Computational Video Editing and Re-editing

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in Electronics and Communication by Research

by

MONEISH KUMAR
201331154
moneish.kumar@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
NOVEMBER 2018

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Computational Video Editing and Re-editing" by MONEISH KUMAR, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. VINEET GANDHI

To my family and F.R.I.E.N.D.S

Acknowledgments

It has been a very enriching 5 years for me here at IIIT-H. I am extremely grateful for the opportunities presented to me here. I would like to thank my advisor Dr. Vineet Gandhi without whom this could not been possible. His guidance and support has been instrumental in shaping this thesis and me.

This research was partially supported by ACM India for presenting at EUROGRAPHICS 2018 in Delft, Netherlands.

I would also like to thank my parents for their constant support and motivation.

Abstract

The amelioration in video capture technology has made recording videos very easy. The introduction of smaller affordable cameras which not only boast high-end specifications but are also capable of capturing videos at very high resolution (4K,8K and even 16K) have made recording of high quality videos accessible to everyone. Although this makes recording of videos very straightforward and effortless, a major part of the video production process - **Editing** is still labor intensive and requires skill and expertise. This thesis takes a step towards automating the editing process and making it less time consuming.

In this thesis, (1) we explore a novel approach of automatically editing stage performances such that both the context of the scene as well as close-up details of the actors are shown. (2) We propose a new method to optimally re-target videos to any desired aspect ratio while retaining the salient regions that are derived using gaze tracking.

Recordings of stage performances are easy to capture with a high-resolution camera, but are difficult to watch because the actors' faces are too small. We present an approach to automatically create a split screen video that transforms these recordings to show both the context of the scene as well as close-up details of the actors. Given a static recording of a stage performance and the tracking information about the actors positions, our system generates videos showing a focus+context view based on computed close-up camera motions using crop-and zoom. The key to our approach is to compute these camera motions such that they are cinematically valid close-ups and to ensure that the set of views of the different actors are properly coordinated and presented. We pose the computation of camera motions as convex optimization that creates detailed views and smooth movements, subject to cinematic constraints such as not cutting faces with the edge of the frame. Additional constraints link the close up views of each actor, causing them to merge seamlessly when actors are close. Generated views are placed in a resulting layout that preserves the spatial relationships between actors. This eliminates the need for manual labour and expertise required for both capturing the performance and later editing it, instead the splitscreen of focus+context views allows the viewer to make an active decision on attending to whatever seems important. We also demonstrate our results on a variety of staged theater and dance performances.

When videos are captured they are captured according to a specific aspect ratio keeping in mind the size of target screen in which they are meant to be viewed, this results in a inferior viewing experience when they are not watched on screens with their native aspect ratio. We present an approach to auto-

matically retarget any given video to any desired aspect ratio while preserving its most salient regions obtained using gaze tracking. Our algorithm performs editing with cut, pan and zoom operations by optimizing the path of a cropping window within the original video while seeking to (i) preserve salient regions, and (ii) adhere to the principles of cinematography. The algorithm has two steps in total. The first step uses dynamic programming to find a cropping window path that maximizes gaze inclusion within the window and also tries to find the location of plausible new cuts (if required). The second step performs regularized convex optimization on the path obtained via dynamic programming to produce a smooth cropping window path comprised of piecewise linear, constant and parabolic segments. We test our re-editing algorithm on a diverse collection of movie and theater sequences. A study conducted with 16 users confirms that our retargeting algorithm results in a superior viewing experience as compared to gaze driven re-editing [30] and letterboxing methods, especially for wide-angle static camera recordings.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With over 1 billion hours of videos being watched daily on YouTube alone [1] and about 7000 feature films being produced world wide this year [2], there has been an tremendous increase in multimedia consumption over the recent years. The enormous growth media consumption has led to an increased demand in video production. Video production involves three stages: pre-production, production, and post-production. Pre-production involves all of the planning aspects of the video production process before filming begins. This includes tasks like scriptwriting, scheduling, and logistics. Production is the phase of video production which captures the video content (moving images / videography) and involves filming the subject(s) of the video. Post-production is the action of selectively combining those video clips through video editing into a finished product that tells a story or communicates a message in it. Post-production **Editing** is largely a very time consuming and labor intensive process that requires a human expert to watch all the recordings and then make a informed decision on how to structure and present all video information.

This thesis explores ways to automate the editing process. We propose (1) a novel method to edit stage performances that provides a comprehensive view of the performance with minimal human intervention. (2) We present an algorithm to optimally re-target videos to any desired aspect ratio while retaining the salient regions that are derived using gaze tracking.

1.1 Jargons

Before we dive into the thesis we will take a look at some of the basic terminology used in the field of cinematography.

[1] https://www.youtube.com/yt/about/press/
[2] https://en.wikipedia.org/wiki/Film_industry

1.1.1 Shot and Shot types

In cinematography and film making, a **shot** is a series of frames that runs for an uninterrupted period of time. There are many ways in which the main subject can be framed within a shot, from seeing their entire body to only their eyes. Generally speaking, Shot types can be broken down into four main shot sizes: Long, Full, Medium, and Close-up. **Long Shot (aka Wide Shot)** is used to show the subject from a distance, or the area in which the scene is taking place. This type of shot is particularly useful for establishing a scene, for this reason it is sometimes also called establishing shot. **Full Shot** frames character from head to toes, with the subject roughly filling the frame. The emphasis tends to be more on action and movement rather than a characters emotional state. **Medium Shot**: Shows part of the subject in more detail. For a person, a medium shot typically frames them from about waist up. This is one of the most common shots seen in films, as it focuses on a character (or characters) in a scene while still showing some environment. **Close-Up** fills the screen with part of the subject, such as a persons head/face. Framed this tightly, the emotions and reaction of a character dominate the scene. Figure 1.1 gives an example of each of the shot type to better explain them.

Figure 1.1 Illustration of different types of shots. (a) Long shot, (b) Full shot, (c) Medium shot and (d) Close-up

©www.wikipedia.org

(a) (b)

Figure 1.2 Illustration of how Aspect ratio affects scene capturing. (a) is captured with a 4:3 aspect ratio where more importance is emphasized on the man and building, while in (b) we see the image captured with 16:9 lays more emphasis on the man and the atmosphere of the place.

1.1.2 Sequence

In cinematography, sequence refers to a series of shots that form a distinct narrative unit, which is usually connected either by unity of location or unity of time.

1.1.3 Composition

Composition in cinematography refers to the frame of the image and how all the arrangement of elements (inside the image) appear in it. When telling a story visually, as in film making it is desired that certain composition guidelines are observed. Rules such as the Lead Room (Figure 1.3) and Rule of thirds (Figure 1.4) are often used in cinematography as they create visually appealing compositions. These rules are however not written on stone, with sufficient motivation and purpose they can be overlooked to conceptualize extraordinary ideas and views.

1.1.4 Aspect ratio

Aspect ratio is an image attribute that describes the proportional relationship between the width of an image and its height. For example, movies, which are usually shot with a wide-angle lens, have an aspect ratio that is typically 16:9, which means that the width of the image area is almost twice its height. Aspect ratio decides the amount and the way environment around the subject is captured. This ability to manipulate the surroundings in different ways gives story tellers more artistic freedom. Figure 1.2 explains with an example how aspect ratio affects composition.

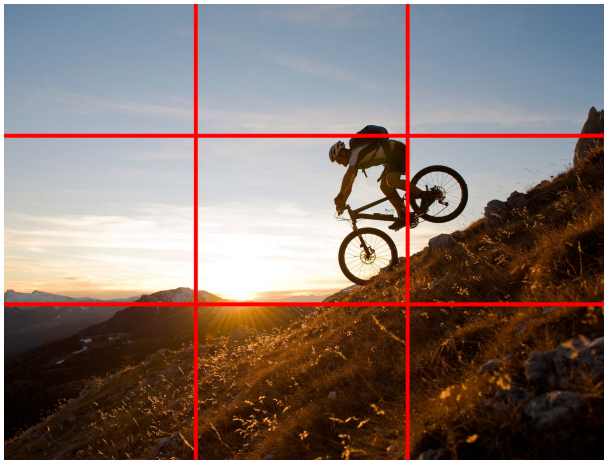
1.1.5 Cut

A cut is an abrupt transition from one sequence to another.

©photo.stackexchange.com

(a) (b)

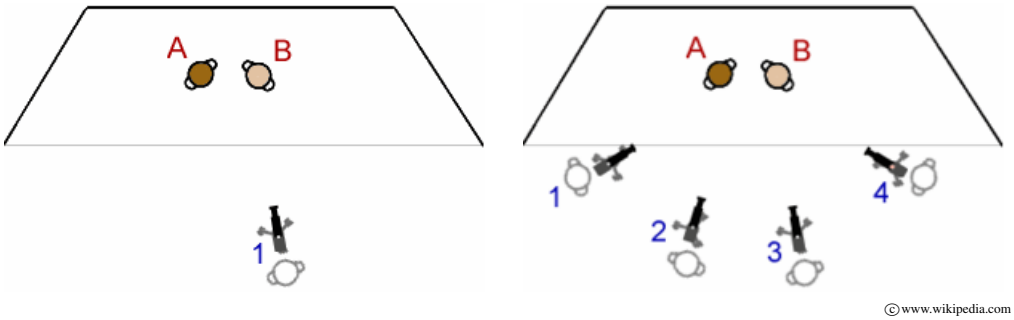
Figure 1.3 Illustration of how Lead space affects composition. Here are two pictures in which we see a plane flying towards right . Figure (a) has space towards right of the subject, this creates a sense of motion from left to right. In Figure (b) where there is space on the left behind the plane there seems to be something missing. There is a sense of motion that has been removed because plane doesn't seem to have anywhere else to go.



©in.pinterest.com

Figure 1.4 Illustration of Rule of thirds. The image is divided into nine equal parts by two imaginary parallel lines drawn along the width and height. The subject of the image is placed along these lines or at their points of intersection.

1.2 A need for Split-Screen Composition



(a) Single-Cam setup (b) Multi-Cam setup

Figure 1.5

It has always been a challenging task to record stage performances such as theatre or dance. A **single camera** production setup (figure 1.5 (a)) which involves capturing the entire scene using just **one** static camera is the simplest and cheapest for such task. The main advantage of this method is that it does not require any camera operator and the entire scene can be recorded automatically. Due to its simplicity and automation this type of setup is mainly used for archiving lectures and plays, rather that professional video production. Commercial video production demands a more engaging and appealing experience than just a prolonged shot from a single view. The most common method to capture stage performances for commercial video production is **Multi-Cam** setup. This method uses more than one camera on one shoot arranged to show the same scene at different angles and shot sizes (Figure 1.5 (b)). Generally, the outer cameras (Figure 1.5 (b) cameras 1,4) capture close-up shots of the active characters on the set at any given time, while the central camera or cameras (Figure 1.5 (b) cameras 2,3) shoot a wider "master shot" to capture the overall action in the scene. The restrictions in term of location (constrained by the field of view of other cameras and the limited number of positions where cameras can be placed inside the theatre) makes it harder to take shots with complex camera movements. This method also requires a larger camera crew and more equipment which add up the production costs.

Recent work by Gandhi et al. [20] provides a method to automatically generate multiple shots suitable for video editing by simulating pan-tilt-zoom camera movements within the frame of a single static camera. It assumes important actors and objects can be localized using computer vision techniques, their method requires only minimal user input to define the subject matter of each sub-clip. The composition of each sub-clip is automatically computed in a novel L1-norm optimization framework.The approach encodes several common cinematographic practices into a single convex cost function minimization problem, resulting in aesthetically pleasing shots. These shots, however have to be manually edited together video editing software to create the final video. Editing (manual) is a very tedious task

that involves an expert to select and assemble parts of these sub-clips into a coherent sequence that best portrays the performance.

This is where we found *split-screen composition* (SSC) to be extremely useful. The shots may be composited together in order to create a SSC. Split-screen compositions give the user the decision of what to attend to, reducing the need for editorial decisions that can be difficult to automate. Creating good split-screen compositions requires creating a set of views that are good individually and can be used together, as well as creating layouts that correctly convey the scene and its details.

In the third chapter, we present our approach for creating split-screen compositions of staged performances. We record a high-resolution, but wide field-of-view, video of the event with a static (unattended) camera. While this easy to create recording may capture the details, it does not necessarily provide a convenient way for a viewer to see both, the whole scene and the important details. Therefore, we provide an automatic system that transforms the video into a split-screen composition of both the wide view of the scene as well as detailed views of the actors' faces (Figure 1.6). The close-up views are created as virtual camera movements by applying panning, cropping and zooming to the source video. The key challenges are to compute the appropriate virtual camera movements in a way that create good close-ups that work together, as well to create proper layouts of these views that preserve the spatial relationships between actors. A single convex optimization is computed for all trajectories over an entire video segment.

The core of our approach is a novel method to determine the camera movements for close-up views of each actor given their position on stage (tracking information). We pose camera trajectory computations as a convex optimization, that an actor should be shown as large and centered in the close-up as possible, given constraints that the entire face must be shown, and the framing should avoid cutting other actors' faces and torso. An $L(1)$ regularization term creates movements that are not only as smooth, but follow the kinds of acceleration patterns preferred by cinematographers.

When multiple actors come close together, the system combines them into a single close-up view to avoid cutting their faces (or torso) with the frame. By adding additional constraints between the camera movements, the different camera paths merge seamlessly as actors approach.

Our system produces videos with a wide "master shot" and a set of closeups of all identified actors. This simple layout avoids making editorial decisions about what is more or less important in the scene. However, it does provide a focus+context view that shows both the overall action as well as the details of actors faces to allow for seeing emotion and dialog. The system presents the close-ups in an arrangement that preserves the spatial relationships between the actors. As actors move, the closeups seamlessly merge as they approach, split as they distance, and re-arrange to preserve ordering.

(a)

(b)

Figure 1.6 Illustration of our approach showing a frame from the input video and its corresponding split screen composition. We can observe that how the split screen view evidently brings out the emotions and expressions from the performance, which are hardly visible in the original view (master shot). The facial expressions play an important role in scenes of theatre and various dance forms like above and highlighting them can extremely enhance the viewing experience.

1.3 Retargeting video using gaze

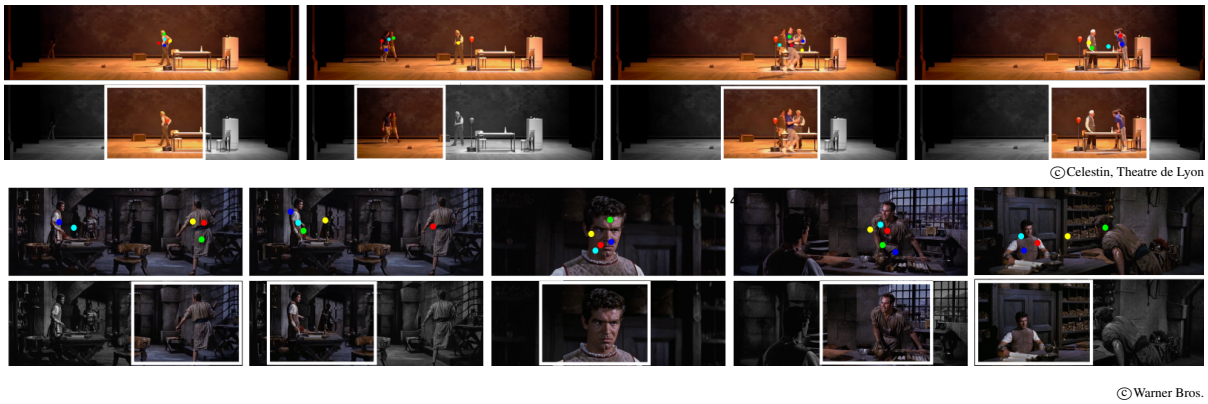


Figure 1.7 We present an algorithm to retarget a widescreen recording to smaller aspect ratios. The original recording with overlaid eye gaze data from multiple users (each viewer is a unique color) and the results computed by our algorithm (white cropping window) are shown. Our algorithm is content agnostic and can be used to edit a theatre recording from a static camera (top) or re-edit a movie scene (bottom).

The phenomenal increase in multimedia consumption has led to the ubiquitous display devices of today such as LED TVs, smartphones, PDAs and in-flight entertainment screens. While viewing experience on these varied display devices is strongly correlated with the display size, resolution and aspect ratio, digital content is usually created with a target display in mind, and needs to be manually re-edited (using techniques like pan-and-scan) for effective rendering on other devices. Therefore, automated algorithms which can *retarget* the original content to effectively render on novel displays are of critical importance.

Retargeting algorithms can also enable content creation for non-expert and resource-limited users. For instance, small/mid level theatre houses typically perform recordings with a wide-angle camera covering the entire stage as costs incurred for professional video recordings are prohibitive (requiring a multi-camera crew, editors *etc.*). Smart retargeting and compositing [36] can convert static camera recordings with low-resolution faces into professional-looking videos with editing operations such as *pan*, *cut* and *zoom*.

