and $pose = right34$ for all videos. It is interesting to notice how $resolution$ is much more important for the low resolution video 3, while $pose$ does not seem to be fundamental, probably because faces with different poses were matched against each other.

We also note that the performance of our selection method is comparable with K-means clustering both in terms of accuracy and efficiency. However, our selection method provides us also with the candidate faces for the speakers visual index, whereas the average faces returned by K-means do not hold any meaning to a human.

4.4.4 Representative Index Extraction

In order to obtain the faces to build the speakers visual index, we took the results of the 3 filters presented in Section 4.3 to all the images in each track and retained the ones returning the best combined scores, following Equation 4.16 (with the modification that $pose = max\,|left34, right34|$, since differently from matching we do not care which direction the face is facing, as long as it is a 3/4 view). The best face among all those in the tracks representing the same speaker (resulting from track matching) is expanded to include a head and shoulder view of the person.

In Figure 4.15 is reported the accuracy of the indexes obtained with different combinations of the three quality measures, as well as their individual performances. Accuracy is measured as the fraction (out of the possible 58 speakers) of selected images representing a 3/4, head and shoulder view of a speaker. In this framework, differently from matching, the $pose$ measure is the predominant factor in performance. This is because from the perspective of the visual index, a full

frontal or full profile view of a person is considered an error. It is also interesting to notice that while in video 1 and 2 $resolution$ seems to hurt in combination with the other two measures, on the lowest resolution video 3 this effect is not registered (lower triangle of the heat map).

Figure 4.16 shows the head and shoulders views of speakers selected for the visual index. The system is able to automatically generate a qualitatively pleasing index consisting of 51 out of the 58 speakers present in the videos. One speaker was never detected, and in some cases the wrong person or view were selected since tracks were not matched properly or *resolution* and *skinRatio* prevailed over *pose*.

We have presented a system to select the most representative faces in unstructured presentation videos with respect to two criteria: to optimize matching accuracy between pairs of face tracks, and for indexing purposes.

Experiments on 3 unstructured presentation videos demonstrate the benefit of our contributions in terms of tracks matching, while reducing the average running time with respect to min-min matching. We were able, by using quality metrics, to build face indexes of 51 out of of 58 speakers with a head and shoulders, 3/4 view, which is the pose preferred by humans.

In the continuation of this work, we plan to conduct user studies to assess the usefulness and likability of our speaker indexes, integrated into a multimodal presentations search engine including also textual and graphical cues.

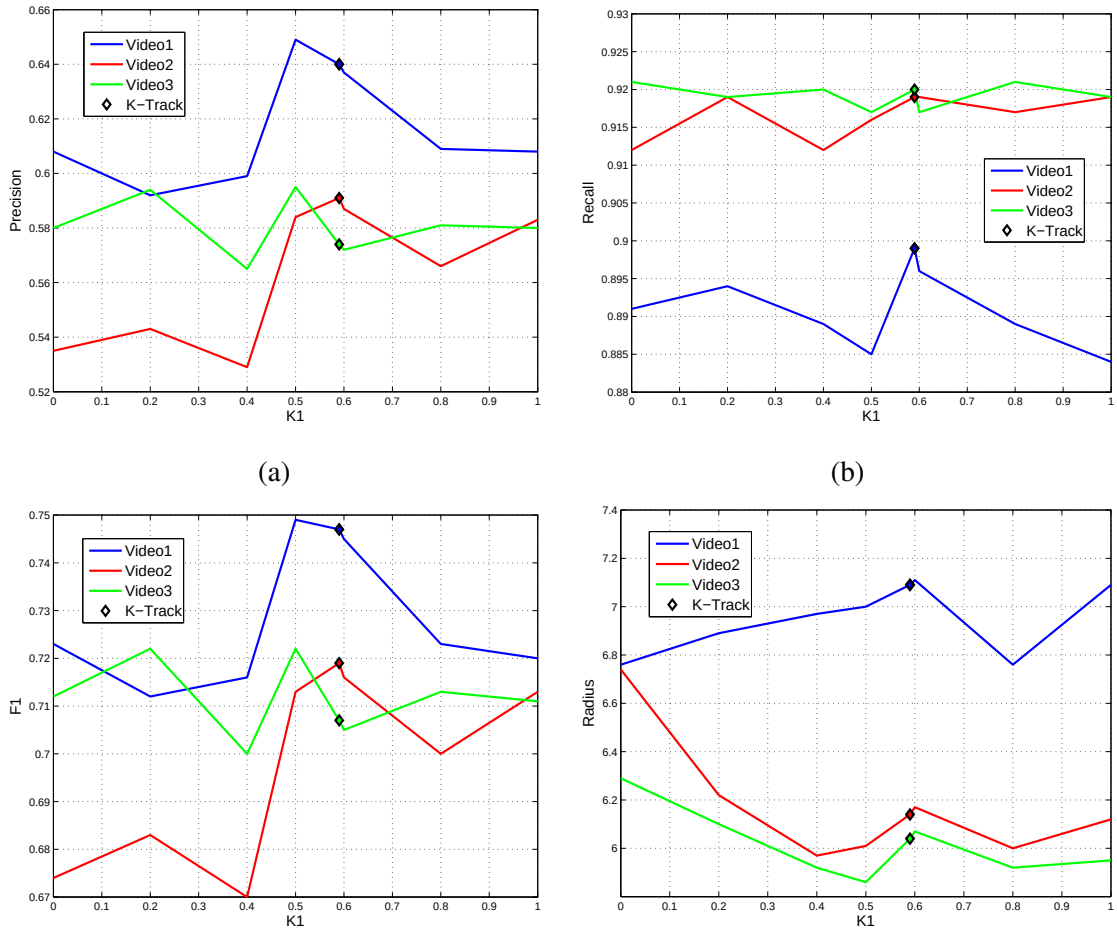


Figure 4.13: Tracking performances on *Presentation* videos as a function of parameter $K1$: (a) precision, (b) recall, (c) F1 and (d) $radius$. Video 1, 2 and 3 are represented with blue, red and green curves respectively. The Kalman framework of *K-Track* allows to pick an "optimal" value for the parameter $K1$ (points with diamond markers). In fact, performances at the selected values of $K1$ are the best or close to the best. Note that for $radius$ (d), the lower the value the better.

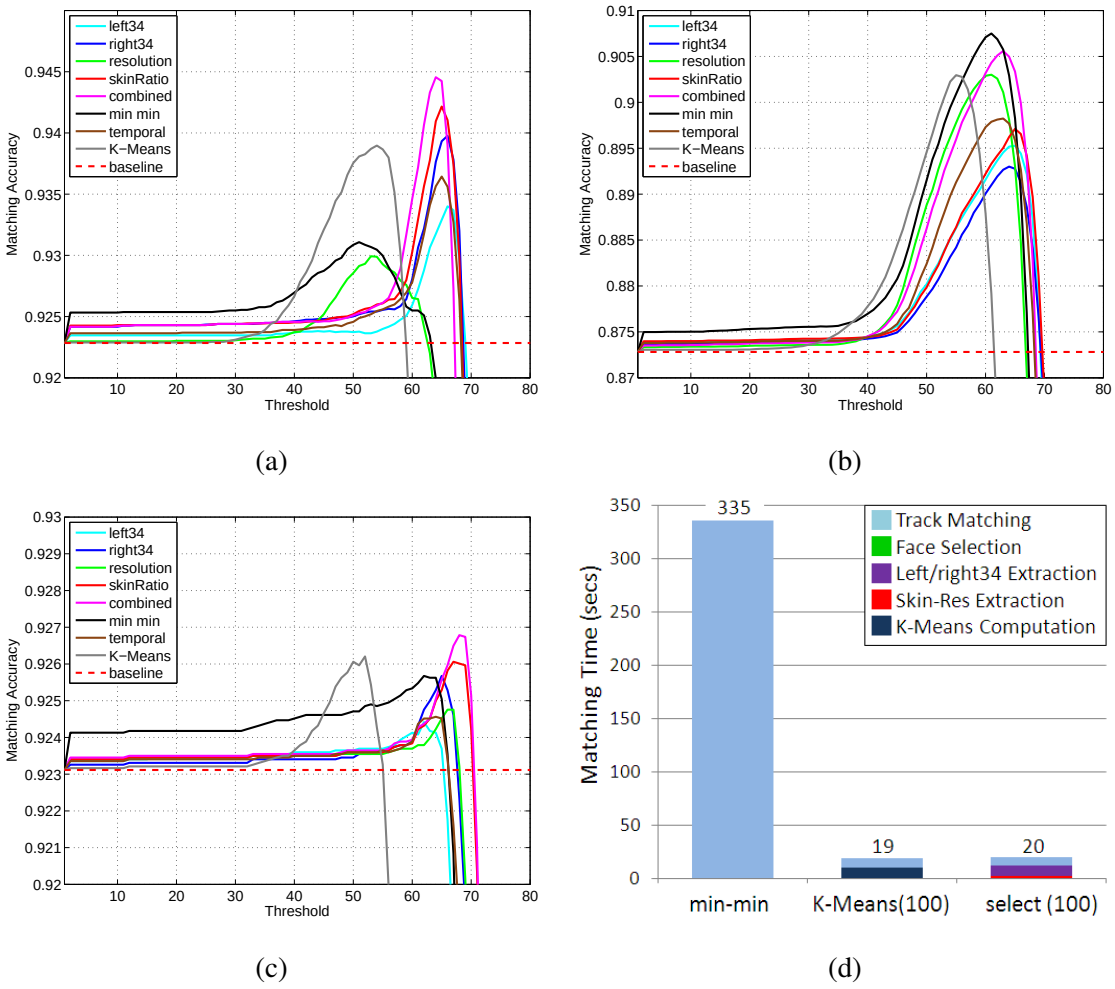


Figure 4.14: Matching accuracy performance for the investigated videos as a function of threshold equal to the maximum distance at which two tracks are considered a match (expressed in percentage of the dynamic range of distances between tracks). Random guess produces 0.5 accuracy. Given the imbalance between matching and non-matching track pairs, a more suitable baseline consists in predicting no matches between any pair of tracks produces the baseline (red dashed line). Results reported for videos 1(a), 2(b) and 3(c). (d) Average processing time (in seconds) for track matching. Comparison between the min-min standard approach, K-means clustering (in dark blue) and the proposed selection method, which is based of 4 steps: skinRatio and image resolution extraction (2.46 seconds, in red), pose classifier evaluation (9.08 seconds, in violet), face selection (0.02 seconds, in green) and track matching (8.18 seconds when the top 100 faces for each track are retained, in light blue). Note that, unlike our proposed method, K-means does not provide face indexes.

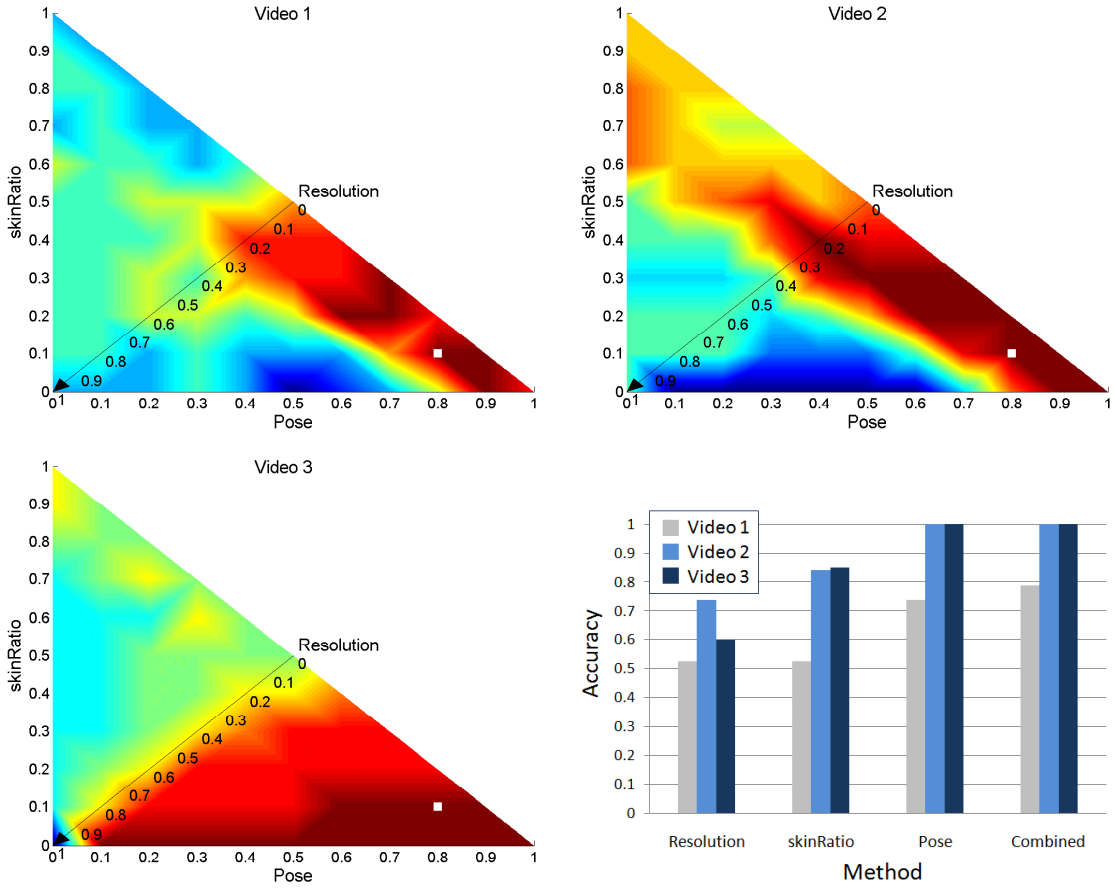


Figure 4.15: Selection accuracy for index building on the three investigated videos. Heat map of the accuracy given combinations of quality measures in Equation 4.16. The white squares represent the optimal combination, which is $\mathbf{w} = (0.8, 0.1, 0.1)^T$, interestingly shared across all three videos. Bottom left: accuracy of face felection for indexing methods, alone or in combination.

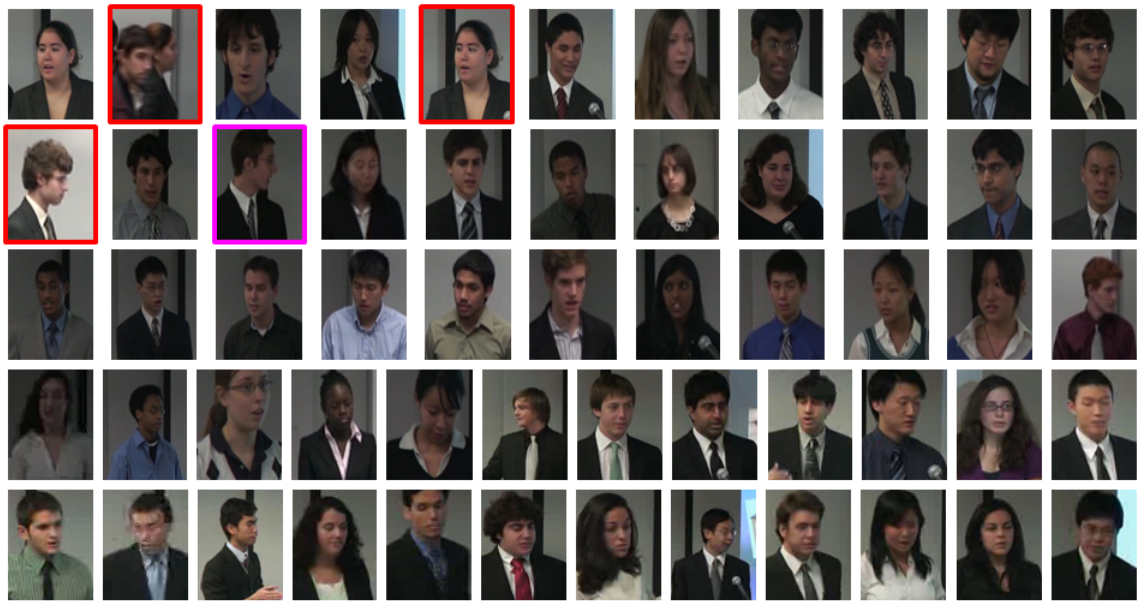


Figure 4.16: Generated visual speaker index. Most of the images show the desired 3/4 head and shoulder view of the speaker. Some fail, either by portraying the wrong person (in red) with respect to the ground truth or by presenting a full profile (in magenta), from which is hard to identify the person.

Chapter 5

Diagram Indexing

We propose a system to index presentation videos based on a new semantic cue: diagrams.

The proposed system first detects diagram regions from frames where a slide occupies the majority of the camera field of view, using standard region detection methods. An online clustering algorithm which combines visual and temporal similarity is employed to group together regions representing the same diagram.

One image is then selected to represent the diagram cluster based on resolution, and finally color corrected with the automatic white balancing Grey-world algorithm [G. and Buchsbaum, 1980]. This is done to generate a visual index of diagram icons which reflect human users preferences collected through Amazon Mechanical Turk experiments, that is, large regions which show as much of the diagram as possible and are white balanced.

