Towards Data-Driven Cinematography and Video Retargeting using Gaze

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

Master of Science
in
Electronics And Communication Engineering
by Research

by

Kranthi Kumar Rachavarapu
201532563
kranthi.rachavarapu@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
April 2019

Copyright © KRANTHI KUMAR RACHAVARAPU, 2019
All Rights Reserved

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Towards Data-Driven Cinematography and Video Retargeting using Gaze" by KRANTHI KUMAR RACHAVARAPU, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. VINEET GANDHI

To

My Family

Acknowledgments

As I submit my MS thesis, I would like to take this opportunity to acknowledge all the people who helped me in my journey at IIIT-Hyderabad.

I would like to express my gratitude to my guide Dr.Vineet Gandhi for his constant support, guidance and understanding throughout my thesis work. This work would not have been possible with out him. I want to thank him for his patience and encouragement while I worked on various different problems before landing on the current work. His knowledge and advices have always helped whenever I faced a problem. I also want to thank him for his support during the highs and lows of my journey at IIIT-Hyderabad.

I thank Harish Yenala for all the discussions we had and for all the work we did together. I also want to thank Syed Nayyaruddin for all the fun and motivating discussions we had during our stay at IIIT-Hyderabad. I want to thank Narendra Babu Unnam, Maneesh Bilalpur, Sai Sagar Jinka and Moneish Kumar for all the wonderful interactions we had in CVIT Lab.

I want to thank all the people who participated in the data collection and user-study process on such a short notice during the course of this thesis work. I specially want to thank Sukanya Kudi for her help in setting up the annotation tool and for all the regular discussion we had in the lab.

Finally, I want to thank my family for their support and understanding in all of my decisions and endeavours.

Abstract

In recent years, with the proliferation of devices capable of capturing and consuming multimedia content, there is a phenomenal increase in multimedia consumption. And most of this is dominated by video content. This creates a need for efficient tools and techniques to create videos and better ways to render the content. Addressing these problems, in this thesis we focus on (a) Algorithms for efficient video content adaptation (b) Automating the process of video content creation.

To address the problem of efficient video content adaptation, we present a novel approach to optimally retarget videos for varied displays with differing aspect ratios by preserving salient scene content discovered via eye 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. Our approach is (a) content agnostic as the same methodology is employed to re-edit a wide-angle video recording or a close-up movie sequence captured with a static or moving camera, and (b) independent of video length and can in principle re-edit an entire movie in one shot.

The proposed retargeting algorithm consists of two steps. The first step employs gaze transition cues to detect time stamps where new cuts are to be introduced in the original video via dynamic programming. A subsequent step optimizes the cropping window path (to create pan and zoom effects), while accounting for the original and new cuts. The cropping window path is designed to include maximum gaze information and is composed of piecewise constant, linear and parabolic segments. It is obtained via $L(1)$ regularized convex optimization which ensures a smooth viewing experience. We test our approach on a wide variety of videos and demonstrate significant improvement over the state-of-the-art, both in terms of computational complexity and qualitative aspects. A study performed with 16 users confirms that our approach results in a superior viewing experience as compared to the state of the art and letterboxing methods, especially for wide-angle static camera recordings. As the retargeting algorithm takes a video and adapts it to a new aspect ratio, we can only use the existing information in the video which limits the applicability.

In the second part of the thesis, we address the problem of automatic video content creation by looking into the possibility of using deep learning techniques for automating cinematography. This type of formulation gives more freedom to the users to create content according to some their preferences. Specifically, we investigate the problem of predicting shot specification from the script by learning this association from real movies. The problem is posed as a sequence classification task using Long Short-

Term Memory (LSTM) network, which takes as input the sentence embedding and a few other high level structural features (such as sentiment, dialogue acts, genre *etc.*) corresponding a line of dialogue and predicts the shot specification for the corresponding line of dialogue in terms of *Shot-Size*, *Act-React* and *Shot-Type* categories.