Commonly employed video retargeting methods are non-uniform scaling (squeezing), cropping and letterboxing [44]. However, squeezing can lead to annoying distortions; letterboxing results in large portions of the display being unused, while cropping can lead to the loss of scene details. Several efforts have been made to automate the retargeting process, the early work by Liu and Gleicher [38] posed retargeting as an optimization problem to select a cropping window inside the original recording. Other advanced methods like content-aware warping, seam carving then followed [48]. However, most of these methods rely on *bottom-up* saliency derived from computational methods which do not consider high-level scene semantics such as emotions, which humans are sensitive to [32, 45]. Recently, Jain *et*

al. [31] proposed Gaze Driven Re-editing (GDR), which preserves human preferences in scene content without distortion via user gaze cues and re-edits the original video introducing novel cut, pan and zoom operations. However, their method has limited applicability due to a) extreme computational complexity and b) the hard assumptions made by the authors regarding the video content– *e.g.*, the authors assume that making more than one cut per shot is superfluous for professionally edited videos, but this assumption breaks down when the original video contains an elongated (or single) shot as with the aforementioned wide-angle theatre recordings. Similarly, their re-editing cannot work well when a video contains transient cuts or fast motion.

To address these problems, we propose a novel retargeting methodology via gaze based re-editing employing convex optimization. Our work is inspired from GDR [31] and also estimates a cropping window path traversing through the original video introducing pan, cut and zoom operations in the process (Figure 1.7). However, our advantages with respect to GDR are that: (1) Our convex optimization framework guarantees a feasible solution with minimal computational complexity; our method requires 40 seconds to edit a 6-minute video, whereas GDR takes around 40 minutes to edit a 30 second video due to computation of non-uniform B-spline blending; (2) Our optimization is $L(1)$ regularized, *i.e.*, it ensures sparse editing motions mimicking professional camera capturing and optimizing viewing experience, whereas spline blending may result in small, unmotivated movements; (3) Our method is agnostic to both video content and video length and it extends GDR to long uncut videos like theatre or surveillance recordings captured with static, large field-of-view cameras.

Once potentially important scene content is captured via gaze data compiled from a few users, our algorithm performs re-editing of the gaze-tracked video in two steps. The first step employs user gaze transitions to identify time stamps in the original video where novel cuts can be introduced via dynamic programming-based optimization. It also estimates a cropping window path within the original recording, such that the cropping window encompasses maximum gaze information. The next step employs an $L(1)$ regularized convex optimization to convert this path into a professional looking one. The $L(1)$ regularization models the virtual camera trajectory via piecewise static, linear and parabolic segments, and ensures that the virtual camera moves frugally and fluidly akin to a professional camera, thereby resulting in a smooth viewing experience. This is confirmed via a user study involving 16 users, where our approach is perceived as enabling optimized viewing of scene emotions and actions, as compared to gaze based re-editing [31] and letterboxing especially for static theatre recordings.

The organization of the thesis is follows:

The second chapter contains literature about the related work done so far. The third chapter presents our approach to automatically creating split-screen, focus+context videos from captured recordings of staged performances. We provide an overview of the visual goals of our system in Section 3.1, the basic notion of the shot and the layout adjustment algorithm is described in Section 3.2. Section 3.3 describes how the virtual camera paths are computed as a convex optimization to be seamlessly composited together. Section 3.4 describes our prototype implementation and results we have produced with it. The fourth chapter provides a detailed discussion of our retargeting approach, associated experiments and

the user study. Section 4.1 explains the clips selected for retargeting and the procedure for collection of gaze data. We present our two stage optimization procedure for retargeting in Sections 4.2 and 4.3. The Results and discussion on them is present in Section 4.4 followed by a comprehensive user study in Section 4.7.

The contributions of this thesis include:

1. An end to end framework to automatically obtain SSC's from a single static master shot and the screen position (tracks) of actors present in the scene.

2. A method to dynamically adjust the partitions of the split screen composition as the actors move around and interact with each other

3. A method to obtain jerk free transitions from a layout to another by using top down constraints on the individual virtual camera simulations.

4. Extensive results generating SSCs on a variety of video sequences from theatre and dance with multiple actors and complex movements to demonstrate the effectiveness of our approach. We also present a novel application of SSC's with partition aware subtitles, which can enhance the perception of staged theatre for hearing challenged viewers.

5. A complete pipeline to optimally retarget videos to any desired aspect ratio using gaze.

6. A through experimentation of the retargeting algorithm on a diverse collection of videos .

7. A complete evaluation of the retargeting algorithm with the help of user analysis.

Chapter 2

Related Work

2.1 Splitscreens, Composite Shots and more

The previous attempts for generating SSC's from a static camera (or a set of static cameras) has been limited to specific scenarios like lecture videos, where a 'picture in picture' setting shows the inset view of speaker over the slides of the lecture. Bianchi's auto-auditorium [5] was one of the first system to demonstrate the automated broadcast of lecture videos. The auto-auditorium system employed a static camera to look at the entire scene and analyzed it to control a robotic camera to follow the speaker to be shown in the inset view (Figure 2.1).

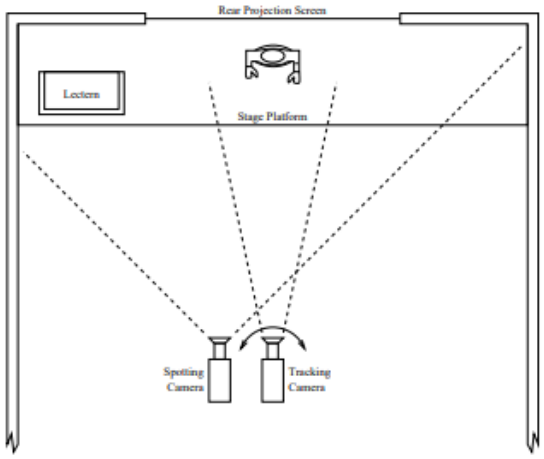


Figure 2.1 Diagram of the overview of Bianchi's auto-auditorium

The robotic camera was replaced by virtual camera simulations in [46] and [28], where the speaker inset view was generated by moving a cropping window inside a high resolution camera frame.

To generate conventional video, Sun et al. in [46] cropped a region of interest (ROI) from the panoramic video. The ROI is located from motion and color information in the uncompressed domain

and macroblock information in the compressed domain, and tracked using a Kalman filter. The resulting virtual camera simulates human controlled video recording.

Heck et al. in virtual videography [28] provided a method specifically aimed for recording classroom lectures. There were four phases in the system (Figure 2.2). (1) Media analysis identified region objects and provided the system with simple events and properties of the original video data . (2) Using the set of region objects supplied by media analysis, the attention model builds an evolving probability for each region specifying the likelihood of this region being a focus of the audiences attention at any instant in time. (3) The computational cinematographer then determines the sequence of shots that should be used to conform to appropriate cinematic rules. (4) Finally, an image synthesis component constructs a final video by synthesizing views and visual effects .

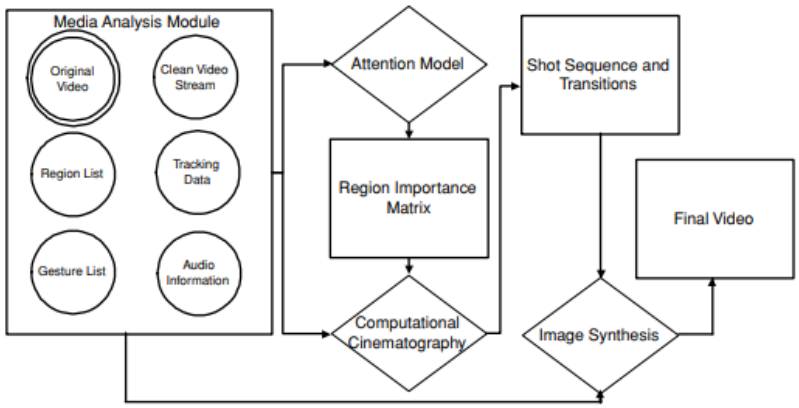


Figure 2.2 A diagram describing the structure of the Virtual Videography system

However, most of these methods have been confined to a simple restricted setting of single presenter in front of a chalkboard or slides. The SSC in these methods is also limited to a single simulated window of the presenter overlaid usually at the bottom right part of the display.

The virtual camera work has also been investigated in the field of sports.

Chen et al. [12] presented a method for personalized production of basketball videos. They use multiple static cameras to track players, referee and the ball. They also compute the identification information based on the jersey numbers . The camera subregion is then computed based on a set of fixed rules and user preferences.

Carr et al. [10] used multiple static cameras to obtain player tracks in Basketball games and used the centroid of the tracks to guide a robotic camera. They showed that smooth, human like camera trajectories can be obtained after post-processing step utilizing non causal filtering.

Daigo et al. [14] try to estimate the areas of interest in a basketball match using audience gaze direction. Static cameras are used to detect rough regions where the players exist (based on motion). The final broadcasting video is then generated using the combination of these two sets of information

by virtual panning in a panoramic video created using four static cameras. They also demonstrate that the individual camera resolution can be preserved by virtual panning in the panoramic videos.

The work in FascinatE project [43] demonstrated real time interactive virtual camera editing in ultra high resolution video recording of soccer games. The project aims to allow end-users to interactively view and navigate around an ultra-high resolution video panorama showing a live event, with the accompanying audio automatically changing to match the selected view. The output would be adapted to their particular kind of device, covering anything from a mobile handset to an immersive panoramic display.

Ariki et al. [3] proposed a method which recognizes game situations or events such as penalty kick or a free kick based on player and ball tracking and then digitally zooms into the video. The tracking information is obtained using background subtraction and the event recognition is performed based on simple rules (for example a penalty kick is detected when the ball is static for several minutes and is placed near the penalty spot). Their method then defines different clipping window sizes based on different events.

A variety of approaches have been proposed to obtain smooth virtual camera trajectories.

Chen and Vleeschouwer [12] use markov random fields (MRF) to obtain smooth camera trajectories. Yokoi and Fujiyoshi [51] used bilateral filtering to reduce the noise from virtual camera trajectory obtained using region of interest (ROI) computed at each frame. Sun et al. [46] used a rule based approach combined with Kalman filter to smooth out noise from initial tracking estimates. Heck et al [28] explicitly modelled the camera movement into three segments (parabolic, linear and then again parabolic). However, most of these methods are limited by ad-hoc rules or lack of generalization. Moreover, the conventional filtering based approaches only suppresses high frequency jitter and may not mimic the behaviour of a professional cameraman.

Our work is closely related to recent L1-norm based camera stabilization/simulation methods [26, 20]. The work in [26] showed that the trajectories of professional camera work comprise of distinct static, linear and parabolic segments and this idea can be exploited to stabilize noisy recordings using a linear optimization framework. Gandhi et al. [20] extended this idea for virtual camera simulation and presented an application of multi-clip editing from single viewpoint. We exploit the generalization of this line of work and show that how this can be extended for the task of generating SSC's by adding additional constraints in the optimization framework.

2.2 Re-editing

Several media Re-editing/retargeting solutions have been proposed, and they can be broadly categorized into three different categories: discrete, continuous and cropping-based approaches. Discrete approaches [4, 41, 40] judiciously remove and/or shift pixels to preserve salient media content.

Avidan et al. in [4] provide a method that can change the size of an image by gracefully carving-out or inserting pixels in different parts of the image. They define "seam" to be a connected path of

low energy pixels crossing the image from top to bottom, or from left to right. Various visual saliency measures (such as gradient magnitude, entropy, visual saliency, eye-gaze movement) are used to define the energy of the pixels. By successively removing or inserting seams of the lowest energy they can reduce, as well as enlarge, the size of an image in both directions.

Pritch et al. in [40] represent geometric rearrangement operation of images (such as image retargeting, inpainting, or object rearrangement) as an optimal graph labeling problem. Shift-map represents the selected label for each output pixel and they solve this problem of graph labeling using graph cuts.

Continuous approaches [17] optimize a warping (mapping) from the source to the target media, with constraints designed to preserve salient media content resulting in a non-homogeneous transformation, with less important regions squeezed more than important ones. Some of these discrete and continuous approaches have been extended to video retargeting [41, 49, 34]. However, discrete or continuous removal of pixels often results in visible artifacts such as squeezed shapes, broken lines or structures and temporal glitches/incoherence.

A third retargeting approach selects a cropping window for each frame inside the original video. This approach eliminates visual artifacts, but some visual information is nevertheless lost in the cropping process. This is in spirit, the 'pan and scan' process, where an expert manually selects a cropping window within a widescreen image to adapt the same to smaller aspect ratios. The goal is to preserve the important scene aspects, and the operator smoothly moves the cropping window as action shifts to a new frame position or introduces new cuts (for rapid transitions).

Several efforts [38, 16, 50] also have been made to automate the 'pan and scan' or re-editing process.

Liu and Gleicher [38] formulate the problem of pan and scan as an optimization to choose a cropping window for each video frame that minimizes information loss by balancing the loss of detail due to scaling with the loss of content and composition due to cropping.The cropping window can be moved during a shot to introduce virtual pans and cuts, subject to constraints that ensure cinematic plausibility.

Deselaers et al. [16] presented a method to automatically fit videos in 16:9 format on 4:3 screens and vice versa. They do this by creating a log-linear model using saliency, optical flow and appearance to obtain important regions in the video frame. Later a cropping window is path is selected using dynamic programming maximizing the important region and keeping the cropping window path smooth (The pipeline is shown in Figure 2.3).

These approaches primarily rely on computational saliency (primarily bottom-up saliency based on spatial and motion cues) to discover important content, and then estimate a smooth camera path that preserves as much key content as possible.

Liu and Gleicher [38] define the virtual camera path as a combination of piecewise linear and parabolic segments.Grundmann *et al.* [27] employ an $L(1)$ regularized optimization framework to produce stable camera movements thereby removing undesirable motion. Gleicher and colleagues [24, 38, 29, 21] relate the idea of a moving cropping window with virtual camera work, and show its application for editing lecture videos, re-editing movie sequences and multi-clip editing from a single sequence.

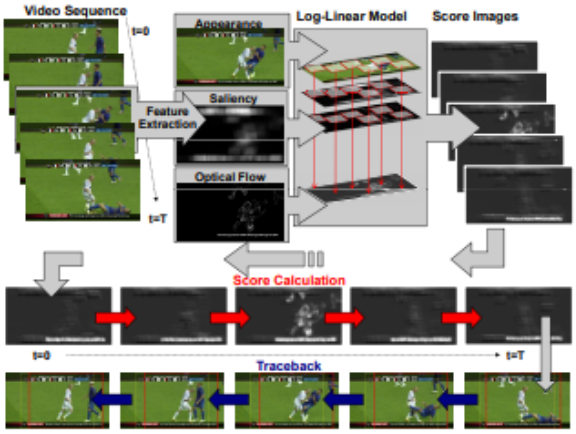


Figure 2.3 A diagram describing the framework of Deselaers' Pan,Scan and Zoom [16]

While human attention is influenced by bottom-up cues, it is also impacted by top-down cues relating to scene semantics such as faces, spoken dialogue, scene actions and emotions which are integral to the storyline [45, 22]. Leake *et al.* [37] propose a computational video editing approach for dialogue-driven scenes which utilizes the script and multiple video recordings of the scene to select the optimal recording that best satisfies user preferences (such as emphasize a particular character, intensify emotional dialogues, *etc.*). A more general effort was made by Galvane *et al.* [18] for continuity editing in the 3D animated sequences. However, the movie script and the high-level cues used in these works may not always be available. A number of other works have utilized *eye tracking* data, which is indicative of the semantic and emotional scene aspects [32, 45], to infer salient scene content.

Santella *et al.* [42] employ eye tracking for photo cropping so as to satisfy any target size or aspect ratio. More recently, Jain *et al.* [31] perform video re-editing based on eye-tracking data, where RANSAC (random sampling consensus) is used to compute smooth B-splines that denote the cropping window path. Nevertheless, this approach is computationally very expensive as B-splines need to be estimated for every RANSAC trial. Also, the methodology involves implicit assumptions relating to the video content, and is susceptible to generating unmotivated camera motions due to imprecise spline blending.

Keeping in mind the shortcomings of the previous methods and taking inspiration from literature in cinematography [47, 6, 39] we discuss the characteristics of an ideal editing algorithm and the criterions that have inspired our algorithmic choices.