5.1 Diagram Extraction System

In order to index all the graphics/diagrams from the video, we need to localize them first, both spatially and temporally. In the following we describe the processing pipeline adopted to first detect frames where a projected slide occupies most of the camera field of view, then localize the graphics regions within the frame and finally cluster together regions detected in different frames but representing the same diagram.

5.1.1 Slide Detection

Since slides provide a natural way to semantically segment a presentation video [He *et al.*, 2000], methods to detect them within frames and to model their transitions become quite important. In our framework we are interested in finding in each video the frames where the focus of the camera is a projected slide. Once a slide frame is found, we can then proceed to extract the information we seek from it (in the context of this work, graphic elements). Many algorithms have been proposed for slide detection. Most of them rely on electronic copies of the slides and align them with video frames using local or global features matching [Fan *et al.*, 2011; Fan *et al.*, 2006; Gigonzac *et al.*, 2007]. Since references or resources besides the recorded video are not always available, we adopt a reference-free method, similarly to [Adcock *et al.*, 2010], who augment a standard keyframe extraction method with face and text detection filters to obtain keyframes where a slide is predominant.

We employ a similar but simpler method, which stems from the observation that in a presentation video recording setting, slides are projected onto a screen thus generating a washed out illumination field for which the camera recording system cannot adapt, unlike from the human vision system. Therefore we compute the average amount of color shift in the frame, either toward a high or low color temperature, as an estimator of the light produced by the projected slide. We compute such shift following the color temperature estimation framework of [Huo *et al.*, 2006], in which a color temperature shift toward low or high temperatures is estimated as the average amount of saturation $\overline{R_{SAT}}$ and $\overline{B_{SAT}}$ of the red or blue channels, respectively:

$$\begin{aligned} \overline{R_{SAT}} &= \left(\sum_{x,y} \frac{R(x,y) - Y(x,y)}{N} \right) = \overline{R} - \overline{Y} \\ \overline{B_{SAT}} &= \left(\sum_{x,y} \frac{B(x,y) - Y(x,y)}{N} \right) = \overline{B} - \overline{Y} \end{aligned} \tag{5.1}$$

where $\overline{C}$ is the average intensity in channel C of an image, N is the total number of pixels and x,y are the pixel coordinates. Y is computed as the average of the RGB channels, and in the model is it represents an approximation to the scene illuminated by white light source (which by definition has equal contribution by all color channels).

The average amount of color shift $\overline{C_S}$ in the frame is then computed as the maximum tempera-

ture shift, as follows:

$$\overline{C_S} = max\left(|\overline{R_{SAT}}|, |\overline{B_{SAT}}|\right) \tag{5.2}$$

Finally, a frame is considered to contain a sufficiently large slide if the average amount of color shift is greater than a threshold θ_s which was estimated on a training set of 20 videos. Details of the training process and the performance of slide frames detection are reported in Section 5.3.1.

5.1.2 Diagram Regions Detection

Slides may contain text and graphical elements such as diagrams, individually or in any combination. Therefore our approach to detect diagrams and images is based on a process of elimination, meaning that we consider a diagram anything that is not labelled as text or background.

The steps of our approach are the following:

- Extract an edge map from the input image using a Laplacian of Gaussian filter
- Group edge connected components into rectangular regions
- Extract geometrical and texture properties from each region, namely coordinates of the center, area, width, height, width/height ratio, density of edges, edge histogram, vertical and horizontal alignment, LBP histogram.
- Separate regions into noise, text-regions, and diagram-regions

In order to perform the last step, we adapt the text detection algorithm presented in Chapter 3.1 based on edge density and geometric constraints. It applies empirically validated thresholds to the extracted features in order to determine which ones are text regions.

Once the text regions are established, diagrams are determined based on a process of elimination (a diagram regions is basically a regions which is *not text*, and *not noise*). The procedure is the following.

- Remove text rectangle regions from the list of candidate regions
- Eliminate rectangles that are too large and with more than 30% area of texts, and also small rectangles contained in others

- Expand each rectangle by 5% in both width and height
- Combine partially overlapping rectangles R^i and R^j if:

$$0.5 < \frac{R^i_{height}}{R^j_{height}} < 1.5 \tag{5.3}$$

$$0.5 < \frac{R^i_{EdgeDensity}}{R^j_{EdgeDensity}} < 1.5 \tag{5.4}$$

$$HO\left(R^i, R^j\right) \geq min\left(R^i_{width}, R^j_{width}\right) + 10 \tag{5.5}$$

$$VO\left(R^i, R^j\right) \geq min\left(R^i_{height}, R^j_{height}\right)/2 \tag{5.6}$$

  where HO and VO represent the horizontal and vertical overlap between regions.

- Eliminate rectangles whose area is less than 0.5% of the image and with aspect ratio larger than 5 or smaller than 0.5.
- Return the remaining rectangle regions as diagrams candidates

5.1.3 Diagram Regions Clustering

Once the candidate graphic regions have been detected, we need to group regions representing the same diagram or graph together. In order to do so we employ an online clustering algorithm based both on visual and temporal similarity, as proposed by [Papka and Allan, 2002]. Using that terminology, we consider each detected graphic region in the slides to represent a *story* regarding a particular *topic*, that is, a unique diagram. Like a topic, a diagram can appear (or be repeated) in multiple separated time intervals. For example one figure can be presented at a certain point of a talk, and then be brought back during the Q&A session. Each occurrence of such figure is then a story regarding the same topic.

Our online clustering approach is presented in Algorithm 2. We represent each detected region $\mathbf{x}_i$ using a normalized concatenation of two global descriptors: one focusing on color and the other on texture:

- **Color histogram**: representing the global color distribution of the region as a 166-dimensional histogram in HSV color space

- **LBP histogram**[Ahonen *et al.*, 2004]: extracted from the greyscale version of the image as a histogram of 8-bits local binary patterns, each of which is generated by comparing the grayscale value of a pixel with those of its 8 neighbors in circular order, and setting the corresponding bit to 0 or 1 accordingly. A pattern is called uniform if it contains at most two bitwise transitions from 0 to 1. The final histogram contains 59 bins, 58 for uniform patterns and 1 for all the non-uniform ones.

Given this representation, when a new graphics region $\mathbf{x}_i$ is detected at time t, we calculate the visual similarity between the region and a cluster $C_j \in \Psi$ (from the initially empty set of existing clusters Ψ) , $1 \leq j \leq |\Psi|$ as the inverse of the χ^2 distance, adopting average linkage (which we found to produce better results than single and complete linkage in our experiments), as follows

$$S(\mathbf{x}_i, C_j) = \frac{1}{|C_j|} \sum_{k=1}^{|C_j|} \frac{1}{\chi^2(\mathbf{x}_i, \mathbf{c}_{jk})} \tag{5.7}$$

where $\mathbf{c}_{jk} \in C_j$, $\forall k = 1, ..., |C_j|$. We also define the average cluster self-similarity as the average similarity among all regions in the cluster:

$$S(C_j) = \frac{2}{|C_j|(|C_j| - 1)} \sum_{k=1}^{|C_j|-1} \sum_{q=k+1}^{|C_j|} \frac{1}{\chi^2(\mathbf{c}_{jk}, \mathbf{c}_{jq})} \tag{5.8}$$

Therefore, the difference in visual similarity becomes

$$\Delta_{vis}(\mathbf{x}_i, C_j) = |S(\mathbf{x}_i, C_j) - S(C_j)| \tag{5.9}$$

The temporal difference is computed as the difference between the timestamp of the detected region and the time of the last region added to the cluster C_j:

$$\Delta_{time}(\mathbf{x}_i, C_j) = t(\mathbf{x}_i) - t(C_j) \tag{5.10}$$

A weighted combination of the visual and temporal differences is computed to obtain a unique difference score $d(\mathbf{x}_i, C_j)\ \forall j = 1, .., |\Psi|$ and then the minimum score is retained.

$$d(\mathbf{x}_i, C_j) = \alpha\Delta_{vis}(\mathbf{x}_i, C_j) + \beta\Delta_{time}(\mathbf{x}_i, C_j) \tag{5.11}$$

The values of α and β were determined empirically through cross-validation, as discussed in Section 5.3.

Finally, a candidate cluster C^* to incorporate the new region is selected as the one with the minimum difference from it

$$C^* = \underset{j}{\operatorname{argmin}} \left(d(\mathbf{x}_i, C_j)\right) \tag{5.12}$$

and the minimum difference is compared against a threshold θ_c, in order to determine if region $\mathbf{x}_i$ should actually be merged with the selected existing cluster C^*, or if a new cluster should be generated

$$d(\mathbf{x}_i, C^*) \gtrless \theta_c \tag{5.13}$$

5.2 Visual Index User Preference Experiments

After we have clustered graphics as described in the previous Section, we need to select one iconic image per cluster to be the representative in the visual index of the video. As a guiding line, we want to select the image that most closely reflects the preferences of human users. To this end, we conducted two experiments to estimate the preference of users with respect to the appearance of the icons used in the visual index of the diagrams found in each video. The first experiment concerns the need and type of color correction/automatic white balancing to apply to the icon, the second to determine what type of view of a diagram (full diagram or detailed blow up of part of it) the users deem more representative/useful. As with the face indexes, for both experiments we employed Amazon Mechanical Turk.

5.2.1 White Balance

The human eye possesses the *color constancy* ability to cope with different lighting conditions and adjust for different colors of the light source [Gijsenij *et al.*, 2011]. Camera sensors, on the other

Algorithm 2 Online Graphics Regions Clustering

1. **Initialize:** Empty set of clusters $\Psi \leftarrow \emptyset$
2. **for** t=1,...,T **do**
   find graphics regions in frame t
   for all detected regions $\mathbf{x}_i$, $\forall C_j \subset \Psi$ **do**
   $\Delta_{vis}(\mathbf{x}_i, C_j) = |S(\mathbf{x}_i, C_j) - S(C_j)|$
   $\Delta_{time}(\mathbf{x}_i, C_j) = t(\mathbf{x}_i) - t(C_j)$
   $d(\mathbf{x}_i, C_j) = \alpha\Delta_{vis}(\mathbf{x}_i, C_j) + \beta\Delta_{time}(\mathbf{x}_i, C_j)$
   end for
   $C^* = \underset{j}{\operatorname{argmin}}\,(d(\mathbf{x}_i, C_j))$
   if $d(\mathbf{x}_i, C^*) < \theta_c$ **then**
   $C^* \leftarrow C^* \cup \mathbf{x}_i$
   else
   Create new cluster $C_{new} \equiv \mathbf{x}_i$
   $\Psi \leftarrow \Psi \cup C_{new}$
   end if
   end for
3. **Result:** $|\Psi|$ distinct clusters representing unique graphics

hand, do not have such property and since the graphics regions are extracted from videos recording projected slides, the illumination tends to shift toward high temperatures. Therefore a white balancing algorithm should be used to restore color constancy and remove the artefacts introduced by the recording process. The goal of this user study is to capture user preferences in terms of the type/amount of white balancing to adopt.

HIT Design

There exist a large number of algorithms to perform automatic white balancing and computational color constancy (for a survey, see [Gijsenij *et al.*, 2011]). We chose a few simple and well known ones to test and present to the users (for details, see Appendix B): original (no correction),

(a) (b) (c) (d) (e) (f) (g) (h)

Figure 5.1: Example of Automatic White Balance Methods on one diagram, sorted according to the AMT experiment user preferences: (a) Grey-world, (b) Original Extracted Diagram, (c) maxRGB, (d) Retinex 10 iterations, (e) Retinex 100 iterations, (f) Grey-world Single Channel, (g) maxRGB Single Channel, (h) Retinex 1 iteration.

Grey-world [G. and Buchsbaum, 1980] with one or two channels corrections, maxRGB [Funt and Shi, 2010] with one or two channels corrections, Retinex [Land and McCann, 1971] with 1, 10 or 100 iterations.

Hence we have a total of 8 possible versions of the same diagram region from which a user can select a favourite. Figure 5.1 shows the effect of the various methods applied to an example diagram region. In order not to bias the results to a specific worker or image ordering, we presented the regions in 3 different random orders and required at least 30 different workers per HIT. Furthermore, we did not provide the workers with any information about the processing that each image underwent, asking them to focus only on their preference for the general appearance of the image. We conducted tests on 40 diagrams coming from different videos, for a total of 3600 HITs.

Results

Of the 3600 HITs, only 20 were not completed successfully, either because the worker did not select any image (in 20 cases) or because he did not provide any explanation of his choice (10 cases).

Despite the distribution of preferences reported in Figure 5.2(a) being more uniform than expected, the simple *Grey-world* algorithm emerges as the most preferred choice. It is also worth noting that the preferences for the Retinex algorithm peak at 10 iterations, finding a compromise between too localized and too global restoration ranges. The choices made by the users were motivated mainly by the appearance of the colors (see distribution of reasons in Figure 5.2(b)). Since we did not reveal how the images were obtained or how the original diagram in the slide looked like, it becomes clearer that some users found the lighting distortion in the *Original* version of the diagram to be visually appealing and considered it to be a pleasant part of the diagram rather than

a noisy artefact introduced by the recording process. Nonetheless, following the results of the user study we apply the *Grey-world* white balancing correction to the selected diagram index icons, as explained in Section 5.2.3.

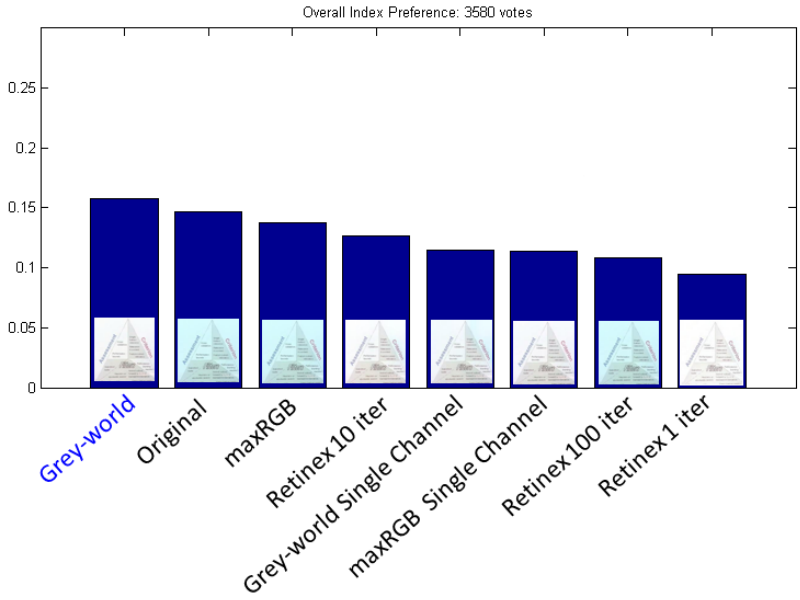


Figure 5.2: Results of user preferences for automatic white balance/color correction algorithms, sorted by preference. The most popular selection was the Grey-world with corrections on both the R and B channels.

5.2.2 Resolution

HIT Design

We found that after the diagram detection and clustering processes described in the previous Sections, mainly two types of diagram regions were recurring in each cluster: one containing the full view of the diagram (referred to as *Full View*), the other a zoomed in view of a detail/part of the diagram itself (referred to as *Detail*). The second view is due mostly to camera motion (in particular zoom in) or diagram detection errors. Even if one intuitively would think that a full view of a diagram would provide a better index icon, that might not always be the case. In fact, due to the low quality of the recorded videos and the distance from the recording camera to the screen where the slides are projected, sometimes the detected full view of a diagram does not have enough resolution to fully discern the meaning of the diagram which is included in some of its details. The zoomed

in version, on the other hand, can provide a clear vision of such details. It must be considered that in the final visual index, we will have to . For example in Figure 5.3 the text in the graph is not readable in its *Full View* version, but part of it is readable in the zoomed in one.

In order to determine which view is preferred by the users, we ran a test in which workers had to choose between the two views for 32 diagrams. We resized both views to a canonical size (the average of their dimensions), since for practical reasons the final visual index for the video will have only fixed size for every item, in the same fashion as standard image search engines show results in canonical fixed size thumbnails. We showed each image pair in both orders and required 50 unique workers per HIT, thus amounting to a total of 3200 HITs.

Results.

Of the 3200 HITs, 91 were not completed successfully. From the results presented in Figure 5.3 clearly emerges that the users prefer the *Full View* of a diagram which was chosen 75% of the time. We asked the users to also specify the reason behind each choice. The motivations follow intuition: *Full View* was mostly chosen because it provides more information, *Detail* was picked because the users could see the details better in it.

5.2.3 Diagram Index Selection

The results of our user studies have evidenced that users prefer a white-balanced, color corrected image that covers as much as possible the whole diagram.

In order to generate on iconic representation from the diagram image clusters detected in the videos, which reflects the user preferences emerged from the experiments, we adopt a two step process. First, we select the region with the highest resolution, that is, the one of largest size. This selection is based on the assumption that for the vast majority of the clusters the regions within a cluster are captured by the camera at approximately the same resolution, therefore larger regions represent larger portions of a given diagram. This in intuition was confirmed through visual inspection of the 20 training videos, and resulted to hold true for the five test sequences as well.