We have conducted a systematic study to find out effect of the combination of features and the effect of input sequence length on the classification accuracy. We propose two different formulations of the same problem using LSTM architecture and extensively studied the suitability of each of them to the current task. We also created a new dataset for this task which consists of 16000 shots and 10000 dialogue lines. The experimental results are promising in terms of quantitative measures (such as classification accuracy and F1-score).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The increase in the demand for video content has given rise to the new digital broadcast players such as Netflix, Amazon etc., and various large production houses. This has created an explosion in filmmaking in terms of the number of films released per year from various different movie industries. The growing interest in this sector along with the rise of large production houses has streamlined the process of filmmaking by creating various specialized roles and responsibilities for a variety of craft such as screenwriter, director, editor, cinematographer, and others.

On the other hand, the advent of digital cameras opened up the possibility of making a featurefilm/ short-film for novice filmmakers. The lack of necessary resources forces the independent filmmaker to take up multiple roles (such as write, direct, edit, shoot etc) as opposed to the industrial model. However, these tasks require a significant knowledge and expertise on each of these crafts along with experience on many software tools (for editing, shooting etc.) and the quality of the output depends on the filmmakers expertise in each of these multiple fields.

In general, these videos are created with a specific targeted audience and specific viewing devices in mind. For examples, all the TV shows have an aspect ratio of 16:9 where as the feature films have 2.35:1 aspect ratio in general. The subjective viewing experience of targeted audience depends on these viewing devices. However, the advent of various devices capable of viewing video content (smart phones, PDA, TV etc) creates a need for efficient rendering techniques for better viewing experience.

Hence, creating automated tools for any of these tasks not only improves the viewing experience but also removes the hurdles of learning and enable the users to create videos that are cinematically good.

1.1 Problem Definition

In this thesis, we focus on the following two problems: (a) Video Retargeting using Gaze which focuses on efficient rendering techniques for the existing video content (b) A data driven approach for automating cinematography for facilitating video content creation.

1.1.1 Problem 1: Video Retargeting Using Gaze

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.