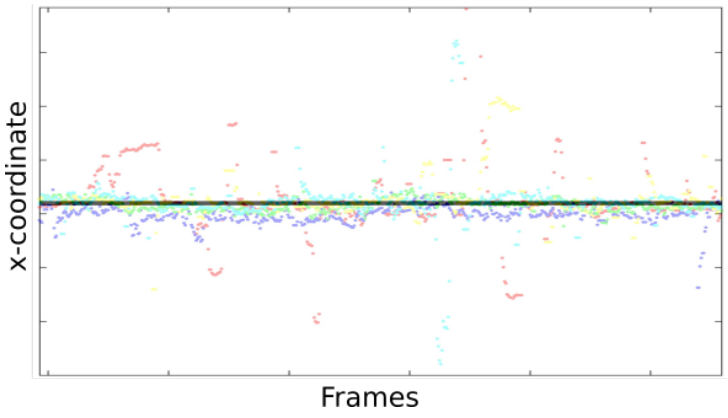
Millerson and Owens [39] stress on the aspect that the shot composition is strongly coupled with what viewers will look at. If viewers do not have any idea of what they are supposed to be looking at, they will look at whatever draws their attention (random pictures produce random thoughts). This motivates the choice of using gaze data in the re-editing process. Although, notable progress has been made in computationally predicting human gaze from images [8], the cognitive gap still needs to be

filled in [9]. For this reason, we explicitly use the collected gaze data in the proposed algorithm as the measure of saliency. The algorithm then aims to align the cropping window with the collected gaze data.

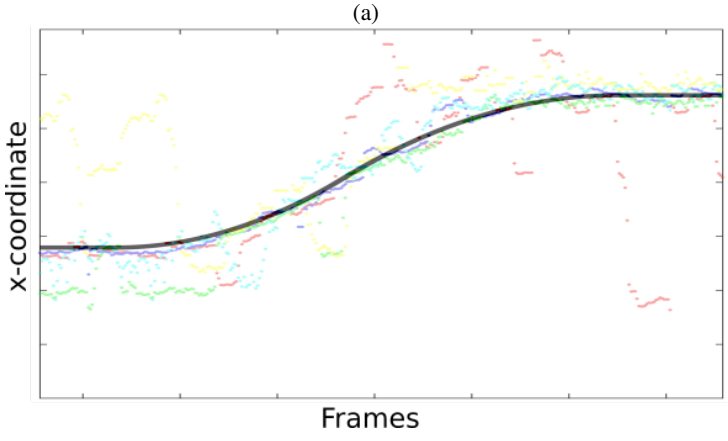
Importance of a steady camera has been highlighted in Thomson and Christopher's 'Grammar of the shot' [47]. They suggest that (a) camera should not move without a sufficient motivation (as it may appear puzzling to the viewer) and brief erratic pans should be avoided and (b) a good pan/tilt movement should comprise of three components: a static period of the camera at the beginning, a smooth camera movement which "leads" the attention of the subject and a static period of the camera at the end. As discussed earlier, ideally we would like to center the cropping window around the most salient region (computed from gaze data), however we allow some relaxation using rectified linear unit function to avoid brief camera movements over small gaze variations. Furthermore, we use a combination of first, second and third order $L(1)$ norm regularization to achieve professional looking smooth pan/zoom (leading to piecewise static, linear and parabolic segments) [27]. Figure 2.4 illustrates the behavior of our algorithm with (a) gaze centered around a location where the algorithm outputs a perfectly static segment and (b) gradually varying gaze, where the algorithm outputs a smooth pan.

The fast panning movements should be avoided, as it may lead to breakup of the movement [39]. We impose constraints on the maximum panning speed in the optimization process, to make sure that fast panning movements do not occur. Apart from panning, another crucial parameter which needs to be controlled is amount of zoom, which is often correlated with the localization and intensity of the scenes [39]. We use variance of gaze fixation data as the indicator of amount of zoom-in. If the gaze data is concentrated around a point for a long interval, it motivates zooming-in at that location and the high variance gaze suggests that a lot is going on in the scene and the camera should zoom out to include more content. Furthermore, to mimic human like behavior, we add a delay in zoom/pan movements (a computational camera starts panning at the instant an actor starts moving, while an actual cameraman takes a moment to respond to the action).

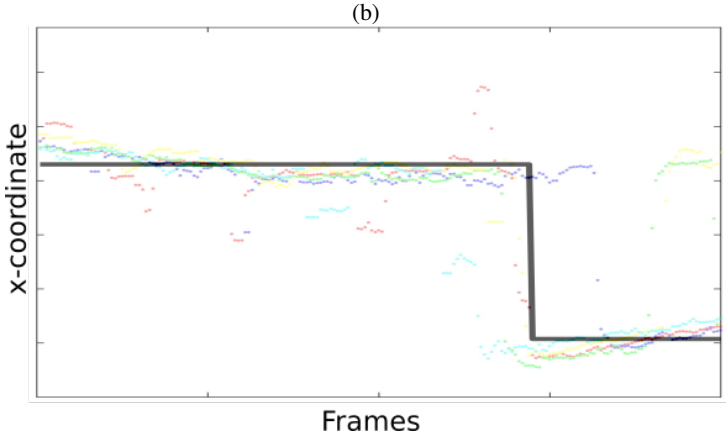
Cuts are another form of transitions which are used when there is a need for quick change in impact. There should always be a reason to make the cut, the two shots before and after the cut should not be too similar (avoiding jump cut) and the pace of the cut should go along with the story (it should not be too fast and too slow) [6]. We argue that frames with abrupt changes in gaze positions are good candidate locations for adding cuts, as that indicates the change in impact. To avoid jump cuts, we use a penalty term in the optimization process, quantifying the overlap in the compositions before and after the cut. Furthermore, we use a decaying exponential penalty to control the pace/rhythm of the cuts. Figure 2.4(c) illustrates an example where abrupt change in gaze location leads to a cut.



(a)



(b)



(c)

Figure 2.4 The recorded eye gaze data from five users and the corresponding output (the optimal x position) of our algorithm. Results with three different type of gaze have been presented i.e. (a) fixation: when the user maintains the eye gaze around a location, our algorithm outputs a static segment; (b) pursuit: a gradual movement of eye gaze results in a pan; (c) saccade: an abrupt transition in gaze leads to cut.

Chapter 3

Automatic Split Screen Generation

This chapter explains our algorithm to create automatic split-screen composition in detail.

3.1 Overview

The proposed focus+context SSC's (Split-screen composition) are defined by the size of the close-up view and the layout of the screen space. The size of the close-up view characterizes the amount of zoom on the subjects i.e. how much to zoom in? Closely zoom only to the face or show the entire upper body. The layout characterizes the partition of the screen space i.e. arrangement of the context view and the focus view. For instance, Figure 1.6 shows a particular layout with screen space partitioned into four segments, the top half showing the overview of the original master shot and the bottom half is split into three parts, each showing one of the actors in a zoomed-in view (four partitions showing four different camera feeds, one original and three virtually simulated).

Given a recording from a static camera (master shot) our method automatically generates such split screen compositions given the position information of the subjects (actor tracks) and the desired size of the focus view. This is an extremely challenging task and extends the domain of the typical multi-camera SSC creation pipeline to scenes like stages theatre where the subjects are moving and interacting with each other. We now describe these challenges:

3.1.1 Challenges

The focus+context SSC's are usually created by combining feeds from multiple cameras manually, where each camera is operated by a professional (focusing on particular subject). The layouts are hard coded, where each partition of screen space is assigned to a particular subject. Hence, SSC's have mostly been prevalent in scenario such as broadcast of debates or discussions of a panel etc., which are recorded in a controlled news studio like environment. The movement of the subjects is minimal in such settings and hence the desired framing can be easily followed and adjusted. And finally, since there is no interaction between subjects, a hard coded layout suffices well.

frame-0375

frame-0494

Figure 3.1 A hard coded SSC does not preserve the left to right order of the actors on stage.

This standard SSC creation process is infeasible in studied scenario of theatre stage recordings, where the subjects move around and interact with each other. The hard coded layout may pose a problem, when actors cross over each other and handling this will require additional editorial decisions. This is illustrated with an example in Figure 3.1, where the close-up view of the male actor is still shown on the left, even after he crosses over to the right side of the stage (the relative order of the actors changes from frame-375 to frame-494, however the order of the close-up view keeps the initial arrangement). This may hinder the correct understanding of the scene. Hence, a primary visual goal of an automatic SSC creation algorithm is to dynamically adjust the relative position of close up view of actors with respect to the actual scene.

The second problem is caused by the interaction of the actors, as it may disturb the individuality of each close up view and output may look chaotic. This is illustrated in Figure 3.2. We can observe the redundancy, where the same actor is shown across multiple screens. Handling such interactions

Figure 3.2 Keeping a fixed hard coded layout causes problems when actors interact with each other.

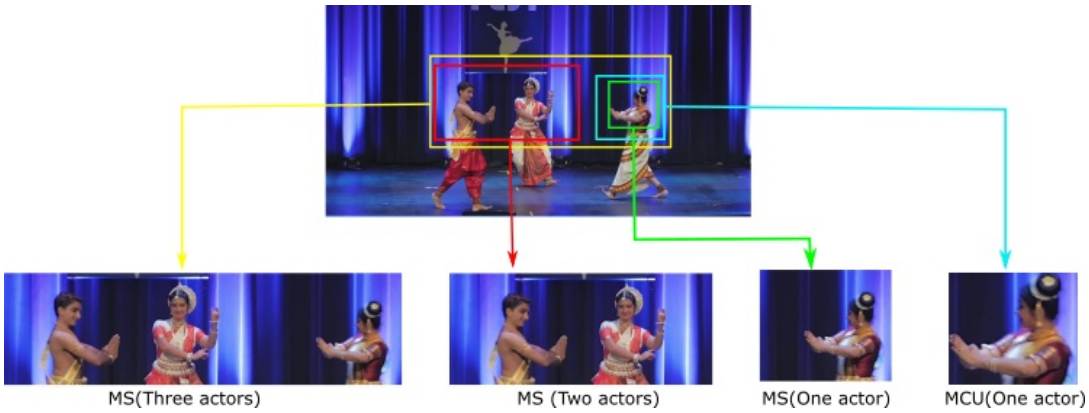


Figure 3.3 A shot is specified using its size and number of actors to be included in it. The figure shows the cropped framing corresponding to different shot specifications. Selected framings with medium shot (MS) and medium close up (MCU) have been shown.

for manual SSC creation in multi-camera setting is a mammoth task and has been one of the main limiting factors for application of SSC's in dynamic settings. We demonstrate that how a computational approach can automatically tackle this challenge by dynamically adjusting the screen layout considering the position of the actors on stage (this is described in Section 3.2.3).

As our method generates the SSC's from a single master shot (instead of manually operated multiple cameras), the next obvious problem is to generate multiple camera views from this single feed. This is achieved by simulating virtual pan/tilt/zoom cameras by moving a cropping window inside the master shot. This is a non trivial task, as the actor tracks are often wobbly and simply following them would lead to unsteady camera feeds. The major challenge here is to emulate the professional camera behaviour. We overcome this using a L-1 regularized optimization framework (explained with detail in Section 3.3). The last hurdle is to provide seamless transitions while changing the layout of the SSC's, which is

tackled by our method using an additional top down constraint in the optimization framework (explained with detail in Section 3.3.4).

Considering the above discussions, we define five major visual goals for an automatic SSC creation algorithm:

1. Each close-up view (insets) should be good individually
2. Virtual camera-work for each close-up view should follow cinematic norms
3. SSC layout should preserve arrangement: all actors are shown + left to right order is preserved
4. Redundancy among different screen partitions (different inset views) should be avoided i.e. no actor appears more than once and no "cut faces or torso"
5. The changes in layout(splitting or merging close-up view of different actors), if any, should be seamless

3.2 Split Screen Specification

In this section, we first explain the notions of shot specification and layout configuration; and then describe the algorithm to dynamically adjust the layout according to the position of the actors on stage.

3.2.1 Shot specification

The literature in video production [7] commonly specifies shots by their subject matter and size. The term 'shot' in the context of focus+context SSC's is the virtually simulated camera feeds, which are to be shown in the close-up view. The subject matter for a shot is typically the list of actors or objects which should be included in the camera view (in this paper we do not consider objects). The shot size basically defines how big things are in the camera view (how much to zoom in). In case of actors, this is typically defined as 'full shot' (covering the entire actor from head to toe); a 'medium shot' (frames an actor from head to waist) or a 'medium close up' (starts at actor's head and cut's off around mid chest).

An example of selected virtual cropping frames corresponding to Medium Shot (MS) and Medium Close Up (MCU) are shown in Figure 3.3. The virtual cropping frame is defined using its centre co-ordinates (x, y), size (s) and aspect ratio A_r (as illustrated in Figure 3.4). It is computed using the input actor tracks (bounding boxes from head to toe, as obtained in usual pedestrian detection algorithms [15]). For example, a medium shot of single actor is obtained by cropping the top half of its tracking bounding box in desired aspect ratio while keeping some empty space above the head (called headroom) for proper framing. Similarly, a medium shot for two actors is computed in such a way that both actors are atleast in a medium shot with a proper head room (framing covers both actors atleast from head to waist).

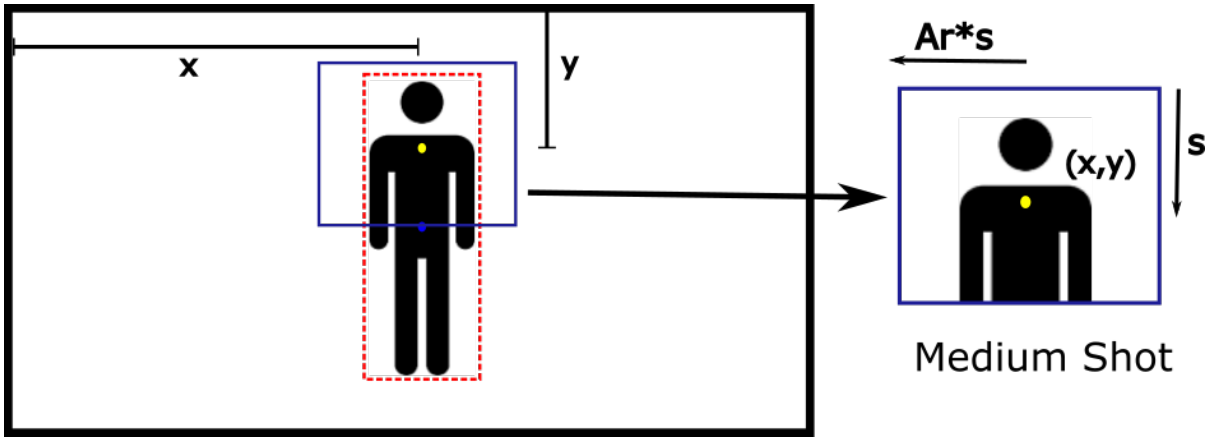


Figure 3.4 Framing convention. The framing (blue rectangle) is obtained using the actor track (red dotted rectangle) and the shot size (MS in this case). It is defined using the center position (w.r.t to original frame), size and the aspect ratio.

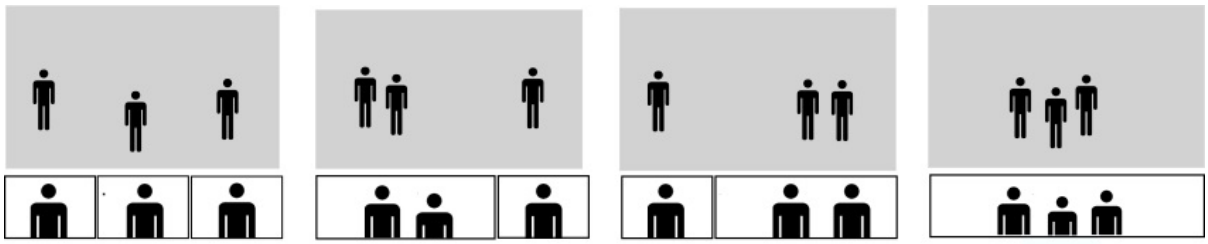


Figure 3.5 We choose different split-screen layouts depending on the grouping of actors. This figure illustrates the four possible layouts for three actors.

The aspect ratio is the width to height ratio of the virtual camera frame and is affected by the shot size, number of actor present in the scene and the layout configuration. For example in Figure 3.3 aspect ratio for a medium shot of an individual actor is defined in such a way, that it covers a third of the width and half of the height. Similarly, a medium shot of two actors, cover two third of the width and half of the height. We discuss the layout configuration with more detail in the proceeding section.

3.2.2 Layout configuration

The layout configuration refers to the way screen space is partitioned into multiple segments, each showing a different camera feed. For example in Figure 1.6(b), we have considered a layout where the screen is first vertically partitioned in two parts (top part showing the overview of the master shot) and then the lower part is further partitioned into three parts (each corresponding to zoomed-in view of an actor).