Once the candidate icon has been selected, we proceed to apply color correction by means of the Grey-World algorithm [G. and Buchsbaum, 1980]. We adopt the diagonal transform model [von Kries, 1970], which achieves the color correction from the pixels in the image with unknown light

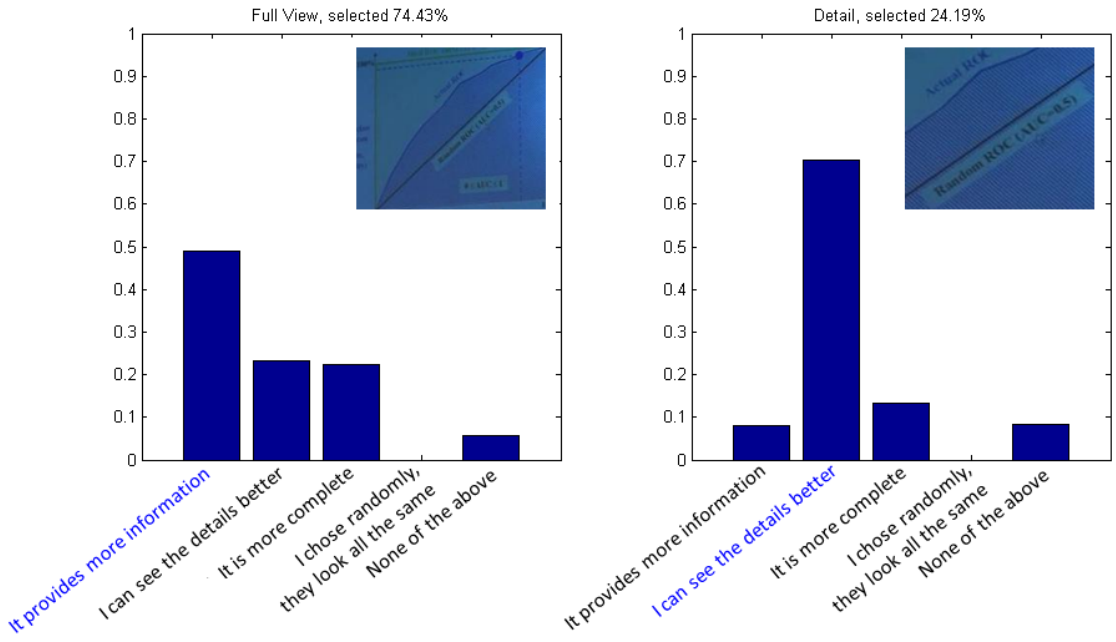


Figure 5.3: Results of user preferences for the resolution of the presented diagrams. The users clearly prefer a full view of the diagram (75% of the selections), even if some details might be too small or blurred to discern. The distribution of motivations for the given choices demonstrates how the workers picked a full view representation to see more information, and a blow up of a part of the diagram when interested in more details.

source I_u to the color corrected image I_c through a multiplication with a diagonal matrix D. The white balancing equation to recover a given pixel $P_u = (R_u, G_u, B_u)^T$ becomes then

$$\begin{pmatrix} R_c \\ G_c \\ B_c \end{pmatrix} = \begin{pmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{pmatrix} \begin{pmatrix} R_u \\ G_u \\ B_u \end{pmatrix} \tag{5.14}$$

Grey-world it assumes that the average intensities of the red, green and blue channels should be equal. Therefore the resulting values of D become

$$d_1 = \overline{G_u}/\overline{R_u}, d_2 = 1, d_3 = \overline{G_u}/\overline{B_u} \tag{5.15}$$

where $\overline{C_u}$ is the average intensity in channel C of I_u.

Figure 5.4 shows an example of the selection process for one cluster in video 3:

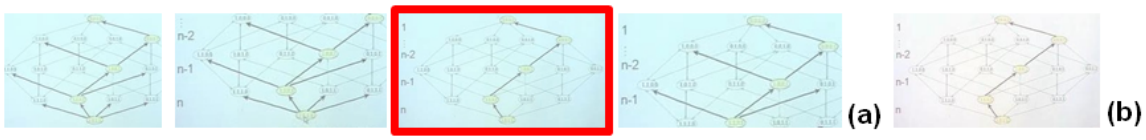


Figure 5.4: Selection process from a given diagram region cluster. (a) The region with the highest resolution (in red) is picked to be the icon representing the diagram. (b) The white balance of the selected region is restored using the Grey-world algorithm.

5.3 Experiments

We tested our proposed approach on five presentation videos retrieved from the Videolectures website[1], for a total of approximately three hours of video and over 10K frames extracted at a rate of on per second. Details on the properties of the individual videos are presented in Table 5.2. Each video was parsed at one frame per second and further processing was conducted on the frame images.

Together with each video we also obtained an electronic copy of the slides and the timestamps which locate each of them temporally in the video. Such resources were used, together with some annotations either generated manually or through AMT, to produce the ground truth against which we evaluated the fully automatic processing blocks of our system. All the experiments were carried on a Pentium 4, 2.33GHz machine.

5.3.1 Slide Detection

We used 20 videos to train a slide frames detection model based on the color saturation score introduced in Section 5.1.1. We extracted one frame per second from each video and manually provided ground truth for each frame, for a total of 48K examples. We computed a ROC curve (see Figure 5.5, in blue) from the true positive and false positive rates on the training data, which produces an area under the curve of 0.9. We selected a threshold θ_s corresponding to the point of maximum combined TP and FP rates. Using θ_s we were able to achieve rates of TP = 0.78 and FP = 0.21 on the 5 test sequences. The good generalization performance of the selected threshold is demonstrated by the location of the achieved rates (red diamond in the Figure) on the ROC curve for the test sequences (red dotted line), which is almost optimal, although not as good as the training performance.

[1] www.videolectures.net

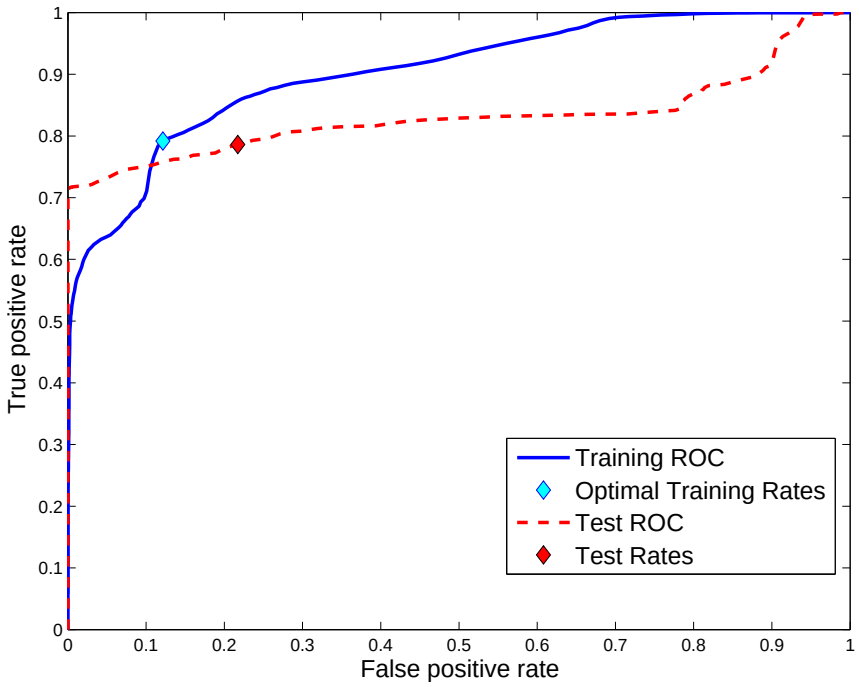


Figure 5.5: Slide detection performances. ROC of the 20 training videos performance (AUC = 0.9), with markers for the training and test TP and FP rates at the threshold θ_s on the color saturation selected during training. Test performance rates (on 5 videos): TP = 0.78, FP = 0.21.

5.3.2 Diagram Detection

In order to evaluate the accuracy of the diagram regions localization algorithm presented in Section 5.1.2, we gathered ground truth locations of all the diagrams appearing in frames where a slide is the focus, by gathering AMT annotations in the form of polygonal regions. Details of the AMT task design and annotations validation are provided in Appendix B.3.

Diagram detection is evaluated to estimate the number of ground truth diagram regions DR_{GT} found by the algorithm, which outputs candidate boxes DR_D. Precision and recall of diagram detection are computed respectively as

$$detPrecision = \frac{\#DR_{GT} \cap DR_D}{\#DR_D}, \quad detRecall = \frac{\#DA_D \cap DA_{GT}}{\#DA_GT} \tag{5.16}$$

where one intersection between two regions $DR_i \cap DR_j$ is counted if DR_i overlaps with DR_j by at least half of its size. For each frame, localization performances are based on the overlap between the ground truth diagram area DA_{GT} and the diagram area detected by our algorithm

DA_D. Localization Overlap, Precision and Recall are then defined as follows:

$$Overlap = \frac{DA_{GT} \cap DA_D}{DA_{GT} \cup DA_D}, \quad Precision = \frac{DA_{GT} \cap DA_D}{DA_D}, \quad Recall = \frac{DA_{GT} \cap DA_D}{DA_{GT}} \tag{5.17}$$

Diagram detection and localization performances are reported in Table 5.1 for all the test videos. The dataset proves to be challenging for the detection algorithm. Localization presents particularly limited Overlap and Recall rates (0.27 and 0.31 on average, respectively). Region detection results are better, as one would expect, given the different amounts of overlap between two regions triggering a true positive. Nevertheless the performances are quite limited in absolute value. Upon inspection of the results, there seem to be two main reasons for such limitations, especially from the localization point of view. The first is inherently structural, since the ground truth is provided under the form of polygons and the algorithm's output is a set of rectangles. The second reason is that the algorithm tends to detect subparts of any given diagram, rather than its full extension. This effect could be mitigated by adjusting some thresholds in merging procedure in Equations 5.3 to 5.6, but this would result in a trade-off between detecting only parts of a diagram versus returning very large regions which enclose the full diagram but also large sections of noise, with the first option being preferable for building the diagram index.

In fact, although the system does not appear to perform extremely well, it must be noted that the final goal is to find all the appearances of unique diagrams in a video, not to spatially locate them in every frame. Consider the extreme case in which our algorithm detects only a very small portion of a unique diagram region in all the frames but one, where it locates the full region correctly. This would result in very poor spatial detection and localization performances. However, given the clustering and selection steps described in Sections 5.1.3 and 5.2.3 respectively, the final goal of temporally identifying a unique diagram and represent it with a full size icon would still be fully achieved.

Therefore, while in the future we plan to adopt more complex and robust algorithms to improve the localization performance, even the current results are acceptable for our purposes.

Video	Overlap	Precision	Recall	detPrecision	detRecall
1	0.37	0.81	0.41	0.58	0.85
2	0.32	0.50	0.36	0.39	0.39
3	0.25	0.49	0.28	0.34	0.44
4	0.17	0.33	0.22	0.23	0.42
5	0.24	0.42	0.27	0.78	0.48
Avg.	0.27	0.51	0.31	0.46	0.52

Table 5.1: Diagram regions detection performance.

5.3.3 Diagram Clustering and Index Construction

Given the sets of detected diagram regions in various frames in the video, our goal is to cluster together regions representing the same diagram, and finally select one representative icon from each cluster to be in the final visual index. We can compare the clusters generated by our system to ground truth clusters, given that we have information where each slide is temporally located in each video and we manually annotated the diagrams within each slide.

Given N regions with $\mathbb{C} = \{c_1, c_2, ..., c_J\}$ ground truth clusters c_j and $\Omega = \{\omega_1, \omega_2, ..., \omega_K\}$ clusters created by the algorithm ω_k, we evaluate clustering performance using two standard measures [Manning *et al.*, 2008], Purity and Normalized Mutual Information (NMI), computed as as follows:

- $Purity(\Omega, \mathbb{C}) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|$
- $NMI(\Omega, \mathbb{C}) = \dfrac{2 \cdot MI(\Omega, \mathbb{C})}{[H(\Omega) + H(\mathbb{C})]}$

where MI and H are the mutual information and entropy measures, respectively computed as

$$MI(\Omega, \mathbb{C}) = \sum_k \sum_j P(\omega_k \cap c_j) log \frac{P(\omega_k \cap c_j)}{P(\omega_k)P(c_j)} \tag{5.18}$$

$$= \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} log \frac{N|\omega_k \cap c_j|}{|\omega_k||c_j|}$$

$$H(\Omega) = -\sum_k P(\omega_k) log P(\omega_k) = -\sum_k \frac{|\omega_k|}{N} log \frac{|\omega_k|}{N} \tag{5.19}$$

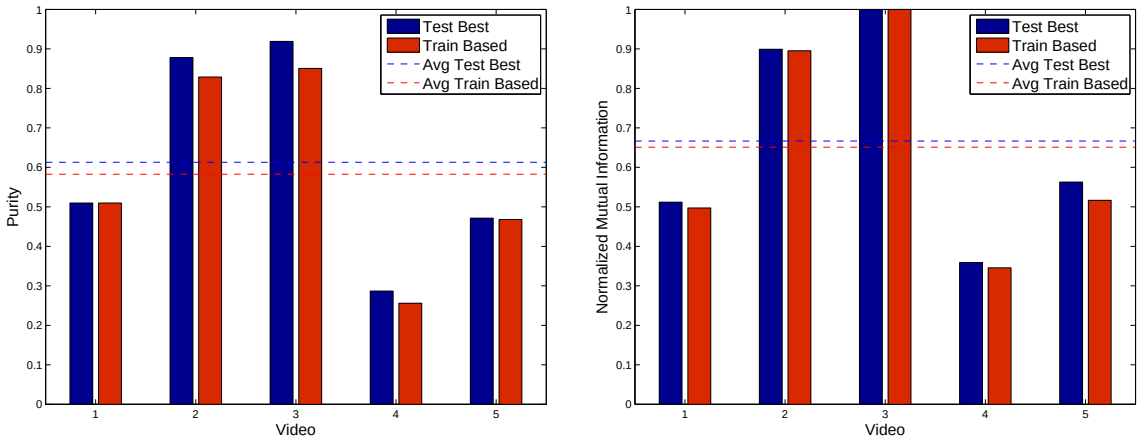


Figure 5.6: Results of the clustering algorithm 2 on the five test videos in terms of Purity (left) and Normalized Mutual Information (right). For both measures the results obtained by selecting the parameters α, β and θ_c from the training set of 20 videos (in red) closely match the best possible performances for the test videos (in blue). On average (dashed lines), Purity is 0.58 versus the optimal 0.61, while NMI is 0.65 versus the optimal 0.66.

Figure 5.6 shows the performances of the online clustering algorithm 2 on the five test videos. The algorithm depends on three parameters, α, β and θ_c, which were set via simple grid search from a separate training set of 20 videos to take the values 0.1, 1 and 0.05, respectively.

We compared the Purity and NMI results obtained by using the parameters α, β and θ_c selected from the training set (in red) to the best possible clustering performances for each the test videos (in blue), given the regions detected by the previous processing steps. On average (dashed lines), Purity is 0.58 versus the optimal 0.61, while NMI is 0.65 versus the optimal 0.66. These results confirm that the three parameters values chosen from the training set generalize well.

Table 5.2 quantitatively summarizes the quality of the diagram indexes for the five test videos in terms of Precision and Recall, while Figure 5.7 provides a more qualitative view of the clustering results as well as the created indexes.

We evaluate the net performance of our full diagram index generation pipeline in terms by comparing the ground truth unique diagrams present in a video (manually extracted from the electronic copies of the slides, presented in the central column of Figure 5.7) with the diagram index of icons automatically generated from the regions detected by our system, clustered, selected based on size

Video	Duration (minutes)	#frames	# Ground Truth Diagrams	# Detected Diagrams	Precision	Recall	Processing Time (sec)
1	18	1078	4	9	0.44	1	123
2	77	4670	33	84	0.33	0.73	576
3	24	1469	12	15	0.67	0.67	187
4	37	2177	25	15	0.73	0.44	210
5	16	975	7	11	0.45	0.71	120
Total	172	10369	81	134	0.52	0.71	1216

Table 5.2: Experiments Videos details and accuracy of the automatically generated diagram index.

and finally automatically white balanced (right column of Figure 5.7)). The left column of the Figure shows the ground truth (in blue) and detected (in red) clusters along the temporal scale. Each row represents a different cluster, which results in one unique entry in the final diagram index. In this representation, optimal clustering would be perfect overlap between blue and red strings, as happens for example in clusters 2, 7 and 10 in the graph for video 3.

From Table 5.2 emerges that our system in general tends to generate more entries in the diagram index than there actually exist in the ground truth. This is due to over-clustering and results in a Precision rate of 0.52. The phenomenon appears clearly in the right column of Figure 5.7, which represents the automatically generated index. In every video, there are icons from a same ground truth diagram are repeated multiple times. The magenta dotted lines group together those icons that were selected from different clusters, but should be merged into a single one, thus are considered false positives. In the same column, in red are highlighted another source of false positives, namely regions that do not represent a diagram, but the system erroneously considers to do.