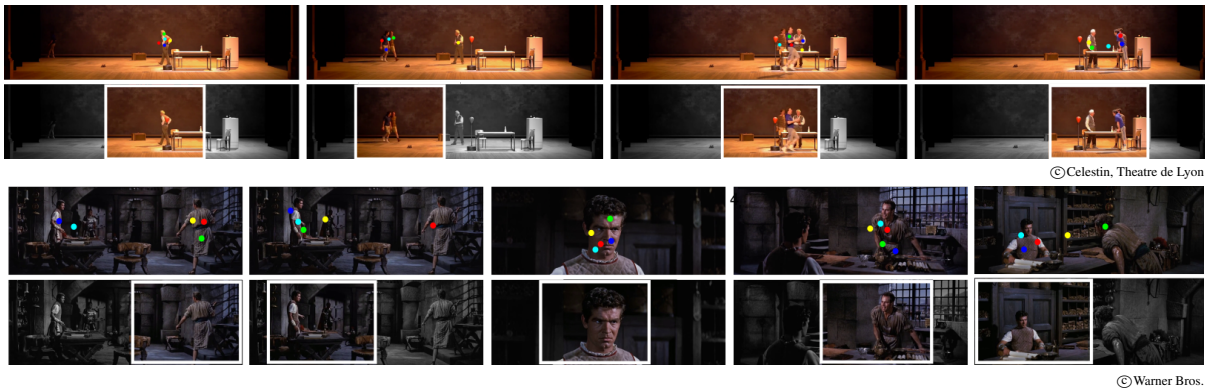


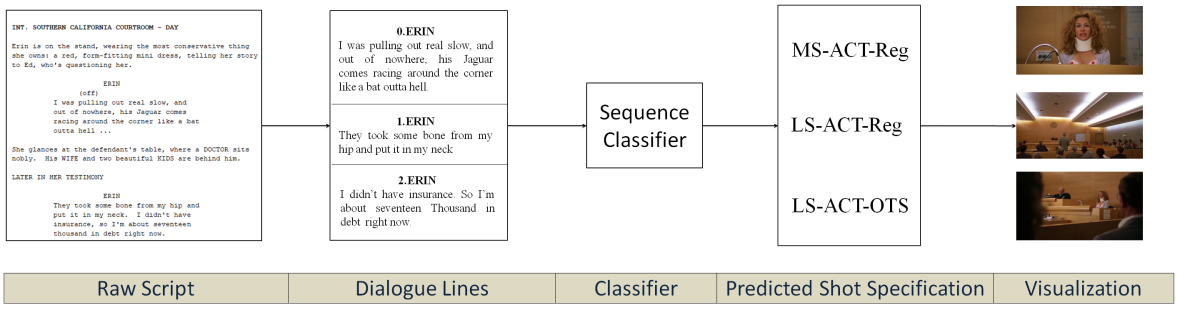
Figure 1.1: We present an algorithm to retarget a widescreen recording to smaller aspect ratios. The original recording with overlaid eye gaze data from multiple users (each viewer is a unique color) and the results computed by our algorithm (white cropping window) are shown. Our algorithm is content agnostic and can be used to edit a theatre recording from a static camera (top) or re-edit a movie scene (bottom).

To address this problem, we propose a novel retargeting methodology via gaze based re-editing employing convex optimization. We use gaze as a measure of importance and estimate a cropping window path that traverses through the original video introducing pan, cut and zoom operations. Figure 3.1 shows two example sequences and the corresponding retargeted output using the proposed method. The main challenge here is to preserve as much of the original video content as possible without compromising on the aesthetics of the video in terms of cinematographic principles and viewing experience. Another challenge is to create an algorithm that can deal with the possible variations in the input video in terms of scene content (such as action scenes, conversation scenes *etc.*) and different camera motion(such as fast-paced videos to high-velocity pans)

1.1.2 Problem 2: A Data Driven Approach for Automating Cinematography for Facilitating Video Content Creation

Recent advancements in 3D animated realistic rendering and gaming environments have created a need for techniques to automate the conventional process of cinematography. There is a growing interest in automating the process reproducing a specific style or genre to some new content.

To address this problem, we look into the possibility of using Deep Learning techniques for automating video content creation. Specifically, we focus on the possibility of predicting shot specification from the script by learning this association from real movies. We pose this problem as sequence classification task using LSTM which will learn the patterns/style in the data and reproduce them on the new data. Figure 1.2 shows the pipeline of the this process. We investigate the extent of such automation by looking into various architectures and combination of features.



©Celestin, Theatre de Lyon

Figure 1.2: We propose a data driven approach for predicting shot specification for each dialogue line in the input script. Dialogue lines are parsed from the input script. A Classifier is trained using the dialogue lines and the corresponding shot specification (obtained from movies) data. *(MS:Meidum-Shot; LS:Long Shot; ACT: Action Shot; Reg: Regular Shot; OTS: Over-The-Shoulder)*

The problem of predicting shot specification from a raw script is a challenging task for the following reasons:

- In general, every director has his own style of making. Hence, the process of learning patterns across genre can be a difficult task for any learning algorithm.
- There are no datasets available online for this kind of task. This forces us to create our own dataset which is the bottleneck in the entire experimentation process.

1.2 Contributions

The main contributions of this thesis are following.

1. **Video Content Adaptation:** We propose a novel, optimization-based framework for video re-editing utilizing minimal user gaze data. To this end, it can work with any type of video (professionally created movies or wide-angle theatre recordings) of arbitrary length to produce a re-edited video of any given size or aspect ratio.

2. **Video Content Creation:** We propose a data-driven approach for facilitating the video content creation by proposing an LSTM based framework for predicting shot specification from raw script.