However, as described earlier it may not always be a good choice to equally partition the lower part horizontally and it may lead to redundancies. In the particular case of Figure 1.6 with three actors

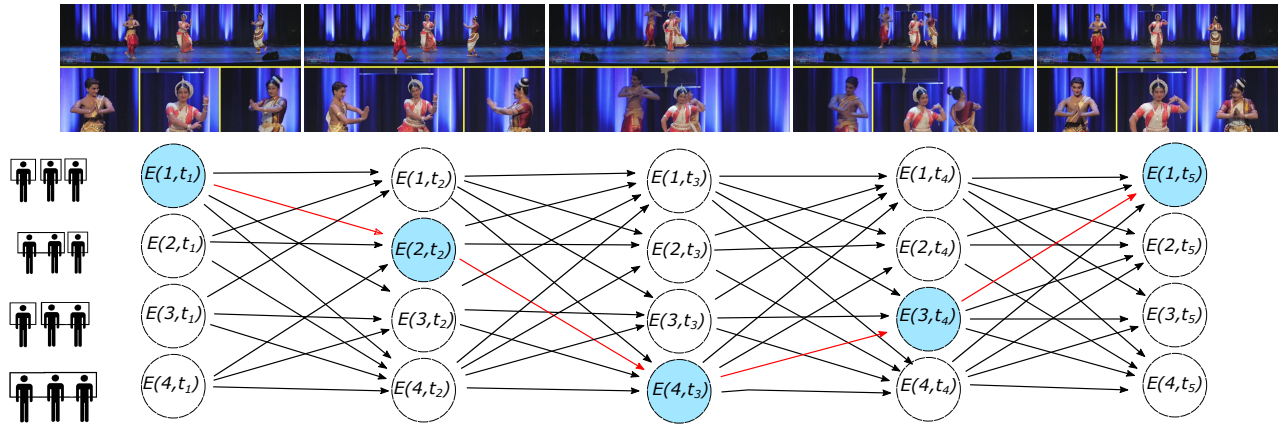


Figure 3.6 Illustration of layout selection algorithm with a realistic three actor example. The algorithm selects a node for each time t from the given possibilities (four in this case). For each node in the graph, we compute the minimum cost $E(r_t, t)$ to reach it. Backtracking is then performed from the minimum cost node in the last column. The blue color shows the selected state, given the positions of the actors at the respective time (images above show the SSC's).

on stage, four different ways of horizontal partitioning are possible, as illustrated in Figure 3.5. First configuration equally divides the horizontal space into three parts, each assigned to an individual actor. However, if two leftmost or rightmost actors are close by each other, two unequal partitions with two actors together and one actor separate may give a better composition (three equal partitions may look awkward in this case, with actors being cut by the partition boundaries as shown in Figure 3.2). Another possibility is to only keep a single partition, if all three actors are close around each other.

The similar idea can be extended for scene with different number of actors, for instance a scene with four actors can take eight different configurations and a scene with two actors only has two possible configurations. A good split screen composition should adjust the layout according to the positions of the actors on stage. The next section describes an algorithm to automatically select the most appropriate layout configuration at a given time in a video sequence. For explanation purposes, we assume the standard format with two equal vertical divisions and the algorithm only adjusts the horizontal divisions in the lower part. However, the method can be easily extended to any reasonable layout (partition of screen space) showing specified shots at particular partitions.

3.2.3 Layout configuration selection

Instead of using a fixed layout throughout the split screen video, the proposed framework dynamically adjusts the horizontal partitions based on the positions of the actors present on stage. The aim is to reflect the meaningful groupings of the actors into the layout configuration. We again take an example scenario with three actors on stage to explain the layout selection algorithm. In this particular case, four possible configurations are possible as described in the previous section (Figure 3.5).

The input to the selection algorithm is the individual framings (assuming equal partitions) for each actor over the entire video and output is a sequence of $\varepsilon = \{r_t\}$ states $r_t \in [1 : k]$, for all frames $t = [1 : N]$. Each configuration denotes a possible state, giving four possible states ($k = 4$) in the considered scenario. The selection algorithm minimizes the following global cost function:

$$E(\varepsilon) = \sum_{t=1}^{N} E_o(r_t) + \lambda \sum_{t=2}^{N} E_s(r_{t-1}, r_t). \tag{3.1}$$

Here E_o is the overlap cost, penalizing any overlap between the framings in the given layout. In case of three actors (a,b and c from left to right), this is defined as follows:

$$E_o(r_t, t) = \begin{cases} O_{a,b}(t) + O_{b,c}(t) & \text{if } r_t = 1\ , \\ -O_{a,b}(t) + O_{b,c}(t) & \text{if } r_t = 2\ , \\ O_{a,b}(t) - O_{b,c}(t) & \text{if } r_t = 3\ , \\ -O_{a,b}(t) - O_{b,c}(t) & \text{otherwise.} \end{cases} \tag{3.2}$$

Here, $O_{a,b}(t)$ denotes the overlap cost between the individual framing of left and middle actors a and b at time t. Given the individual framings (x_t^a, y_t^a, s_t^a) and (x_t^b, y_t^b, s_t^b) at time t for actors a and b respectively, it is defined as sum of width minus the distance between the center of the two framings:

$$O_{a,b}(t) = A_r(s_t^a + s_t^b) - |x_t^a - x_t^b| \tag{3.3}$$

The above term is negative if there is no overlap between the individual framing of actor a and actor b. It takes a positive value in case of overlapping framings. The term $O_{b,c}(r_t, t)$ is similarly defined as the overlap cost between the individual framing of right actor c and the middle actor b. We can observe in Equation 3.2, the first state will be the preferred state when all three actors are further apart, with both $O_{a,b}$ and $O_{b,c}$ taking negative values. State two will be preferred state when there is overlap between actor a and actor b, but actor c is further apart (with $O_{a,b}$ taking a positive value and $O_{b,c}$ taking a negative value). Similarly, state four will be preferred when all three actors are close to each other(both $O_{a,b}$ and $O_{b,c}$ taking positive values).

The smoothness term E_s penalizes frequent transitions (momentary transitions may distract the user), it is defined as follows:

$$E_s(r_t, r_{t-1}) = \begin{cases} 0 & \text{if } r_t = r_{t-1}\ , \\ 1 & \text{if } r_{t-1} = 1 \text{ and } r_t \in \{2, 3\} \text{ or vice versa,} \\ 1 & \text{if } r_{t-1} \in \{2, 3\} \text{ and } r_t = 3 \text{ or vice versa,} \\ 2 & \text{if } r_{t-1} = 3 \text{ and } r_t = 1 \text{ or vice versa.} \end{cases} \tag{3.4}$$

No penalty is incurred if the layout configuration does not change. A change from state one to either state two or three is penalized with an unit cost. A transition from state one to state four directly is penalized with twice the unit cost. The unit cost is defined by the parameter λ in Equation 3.1.

Please note that we have taken an example scenario of three actors for explanation purposes, but the system can be easily extended to handle split screen compositions with different number of actors. In general, the number of possible states is equivalent to finding number of subset of the $(n-1)$ separators between n actors, which equals 2^{n-1} possibilities (each separator is either active or not). For example in case of two actors there are two possible states and we only need to handle a single overlap term and single transition. Similarly, for four actors, there are eight possible states which need to be handled with three overlap terms and sixty four possible transitions.

Finally, we minimize Equation 3.1 using dynamic programming. The algorithm selects a state r_t for each time t from the given possibilities (k in this case). We build a cost matrix $C(r_t, t)$ (where $r_t \in [1:k]$ and $t \in [1:N]$). Each cell in this table is called a $node$. The recurrence relation used to construct the DP cost matrix is a result of the above energy function and is as follows:

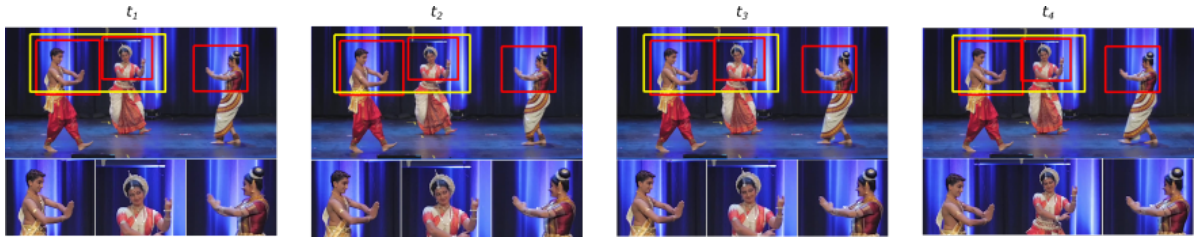
$$C(r_t, t) = \begin{cases} E_o(r_t) & t = 1 \\ \min\limits_{r_{t-1}} \left[E_o(r_t) + \lambda * E_s(r_{t-1}, r_t) + C(r_{t-1}, t-1) \right] & t > 1 \end{cases}$$

For each node (r_t, t) we compute and store the minimum cost $C(r_t, t)$ to reach it. A cut c_t is introduced at frame t, if the accumulated cost is lower for introducing a cut than keeping the position constant or panning the window. Backtracking is then performed from the minimum cost node in the last column to retrieve the desired path. Finally, the output of the algoritm is the optimized cropping window path $\epsilon = \{\mathrm{r}_t\}$ and the set of newly introduced cuts $\{c_t\}$. The time complexity of the algorithm is $O(k^2 N)$ and the space complexity is $O(kN)$, which are both linear with N. The process is illustrated with a synthetic example in Figure 3.6.

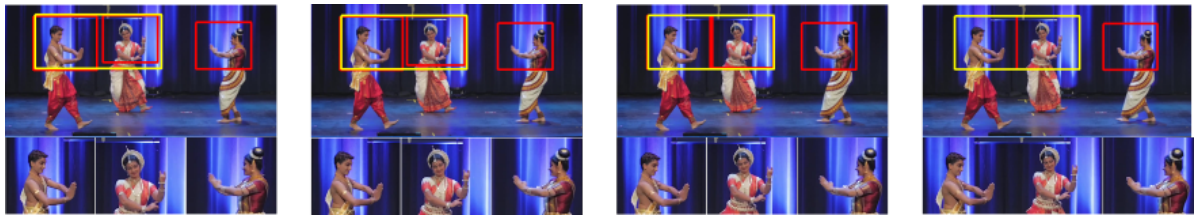
3.3 Split screen optimization

Given the appropriate split screen layout configuration at each frame of the video, the next step is to simulate multiple camera feeds which can be shown at the corresponding partition. For example, in state one in Figure 3.5 we need to simulate three virtual cameras for each actor individually and display them side by side. However, if the layout transitions to state two, we will need two camera feeds, one showing left two actors in medium shot and another showing the rightmost actor in a medium shot separately. Given the shot specification and the desired aspect ratio, we now describe the optimization procedure to obtain the virtual camera feeds. We integrate layout transitions constraints in an optimization framework inspired by the work in [20], to obtain smooth individual camera movements with seamless transitions between different layout configurations.

The optimization algorithm takes as input the actor tracks, the aspect ratio and the shot specification. The actor track bounding box at time t for a particular actor a is defined by its center bx_t^a, by_t^a, half width bw_t^a and half height bh_t^a. The initial virtual camera cropping window is individually computed for each frame of the video using the actor tracks. This initial computed virtual camera frame f_t is denoted by its center and size (x_t, y_t, s_t) for each frame t (as illustrated in Figure 3.4). The output of the algorithm



(a) SSC without the layout transition constraint.



(b) SSC with layout transition constraint.

Figure 3.7 An example of layout transition. The two partitions on the left merge into a single one as the left dancer moves towards the screen right. The red rectangles shows the individual medium shots for each dancer and the yellow rectangle shows the combined medium shot of two actors on the left.

is the optimized set of framing coordinates $\xi = \{x_t^*, y_t^*, s_t^*\}$ following the cinematographic conventions for composition and movement of the camera.

3.3.1 Shot size penalty

It is desired that the simulated virtual camera feed should abide to the required shot specification (when asked for a medium shot, the cropped framing should approximately cover the actors from head to waist). The shot size penalty thus penalizes any deviation from the desired shot size. It is defined as follows:

$$D(\xi) = \sum_{t=1}^{N}((x_t^* - x_t)^2 + (y_t^* - y_t)^2 + (s_t^* - s_t)^2). \tag{3.5}$$

The function increases the cost if the optimized virtual cropping frame deviates from the initial cropping window individually computed for each frame.

3.3.2 Inclusion constraints

We introduce two sets of hard constraints, first that the virtual cropping window should always lie within the master shot and second that the appropriate part of the actor track should be included inside the virtual cropping window. The first constraint ensures that we always get a feasible solution and second constraint ensures that the actors are not strangely chopped off by the virtual cropping window. For instance, the left most coordinate of the cropping window $x_t - A_r * s_t$ should be greater than zero

and less than the left boundary of the actor bounding box $bx_t - bw_t$. Similarly, the rightmost coordinate of the cropping window $x_t + A_r * s_t$ should be greater than the right boundary of the actor bounding box $bx_t + bw_t$ and should be less than the width of the master frame W. Formally, we define the horizontal constraints as:

$$0 < x_t - A_r * s_t \leq bx_t - bw_t \text{ and } bx_t + bw_t \geq x_t + A_r * s_t \leq W \tag{3.6}$$

In case of shot with multiple actors, the leftmost actor is considered for the left constraint and the right most actor in the shot is considered for the right constraint. The vertical constraints are similarly defined for the top and the lower virtual frame boundaries. The lower constraint is adjusted according to the shot size.

3.3.3 Movement regularization

Simply computing the framing for each individual frame may lead to a noisy virtual camera motion. According to professional cinematographic practices [7], a steady camera behaviour is necessary for pleasant viewing experience. A camera movement without enough motivation may appear irritating to the viewer, hence the camera should remain static in case of small and unmotivated movements. When the camera moves, it should start with a segment of constant acceleration followed by a segment of constant velocity and should come to a static state with a segment of constant deceleration.

We introduce three different penalty terms to obtain the desired camera behaviour. The first term tends to keep the camera static by penalizing the L-1 norm over the first order derivative:

$$M_1(\xi) = \sum_{t=1}^{N-1} (|x^*_{t+1} - x^*_t| + |y^*_{t+1} - y^*_t| + |s^*_{t+1} - s^*_t|). \tag{3.7}$$

The second term drives constant velocity segments and it is defined as follows:

$$\begin{aligned} M_2(\xi) = \sum_{t=1}^{N-2} (|x^*_{t+2} - 2x^*_{t+1} + x^*_t| + |y^*_{t+2} - 2y^*_{t+1} + y^*_t| \\ +|s^*_{t+2} - 2s^*_{t+1} + s^*_t|). \end{aligned} \tag{3.8}$$

The third term optimizes for constant acceleration segments and it is defined as follows:

$$\begin{aligned} M_3(\xi) = \sum_{t=1}^{N-3} (|x^*_{t+3} - 3x^*_{t+2} + 3x^*_{t+1} - 3x^*_t| \\ +|y^*_{t+3} - 3y^*_{t+2} + 3y^*_{t+1} - 3y^*_t| \\ +|s^*_{t+3} - 3s^*_{t+2} + 3s^*_{t+1} - 3s^*_t|). \end{aligned} \tag{3.9}$$

Combining, the three penalties gives an output with the camera movement consisting of distinct static, linear and parabolic segments.

3.3.4 Split and merge constraints

The split screen composition is obtained by simulating multiple virtual camera feeds and displaying them at the respective positions. For example in state one in Figure 3.6, three different camera feeds corresponding to each individual actor is concatenated to form the lower part of the SSC. However, the layout may differ as the actors change their positions on stage as described in Section 3.2.3. For instance state two in Figure 3.6, will require concatenation of only two virtual camera feeds i.e. a medium shot of two leftmost actors and a medium shot of right actor. Overall, we need to simulate all possible shot specifications which may be required for rendering the SSC for the entire video. With example case of three actors, we will need to generate seven different camera feeds (three individual shots, three two shots and a three shot).

However, individually generating each of these seven shots and then concatenating them will cause jerks in event of transition from one layout to another. This is illustrated in top row of Figure 3.7. At time t_1, all three actors are far apart and split screen composition shows their individual camera feeds side by side. The layout transition happens at time t_4, as the actor on the left moves closer to the middle actor. The computed virtual camera framings for medium shot of all three actors (red rectangles) and the medium shot of the left two actors are also shown (yellow rectangle). We can observe that switching from the two individual medium shots to the combined medium shot of left two actors (at transition point t_4), will cause jerk (due to the discontinuity).

We address this problem, using a top down approach for virtual camera simulation i.e. we first optimize the highest order shot (including maximum number of actors) in the scene and use it to apply additional constraints on the lower order shots (with lesser number of actors) at the point of transitions. We just need to make sure that at the point of transition, the higher order shot should exactly be equal to the union of its subset shots. We introduce this as a hard constraint to obtain fluid transitions from one layout configuration to another. The example in lower row of Figure 3.7 illustrates the output with top-down constraints, which makes sure that the individual framings of left two actors (red bounding boxes) exactly match the yellow rectangle at the point of transition.