On the other hand, the number of false negatives, that is, ground truth diagrams which are not represented in our final index, is limited. They are highlighted in magenta in the central column of Figure 5.7. Hence the Recall rate is much better than the Precision one and achieves a level of 0.71. The false negatives rate is problematic only for video 4, which in fact has a Recall of only 0.44. In this case, the failure is due mostly to the low region detection rate, rather than the clustering. In fact, looking at the temporal layout of the clusters for this specific video in the left column of Figure 5.7, we notice that there are multiple ground truth diagrams, temporally located toward the end of the

video, which are never detected by the algorithm (top right corner). This is the only case where the suboptimal performance of the detection algorithm heavily conditions the final index result.

A peculiar case is registered in video 3, in which one of the diagrams found in the electronic copies of the slides (highlighted in blue in the appropriate row of the central column in Figure 5.7) never actually appears in the video footage of the recorded lecture. On the other hand, a diagram which is not part of the deck of slides is duly registered by our system (highlighted in green in the appropriate row of the right column of the Figure). This happened because during the presentation an extra slide, not part of the original deck and containing the mentioned diagram, was shown and hence video-recorded.

Overall, 57 out of 81 unique diagrams were properly identified, selected and color-corrected (even if with some redundancies) from the three hours of videos containing five different presentations.

Finally, it must be noted that the computational complexity of our diagram generation algorithm is very low. On average, the system processes one hour of video in little over 7 minutes on a single core 2.33GHz machine, which is several times faster than real time. Therefore the system can be easily employed to process large scale collections.

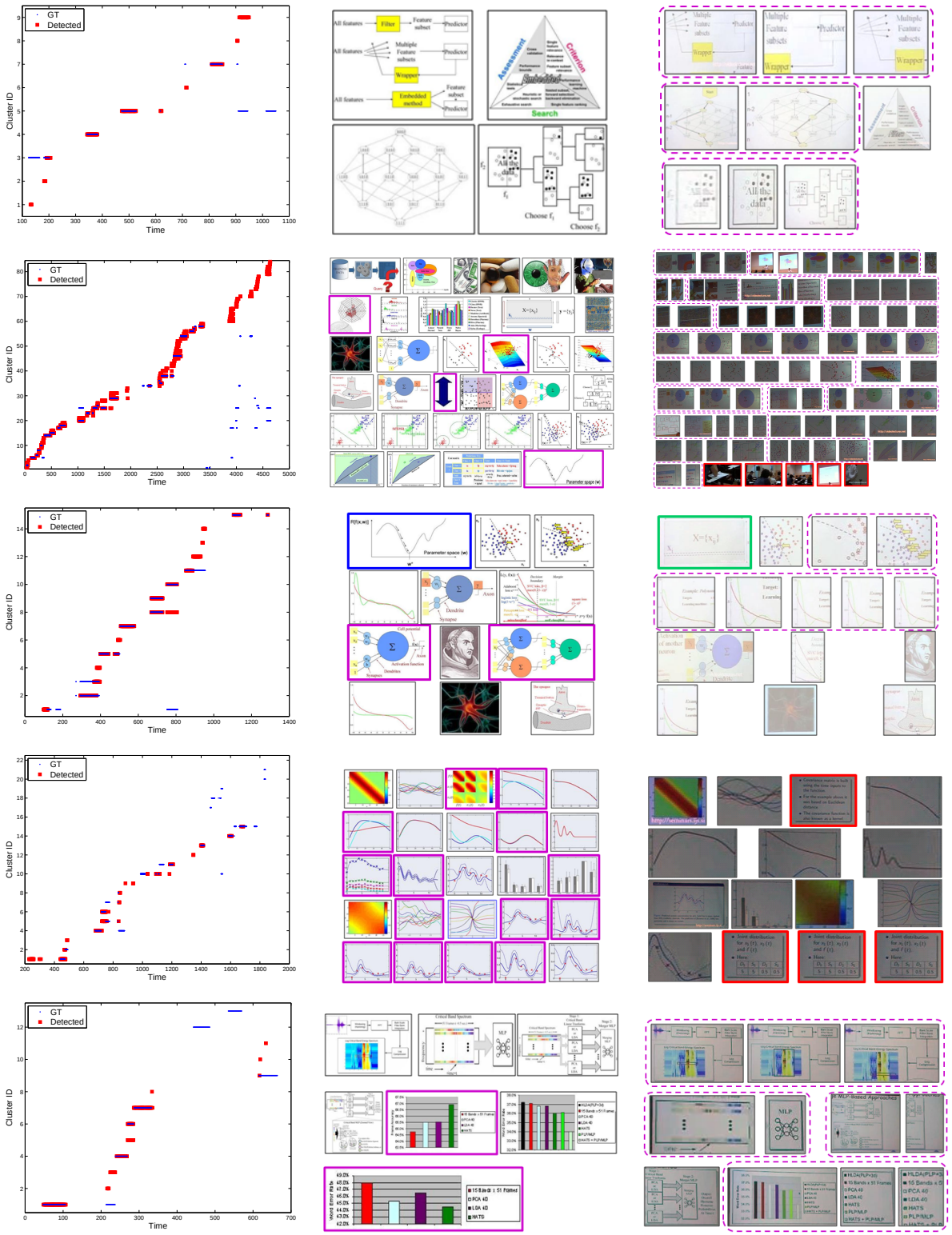


Figure 5.7: Results of the clustering and automatically generated visual indexes on the five test videos (one video per row). Left: temporal scale with ground truth (in blue) and detected (in red) diagrams clusters. Middle: visual index of ground truth clusters. False negatives are highlighted in magenta. Right: automatically generated visual index, after color correction. False positives are highlighted in red.

Chapter 6

Conclusions

Presentation videos represent an interesting and challenging class of unstructured, "wild" videos. Since presentations are by definition a way of conveying an intelligible message, this class of videos represent the ideal setting to develop indexes which are semantically richer than standard keyframe or tag based ones.

6.1 Contributions

In this thesis we have introduced four novel semantic cues to index and cross-reference presentation videos in a multi-modal manner: text, speaker faces, graphics, and mosaics. We have presented algorithms to generate these indexes in the particularly challenging "unsourced" domain, in which no other source of information is available besides the video itself. Our indexes allow to perform search not only inside video collection, but also within individual clips. Furthermore, they provide a multi-modal summary of a video.

We have developed a textual index of words recognized from the slides projected in the video. We have introduced a novel binarization algorithm, Local Adaptive Otsu (LAO), to explicitly deal with the low quality of the video and detected scene text regions. We demonstrated the usefulness of the LAO algorithm by almost doubling the character recognition rate (up to 74%) of the Tesseract open source OCR engine on 8 presentations spanning over 1 hour and 45 minutes of video. We could then use the recognized words from the projected slides not only to build the textual index of the videos, but also to semantically segment them into shots.

We have presented a combination of a keyframe sampling method, proportional to estimated PTZ camera motion, and of a local-features based image stitching algorithm, in order to build an index of mosaics for the semantic video shots.

In order to establish the criteria for building of the visual indexes, in particular the face and diagram-based ones, we have adopted a user centric perspective. To this end, we have employed Amazon Mechanical Turk HITs as surveys to gauge user preferences in terms of the visual appearance of the icons in each of the two indexes. The results have suggested that icons in a speaker face index should present a three-quarter, head and shoulder view of a person. For a diagram index, users have shown to prefer a white-balanced, color corrected image that covers as much as possible the whole area of the diagram, even if its details are not as clear as one of its sub-parts.

Within a standard processing pipeline to extract faces of speakers in videos, we have introduced a tracking algorithm which integrates a generic object/face tracker as a noisy prediction in a simplified version of a Kalman filter named *K-Track*, which uses object/face detections as noisy observations. *K-Track* is used to mitigate the drifting effect, which typically affects appearance based tracking algorithms. Using our tracking framework we have registered up to 5.7% relative improvement in tracking precision with respect to a state of the art multiple instance learning tracker [Babenko *et al.*, 2009] on 3 unstructured presentation videos with a total of more than a quarter million frames.

We have presented the use of three quality measures, namely resolution, amount of skin, and pose, in order to simultaneously perform two selection tasks needed within the face indexing framework. The first selection process is necessary for tracks matching, in order to avoid the computational burden of comparing every pair of faces in each track. The second selection is needed for choosing a unique speaker face icon to be used in the final index, with the goal of closely matching the human preferences recorded through the Mechanical Turk surveys. We were able to automatically build a face index from three unstructured presentation videos of approximately 45 minutes each, which showed 87% accordance (51 out of 58 speakers) with such human preferences.

Finally, we have introduced the first presentation video index based on diagrams. In order to generate such index, we have described a processing pipeline consisting of four blocks: slide detection, diagram region localization, diagram clustering and diagram icon selection. Diagram regions are localized using standard region detection methods, based on edge density and geometric

constraints, within preselected frames. Such frames are identified as showing a projected slide that occupies the majority of the camera field of view, by estimating the amount of color shift toward a high or low temperature. An online clustering algorithm, which combines visual and temporal similarity, is employed to group together regions representing the same diagram. Finally, one image is selected to represent the diagram cluster based on resolution, and then color corrected with the automatic white balancing Grey-world algorithm [G. and Buchsbaum, 1980]. We have demonstrated that our system is capable, completely in automatic and without any reference source besides the video itself, of generating indexes of high resolution, white balanced diagrams from five videos of approximately three hours, with 71% accuracy (57 out of 81 diagrams).

6.2 Future Work

We plan to further extend the work of this thesis in two main directions. The first consists in integrating our indexes into an augmented video browser. This will give us the platform to perform user studies to assess the usability and importance of the indexes for end users, as well as determining which type of visualization is most suitable for a specific task. The second directions regards a further semantic analysis of the visual indexes, in particular the diagram and face ones.

In the following we discuss in detail the possible extensions to our work.

- **Integration with Video Browser** We plan to integrate our indexes into an augmented video browser. In this context, the VastMM [Haubold and Kender, 2007] consists in a natural choice for our semantic indexes, especially if extended to be a web application. This will give us the platform to perform user studies to assess the usability and importance of the indexes for end users, by comparing their performance to other standard indexes (for example, playback) in terms of time and accuracy for search and retrieval of useful segments within the videos.

  One type of user study we could consist in providing the system as an assistive tool for students to review class material and prepare for exams. We could study the use of the browser by regular students during the week before finals, assessing how much the given indexes are employed. It would be interesting to compare the grades of students who used our tool to the ones of those who had only access to the streaming video and/or the regular class material (electronic copies of the slides, web pages) to verify its usefulness. We also plan

to record which type of index (or combination of indexes) were used the most, so to verify which controls are actually more useful. Finally, we also plan to collect feedback from users regarding their subjective experience in using the tool, to verify the likeability of the various indexes and controls.

A second batch of user studies will be performed to evaluate speed and accuracy of retrieval of semantic concepts given the built indexes. We will ask users to retrieve segments in which a particular concept is explained or slide is presented, possibly by a specific presenter, and measure the duration and completion rate of the task given different configurations of indexes available: pure video stream, keyframe based representation, mosaic representation, textual index, graphics indexes. This type of user studies could be conducted on large scale using Amazon Mechanical Turk.

- **Diagram Classification** We plan to provide a semantically richer representation of the diagrams found in presentation videos by classifying them into diagram categories, i.e. line plots, bar charts, tables, etc. Existing work on diagram classification [Prasad *et al.*, 2007; Savva *et al.*, 2011] is limited to had hoc, small scale datasets. By leveraging the growing quantity of presentation videos available on the web and our diagram region extraction pipeline, we could discover diagram categories through unsupervised clustering. We could then apply category labels to diagrams detected in new videos.

- **Speaker Identification** Our proposed speaker face indexing process is completely anonymous, since it simply retrieves the faces of the people presenting in a video, without any information about their identity. It could be interesting to associate faces to their names. This could be done for example by matching the faces detected in the video to images of professors takes from department websites of universities or researchers from a company website. Once a speaker has been identified, we could provide a richer information in the visual index, for example the link to a professor's webpage.

- **Enhancement of Semantic Elements in Mosaics** Mosaics are a meaningful representation of semantic content that situates it into its video context and therefore goes beyond the usual keyframe or video playback based displays. Some systems try to offer a clearer visualization by simply presenting a copy of the electronic slides besides the window with the video

playback or directly enhance the video by superimposing the slides on the proper regions. Such systems introduce a split-attention effect [Friedland and Rojas, 2008], where two areas of the screen (one with the slides, another with a small video of the presenter) are competing for the viewers attention. Our mosaic index overcomes such limitation, since all the information is integrated in a single representation. Furthermore, we plan to enhance the mosaics by highlighting, enlarging, crispening and superimposing meaningful text and diagrams on them. For example, when the user will hover over the slide portion of a mosaic, a window appears showing the slide with the recognized text overlaid and clearly printed, as well as the diagrams sharpened and enlarged for better visualization. By clicking on different locations on the mosaic, the user will also be able to control the video playback, as the portion of the shot containing the corresponding keyframe will be played.

- **Extension to other domains**. While the work presented in this thesis is mostly focused on the specific domain of presentation videos, it would be interesting to apply these semantic indexing capabilities to other domains, for example generic consumer videos. In particular the face and text indexes could be employed directly or as features for video classification or retrieval systems. The words recovered by the text indexing module from scene or overlaid text could be collected into a bag of words histogram representation of a given video. The face index construction model could be instead exploited to extract face trajectories, or simple representations describing the number of people present in a clip and their co-occurrence in different segments of a video.

Appendix A

Steady State Kalman Filter Derivation Details

In this Appendix we provide some details of the mathematical formulation of the proposed steady state Kalman filter framework. First, we provide the standard Kalman Filter Equations to be used as reference to their counterparts presented in Chapter 4.2.2. Then we describe in detail the derivation of Equation 4.13 from the same Chapter.

A.1 General Kalman Filter Framework

The Kalman filter tries to estimate the state $\boldsymbol{x} \in \mathbb{R}^n$ of a process governed by the differential equation

$$\boldsymbol{x}_t = A\boldsymbol{x}_{t-1} + B\boldsymbol{u}_{t-1} + \boldsymbol{w}_{t-1} \tag{A.1}$$

where the random variable $\boldsymbol{u} \in \mathbb{R}^l$ represents an optional control input, the random variable $\boldsymbol{w}$ represents the process noise, and the matrix A relates the state at the current time $\boldsymbol{x}_t$ to the previous time step $\boldsymbol{x}_{t-1}$ in the absence of a driving function or process noise. The measurement $\boldsymbol{x}_t^O \in \mathbb{R}^m$ of the state is represented as

$$\boldsymbol{x}_t^O = H\boldsymbol{x}_t + \boldsymbol{v}_t \tag{A.2}$$

The random variable $\boldsymbol{v}$ represents measurement noise. Both $\boldsymbol{w}_t$ and $\boldsymbol{v}_t$ are assumed to be white, independent of each other, and have normal probability distributions

$$p(\boldsymbol{w}) \sim N(0, Q) \quad , \quad p(\boldsymbol{v}) \sim N(0, R) \tag{A.3}$$

Q and R represent the process noise and the measurement noise covariance matrices, respectively. The Kalman filter state estimation at step t is based on two estimates: the *a priori* state estimate, given knowledge of the process prior to step t, defined as $\tilde{\boldsymbol{x}}_t \in \mathbb{R}^n$, and the *a posteriori* state estimate, given the measurement $\boldsymbol{x}_t^O$, defined as $\hat{\boldsymbol{x}}_t \in \mathbb{R}^n$. The *a priori* and *a posteriori* estimate errors and their covariances at step t can be defined as

$$\tilde{\boldsymbol{e}}_t = \boldsymbol{x}_t - \tilde{\boldsymbol{x}}_t \quad , \quad \hat{\boldsymbol{e}}_t = \boldsymbol{x}_t - \hat{\boldsymbol{x}}_t \tag{A.4}$$

$$\tilde{P}_t = E\left[\tilde{\boldsymbol{e}}_t \tilde{\boldsymbol{e}}_t^T\right] \quad , \quad \hat{P}_t = E\left[\hat{\boldsymbol{e}}_t \hat{\boldsymbol{e}}_t^T\right] \tag{A.5}$$

Under these assumptions, it can be shown that the discrete Kalman filter time update equations result in

$$\tilde{\boldsymbol{x}}_t = A\hat{\boldsymbol{x}}_{t-1} + B\boldsymbol{u}_{t-1} \tag{A.6}$$

$$\tilde{P}_t = A\hat{P}_{t-1}A^T + Q \tag{A.7}$$

and the measurement update equations result in

$$K_t = \tilde{P}_t H^T \left(H\tilde{P}_t H^T + R\right)^{-1} \tag{A.8}$$

$$\hat{\boldsymbol{x}}_t = \tilde{\boldsymbol{x}}_t + K_t\left(\boldsymbol{x}_t^O - H\tilde{\boldsymbol{x}}_t\right) \tag{A.9}$$

$$\hat{P}_t = \left(I - K_t H\right)\tilde{P}_t \tag{A.10}$$

where K_t is referred to as the Kalman gain.

A.2 Derivation of Equation 4.13

We start from Equations

$$\tilde{P} = A\hat{P}A^T + G\sigma_a^2 G^T \tag{A.11}$$

$$\hat{P} = \left(I - KH\right)\tilde{P} \tag{A.12}$$

Following [Ramachandra, 2000], we can combine Equations A.11 and A.12, and obtain notations for $\hat{P}$, $\tilde{P}$ and K, as follows. Equation A.12 can be rewritten as

$$\hat{P}^{-1} = \tilde{P}^{-1} + H^T R^{-1} H \tag{A.13}$$

Combining Equations A.11 and A.13 we obtain

$$\hat{P} - G\sigma_a^2 G^T = A(\hat{P}^{-1} + H^T R^{-1} H)^{-1} A^T \tag{A.14}$$

If we define $\hat{P} = \begin{bmatrix} p_1 & p_2 \\ p_2 & p_3 \end{bmatrix}$, we can rewrite Equation A.14 as

$$\begin{bmatrix} p_1 - \frac{\sigma_a^2}{4} & p_2 - \frac{\sigma_a^2}{2} \\ p_2 - \frac{\sigma_a^2}{2} & p_3 - \sigma_a^2 \end{bmatrix} = \frac{1}{1 + \frac{p_1}{\sigma_o^2}} \begin{bmatrix} p_1 + 2p_2 + p_3 + \frac{\Delta}{\sigma_o^2} & p_2 + p_3 + \frac{\Delta}{\sigma_o^2} \\ p_2 + p_3 + \frac{\Delta}{\sigma_o^2} & p_3 + \frac{\Delta}{\sigma_o^2} \end{bmatrix} \tag{A.15}$$

with $\Delta = p_1 p_3 - {p_2}^2$ being the determinant of $\tilde{P}$. From Equation A.15 we obtain the following system of Equations:

$$\begin{cases} \left(p_1 - \frac{\sigma_a^2}{4}\right)\left(1 + \frac{p_1}{\sigma_o^2}\right) = p_1 + 2p_2 + p_3 + \frac{\Delta}{\sigma_o^2} \\ \left(p_2 - \frac{\sigma_a^2}{2}\right)\left(1 + \frac{p_1}{\sigma_o^2}\right) = p_2 + p_3 + \frac{\Delta}{\sigma_o^2} \\ \left(p_3 - \sigma_a^2\right)\left(1 + \frac{p_1}{\sigma_o^2}\right) = p_3 + \frac{\Delta}{\sigma_o^2} \end{cases} \tag{A.16}$$

which, solving and substituting, produce the following notations of $\hat{P}$, $\tilde{P}$ and K

$$\tilde{P} = \begin{bmatrix} \frac{\sigma_o^2 d(d+1)^2}{r^2} & \frac{\sigma_o \sigma_a (d+1)^2}{2r} \\ \frac{\sigma_o \sigma_a (d+1)^2}{2r} & \frac{\sigma_a^2 (d+1)}{2} \end{bmatrix}, \quad \hat{P} = \begin{bmatrix} \frac{\sigma_o^2 d(d-1)^2}{r^2} & \frac{\sigma_o \sigma_a (d-1)^2}{2r} \\ \frac{\sigma_o \sigma_a (d-1)^2}{2r} & \frac{\sigma_a^2 (d-1)}{2} \end{bmatrix}$$

$$K = \begin{bmatrix} \frac{d(d-1)^2}{r^2} \\ \frac{2(d-1)^2}{r^2} \end{bmatrix} \tag{A.17}$$

with $r = \frac{4\sigma_o}{\sigma_a}$ and $d = \sqrt{1 + 2r}$.

Appendix B

Mechanical Turk Experiments Details

In this Appendix we discuss in detail the setup and results of the Amazon Mechanical Turk experiments that we conducted for two reasons. The first was to obtain human preferences for the appearance of the icons to be employed in our visual indexes, fro both the faces and diagrams. The second was to annotate the locations of diagrams inside video frames, so to obtain ground truth against which to evaluated our algorithms.

B.1 Faces Index Preferences

This AMT experiment was designed as a user study to evaluate the preferences of people in terms of how the face of a speaker should be presented in the visual index of a video, with respect to two criteria: head pose and context.

The experimental setup was the following. Each HIT presented the user with two type of face views of a speaker from our presentation videos: one type showing only the face, the other with a head and shoulder view. For each type, we presented 5 poses: -90^o, -45^o, 0^o (frontal), $+45^o$, $+90^o$. Figure B.1 shows the appearance of the HIT, in which the views of a person's face are randomized. The full set of images and speakers used in the experiment is reported in Figure B.2.

In order to avoid preferences based on a specific speaker or a specific ordering in which the views were shown, we conducted experiments with 15 different speakers and 3 random views orderings. Furthermore, we requested 35 different workers to complete each combination of speaker and views order, therefore amounting a total of 1575 unique HITs.

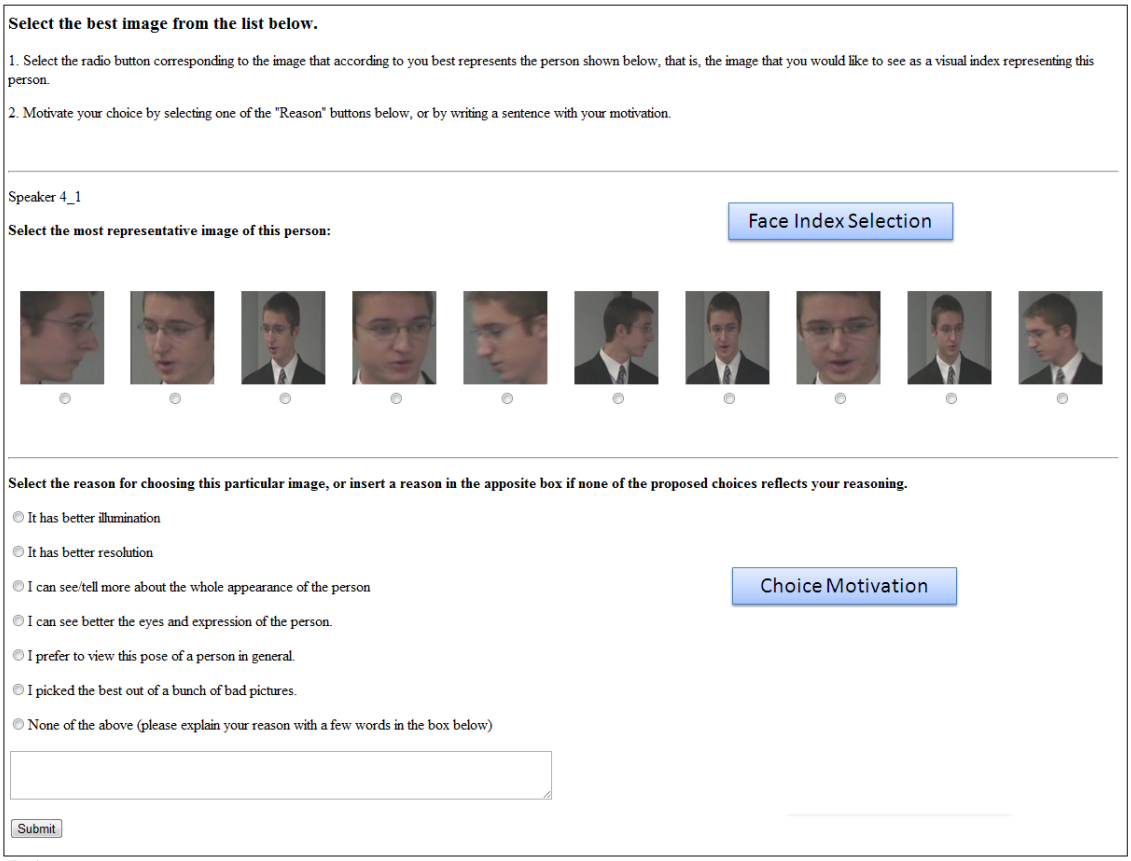
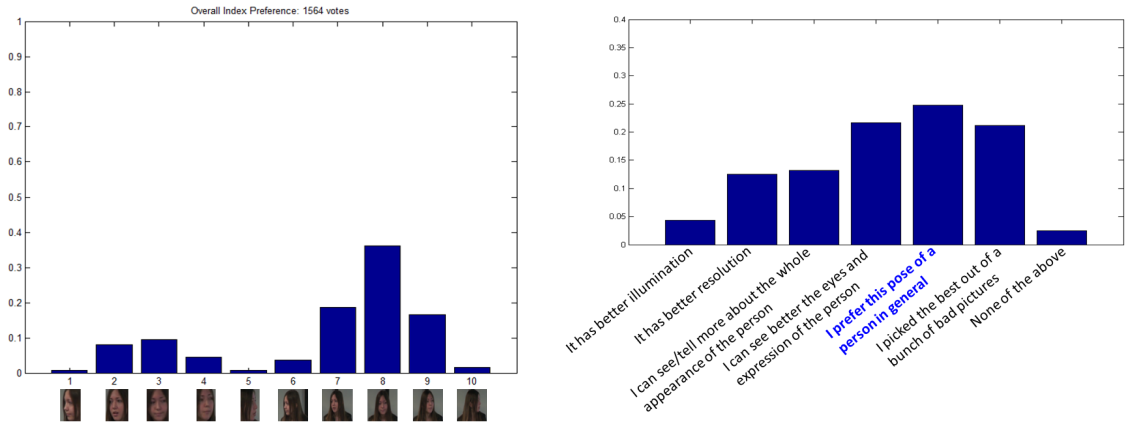


Figure B.1: Face Quality Selection HIT Example.

We also requested the workers to provide a motivation for their choices, giving the following set of predefined choices (lower part of Figure B.1), with the possibility to leave personal comments.

1. It has better illumination
2. It has better resolution
3. I can see/tell more about the whole appearance of the person
4. I can see better the eyes and expression of the person
5. I prefer this pose of a person in general
6. I picked the best out of a bunch of bad pictures
7. None of the above

Figure B.2: Full dataset of 15 speakers used in the experiment. For each speaker the workers had to select one out of ten different views.



(a) (b)

Figure B.3: Face Quality Selection overall results (left) and motivations (right).

All HITs were completed within two and a half hours, with an average time per assignment of 19 seconds. 69 unique workers participated to the experiment. Out of the possible 1575 votes, 11 were invalid, leaving a total of 1564 valid entries. The results, reported in Figure B.3 (a) showed a strong preference for a head and shoulder rather than face only view (76% versus 24%). The single most selected pose was the frontal one (45%). However, the combination of the left and right 3/4 poses amounted to 47% of the votes.

Looking at the distribution of choices for the individual speakers in Figure B.4, we notice that only for one out of 15 speakers the preference was given to a face only view (circle in red). For two out of 15 speakers the preference was given directly to a three quarter view, while if we combine Left and Right three quarters view, this optin was chosen for five out of 15 speakers (circles in blue).

Some considerations can be made about the motivations behind the choice of a particular view, which are detailed in Figures B.3 (b) and B.5. First, the reason behind most of the choices was a general preference for the particular view of a person (reason 5), which reflects exactly the goal of the user study. Second, as expected, the choice of a face-only view was mostly motivated by reason 4, for which it is possible to better discern the eyes and expression of a person. It interesting to notice that

The workers left a total of 48 comments, 36 of which expanding on the “None of the above” reason for a particular selection. Some comments were repeated, and it is interesting to see how most of the comments express remarks on the physical appearance or attractiveness of the person,

rather than its view angle or pose. The full list is the following: *looks smart, his smile is good, he is expressing some expression, his reaction is nice, it is the most attractive picture and takes in the person's whole face, it looks good, nice hair style*.

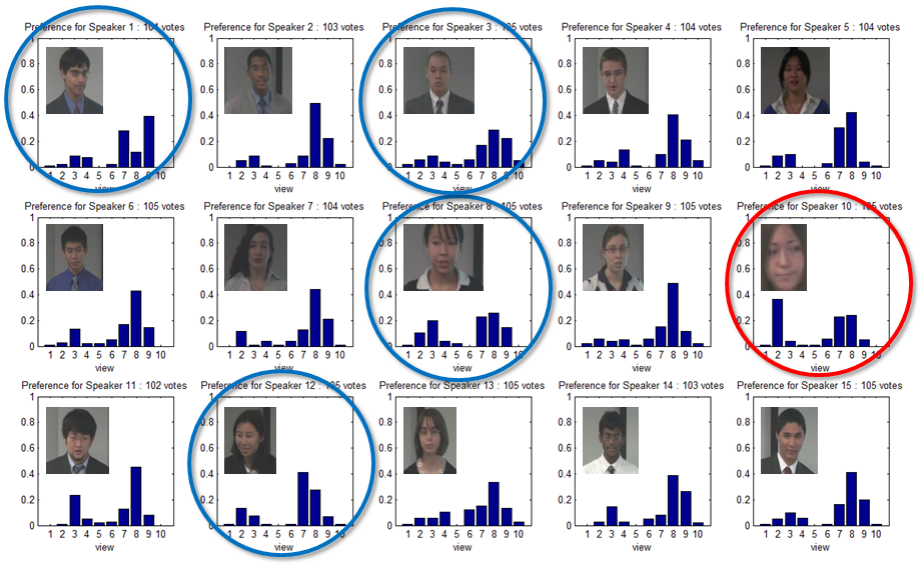


Figure B.4: Face Quality Selection results, per individual.

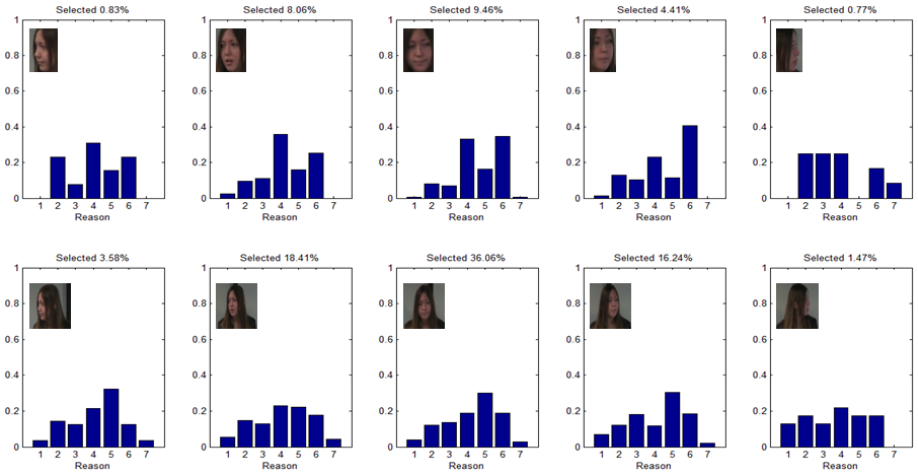


Figure B.5: Face Quality Selection motivations.

B.2 Diagrams Index Preferences

B.2.1 White Balance

The human eye possesses the *color constancy* ability to cope with different lighting conditions and adjust for different colors of the light source [Gijsenij *et al.*, 2011]. Camera sensors, on the other hand, do not have such property and since the graphics regions are extracted from videos recording projected slides, the illumination tends to shift toward high temperatures. Therefore a white balancing algorithm should be used to restore color constancy and remove the artefacts introduced by the recording process. The goal of this user study is to capture user preferences in terms of the type/amount of white balancing to adopt.

HIT Design

There exist a large number of algorithms to perform automatic white balancing and computational color constancy (for a survey, see [Gijsenij *et al.*, 2011]). We chose a few simple and well known ones to test and present to the users. For all methods but *Original* and *Retinex* we adopt the diagonal transform model [von Kries, 1970], which achieves the color correction from the pixels in the image with unknown light source I_u to the color corrected image I_c through a multiplication with a diagonal matrix D. The white balancing equation to recover a given pixel $P_u = (R_u, G_u, B_u)^T$ becomes then

$$\begin{pmatrix} R_c \\ G_c \\ B_c \end{pmatrix} = \begin{pmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{pmatrix} \begin{pmatrix} R_u \\ G_u \\ B_u \end{pmatrix} \tag{B.1}$$

Different methods employ different assumptions to derive the values of the diagonal elements of D. The following is a list of the methods that we tested, which are mostly standard.

- **Original**: the graphic region as detected in the video, without any post-processing/color correction. In this case $D = I$, I being the identity matrix.

- **Grey-world** [G. and Buchsbaum, 1980]: it assumes that the average intensities of the red, green and blue channels should be equal. Therefore the resulting values of D become

  $$d_1 = \overline{G_u}/\overline{R_u}, d_2 = 1, d_3 = \overline{G_u}/\overline{B_u}$$

  where $\overline{C_u}$ is the average intensity in channel C of I_u.