More formally, assuming that a higher order shot f_t^H get partitioned into a set of two lower order shots $\{f_t^{s_l}, f_t^{s_r}\}$ at point of layout transition t (or the lower order shots get merged into higher order shot). For instance, a medium shot of two actors together gets partitioned into two individual medium shots or a medium shot of three actors together gets partitioned into one medium shot of single actor and one medium shot of two actors together. The following is the layout transition constraint:

$$f_t^{s_l} \bigcup f_t^{s_r} = f_t^H \text{ and } f_t^{s_l} \bigcap f_t^{s_r} = \phi, \tag{3.10}$$

where ϕ is the null set. This constraint is added at all points of layout transitions τ, which are pre-computed and are known prior to the virtual camera optimization. The above equation implies that there should be no overlap between the lower order subset views and their union should exactly comprise the higher order view. Let $f_t^{s_l}$ denotes the left side partition and $f_t^{s_r}$ denote the right side partition,

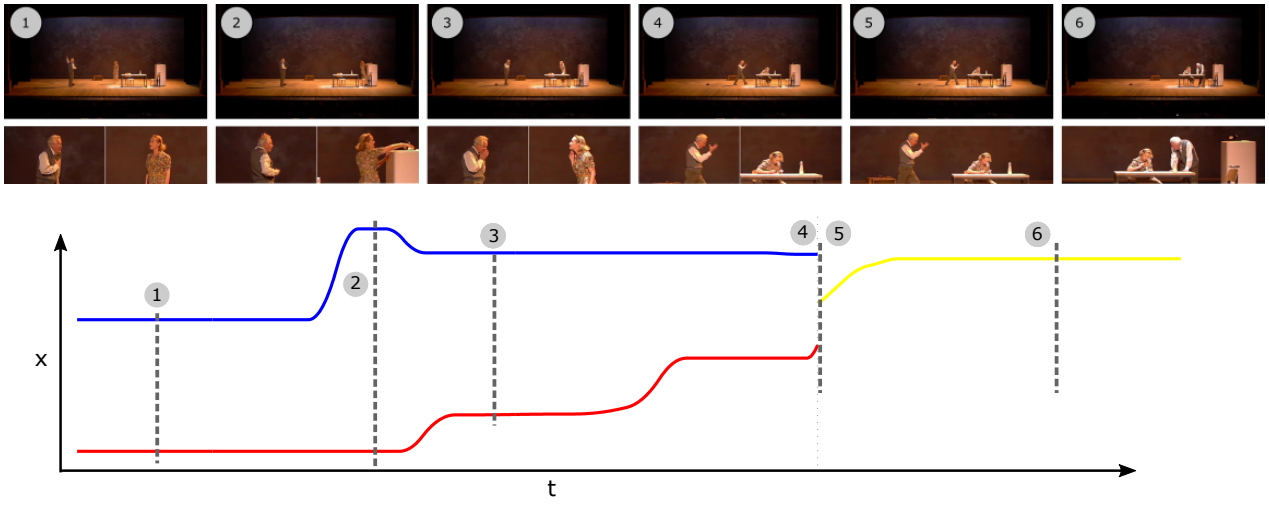


Figure 3.8 Trajectories of the centre x-coordinate of the computed virtual framings (for MS) over a theatre sequence with two actors. Some selected images and the corresponding SSC's are also shown. The trajectory for individual framing of male actor 'Willy' and female actor 'Linda' are shown in blue and red color respectively. The yellow color presents the trajectory of the both actors together. The dotted black lines show the time instances where images were taken. The layout transition happens at frame 4-5.

then the Equation 3.10 can be defined as a set of linear constraints *Split And Merge* linear constraints $SAM(f^{s_l}, f^{s_r}, f^H, \tau)$:

$$\begin{aligned} x_t^{s_l} + Ar^{s_l} h_t^{s_l} &= x_t^H, \\ x_t^{s_r} - Ar^{s_r} h_t^{s_r} &= x_t^H, \\ y_t^{s_l} = y_t^{s_r} &= y_t^H, \\ h_t^{s_l} = h_t^{s_r} &= h_t^H, \ t \in \tau. \end{aligned} \tag{3.11}$$

These conditions ensure that the (x, y)-coordinates and height of the lower order shots are correctly aligned with the higher order shots, at the point of of transitions to avoid any jerk. The constraints can be easily extended to higher order splits (such as a three shot splitting into three individual one shots), as a higher order split can be defined as multiple binary splits.

3.3.5 Energy minimization

Finally, the problem of finding the virtual camera trajectory, can be simply be stated as a problem of minimizing a convex cost function with linear constraints. The overall optimization function is defined as follows:

$$\underset{x,y,s}{\text{minimize}}\ (D(\xi) + \lambda_1 M_1(\xi) + \lambda_2 M_2(\xi) + \lambda_3 M_3(\xi))$$

subject to

$$\begin{aligned} & 0 < x_t^* - A_r * s_t^* \leq bx_t - bw_t, \\ & bx_t + bw_t \leq x_t^* + A_r * s_t^* \leq W, \\ & 0 < y_t^* - s_t^* \leq by_t - bh_t, \\ & by_t' \leq y_t^* - s_t^* \leq H, \\ & SAM(f^{s_l}, f^{s_r}, f^H, \tau),\ t = 1, \ldots, N. \end{aligned} \tag{3.12}$$

The variables λ_1, λ_2 and λ_3 are parameters. The optimization is done for each shot separately from higher order to lower (shots with more actor are optimized first). The SAM function, derives the hard layout transition constraints from the higher order shots depending on the configuration (no layout constraints are applied on the shot with all actors). The overall optimization function (including the layout constraints) can be optimized using any off the shelf convex optimization toolbox, we use cvx [25].

3.4 Experimental results

We demonstrate results on seven different sequences from Arthur Miller's play 'Death of a salesman'; two sequences from Tennessee Williams play 'Cat on a hot tin roof' and an Indian classical dance sequence. The play sequences were recorded from same viewpoint in Full HD (1920×1080) and the dance sequence was recorded in 4K (3840×2160) resolution. To show the versatility of our approach, we have chosen sequences with two, three and four actors. For each of the given master sequence, we generate split screen compositions by rendering multiple virtual camera feeds from it corresponding to user defined shot size. All the rendered videos are provided in the supplementary material. Comparison between a naive approach (of simply computing the framing for each actor independently at each time) and the proposed approach has also been provided in the supplementary material with the help example videos. Additionally, example frames from the SSC's of different sequences are shown in Figure 3.8, Figure 3.11 and Figure 3.9. We now discuss the obtained results over various aspects:

Layout transitions: Results on a two actor theatre sequence are shown in Figure 3.8. The Figure illustrates the selected images from the original video and their corresponding split screen compositions with the x-coordinate of the computed virtual framings over the entire sequence. We can observe the seamless transition of layout from two partitions to a single partition at frame-5 (the x-coordinate of the yellow trajectory corresponding to a combined medium shot is exactly at the mid-point of the two individual framings, at the point of transition). The fluidness of the transition can also be observed in the rendered images of the SSC where the union of two partitions at frame-4, exactly constitutes the two shot framing in the next frame. Similar transitions can also be observed at *sequence2*-1177/1178, *sequence4*-

(a) (b)

Figure 3.9 Split screen compositions with subtitles placed selectively in the partition of the actor who is speaking. Such position aware subtitles can further enhance the perception of the staged events like theatre (usually recorded from a static wide angle camera), especially for the hearing challenged viewers.

0303/0304 and *sequence7*-0146/0147 in Figure 3.11. Moreover, the example SSC's in Figure 3.11 clearly demonstrate the versatility of our approach, covering results comprising of different layouts with different aspect ratios; with different number of actors undergoing significant movements/interactions. The correctly done layout transitions also takes care of redundancy problem among screen partitions and preserves the actual left to right order of the actors.

Camera Movement: The centre x-coordinate trajectories of virtual framings in Figure 3.8 demonstrate the desired camera behavior comprising of smooth transitions between long static segments (when the camera moves, it moves nicely or it remains fully static). For example, observe how the virtual camera frame of Linda (blue trajectory) smoothly accelerates and then decelerates as she moves towards the fridge and comes to rest again. We can also observe that there is no jitter in the trajectories (due to small movements), for instance the right movement of Willy also comprises of two smooth strides, with static segments in-between.

Composition: Another aspect to observe in the presented results is how the nice compositions are maintained with desired shot specifications, even in presence of large movements and interaction of actors, including cases when they cross each other (*sequence3*, *sequence5* and *sequence7* in Figure 3.11). We can observe that the virtual camera frame avoids cropping the actors and maintains a proper headroom, even with changes in aspect ratio; shot size or number of actors in it. The composition in also maintained well with changes in postures and poses (Figure 3.8 and *sequence6* in Figure 3.11). Integrating more information into the tracker like hand positions or motion saliency etc. can further help to improve the compositions for better perception of the gestures.

Viewing Experience: It is evident in almost all examples, that it is difficult to perceive emotions and facial expressions in a wide shot of the entire stage. In fact, in some cases, it can even be difficult to recognize and locate speakers. As we observe in Figure 3.11, SSC's could be one way to resolve this issue as they clearly bring out the emotions and expressions of the actors. This makes our method

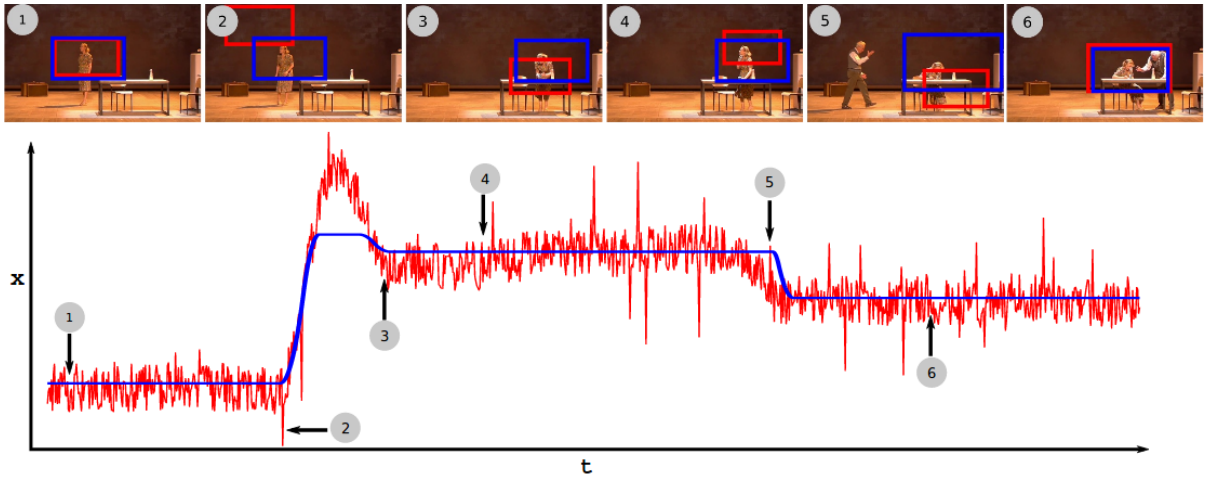


Figure 3.10 The figure illustrates the performance of our system when the input actor position estimates are noisy. The x-coordinate of independent per frame close-up view estimation (red) for Linda is compared with the optimized version (blue) for sequence1. Some selected images with estimated close-up view bounding boxes are also shown.

attractive for digital heritage projects of the performing arts in general. Our method is also generally applicable to other domains such as sports and social events. One extremely interesting application of SSC is to enhance the viewing perception for hearing challenged people, with the help of partitioned subtitles. An example is illustrated in Figure 3.9. We can notice that, not only the SSC helps tracking the lip movements using zoomed-in views, putting the subtitles in correct partition can further help to enhance the understanding of the scene for hearing challenged viewers.

Performance with noisy actor tracks: In order to test the robustness of the system, we add synthetic noise to the input actor tracks to simulate the effects of inaccurate tracking. The noise is added in the form of uniformly distributed pseudo random integers with sample interval of [-40, 40] pixels. We also add larger errors at random positions (shift up to 200 pixels) to simulate momentary tracking failures. We remove the hard constraint on actor bounding box in Equation 3.12 to allow convergence of the optimization function. The per frame close-up view estimation from noisy tracks are compared with the optimized ones in Figure 3.10. The figure shows that the optimized result (blue rectangle) is well composed even with large errors in tracking information where the per frame estimations (red rectangle) are completely off the target (for instance in frame2 and frame5). The video results for this experiment are provided in the supplementary material

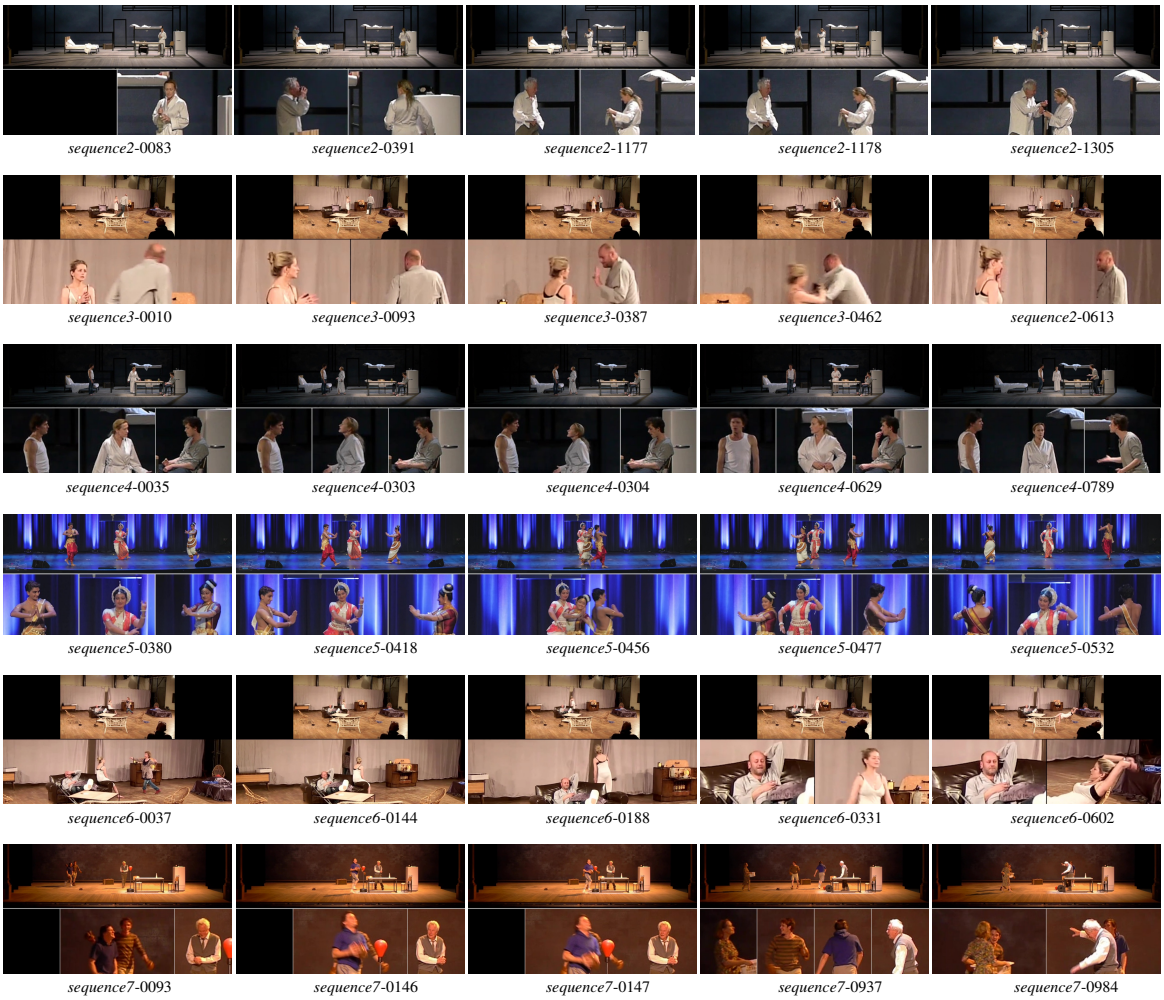


Figure 3.11 Examples frames from SSC over different video sequences (*sequence2*, *sequence3*, *sequence4*, *sequence6*, *sequence7* are from theatre and *sequence5* is of a dance performance). This illustration includes cases with two (*sequence2*, *sequence3*), three (*sequence4*, *sequence5*, *sequence6*) and four (*sequence7*) actors.

Chapter 4

Retargeting video using gaze

In this chapter we present a novel, optimization-based framework for video retargeting which utilizes minimal user gaze data. It is capable of working with any type of input video (professionally created feature films or even unmanned wide-angle recordings of theatre/dance performances) of arbitrary length to produce a retargeted video of any given aspect ratio or size. $L(1)$ regularized optimization that we utilize, economizes and smoothens virtual camera motion to mimic professional camera capture behavior, and ensures a smooth viewing experience. Our method requires only approximate rather than accurate information regarding salient scene regions, we use eye-tracking data recorded with a low-end and affordable ($\approx$ 100 $euros$) eye tracker. We demonstrate that our method outperforms the state-of-the-art [31] via experiments and a user study reflecting subjective human expressions concerning the quality of the retargeted video.