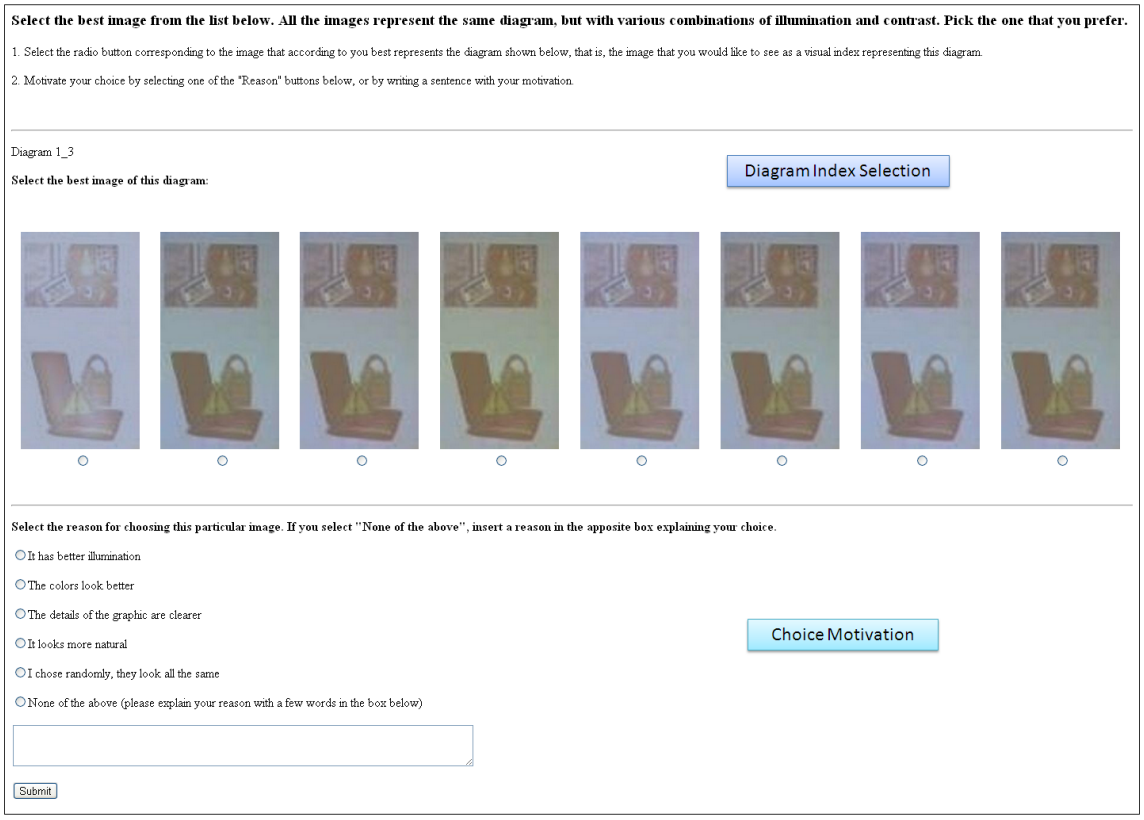


Figure B.6: Layout of the Amazon Mechanical Turk HIT used to estimate user preferences for the graphics visual index in terms of White Balance. The HIT consists of two parts: one to *select* the preferred representation of a same diagram (top) and the other to *motivate* a given choice (bottom).

- **maxRGB**[Funt and Shi, 2010]: it follows the *white-patch* assumption, which states that the maximum response in the RGB channels is caused by perfect reflectance. Under such assumption, the values of D can be computed as follows:

$$d_1 = max(G_u)/max(R_u), d_2 = 1, d_3 = max(G_u)/max(B_u)$$

- **Single Channel Adjustments**: from the slide detection algorithm used in Section 5.1.1 we estimated which color channel (between red and blue) was saturated due to the light shift generated by the slide projection and camera recording system. We tried to employ that information to color correct only the saturated channel instead both red and blue like in the previous approaches. Therefore, according to Equation 5.1, we set only one between d_1 or d_3 depending on which is maximum between $\overline{R_{SAT}}$ and $\overline{B_{SAT}}$. To set the value in D we

used either the *Grey-world* or the *maxRGB* equations, thus obtaining the *Grey-world Single Channel* and *maxRGB Single Channel* white balancing algorithms. For *Grey-world Single Channel*, D is computed as follows

$$d_1 = \begin{cases} \overline{G_u}/\overline{R_u} & if \overline{R_{SAT}} > \overline{B_{SAT}} \\ 1 & otherwhise \end{cases}$$

$$d_2 = 1$$

$$d_3 = \begin{cases} 1 & if \overline{R_{SAT}} > \overline{B_{SAT}} \\ \overline{G_u}/\overline{B_u} & otherwhise \end{cases}$$

The values for *maxRGB Single Channel* are computed similarly.

- **Retinex** [Land and McCann, 1971]: we used the McCann99 Retinex [Funt *et al.*, 2000] implementation. We tested the algorithm with $N = 1, 10$ and 100 iterations. [Funt and Shi, 2010] noted that *maxRGB* is a special and extremely limited case of *Retinex*. In particular, it corresponds to McCann99 Retinex when the number of iterations is infinite.

Once the workers selected a preferred icon, we also asked them to specify the reason behind each choice, giving the following pre-specified options or allowing them to explain their motivation in their own words:

1. It has better illumination
2. The colors look better
3. The details of the graphic are clearer
4. It looks more natural
5. I chose randomly, they look all the same
6. None of the above (please explain your reason with a few words in the box below)

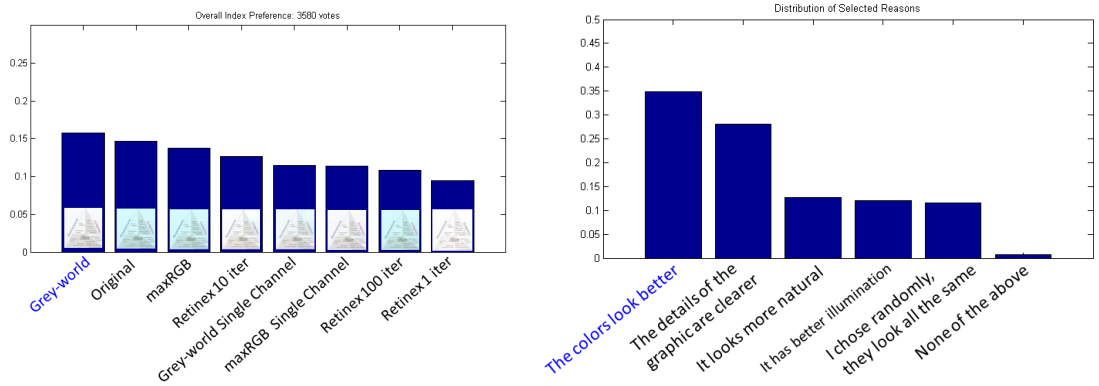
Results

Of the 3600 HITs, only 20 were not completed successfully, either because the worker did not select any image (in 20 cases) or because he did not provide any explanation of his choice (10 cases), as reported in Table B.1.

Experiment	#Diag.	#Choices	#Perms	#Assign. per HIT	Total #HITs	Avg HIT Time(secs.)	#Unique Workers	#Missing Votes	#Missing Reason
AWB	40	8	3	30	3600	18.7	84	20	10
Resolution	32	2	2	50	3200	15.8	101	91	43

Table B.1: Summary of AMT Experiments on user preferences for diagram index icons in terms of white balance correction and resolution.

Despite the distribution of preferences reported in Figure B.7(a) being more uniform than expected, the simple *Grey-world* algorithm emerges as the most preferred choice. It is also worth noting that the preferences for the Retinex algorithm peak at 10 iterations, finding a compromise between too localized and too global restoration ranges. The choices made by the users were motivated mainly by the appearance of the colors (see distribution of reasons in Figure B.7(b)). Since we did not reveal how the images were obtained or how the original diagram in the slide looked like, it becomes clearer that some users found the lighting distortion in the *Original* version of the diagram to be visually appealing and considered it to be a pleasant part of the diagram rather than a noisy artefact introduced by the recording process. Nonetheless, following the results of the user study we apply the *Grey-world* white balancing correction to the selected diagram index icons, as explained in Section 5.2.3.



(a) (b)

Figure B.7: Results of user preferences for automatic white balance/color correction algorithms, sorted by preference. (a) The most popular selection was the Grey-world with corrections on both the R and B channels. (b) Distribution of motivations for the given choices. The workers chose predominantly based on the appearance of the colors.

B.2.2 Resolution

HIT Design

We found that after the diagram detection and clustering processes described in the previous Sections, mainly two types of diagram regions were recurring in each cluster: one containing the full view of the diagram (referred to as *Full View*), the other a zoomed in view of a detail/part of the diagram itself (referred to as *Detail*). The second view is due mostly to camera motion (in particular zoom in) or diagram detection errors. Even if one intuitively would think that a full view of a diagram would provide a better index icon, that might not always be the case. In fact, due to the low quality of the recorded videos and the distance from the recording camera to the screen where the slides are projected, sometimes the detected full view of a diagram does not have enough resolution to fully discern the meaning of the diagram which is included in some of its details. The zoomed in version, on the other hand, can provide a clear vision of such details. It must be considered that in the final visual index, we will have to . For example in Figure 5.3 the text in the graph is not readable in its *Full View* version, but part of it is readable in the zoomed in one.

In order to determine which view is preferred by the users, we ran a test in which workers had to choose between the two views for 32 diagrams. We resized both views to a canonical size (the average of their dimensions), since for practical reasons the final visual index for the video will have only fixed size for every item, in the same fashion as standard image search engines show results in canonical fixed size thumbnails. We showed each image pair in both orders and required 50 unique workers per HIT, thus amounting to a total of 3200 HITs.

We asked the users to also specify the reason behind each choice, giving the following 4 options or specifying their reasoning in case none of the offered options matched their choice motivation:

1. It provides more information
2. I can see the details better
3. It is more complete
4. I chose randomly, they look all the same
5. None of the above (please explain your reason with a few words in the box below)

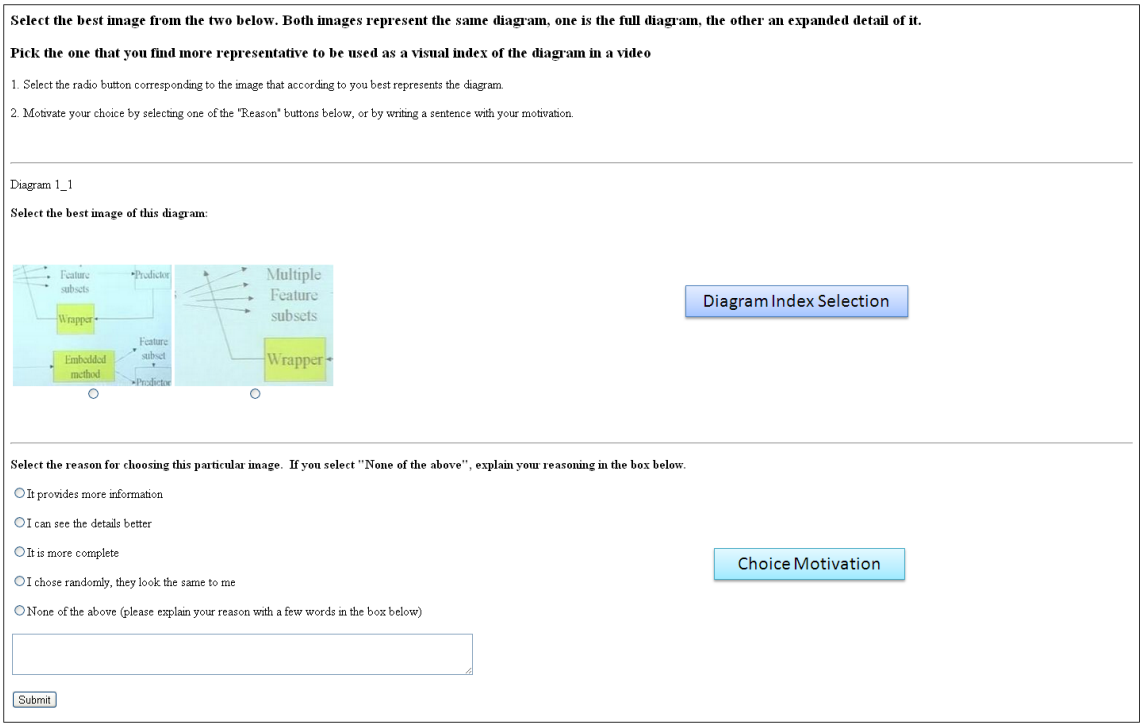


Figure B.8: Layout of the Amazon Mechanical Turk HIT used to estimate user preferences for the graphics visual index in terms of Resolution. The HIT consists of two parts: one to *select* the preferred representation of a same diagram (top) and the other to *motivate* a given choice (bottom).

Results

Of the 3200 HITs, 91 were not completed successfully, and in 43 of those the user did not provide any explanation of his choice (or lack of it, see Table B.1). From the results presented in Figure B.9 clearly emerges that the users prefer the *Full View* of a diagram which was chosen 75% of the time. The motivations for the choices follow intuition: *Full View* was mostly chosen because it provides more information, *Detail* was picked because the users could see the details better in it.

B.3 Diagrams Regions Localization

In order to evaluate the diagram region detection and localization performances of our algorithms we needed ground truth annotations from the frames in the five test videos. Therefore we prepared an Amazon Mechanical Turk region annotation task.

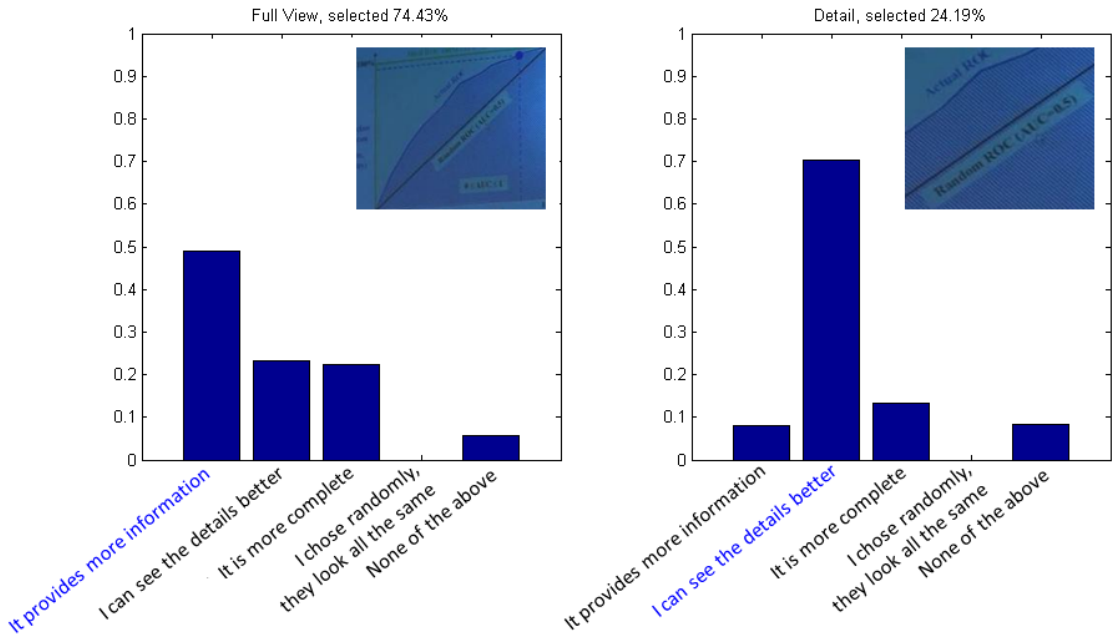


Figure B.9: Results of user preferences for the resolution of the presented diagrams. The users clearly prefer a full view of the diagram (75% of the selections), even if some details might be too small or blurred to discern. The distribution of motivations for the given choices demonstrates how the workers picked a full view representation to see more information, and a blow up of a part of the diagram when interested in more details.

We created HITs for every valid diagram-frame pair in the the five test videos. All the 81 diagrams were considered, while a preprocessing stage was employed to select a subset of frames to annotate. First, we selected only frames where the slide containing the diagram of interest them appeared, according to the VideoLecture.net temporal synchronization information. We further selected from that pool only those frames where the slide occupied the majority of the camera field of view, using the manual annotation employed to evaluate slide detection in Chapter 5.3.1. The selection left 2953 frames, from the overall 10K of the videos. We required three separate workers to annotate the location of the diagram in a frame by drawing a polygonal region around it. The total number of HITs amounted to 15798.

The interface of the HIT is shown in Figure B.10. It is didvided into two parts: the top one contains instructions and a reference image showing the diagram to annotate, to bottom one with

the annotation interface, which we was the flash application developed by Alexander Sorokin[1], adapted for our purposes. To perform the annotation, a worker could click on multiple points in the frame, as the interface drew lines connecting the dots and tracing the contour of the polygon. The annotation was editable at any point, before submission. The worker had also the option of labeling the frame as not containing the given diagram. This option was used because in certain cases the temporal alignment provided by Videolectures.net was not always perfect.