The proposed algorithm takes as input (a) sequence of frames $t = [1 : N]$ of the input video, where N is the total number of frames; (b) the raw gaze points of multiple users, g_t^i, for each frame t and subject i and (c) the desired output aspect ratio. The algorithm outputs the edited sequence to the desired aspect ratio, which is characterized by a cropping window parametrized by the x-position (x_t^*) and zoom (z_t) at each frame. The edited sequence introduces new panning movements and cuts within the original sequence, aiming to preserve the cinematic and contextual intent of the video.

Our algorithm consists of two steps.The first step uses dynamic programming to detect a path ($\epsilon = \{r_t\}_{t=1:N}$) for the cropping window which maximizes the amount of gaze and time stamps appropriate to introduce new cuts which are cinematically plausible. The second step optimizes over the path ($\{\mathrm{r}_t\}$) to mimic the professional cameraman behavior, using a convex optimization framework (converting the path into piecewise linear, static and parabolic segments, while accounting for the original cuts in the sequence and the newly introduced ones). We will first explain the data collection process and then describe the two stages of our algorithm. In last part of the chapter we present the result of over algorithm over a variety of sequences and also perform qualitative and quantitative assessment.

4.1 Data collection

We selected a variety of clips from movies and live theatre. A total of 12 sequences are selected from four different feature films and cover diverse scenarios like dyadic conversations, conversations in crowd, action scenes, *etc.* The clips include a variety of shots such as close ups, distant wide angle shots, stationary and moving camera *etc.* The native aspect ratio of all these sequences is either 2.76:1 or 2.35:1. The pace of the movie sequences vary from a cut every 1.6 seconds to no-cuts at all in a 3 minute sequence. The live theatre sequence were recorded from dress rehearsals of Arthur Miller's play 'Death of a salesman' and Tennessee Williams' play 'Cat on a hot tin roof'. All the 4 selected theatre sequences were recorded from a static wide angle camera covering the entire stage and have an aspect ratio of 4:1. These are continuous recordings without any cuts. The combined set of movie and live theatre sequences amount to a duration of about 52 minutes (minimum length of about 45 seconds and maximum length of about 6 minutes).

Five naive participants with normal vision (with or without lenses) were recruited from student community for collecting the gaze data. The participants were asked to watch the sequences resized to a frame size of 1366 $\times$ 768 on a 16 inch screen. The original aspect ratio was preserved during the resizing operation using letterboxing. The participants sat at approximately 60 cm from the screen. Ergonomic settings were adjusted prior to the experiment and system was calibrated. PsychToolbox [33] extensions for MATLAB were used to display the sequences. The sequences were presented in a fixed order for all participants. The gaze data was recorded using the 60 Hz Tobii Eyex, which is an easy to operate, low cost eye-tracker.

4.2 Gaze as an indicator of importance

The basic idea of video retargeting is to preserve what is important in video by removing what is not. We explicitly use gaze as the measure of importance and propose a dynamic programming optimization, which takes as input the gaze tracking data from multiple users and outputs a cropping window path which encompasses maximal gaze information. The algorithm also outputs the time stamps to introduce new cuts (if required) for more efficient storytelling. Whenever there is an abrupt shift in the gaze location, introducing a new cut in the cropping window path is a preferable option over panning movement (as fast panning would appear jarring to the viewer). However, the algorithm penalizes jump cuts (ensuring that the cropping window locations, before and after the cut are distinct enough) as well as many cuts in short succession (it is important to give the user sufficient time to absorb the details before making the next cut).

More formally, the algorithm takes as input the raw gaze data g_t^i of i^{th} user for all frames $t = [1 : N]$ and outputs a state $\epsilon = \{\mathrm{r}_t\}$ for each frame. Where the state $r_t \in [1 : W_o]$ (where W_o is width of the original video frames) selects one among all the possible cropping window positions. The optimization

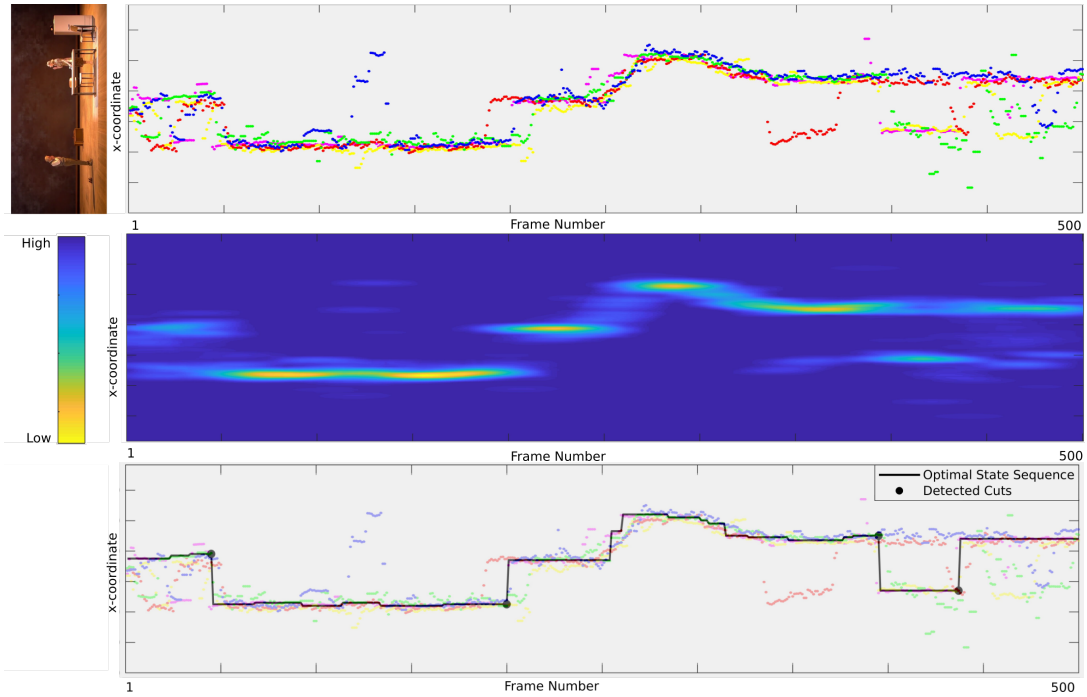


Figure 4.1 The x-coordinate of the recorded gaze data of 5 users for the sequence "Death of a Salesman" (top row). Gaussian filtered gaze matrix over all users, used as an input for optimization algorithm (middle row). The output of optimization: The optimal state sequence (black line) along with the detected cuts (black circles) for an output aspect ratio of 1:1(bottom row).

problem aims to minimize the following cost function:

$$E(\epsilon) = \sum_{t=1}^{N} E_s(r_t) + \lambda \sum_{t=2}^{N} E_t(r_{t-1}, r_t, d) \tag{4.1}$$

where the first term E_s penalizes the deviation of r_t from the gaze data at each frame and E_t is the transition cost, which computes the cost of transition from one state of another (considering both the options of camera movement and cut). Given the raw gaze data g_t^i, the unary term E_s is defined as follows:.

$$E_s(r_t) = M_s(r_t, t)$$

$$\text{where} \quad M_s(x,t) = \left(\sum_{i=1}^{u} M^i(x,t) \right) * \mathcal{N}(0,\, \sigma^2)$$

$$M^i(x,t) = \begin{cases} -1 & \text{if } x = g_t^i \\ 0 & \text{otherwise} \end{cases}$$

Here, $M^i(x,t)$ is a $W_o \times N$ matrix of i^{th} user gaze data and $M_s(x,t)$ is the sum of gaussian filtered gaze data over all users. Figure 4.1 (middle row) shows an example of matrix M_s computed from the corresponding gaze data (top). Essentially $E_s(r_t)$ is low, if r_t is close to the gaze data and increases as

r_t deviates from the gaze points. Using the combination of multiple user gaze data, makes the algorithm robust to both the noise induced by the eye-tracker and the momentary erratic eye movements of some users. We use a Gaussian filter (with standard deviation σ) over the raw gaze data to better capture the overlap between multi-user data, giving a lowest unary cost to areas where most users look at.

The pair-wise cost E_t considers a case wise penalty. The penalty differs if there is a new cut introduced or not. If there is no cut introduced, it is desired that the new state be closer to the previous state. If a new cut is introduced, it is desired to avoid a jump cut and also leave sufficient time from the previous cut. The term E_t is defined as follows:

$$E_t(r_{t-1}, r_t, d) = \begin{cases} \left(1 - e^{\frac{-4|r_t - r_{t-1}|}{W}}\right) & |r_t - r_{t-1}| \leq W \\ \left(1 + e^{\frac{-|d|}{D}}\right) & |r_t - r_{t-1}| > W, \end{cases} \tag{4.2}$$

where d is the duration from the previous cut and D is a parameter which controls the cutting rhythm and can be tuned for faster or slower pace of the scene. We set the value of D to 200 frames, which is roughly the average shot length used in movies between the 1983 and 2013 [13]. The first case in Equation 4.2 is considered, when the difference in consecutive states is less than W, the minimum width to avoid jump cut (we assume occurrence of jump cut if overlap between consecutive cropping windows is more than 25%). The cost is 0 when $r_t = r_{t-1}$ and gradually saturates towards 1, when $|r_t - r_{t-1}|$ approaches W. A transition of more than W indicates possibility of a cut, and then the pairwise cost is driven by the duration from the previous cut. The cost gradually decreases with increase in duration from the previous cut.

Finally, we solve Equation 4.1 using Dynamic Programming (DP). The algorithm selects a state r_t for each time t from the given possibilities (W_o in this case). We build a cost matrix $C(r_t, t)$ (where $r_t \in [1 : W_o]$ and $t \in [1 : N]$). Each cell in this table is called a $node$. The recurrence relation used to construct the DP cost matrix is a result of the above energy function and is as follows:

$$C(r_t, t) = \begin{cases} E_s(r_t) & t = 1 \\ \min\limits_{r_{t-1}} \left[E_s(r_t) + \lambda * E_t(r_{t-1}, r_t, d) + C(r_{t-1}, t-1)\right] & t > 1 \end{cases}$$

For each node (r_t, t) we compute and store the minimum cost $C(r_t, t)$ to reach it. A cut c_t is introduced at frame t, if the accumulated cost is lower for introducing a cut than keeping the position constant or panning the window. Backtracking is then performed from the minimum cost node in the last column to retrieve the desired path. Finally, the output of the algoritm is the optimized cropping window path $\epsilon = \{\mathrm{r}_t\}$ and the set of newly introduced cuts $\{c_t\}$. The time complexity of the algorithm is $O({W_o}^2 N)$ and the space complexity is $O(W_o N)$, which are both linear with N. An example of the generated optimization result is illustrated in Figure 4.1 (bottom row).

4.3 Optimization for cropping window sequence

The output of the dynamic programming optimization gives a cropping window path which maximizes the inclusion of gaze data inside the cropping window and the location of the cuts. However, this cropping window path does not comply with cinematic principles (leading to small erratic and incoherent movements). We further employ an L(1) regularized convex optimization framework, which aims to convert the rough camera position estimates into smooth professional looking camera trajectories, while accounting for cuts (original and newly introduced ones) and other relevant cinematographic aspects, as discussed in Section 2.2. This optimization takes as input, the original gaze data; the initial path estimate ($\epsilon = \{r_t\}_{t=1:N}$); the original cuts in the sequence, if any (computed using [1]); the newly introduced cuts c_t and outputs the optimized virtual camera trajectory, $\xi = \{(x_t^*, z_t)\}_{t=1:N}$. The optimization consists of several cost terms and constraints and we describe each of them in detail:

4.3.1 Data term:

The data term penalizes deviation of the virtual camera path (cropping window path) from the initial estimates (which eventually is capturing the gaze behavior). The term is defined as follows:

$$D(\xi) = \sum_{t=1}^{N} \left(max\left[|x_t^* - r_t| - \tau, 0\right]\right)^2 \tag{4.3}$$

The function penalizes if the optimized sequence x_t^* deviates from the initial estimate of the camera position r_t. However, it is relaxed with a rectifier linear unit function to avoid the penalty for small gaze movements and in turn to avoid brief and erratic camera movements. To summarize, the above cost function incurs a penalty only if the optimal path x_t^* varies from r_t, with more than a threshold, τ.

4.3.2 Movement regularization

As discussed in smooth and steady camera movement is necessary for pleasant viewing experience [47]. Professional cameramen avoid unmotivated movements and keep the camera as static as possible. When the camera is moved, it should start with a segment of constant acceleration followed by a segment of constant velocity and should come to a static state with a segment of constant deceleration. Early attempts modeled this behavior with heuristics [23], however recent work by Grundmann *et al.* [27] showed that such motions could be computed as the minima of an $L(1)$ optimization. In the similar spirit, we introduce three different penalty terms to obtain the desired camera behavior.

When $L(1)$ norm term is added to the objective to be minimized, or constrained, the solution typically has the argument of the $L(1)$ norm term sparse (i.e., with many exactly zero elements). The first term, penalizes the $L(1)$ norm over the first order derivative, inducing static camera segments:

$$M_1(\xi) = \sum_{t=1}^{N-1} (|x_{t+1}^* - x_t^*|). \tag{4.4}$$

The second term induces constant velocity segments by minimizing accelerations:

$$M_2(\xi) = \sum_{t=1}^{N-2} (|x^*_{t+2} - 2x^*_{t+1} + x^*_t|). \tag{4.5}$$

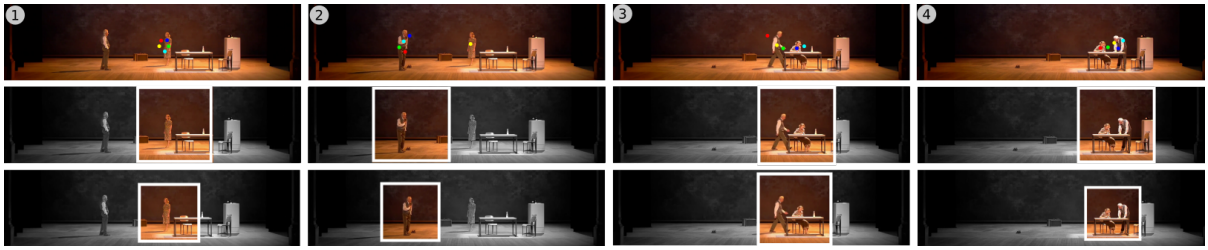
The third term minimizes jerk, leading to segments of constant acceleration:

$$M_3(\xi) = \sum_{t=1}^{N-3} (|x^*_{t+3} - 3x^*_{t+2} + 3x^*_{t+1} - 3x^*_t|). \tag{4.6}$$

Combining these three penalties yields camera movements consisting of distinct static, linear and parabolic segments.

4.3.3 Zoom

We perform zoom by varying the size of the cropping window (decreasing the size of the cropping window results in a zoom-in operation, as it makes the scene look bigger). The amount of zoom is decided based on the standard deviation of the gaze data, taking inspiration from previous work on gaze driven editing [31]. However, we use gaze fixations instead of the raw gaze data for computing the standard deviation. We observed that using fixation gives added robustness over the outliers/momentary noise. We use the EyeMMV toolbox [35] for computing the fixation, with a duration of 200 ms.



©Celestin, Theatre de Lyon

Figure 4.2 An example sequence where our method performs zoom-in and zooms-out action. The original sequence with overlaid gaze data (Top). The output of our algorithm without zoom (Middle) and with zoom (Bottom).

Let σ_t be the standard deviation of the gaze data at the fixation points at each frame. The ratio

$$\rho_t = 1 - 0.3 * \left(1 - \frac{\sigma_t}{\underset{k}{max}(\sigma_k)}\right) \tag{4.7}$$

$\rho_t \in [0.7, 1]$ is used as an indicator for the amount of zoom at each frame ($\rho_t = 1$ corresponds to the largest feasible cropping window and zoom-in happens as value of ρ_t decreases). We add the following penalty terms in the optimization:

$$Z(\xi) = \sum_{t=1}^{N} (z_t - \rho_t)^2, \tag{4.8}$$

which penalizes the deviation of zoom from the value σ_t at each frame. We further add $L(1)$ regularization terms (first order (M_1^z), second order (M_2^z) and third order (M_3^z)) over z_t to the objective function to avoid small erratic zoom-in zoom-out movements and ensure that whenever zoom action takes place, it occurs in a smooth manner.

4.3.4 Inclusion and panning constraints:

We introduce two sets of hard constraints. The first constraint enforces the cropping window to be always within the original frame i.e. $\frac{W_r}{2} < x_t^* < W_o - \frac{W_r}{2}$, which ensures a feasible solution (where W_r is the width of the retarget video). We add second constraint as upper bound on the velocity of the panning movement, to avoid fast panning movements. Cinematographic literature [39] suggests that a camera should pan in such a way that it takes an object at least 5 seconds to travel from one edge of the screen to the other. This comes out to roughly 6 pixels/frame for the video resolution used in our experiments and leads to following constraint: $|x_t^* - x_{t-1}^*| \leq 6$.