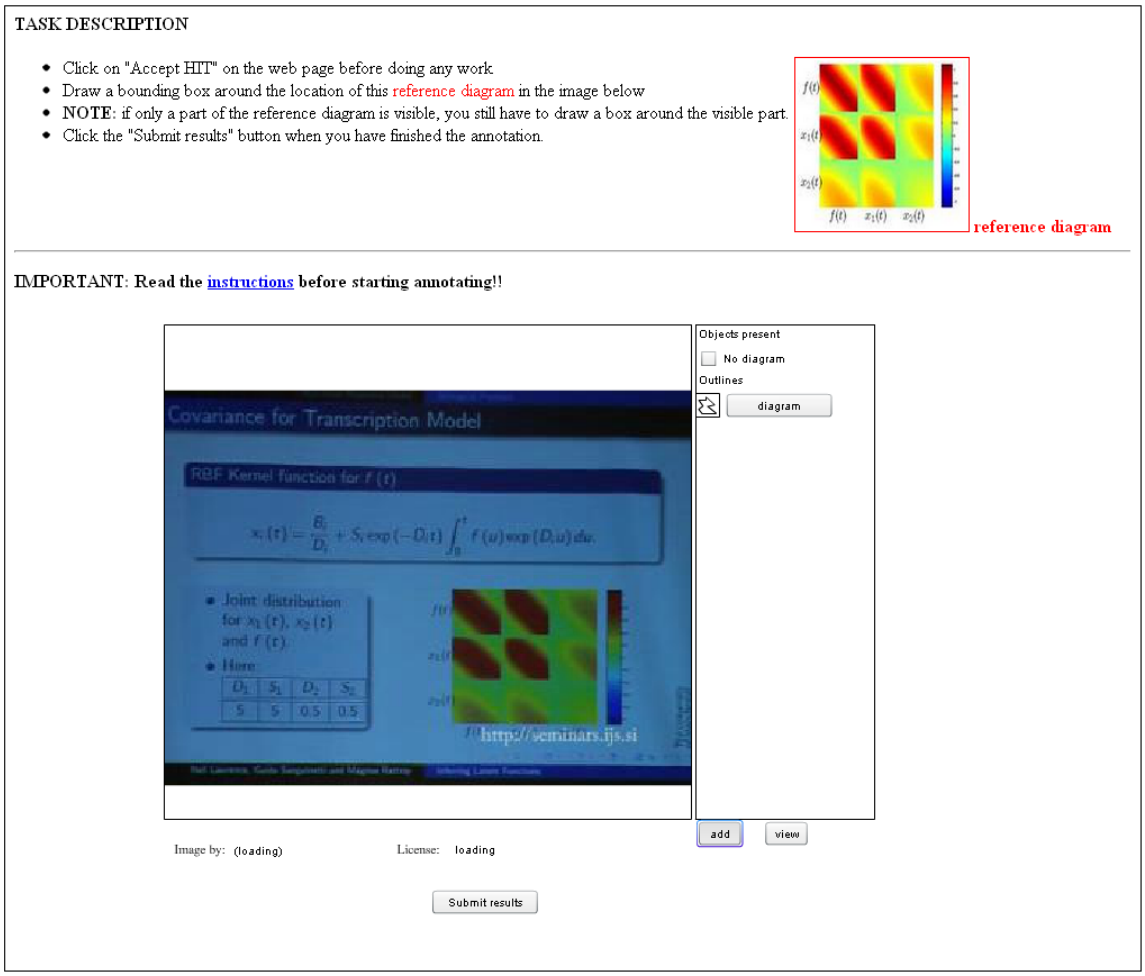
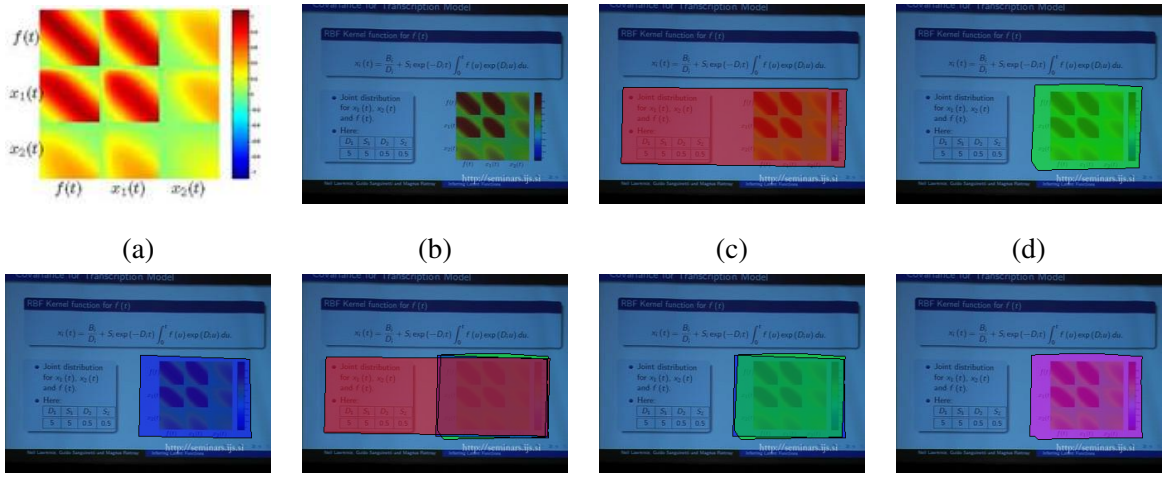


Figure B.10: Layout of the two Amazon Mechanical Turk HIT used to gather ground truth locations of diagrams in video frames. Top: instructions and reference image of the diagram to be annotated. Bottom: annotation interface showing the frame to be annotated.

[1] http://vision.cs.uiuc.edu/annotation/tools/annotation_instructions.html



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

(e) (f) (g) (h)

Figure B.11: Annotations for a given diagram/frame pair. (a) Diagram to locate. (b) Frame to be annotated. (c)-(e) Annotations from workers 1, 2 and 3, respectively. (f) Overlap of the three annotations. (g) Matching annotations. (h) Final result retained as ground truth R_{GT}.

A separate page was also prepared to provide detailed instructions to the workers, directions on how to use the commands in the labeling interface, and most importantly showing showing examples of good and bad annotations (see details in Figure B.12).

The annotation process was much longer in comparision with the previous AMT tasks, as it took four days to be fully completed. The average HIT completion time was 36 seconds.

Once all the annotations were gathered, we performed a manual verification for 1% of the HITs, which showed an overall good quality of the annotations for at least 2 out 3 workers in each frame. We then converted the annotations in each frame to the final ground truth regions in the following way. Each diagram-frame pair had three polygonal regions R_i, $i = 1, 2, 3$ annotated by different workers, as shown in Figure B.11 (c), (d) and (e). We compute the overlap between each pair of annotated regions R_i and R_j, and consider the pair to match if the intersection and the union of the regions respect the following constraint:

$$|R_i \cap R_j| > 0.7 * |R_i \cup R_j|$$

We then selected the final ground truth region R_{GT} based on the number of matching pairs:

- 0 matching pairs: wrong annotation. In this case, we proceeded to manually annotate the

frames. Only 165 HITs presented this result.

- 1 matching pair (i, j): $R_{GT} = |R_i \cup R_j|$
- 2 matching pairs (i, j) and (i, k) : $R_{GT} = R_i$
- 3 matching pairs: $R_{GT} = |R_1 \cup R_2 \cup R_3|$

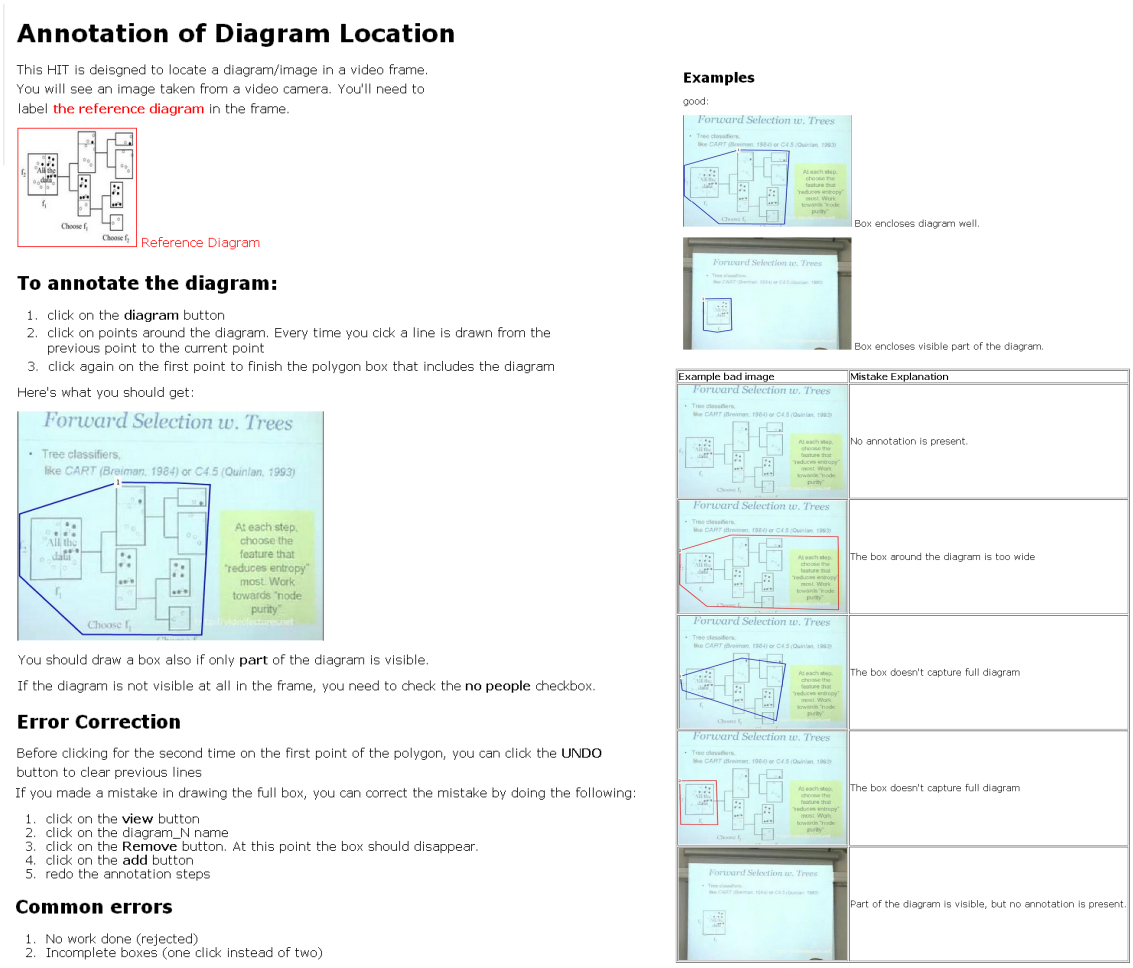


Figure B.12: Instructions for the diagram region localization task. Left: general instructions and interface commands explanation. Right: good and bad annotations examples.

B.4 Lessons Learned

In the following we present a short list of lessons that we have learned from our experience with Amazon Mechanical Turk experiments, which we hope can be useful for researchers interested in running similar sets of experiments:

- Clarity of the task is fundamental. Showing examples of correct and incorrect selections clearly augments the proper completion rate. This was particularly relevant for the diagram region localization task.

- When conducting users studies, it is useful to run a small mock test, maybe with approximately 50 HITs, and gather comments from the workers. This can be fundamental to ensure that the choices are conditioned only by the factors that one wants to estimate with the user study, and not some external factors. This process was fundamental for the speaker face appearance experiment. After running the mock test, most of the workers commented that the images were overall too dark and many of them picked a view simply because it had a better illumination. This allowed us to correct the illumination of the images before submitting the real run, thus.

- The monetary reward is directly related to the completion time of the experiment. The higher the reward, the faster all HITs will be completed. However, it does not provide a guarantee of the quality of the work. We found that clarity of instructions and most interestingly the overall quality of the images presented in the HITs was more important to obtain good results. A higher reward incentives more workers to complete the jobs maximum amount of jobs in the shortest amount of time. However, most workers concentrate on the quality of their job if they find the work to be pleasant. Some workers even explicitly requested new jobs of the same type after they completed the assigned tasks.

- A degree of manual verification is still needed after the work is completed. While individual workers which consistently perform bad tasks are relatively easy to detect, it is still important to look at a subset of randomly selected HITs to make sure they comply with the expectations of the task.

Bibliography

[Adam *et al.*, 2006] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, volume 1, pages 798 – 805, 2006.

[Adcock *et al.*, 2010] John Adcock, Matthew Cooper, Laurent Denoue, Hamed Pirsiavash, and Lawrence A. Rowe. Talkminer: a lecture webcast search engine. In *Proceedings of the international conference on Multimedia*, MM '10, pages 241–250, 2010.

[Ahonen *et al.*, 2004] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face recognition with local binary patterns. In *ECCV*, pages 469–481, 2004.

[Amir *et al.*, 2004] Arnon Amir, Gal Ashour, and Savitha Srinivasan. Automatic generation of conference video proceedings. *J. Visual Communication and Image Representation*, 15(3):467–488, 2004.

[Anderson *et al.*, 2004] Richard Anderson, Crystal Hoyer, Craig Prince, Jonathan Su, Fred Videon, and Steve Wolfman. Speech, ink, and slides: the interaction of content channels. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 796–803, New York, NY, USA, 2004. ACM.

[Aner-Wolf and Kender, 2004] Aya Aner-Wolf and John R. Kender. Video summaries and cross-referencing through mosaic-based representation. *Comput. Vis. Image Underst.*, 95(2):201–237, 2004.

[Arandjelovic and Zisserman, 2005] O. Arandjelovic and A. Zisserman. Automatic face recognition for film character retrieval in feature-length films. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, pages 860–867, 2005.

[Babenko *et al.*, 2009] B. Babenko, Ming-Hsuan Yang, and S. Belongie. Visual Tracking with Online Multiple Instance Learning. In *CVPR*, 2009.

[Badekas *et al.*, 2007] E. Badekas, N. Nikolaou, and N. Papamarkos. Text binarization in color documents. *International Journal of Imaging Systems and Technology (IJIST)*, 16(6):262–274, 2007.

[Baker and Matthews, 2002] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework: Part 1. Technical Report CMU-RI-TR-02-16, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2002.

[Bevilacqua and Azzari, 2007] Alessandro Bevilacqua and Pietro Azzari. A fast and reliable image mosaicing technique with application to wide area motion detection. In *ICIAR*, pages 501–512, 2007.

[Birchfield, 1998] Stan Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *CVPR*, pages 232–237, 1998.

[Boutellier and Silvn, 2006] J. Boutellier and O. Silvn. Panoramas from partially blurred video. In *International Workshop on Intelligent Computing in Pattern Analysis/Synthesis, IWICPAS 2006 Proceedings*, volume 4153, pages 300–307, 2006.

[Boutellier *et al.*, 2008] Jani Boutellier, Olli Silvn, Marius Tico, and Lassi Korhonen. Objective evaluation of image mosaics. *Computer Vision and Computer Graphics. Theory and Applications*, pages 107–117, December 2008.

[Breitenstein *et al.*, 2010] Michael D Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. Online multi-person tracking-by-detection from a single, uncalibrated camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99:1–14, 2010.

[Brown and Lowe, 2003] M. Brown and D. G. Lowe. Recognising panoramas. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 1218, Washington, DC, USA, 2003. IEEE Computer Society.

[Brown *et al.*, 20 25 June 2005] M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale oriented patches. *CVPR 2005*, 1:510–517 vol. 1, 20-25 June 2005.

[Burke *et al.*, 2007] Darren Burke, Jessica Taubert, and Talia Higman. Are face representations viewpoint dependent? a stereo advantage for generalizing across different views of faces. *Vision Research*, 47(16):2164 – 2169, 2007.

[Burton and Bindemann, 2009] A. Mike Burton and Markus Bindemann. The role of view in human face detection. *Vision Research*, 49(15):2026 – 2036, 2009.

[Chen and Heng, 2003] Yu Chen and Wei Jyh Heng. Automatic synchronization of speech transcript and slides in presentation. *Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on*, 2:II–568–II–571 vol.2, 2003.

[Chen and Yuille, 2004] Xiangrong Chen and A.L. Yuille. Detecting and reading text in natural scenes. *Proceedings of the 2004 Computer Vision and Pattern Recognition (CVPR)*, 2:II–366–II–373, June-2 July 2004.

[Chen *et al.*, 2002] Xilin Chen, Jie Yang, Jing Zhang, and Alex Waibel. Automatic detection of signs with affine transformation. In *WACV '02: Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision*, page 32, 2002.

[Chen *et al.*, 2007] Tse-Wei Chen, Shou-Chieh Hsu, and Shao-Yi Chien. Automatic feature-based face scoring in surveillance systems. In *Proceedings of the Ninth IEEE International Symposium on Multimedia*, pages 139–146, 2007.

[Chong-wah Ngo and Huang, 2002] Ting-Chuen Pong Chong-wah Ngo and Thomas S. Huang. Detection of slide transition for topic indexing. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 533–536, 2002.

[Chowdhury *et al.*, 2003] S. P. Chowdhury, S. Mandal, A. K. Das, and Bhabatosh Chanda. Automated segmentation of math-zones from document images. *Document Analysis and Recognition, International Conference on*, 2:755, 2003.

[Del Bimbo *et al.*, 2009] Alberto Del Bimbo, Fabrizio Dini, and Giuseppe Lisanti. A real time solution for face logging. In *Proc. of International Conference on Imaging for Crime Detection and Prevention (ICDP)*, 2009.