4.3.5 Accounting for cuts : original and newly introduced

Our algorithm is agnostic to the length and type of the video content; this means that the original video may include arbitrary number of cuts (original and newly introduced). This is in contrast to previous approaches [31], which solve the problem on a shot-by-shot basis. This generalization is achieved by relaxing the movement regularization around cuts. The following two properties are desired in the periphery of a cut: (a) The transition at the cut should be sudden; and (b) The camera trajectory should be static just before and after the cut, as cutting with moving cameras can cause the problem of motion mismatch [6];

To induce sudden transition, we make all penalty terms zero at the point of cut. We also make the data term zero, p frames before and after every cut to account for the delay a user takes to move from a gaze location to another (although the change of focus in the scene is instantaneous, in reality the viewer takes a few milliseconds to shift his gaze to the new part of the screen). Similarly to induce static segments before and after the cut, we make the second and third order $L(1)$ regularization zero in the same interval. However, we keep the first order $L(1)$ regularization term non zero on the entire optimization space, except at the exact point to cut, to allow for the transition. The parameter p is set to 5, because we use third order $L(1)$ term which uses 4 previous values.

4.3.6 Energy Minimization:

Finally, the problem of finding the optimal cropping window sequence can be simply stated as a problem of minimizing a convex cost function with linear constraints. The overall optimization function

is defined as follows:

$$\begin{aligned} \underset{x^*,z}{\text{minimize}} \quad & D(\xi) + \lambda_1 M_1(\xi) + \lambda_2 M_2(\xi) + \lambda_3 M_3(\xi) + \\ & Z(\xi) + \lambda_1 M_1^z(\xi) + \lambda_2 M_2^z(\xi) + \lambda_3 M_3^z(\xi) \\ \text{subject to} \quad & \frac{W_r}{2} \leq x_t^* \leq W_o - \frac{W_r}{2} \\ & |x_t^* - x_{t-1}^*| \leq 6, \\ & 0.7 \leq z_t \leq 1,\ t = 1, \ldots, N-1. \end{aligned} \tag{4.9}$$

As discussed in Section 2.2, the optimal cropping window, x_t^* usually starts panning, at the instant the actor starts moving, while the actual cameraman takes a moment to respond for the action. To account for this, we delay the optimal cropping window path, x_t^*, by 10 frames for each shot. The parameters, λ_1, λ_2, λ_3 can be changed to vary the motion model. Currently, we keep λ_1 higher, preferring static segments.

4.4 Results

The results are computed on all the 12 clips (8 movie sequences & 4 theatre sequences) mentioned in Section 4.1.

All the sequences were retargeted from their native aspect ratio to 4:3 and 1:1 using our algorithm. We also compute results using Gaze Driven Editing (GDR) [31] algorithm by Jain *et al.* for the case of 4:3 aspect ratio, over all the sequences. The results with GDR were computed by first detecting original cuts in the sequences and then applying the algorithm shot by shot. Some of the example results and comparisons are shown in Figure 4.2, Figure 4.3 and Figure 4.4. An explicit comparison on output with and without zoom is shown in Figure 4.2. All the original and retargeted sequences are provided in the supplementary material.

We used CVX [25] toolbox with MOSEK [2] for convex optimization. The parameters used for the algorithm are given in Table 4.1. Same set of parameters are used for all theatre and movie sequences.

4.5 Runtime

The proposed algorithm optimizes the cropped window for any length of video sequence with arbitrary number of cuts. Hence, it is independent of the video resolution and number of shots/cuts. The proposed algorithm takes around 40 sec for a 6 min video sequence (processing around 220 frames per second) on a laptop with i3 processor and 4GB RAM whereas [31] takes around 40 min for 30 sec sequence. We empirically observed that the complexity of our algorithm increases linearly with number of frames and takes about 10 minutes for retargeting a 90 minute movie. Dynamic programming has complexity of $O(W_o N)$, which grows linearly with the number of frames N.The convex optimization

Parameter	λ	σ	D	W	λ_1	λ_2	λ_3	τ
Values	2	15	200	$0.75W_r$	5000	500	3000	$0.1W_r$

Table 4.1 Parameters for path optimization algorithm and cut detection algorithm.

Type	**Our Method**	**GDR**
All	$89.63\%(\sigma = 4.96)$	$76.26\%(\sigma = 9.71)$
Movies	$90.28\%(\sigma = 4.63)$	$77.76\%(\sigma = 10.3)$
Theatre	$85.95\%(\sigma = 4.43)$	$74.02\%(\sigma = 6.27)$

Table 4.2 Comparing our method with GDR based on the mean percentage of gaze data included within the retargeted video at 4:3 aspect ratio.

solved using CVX-MOSEK [2] has a theoretical worst case complexity of $O(N^3)$ for each iteration. In practice it is extremely fast as the constraint-matrix in our case is sparse (it has non-zero coefficients only around the diagonals because of L1-regularization terms). MOSEK usually takes less than 100 iterations to solve any of the supported optimization problems. Hence, the proposed method works in real-time, processing around 150-200 frames per second.

4.6 Included gaze data

One measure for evaluating retargeting performance is to compute the percentage of gaze data included inside the cropped window, as suggested in [11] and [31]. Table 4.2 shows the average percentage of gaze data included over all the retargeted videos at 4:3 aspect ratio with our method and GDR. The global perspective and flexibility of our method allows it to capture considerably more gaze data than GDR. On average, our method is able to include about 13% more gaze data and the numbers reflect for both movie and theatre sequences. Smaller deviation ($\sigma = 4.96$ for our method, compared to $\sigma = 9.71$ for GDR) across sequences also confirms content agnosticism of our algorithm. The proportion of included gaze reduces to about 81.8% when the retargeted videos are rendered at 1:1 aspect ratio. In other words, when retargeting a video from 2.76:1 to 1:1, our algorithm preserves around 81% of gaze data while losing around 63% of the screen space.

4.7 User study evaluation

The primary motivation behind employing an $L(1)$ regularized optimization framework is to produce a smooth viewing experience. In order to examine whether our gaze-based retargeting approach positively impacted viewing experience, we performed a study with 16 users who viewed 20 original

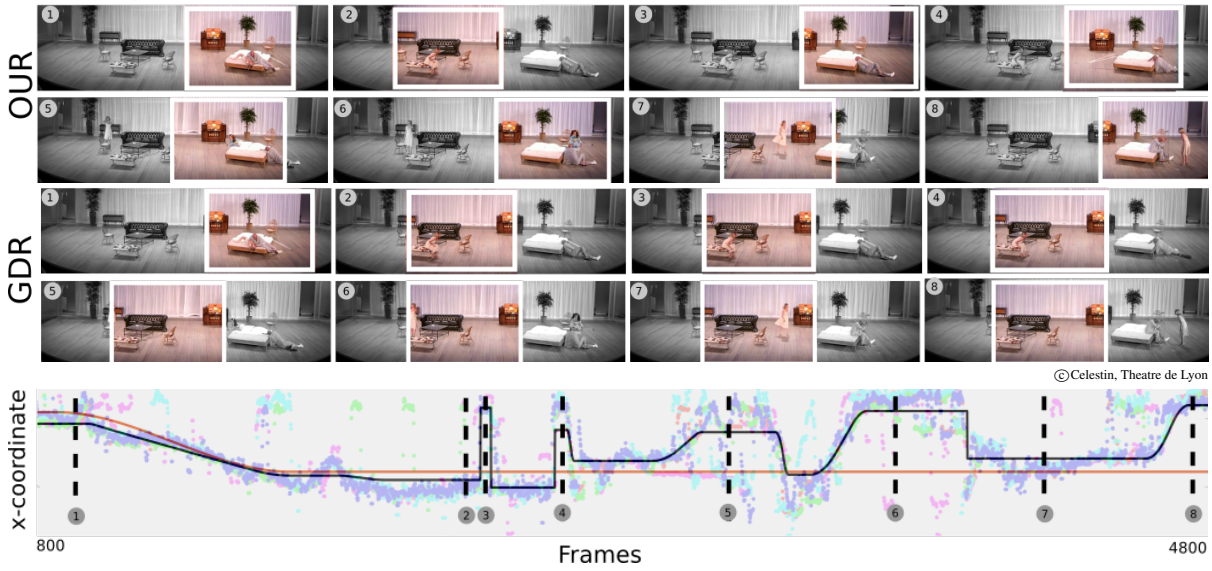


Figure 4.3 The figure shows original frames and overlaid outputs from our method and GDR (coloured, within white rectangles) on a long sequence. Plot shows the x-position of the center of the cropping windows for our method (black curve) and GDR (red curve) over time. Gaze data of 5 users for the sequence are overlaid on the plot. Unlike GDR, our method does not involve hard assumptions and is able to better include gaze data (best viewed under zoom).

and re-edited snippets from the 12 videos used in our study. The re-edited snippets were generated via (a) letterboxing, (b) the gaze driven re-editing (GDR) of Jain *et al.* [31] and (c) our approach. Details of the user study are presented below. The output aspect ratio was 4:3 in both the cases of GDR and ours, without considering the zoom in both cases.

4.7.1 Materials and methods

16 viewers (22–29 years of age) viewed 20 snippets of around 40 seconds length initially in their original form, followed by re-edited versions produced with letterboxing, GDR and our method shown in random order. Similar to the eye-tracking study, viewers watched the videos on a 16 inch screen at 1366 $\times$ 768 pixel resolution from around 60 cm distance. We followed a randomized 4$\times$4 Latin square design so that each viewer saw a total of five snippets (and each of them in four different forms), and such that all 20 snippets were cumulatively viewed by the 16 viewers.

We slightly modified the evaluation protocol of Jain *et al.* [31], who simply asked viewers regarding their preferred version of the original clip at a reduced aspect ratio. We instead asked viewers to provide us with real-valued ratings on a scale of 0–10 in response to the following questions:

$Q1$. Rate the edited video for how effectively it conveyed the scene content with respect to the original (scene content effectiveness or SCE).

$Q2$. Rate the edited video for how well it enabled viewing of actors' facial expressions (FE).

$Q3$. Rate the edited video for how well it enabled viewing of scene actions (SA).

$Q4$. Rate the edited video for viewing experience (VE).

These four questions were designed to examine how faithfully the examined video re-editing methods achieve the objective of the 'pan and scan' approach. $Q1$ relates to how well the re-editing process preserves the scene happenings and semantics. $Q2$ and $Q3$ relate to the cropping window movements, and how well they capture the main scene actors and their interactions (*e.g.*, when feasible, it would be preferable to watch both boxers in a boxing match rather than only the one who is attacking or defending). $Q4$ was added to especially compare the smoothness of the cropping window trajectory in the GDR and our approaches, and with the larger objective that re-editing approaches should not only capture salient scene content but should also be pleasing to the viewer's eyes by enforcing (virtual) camera pan and zoom only sparingly and when absolutely necessary.

Of the 20 snippets, 14 were extracted from movie scenes, and the remaining from theatre recordings. Since the content of these scenes varied significantly (professionally edited vs wide angle static camera recorded), we hypothesized that the re-editing schemes should work differently, and have different effects on the two video types. Overall, the study employed a 3×2 within-subject design involving the *re-editing* technique (letterboxing, GDR or our) and the *content type* (movie or theatre video) as factors.

4.7.2 User data analysis

Figure 4.5 presents the distribution of user scores in response to questions $Q1 - Q4$ for *all*, *theatre* and *movie* clips. In order to examine the impact of the snippet content and re-editing techniques, we performed a 2-way unbalanced ANOVA (given the different number of movie and theatre snippets) on scores obtained for each question.

For $Q1$, Figure 4.5 (left) shows that letterboxing scores are highest followed by our method and GDR respectively. This is to be expected as letterboxing preserves all the scene content with only a loss of detail, while re-editing techniques are constrained to render only a portion of the scene that is important. ANOVA revealed the main effect of both editing technique and content type. A post-hoc Tukey test further showed that the SCE scores were significantly different for Letterboxing (mean SCE score = 8.7) and GDR (mean SCE score = 7) at $p < 0.0001$, as well as our method (mean SCE score = 8.1) and GDR at $p < 0.001$, while the scores for our method and letterboxing did not differ significantly. These results suggest that ***our retargeting method preserves scene content better than GDR, but only slightly worse than letterboxing***.

Scores for $Q2$ in Figure 4.5 (left) show that our method performs better than the two competitors. ANOVA revealed the significant effects of the content type (p¡0.05) and re-editing technique (p¡0.0001) in this case. A Tukey test showed that the FE scores for our method (mean FE score = 8.3) were significantly different from either GDR (mean FE score = 7.5) or letterboxing (mean FE score = 7). These result reveal that ***our retargeting reveals actors' expressions most effectively, while letterboxing***

Type	Our	Letterboxing	GDR
All	43.6	39.3	17.1
Movies	35.9	46.6	17.5
Theatre	65.2	18.8	16

Table 4.3 User preferences (denoted in %) based on averaged and z-score normalized user responses for questions Q1-Q4.

performs worst in this respect due to loss of scene detail. For scene actions however, letterboxing again scored the highest while GDR scored the lowest. Tukey test for SA showed that both letterboxing (mean SA score = 8.2) and our approach (mean SA score = 8) performed significantly better than GDR (mean SA score = 7.2), while the difference between letterboxing and our approach was insignificant. ***So, our retargeting performs only slightly worse than letterboxing with respect to preserving scene actions***. Finally, ***our method and letterboxing achieve very comparable scores for viewing experience***, with both receiving significantly higher scores (mean VE score $\approx$ 8) than GDR (mean VE score = 6.9).

Figures 4.5 (middle) and (right) show the user score bar plots corresponding to theatre and movie snippets. Quite evidently, our FE, SA and VE scores are the highest for theatre clips, and are found to be significantly higher than either GDR or letterboxing via Tukey tests. Nevertheless, the superiority of our method diminishes for movie clips, with letterboxing and our approach performing very comparably in this case. Except for FE with theatre videos, GDR scores the least for all other conditions. These results again show that our method is able to capture theatre scenes best, and the main difference between theatre and movie scenes is that action is typically localized to one part of the stage in theatre scenes, while directors tend to effectively utilize the entire scene space in their narrative for movie scenes. Since our method is inherently designed to lose some scene information due to the use of a cropping window, it generates an output comparable to letterboxing in most cases. However, GDR performs the worst among the three considered methods primarily due to unmotivated camera movements and heuristic motion modeling which fails on longer shots. Table 4.3 tabulates the percentage of viewers that preferred our method over letterboxing and GDR upon z-score normalizing and averaging responses for $Q1 - Q4$. The numbers again reveal that our method is most preferred for theatre, but loses out to letterboxing for movie clips. Cumulatively, ***subjective viewing preferences substantially favor our method as compared to gaze based re-editing***.

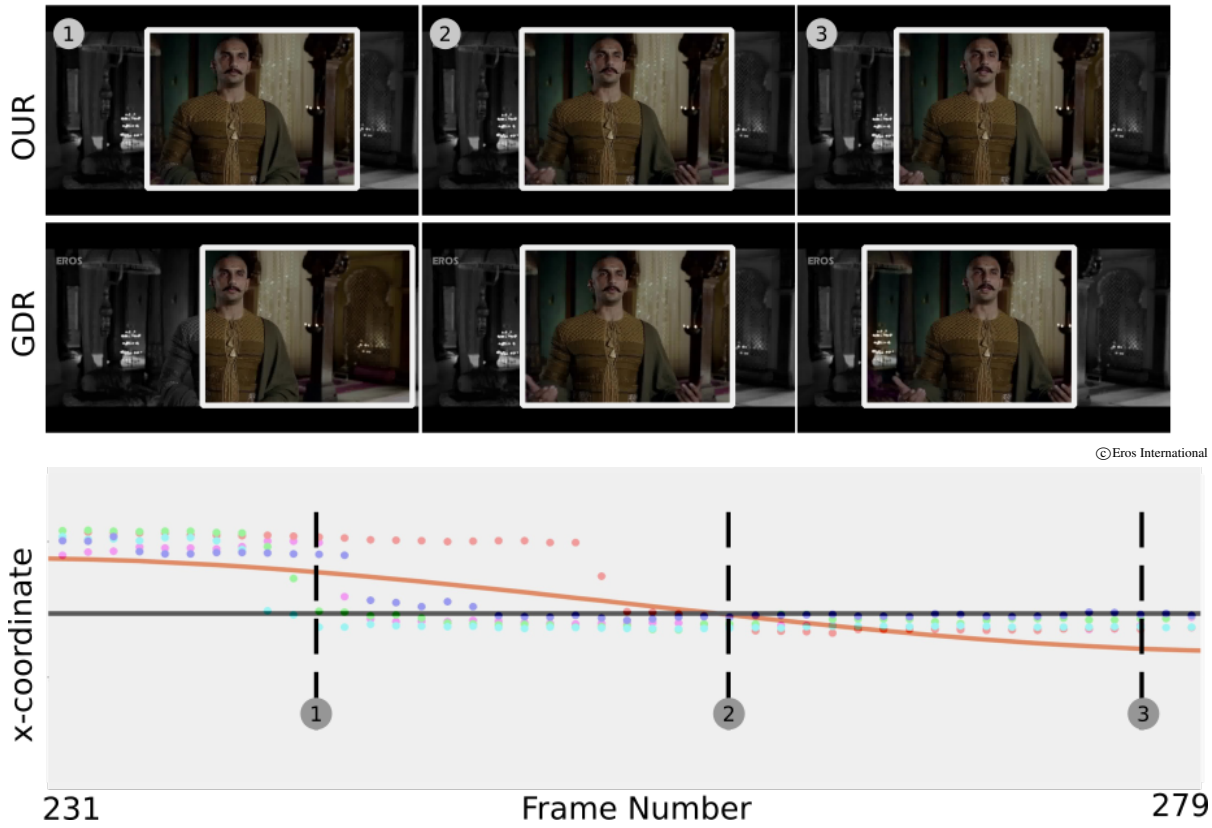


Figure 4.4 Frames from the original sequence cropped by the output of our algorithm and GDR (white rectangles). Corresponding gaze data (below) reveals that gaze is broadly cluttered around the shown character. Our algorithm produces a perfectly static virtual camera segment (black curve), while GDR results in unmotivated camera movement (red curve).