[Djordjevic and Ghani, 2010] Divna Djordjevic and Rayid Ghani. Graphics classification for enterprise knowledge management. *Data Mining Workshops, International Conference on*, 0:562–569, 2010.

[Duffner and Odobez, 2011] Stefan Duffner and Jean-Marc Odobez. Exploiting long-term observations for track creation and deletion in online multi-face tracking. In *IEEE Conference on Automatic Face and Gesture Recognition*, 2011.

[Everingham *et al.*, 2006] M. Everingham, J. Sivic, and A. Zisserman. "Hello! My name is... Buffy" – automatic naming of characters in TV video. In *Proceedings of the British Machine Vision Conference*, 2006.

[Everingham *et al.*, 2009] Mark Everingham, Josef Sivic, and Andrew Zisserman. Taking the bite out of automated naming of characters in tv video. *Image and Vision Computing*, 27(5):545 – 559, 2009.

[Fan *et al.*, 2006] Quanfu Fan, Kobus Barnard, Arnon Amir, Alon Efrat, and Ming Lin. Matching slides to presentation videos using sift and scene background matching. In *8th ACM international workshop on Multimedia information retrieval (MIR)*, pages 239–248, New York, NY, USA, 2006. ACM.

[Fan *et al.*, 2011] Quanfu Fan, K. Barnard, A. Amir, and A. Efrat. Robust spatiotemporal matching of electronic slides to presentation videos. *Image Processing, IEEE Transactions on*, 20(8):2315 –2328, aug. 2011.

[Feris *et al.*, 2007] Rogerio Feris, Ying-Li Tian, and Arun Hampapur. Capturing people in surveillance video. In *CVPR*, pages 1–8, 2007.

[Fourney and Laganiere, 2007] Adam Fourney and Robert Laganiere. Constructing face image logs that are both complete and concise. In *CRV*, pages 488–494, 2007.

[Friedland and Rojas, 2008] Gerald Friedland and R. Rojas. Anthropocentric video segmentation for lecture webcasts. In *EURASIP Journal on Image and Video Processing, Hindawi*, 2008.

[Friedland, 1973] B. Friedland. Optimum steady-state position and velocity estimation using noisy sampled position data. *Aerospace and Electronic Systems, IEEE Transactions on*, AES-9(6):906–911, 1973.

[Funt and Shi, 2010] B. Funt and L. Shi. The rehabilitation of MaxRGB. In *Proceedings of the Eighteenth IS&T Color Imaging Conference (Society for Imaging Science and Technology)*, pages 256–259, 2010.

[Funt *et al.*, 2000] Brian Funt, Florian Ciurea, and John Mccann. Retinex in matlab. In *Journal of Electronic Imaging*, pages 112–121, 2000.

[G. and Buchsbaum, 1980] G. and Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin Institute*, 310(1):1–26, 1980.

[Gigonzac *et al.*, 2007] G. Gigonzac, F. Pitie, and A. Kokaram. Electronic slide matching and enhancement of a lecture video. *Visual Media Production, 2007. IETCVMP. 4th European Conference on*, pages 1–7, Nov. 2007.

[Gijsenij *et al.*, 2011] A. Gijsenij, T. Gevers, and J. van de Weijer. Computational color constancy: Survey and experiments. *IEEE Transactions on Image Processing*, 20(9):2475–2489, 2011.

[Godec *et al.*, 2010] M. Godec, C. Leistner, A. Saffari, and H. Bischof. On-line random naive bayes for tracking. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 3545 –3548, 2010.

[Gomez and Morales, 2002] Giovani Gomez and Eduardo F. Morales. Automatic feature construction and a simple rule induction algorithm for skin detection. In *ICML Workshop on Machine Learning in Computer Vision*, pages 31–38, 2002.

[Haubold and Kender, 2005] Alexander Haubold and John R. Kender. Augmented segmentation and visualization for presentation videos. In *MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia*, pages 51–60, New York, NY, USA, 2005. ACM.

[Haubold and Kender, 2007] Alexander Haubold and John R. Kender. VAST MM: multimedia browser for presentation video. In *CIVR*, pages 41–48, 2007.

[He *et al.*, 2000] Liwei He, Elizabeth Sanocki, Anoop Gupta, and Jonathan Grudin. Comparing presentation summaries: slides vs. reading vs. listening. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 177–184, New York, NY, USA, 2000. ACM.

[Hirakawa *et al.*, 2002] M. Hirakawa, K. Uchida, and A. Yoshitaka. Content-based video retrieval using mosaic images. *Cyber Worlds, 2002. Proceedings. First International Symposium on*, pages 161–167, 2002.

[Huang *et al.*, 2007] G.B. Huang, M. Ramesh, T. Berg, and E. Learned Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In *UMass Technical Report 07-49*, 2007.

[Huo *et al.*, 2006] Jun-yan Huo, Yi-lin Chang, Jing Wang, and Xiao-xia Wei. Robust automatic white balance algorithm using gray color points in images. *IEEE Transactions on Consumer Electronics*, 52(2):541–546, 2006.

[Irani *et al.*, 1996] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S. Hsu. Efficient representations of video sequences and their applications. *Signal Processing : Image Communication*, 8:327–351, 1996.

[Jung *et al.*, 2004] Keechul Jung, Kwang In Kim, and Anil K. Jain. Text information extraction in images and video: A survey. *Pattern Recognition*, 37(5):977–997, 2004.

[Kalal *et al.*, 2010] Z Kalal, J Matas, and K Mikolajczyk. P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints. 2010.

[Kender, 2000] John R. Kender. On the structure and analysis of home videos. In *In Proceedings of the Asian Conference on Computer Vision*, 2000.

[Kim *et al.*, 2008] Minyoung Kim, Sanjiv Kumar, Vladimir Pavlovic, and Henry Rowley. Face tracking and recognition with visual constraints in real-world videos. In *CVPR*, 2008.

[Krüger and Zhou, 2002] Volker Krüger and Shaohua Zhou. Exemplar-based face recognition from video. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part IV*, 2002.

[Land and McCann, 1971] Edwin H. Land and John J. McCann. Lightness and retinex theory. *Journal of the Optical Society of America*, 61(1):1–11, Jan 1971.

[Li and Wang, 2008] Jiangwei Li and Yunhong Wang. Face indexing and searching from videos. In *ICIP*, pages 1932–1935, 2008.

[Li *et al.*, 2007] Pengxu Li, Haizhou Ai, Yuan Li, and Chang Huang. Video parsing based on head tracking and face recognition. In *CIVR*, pages 57–64, 2007.

[Lienhart and Wernicke, 2002] R. Lienhart and A. Wernicke. Localizing and segmenting text in images and videos. *Circuits and Systems for Video Technology, IEEE Transactions on*, 12(4):256–268, Apr 2002.

[Liew and Kan, 2008] Guo Min Liew and Min-Yen Kan. Slide image retrieval: a preliminary study. In *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '08, pages 359–362, 2008.

[Lin *et al.*, 2004] R.-S. Lin, D. Ross, J. Lim, and M.-H. Yang. Adaptive discriminative generative model and its applications. In *NIPS*, pages 801–808, 2004.

[Liu *et al.*, 2002] Tiecheng Liu, R. Hjelsvold, and J.R. Kender. Analysis and enhancement of videos of electronic slide presentations. In *IEEE International Conference on Multimedia and Expo (ICME)*, volume 1, pages 77–80, 2002.

[Liu *et al.*, 2006] Xiaoming Liu, Jens Rittscher, and Tsuhan Chen. Optimal pose for face recognition. In *CVPR*, pages 1439–1446, 2006.

[Liu *et al.*, 2009] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos "in the wild". *CVPR*, 2009.

[Liyanage and Sasase, 2009] M. Liyanage and I. Sasase. Steady-state kalman filtering for channel estimation in ofdm systems utilizing snr. In *Communications, 2009. ICC '09. IEEE International Conference on*, pages 1 –6, june 2009.

[Longmore *et al.*, 2008] Christopher A. Longmore, Chang Hong Liu, and Andrew W. Young. Learning faces from photographs. *Journal if Experimental Psychology: Human Perception and Performance*, 34:77 – 100, 2008.

[Lyu *et al.*, 2005] M.R. Lyu, Jiqiang Song, and Min Cai. A comprehensive method for multilingual video text detection, localization, and extraction. *Circuits and Systems for Video Technology, IEEE Transactions on*, 15(2):243–255, Feb. 2005.

[Mackiewicz, August 2007] Jo Mackiewicz. Audience perceptions of fonts in projected powerpoint text slides. *Technical Communication*, 54:295–307(13), August 2007.

[Malik *et al.*, 2011] W.Q. Malik, W. Truccolo, E.N. Brown, and L.R. Hochberg. Efficient decoding with steady-state kalman filter in neural interface systems. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 19(1):25 –34, feb. 2011.

[Manning *et al.*, 2008] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[Martin *et al.*, 2007] T. Martin, A. Boucher, J.-M. Ogier, M. Rossignol, and E. Castelli. Multimedia scenario extraction and content indexing for e-learning. pages 204–211, June 2007.

[Mau *et al.*, 2010] Sandra Mau, Shaokang Chen, Conrad Sanderson, and Brian Lovell. Video face matching using subset selection and clustering of probabilistic multi-region histograms. In *ICIVC*, pages 1 – 8, November 2010.

[Merler and Kender, 2009] M. Merler and J.R. Kender. Semantic keyword extraction via adaptive text binarization of unstructured unsourced video. In *ICIP09*, pages 261–264, 2009.

[Merler and Kender, 2011] M. Merler and J.R. Kender. Selecting the best faces to index presentation videos. In *ACM MM*, 2011.

[Merler *et al.*, 2007] M. Merler, C. Galleguillos, and S. Belongie. Recognizing groceries in situ using in vitro training data. In *SLAM*, June 2007.

[Misra and Sural, 2006] Chinmaya Misra and Shamik Sural. Content based image and video retrieval using embedded text. In *ACCV06*, 2006.

[Morris and Kender, 2009] Mitchell J. Morris and John R. Kender. Sort-merge feature selection and fusion methods for classification of unstructured video. In *ICME'09: Proceedings of the 2009 IEEE international conference on Multimedia and Expo*, pages 578–581, 2009.

[Nasrollahi and Moeslund, 2008] Kamal Nasrollahi and Thomas B. Moeslund. Face quality assessment system in video sequences. *Biometrics and Identity Management: First European Workshop, BIOID 2008*, pages 10–18, 2008.

[Nasrollahi and Moeslund, 2009] K. Nasrollahi and T.B. Moeslund. Complete face logs for video sequences using face quality measures. *Signal Processing, IET*, 3:289 – 300, 2009.

[Otsu, 1979] N. Otsu. A threshold selection method from gray level histograms. *IEEE Trans. Systems, Man and Cybernetics*, 9:62–66, mar 1979.

[Paalanen *et al.*, 2007] P. Paalanen, J.-K. Kmrinen, and H. Klviinen. Image based qunatitative mosaic evaluation with artificial video. *Lappeenranta University of Technology, Research Report 106*, 2007.

[Papka and Allan, 2002] Ron Papka and James Allan. Topic detection and tracking: Event clustering as a basis for first story detection. In *Advances in Information Retrieval*, volume 7 of *The Information Retrieval Series*, pages 97–126. Springer US, 2002.

[Patricio and Gómez-Allende, 2000] M. A. Patricio and Darío Maravall Gómez-Allende. Segmentation of text and graphics/images using the gray-level histogram fourier transform. In *Proceedings of the Joint IAPR International Workshops on Advances in Pattern Recognition*, pages 757–766, 2000.

[Porikli, 2005] Fatih Porikli. Integral histogram: A fast way to extract histograms in cartesian spaces. *CVPR Vol. 1*, pages 829–836, 2005.

[Porter, 1980] M.F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[Prasad *et al.*, 2007] V.S.N. Prasad, B. Siddiquie, J. Golbeck, and L.S. Davis. Classifying computer generated charts. In *Content-Based Multimedia Indexing, 2007. CBMI '07. International Workshop on*, pages 85–92, june 2007.

[Ramachandra, 2000] K.V. Ramachandra. Kalman filtering techniques for radar tracking. *Marcel Dekker, New York*, 2000.

[Ramanan *et al.*, 2007] Deva Ramanan, Simon Baker, and Sham Kakade. Leveraging archival video for building face datasets. In *CVPR*, 2007.

[Saffari *et al.*, 2010] A. Saffari, M. Godec, T. Pock, C. Leistner, and H. Bischof. Online multi-class lpboost. In *CVPR*, pages 3570 –3577, 2010.

[Saidane and Garcia, 2008] Zohra Saidane and Christophe Garcia. An automatic method for video character segmentation. In *ICIAR '08: Proceedings of the 5th international conference on Image Analysis and Recognition*, pages 557–566, 2008.

[Santner *et al.*, 2010] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof. Prost: Parallel robust online simple tracking. In *CVPR*, pages 723 –730, 2010.

[Savva *et al.*, 2011] Manolis Savva, Nicholas Kong, Arti Chhajta, Li Fei-Fei, Maneesh Agrawala, and Jeffrey Heer. ReVision: Automated classification, analysis and redesign of chart images. In *UIST*, 2011.

[Shafait *et al.*, 2008] Faisal Shafait, Daniel Keysers, and Thomas M. Breuel. Efficient implementation of local adaptive thresholding techniques using integral images. In *Document Recognition and Retrieval XV*, San Jose, CA, Jan 2008.

[Shi and Tomasi, 1994] Jianbo Shi and C. Tomasi. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994*, pages 593 – 600, jun 1994.

[Simon *et al.*, 2007] Ian Simon, Noah Snavely, and Steven M. Seitz. Scene summarization for online image collections. *ICCV*, pages 1–8, 2007.

[Sivic *et al.*, 2009] J. Sivic, M. Everingham, and A. Zisserman. "Who are you?" – Learning person specific classifiers from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[Smith, 2007] R. Smith. An overview of the tesseract ocr engine. In *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2*, pages 629–633, Washington, DC, USA, 2007. IEEE Computer Society.

[Steedly *et al.*, 2005] Drew Steedly, Chris Pal, and Richard Szeliski. Efficiently registering video into panoramic mosaics. *ICCV*, 2:1300–1307, 2005.

[Szeliski, 2006] R. Szeliski. Image alignment and stitching: A tutorial. Technical Report MSR-TR-2004-92, Microsoft Research, December 2006.

[Tang and Kender, 2005] L. Tang and J.R. Kender. Semantic indexing for instructional video via combination of handwriting recognition and information retrieval. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 920–923, July 2005.

[Vinciarelli and Odobez, Oct 2006] A. Vinciarelli and J.-M. Odobez. Application of information retrieval technologies to presentation slides. *Multimedia, IEEE Transactions on*, 8(5):981–995, Oct. 2006.

[Viola and Jones, 2002] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 2002.

[von Kries, 1970] J. von Kries. Influence of adaptation on the effects produced by luminous stimuli. *Sources of Color Vision*, pages 109–119, 1970.

[Wang and Chen, 2009] Hongye Wang and Youbin Chen. Logo detection in document images based on boundary extension of feature rectangles. In *ICDAR '09: Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, pages 1335–1339, 2009.

[Wang and Kan, 2006] Fei Wang and Min-Yen Kan. Npic: Hierarchical synthetic image classification using image search and generic features. In *CIVR*, pages 473–482, 2006.

[Wang *et al.*, 2003] Feng Wang, Chong-Wah Ngo, and Ting-Chuen Pong. Synchronization of lecture videos and electronic slides by video text analysis. In *MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia*, pages 315–318, New York, NY, USA, 2003. ACM.

[Wang *et al.*, 2007] Feng Wang, Chong-Wah Ngo, and Ting-Chuen Pong. Lecture video enhancement and editing by integrating posture, gesture, and text. *IEEE Transactions on Multimedia*, 9(2):397 –409, February 2007.

[Wang *et al.*, 2008] Feng Wang, Chong-Wah Ngo, and Ting-Chuen Pong. Structuring low-quality videotaped lectures for cross-reference browsing by video text analysis. *Pattern Recognition*, 41(10):3257–3269, 2008.

[Wang *et al.*, 2011] Kai Wang, Boris Babenko, and Serge Belongie. End-to-end scene text recognition. In *IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, 2011.

[Yang *et al.*, 2005] Jun Yang, Rong Yan, and Alexander G. Hauptmann. Multiple instance learning for labeling faces in broadcasting news video. In *ACM MM*, pages 31–40, 2005.

[Yin *et al.*, 2007] Fei Yin, D. Makris, and S. A. Velastin. Performance evaluation of object tracking algorithms. In *10th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS2007)*, October 2007.

[Zhang *et al.*, 2008] Cha Zhang, Yong Rui, Jim Crawford, and Li-Wei He. An automated end-to-end lecture capture and broadcasting system. *ACM Trans. Multimedia Comput. Commun. Appl.*, 4(1):1–23, 2008.

[Zhu and Doermann, 2007] Guangyu Zhu and D. Doermann. Automatic document logo detection. volume 2, pages 864–868, 2007.