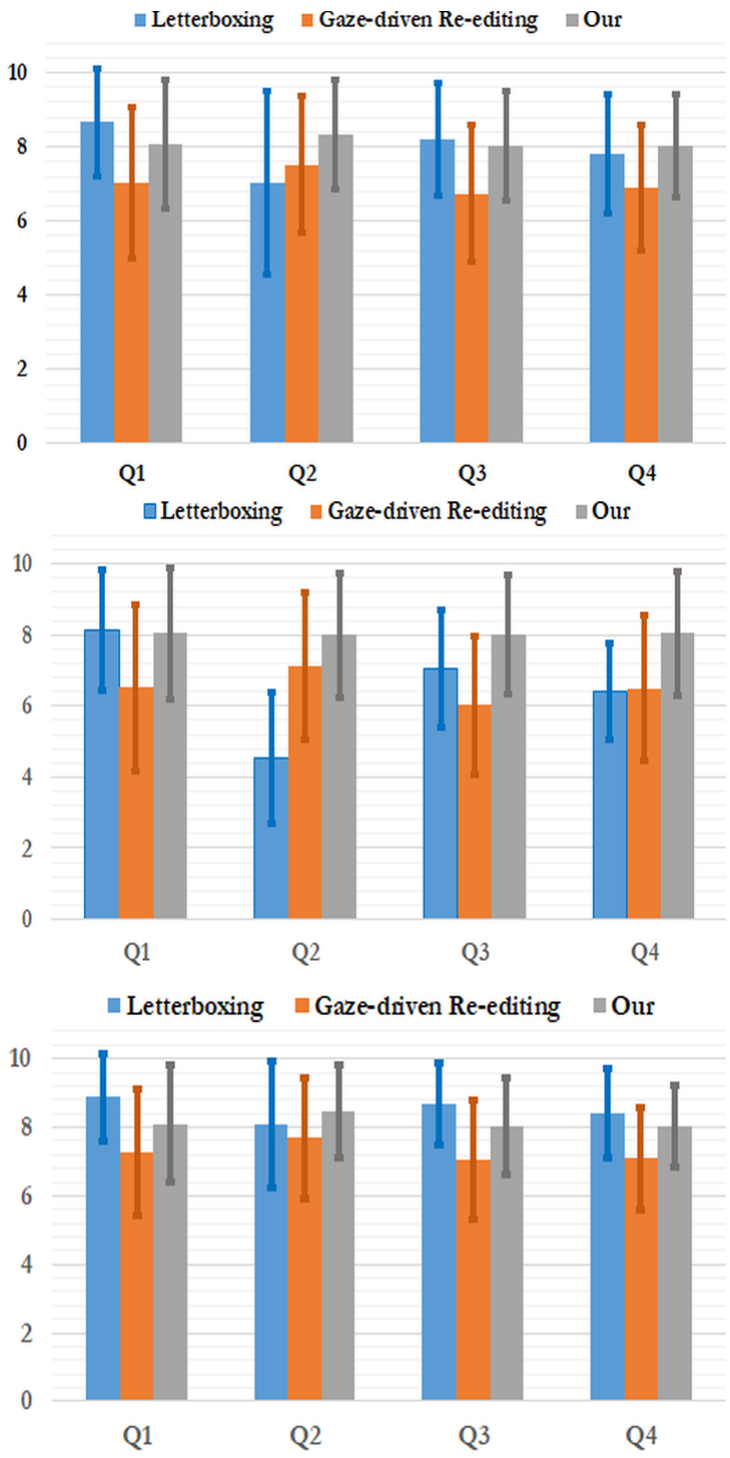


Figure 4.5 Bar plots showing scores provided by users in response to the four questions (Q1-Q4) posed for (left) *all* clips, (middle) *theatre* clips and (right) *movie* clips. Error bars denote unit standard deviation.

Chapter 5

Conclusion

In this Thesis, we looked at (1) an end-to-end framework to automatically computing split-screen compositions from a single master shot and (2) an approach to optimally retarget videos to any desired aspect ratio using the gaze data collected with the help of a gaze tracker.

Our experiments on split-screen compositions have shown that this is a difficult task in general, requiring many adjustments for correctly matching the positions and movements of actors in the different subframes at all times, and controlling transitions from one screen layout to another, depending on the actors groupings. By carefully analyzing the constraints that must be observed, we have cast this problem of split-screen composition as a series of nested convex optimization problems, which have a unique solution and can be solved efficiently offline. Experimental results demonstrate the generality and quality of the results and make our method suitable for displaying ultra high definition video on a variety of screen formats with optimal viewing comfort.

The retargeting algorithm we propose comprises of a two stage optimization to retain maximum gaze while complying with cinematic principles concerning pans, zooms and cuts. Our method stands out from the previous methods [31] as we use a heuristic-free motion modeling and avoid making any assumptions on the length or type of the input sequence. This makes our method capable of re-editing existing video/movie sequences as well as edit a entire new raw sequence. The versatility to edit a raw sequence allows for new scope in the field of video retargeting. A user can simply record the scene from a static/moving camera covering the entire scene and can later use a our algorithm to edit the recording based on gaze data. We motivate this application based on examples recorded from content rich theatre recordings and the user study confirms that the retargeted version better conveys the important details and significantly improves the overall viewing experience.Our algorithm is robust to noise and hence allows us to retarget video using gaze data obtained from a low cost Tobii Eyex ($100 euros$) eye tracker. These eye trackers can be easily connected to any computer/laptop and in fact may come integrated with laptops in future (https://imotions.com/blog/smi-apple-eye-tracking/), which creates the possibility of (a) creating personalized edits and (b) crowd sourcing gaze data for more efficient video editing. The current computational time of out algorithm is about 10 minutes to retarget a 90 minute video, which makes it suitable for directly editing full-length movies.

The user study we perform confirms that our approach allows users to obtain a better view of the scene emotion and actions, and in particular enhances viewing experience for static theatre recordings.

5.1 Limitations and future work

Both our approaches are currently limited to offline processing as it requires multiple stages of processing. A promising avenue for future research would be to learn to generate split screen compositions and re-edit videos online using training examples from the results generated by our methods.

SCC generation method was tested and evaluated with qualitative, subjective assessment on the special case of stage performances. We are planning to investigate other application scenarios (such as social events, conferences, meetings) as well as quantitative validation methods such as eye tracking studies in future work.Tracking is another source of limitations. In our implementation, we used an offline tracking method based on a generative model of actor's appearances [19] which correctly tracks the actor's upper bodies but fails to detect their hands, which can cause artifacts. Tracking the actors pose would bring substantial improvements to our method.

Our **retargeting** approach only optimizes over the x-position and zoom. The y-position is remains unaltered, which limits our algorithm to (a) retarget to any arbitrary aspect ratios and (b) to freely manipulate the compositions (for instance, retargeting from a long shot to a medium shot, *etc.*). The current version of our algorithm may result in videos where faces or body are cut by the frame boundary. However, it avoids cutting objects that are attended upon, and this problem can be partially handled via additional constraints based on human/object detection.

Related Publications

- **Moneish Kumar**, Vineet Gandhi, Remi Ronfard and Michael Gleicher.2017. **Zooming On All Actors: Automatic Focus+Context Split Screen Video Generation**. In *Eurographics 2017*
- **Moneish Kumar**, Kranthi Kumar Rachavarapu, Vineet Gandhi and Ramanathan Subramanian. 2018. **Watch to Edit: Video Retargeting using Gaze**. In *Eurographics 2018*

Bibliography

[1] E. Apostolidis and V. Mezaris. Fast shot segmentation combining global and local visual descriptors. In *ICASSP*, 2014.

[2] M. ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28).*, 2015.

[3] Y. Ariki, S. Kubota, and M. Kumano. Automatic production system of soccer sports video by digital camera work based on situation recognition. In *Multimedia, 2006. ISM'06. Eighth IEEE International Symposium on*, 2006.

[4] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM Trans. Graph.*, 26(3):10, 2007.

[5] M. Bianchi. Automatic video production of lectures using an intelligent and aware environment. In *Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia*, 2004.

[6] C. J. Bowen and R. Thompson. *Grammar of the Edit*. Focal Press, 2010.

[7] C. J. Bowen and R. Thompson. *Grammar of the Shot*. Taylor & Francis, 2013.

[8] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, and A. Torralba. Mit saliency benchmark. http://saliency.mit.edu/.

[9] Z. Bylinskii, A. Recasens, A. Borji, A. Oliva, A. Torralba, and F. Durand. Where should saliency models look next? In *European Conference on Computer Vision*, 2016.

[10] P. Carr, M. Mistry, and I. Matthews. Hybrid robotic/virtual pan-tilt-zom cameras for autonomous event recording. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 193–202, 2013.

[11] C. Chamaret and O. Le Meur. Attention-based video reframing: validation using eye-tracking. In *ICPR*, 2008.

[12] F. Chen and C. De Vleeschouwer. Personalized production of basketball videos from multi-sensored data under limited display resolution. *Computer Vision and Image Understanding*, 114(6):667–680, 2010.

[13] J. E. Cutting and A. Candan. Shot durations, shot classes, and the increased pace of popular movies, 2015.

[14] S. Daigo and S. Ozawa. Automatic pan control system for broadcasting ball games based on audience's face direction. In *Proceedings of the 12th Annual ACM International Conference on Multimedia*, MULTIMEDIA '04, pages 444–447, 2004.

[15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[16] T. Deselaers, P. Dreuw, and H. Ney. Pan, zoom, scan - time-coherent, trained automatic video cropping. In *CVPR*, 2008.

[17] R. Gal, O. Sorkine, and D. Cohen-Or. Feature-aware texturing. In *Proceedings of EUROGRAPHICS Symposium on Rendering*, pages 297–303, 2006.

[18] Q. Galvane, R. Ronfard, C. Lino, and M. Christie. Continuity editing for 3d animation. In *AAAI*, 2015.

[19] V. Gandhi and R. Ronfard. Detecting and Naming Actors in Movies using Generative Appearance Models. In *Computer Vision and Pattern Recognition*, 2013.

[20] V. Gandhi, R. Ronfard, and M. Gleicher. Multi-clip video editing from a single viewpoint. In *Proceedings of the 11th European Conference on Visual Media Production*, 2014.

[21] V. Gandhi, R. Ronfard, and M. Gleicher. Multi-Clip Video Editing from a Single Viewpoint. In *CVMP 2014 - European Conference on Visual Media Production*, 2014.

[22] S. O. Gilani, R. Subramanian, Y. Yan, D. Melcher, N. Sebe, and S. Winkler. Pet: An eye-tracking dataset for animal-centric pascal object classes. In *ICME*, 2015.

[23] M. Gleicher and F. Liu. Re-cinematography: Improving the camerawork of casual video. *ACM Transactions on Multimedia Computing Communications and Applications (TOMCCAP)*, 5(1):1–28, 2008.

[24] M. Gleicher and J. Masanz. Towards virtual videography (poster session). In *ACM Multimedia*, 2000.

[25] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, Mar. 2014.

[26] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *CVPR*, 2011.

[27] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *CVPR*, 2011.

[28] R. Heck, M. Wallick, and M. Gleicher. Virtual videography. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 3(1):4, 2007.

[29] R. Heck, M. Wallick, and M. Gleicher. Virtual videography. *ACM Transactions on Multimedia Computing Communications and Applications*, 3(1), 2007.

[30] E. Jain, Y. Sheikh, A. Shamir, and J. Hodgins. Gaze-driven video re-editing. *ACM Transactions on Graphics (TOG)*, 34(2):21, 2015.

[31] E. Jain, Y. Sheikh, A. Shamir, and J. Hodgins. Gaze-driven video re-editing. *ACM Transactions on Graphics (TOG)*, 34(2):21, 2015.

[32] H. Katti, R. Subramanian, M. Kankanhalli, N. Sebe, T.-S. Chua, and K. R. Ramakrishnan. Making computers look the way we look: exploiting visual attention for image understanding. In *ACM international conference on Multimedia*, pages 667–670, 2010.

[33] M. Kleiner, D. Brainard, D. Pelli, A. Ingling, R. Murray, C. Broussard, et al. What's new in psychtoolbox-3. *Perception*, 36(14):1, 2007.

[34] P. Krähenbühl, M. Lang, A. Hornung, and M. H. Gross. A system for retargeting of streaming video. *ACM Transactions on Graphics (TOG)*, 28(5), 2009.

[35] V. Krassanakis, V. Filippakopoulou, and B. Nakos. Eyemmv toolbox: An eye movement post-analysis tool based on a two-step spatial dispersion threshold for fixation identification. *Journal of Eye Movement Research*, 7(1), 2014.

[36] M. Kumar, V. Gandhi, R. Ronfard, and M. Gleicher. Zooming On All Actors: Automatic Focus+Context Split Screen Video Generation. In *Eurographics Workshop on Intelligent Cinematography and Editing*, 2017.

[37] M. Leake, A. Davis, A. Truong, and M. Agrawala. Computational video editing for dialogue-driven scenes. *ACM Transactions on Graphics*, 36(4):130:1–130:14, 2017.

[38] F. Liu and M. Gleicher. Video retargeting: Automating pan and scan. In *ACM Multimedia*, 2006.

[39] G. Millerson and O. Jim. *Video Production Handbook*. Focal Press, 2008.

[40] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *ICCV'09*, pages 151–158, Kyoto, Sept 2009.

[41] M. Rubinstein, A. Shamir, and S. Avidan. Improved seam carving for video retargeting. *ACM transactions on graphics (TOG)*, 27(3), 2008.

[42] A. Santella, M. Agrawala, D. DeCarlo, D. Salesin, and M. Cohen. Gaze-based interaction for semi-automatic photo cropping. In *Conference on Human Factors in Computing Systems*, CHI '06, pages 771–780, 2006.

[43] O. Schreer, I. Feldmann, C. Weissig, P. Kauff, and R. Schäfer. Ultrahigh-resolution panoramic imaging for format-agnostic video production. *Proceedings of the IEEE*, 101(1):99–114, 2013.

[44] A. Shamir and O. Sorkine. Visual media retargeting. In *ACM SIGGRAPH ASIA 2009 Courses*, pages 11:1–11:13, 2009.

[45] R. Subramanian, D. Shankar, N. Sebe, and D. Melcher. Emotion modulates eye movement patterns and subsequent memory for the gist and details of movie scenes. *Journal of vision*, 14(3):1–18, 2014.

[46] X. Sun, J. Foote, D. Kimber, and B. Manjunath. Region of interest extraction and virtual camera control based on panoramic video capturing. *Multimedia, IEEE Transactions on*, 7(5):981–990, 2005.

[47] R. Thomson and C. J. Bowen. *Grammar of the shot*. Focal Press, 2010.

[48] D. Vaquero, M. Turk, K. Pulli, M. Tico, and N. Gelfand. A survey of image retargeting techniques. In *Proc.SPIE*, pages 1–15, 2010.

[49] L. Wolf, M. Guttmann, and D. Cohen-Or. Non-homogeneous content-driven video-retargeting. In *ICCV*, 2007.

[50] Y.-Y. Xiang and M. S. Kankanhalli. Video retargeting for aesthetic enhancement. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 919–922. ACM, 2010.

[51] T. Yokoi and H. Fujiyoshi. Virtual camerawork for generating lecture video from high resolution images. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 4–pp, 2005.