3. **Dataset:** We have created a new movie-dataset for computational cinematography from 6 Hollywood movies. These movies span different genres such as action, drama, thriller, comedy, romance *etc.*The dataset contains 16000 movie shots with 10000 dialogue sequences in them. This dataset can be used for many other tasks such as automating the pipeline for further dataset creation, analyzing script to shot relation *etc.*

1.3 Thesis Outline

The rest of the thesis is organized as follows. In Chapter 2, we provide the necessary background useful for further chapters. We briefly discuss the process of film making and the complexity of automating such process. In Chapter 3, we look into the problem of video retargeting. We propose a novel optimization framework which is efficient in terms of both run-time and the quality of the retargeted video. Chapter 4 focuses on the possibility of using deep learning, specifically LSTM, for automating cinematography. Here we look into the sub-problem of prediction the shot specification from a raw script. We discuss in detail about the extent of automation and limitations. Chapter 5 provides a summary of our work with a way forward.

Chapter 2

Background

2.1 Cinematography Background

In this section, we provide a basic outline of filmmaking process and familiarize the readers with the common terminology of cinematography which will be used in the rest of the thesis.

2.1.1 Script and Storyboards

The process of making a film starts with an initial idea about the story. The screenwriter works on the story to create a script. A script is a written document that focusses on narrative aspects of story. Here, the story is logically broken down into scenes with information about the visual elements, dialogues, actors, actions, emotions and setting of the scene *etc.*.

Every director will have their own distinct style of film making (Directing and/or writing). The script will have information regarding narrative aspects, not about the visual aspects. Hence, they assist to create storyboard or directors script which describes how to shoot each element of screenplay in terms of composition and sequence. These visualization tools helps in fine tuning the narrative ideas before the actual shooting begins.

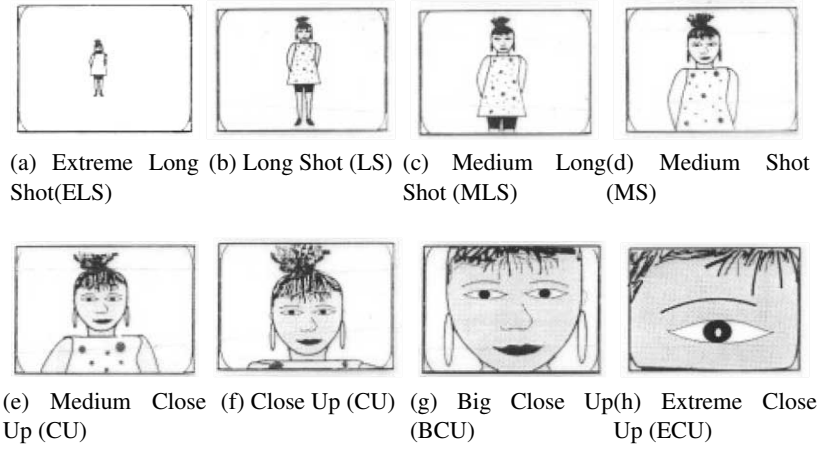
2.1.2 Filming and Composition

This stage of the film-making is called Production and this is where the actual film will be shot in parts. In general, the films were shot out of continuity. And the only guide in this non-linear order of shooting is the film script and/or storyboard. The director uses the script to guide and actors and the cinematographer uses the storyboard to capture the shot by controlling camera and lighting.

In filmmaking, a shot is the basic building block, which refers to an uninterrupted camera recording. A scene refers to the set of events/shots that occur in a single location during a continuous time. In films, each shot composed to express an idea or an element of the narrative aspect. And shot composition plays a major role in conveying this idea. There are different elements in a shot composition. A few of them are given below:

2.1.2.1 Shot Size

Shot size is one of the universally used elements of shot composition. It specifies the relationship between the frame and the subject in the frame. Fig 2.1 shows an example of different shot sizes for human as the subject. However, shot size can be defined for any other object or location.



(a) Extreme Long Shot(ELS) (b) Long Shot (LS) (c) Medium Long Shot (MLS) (d) Medium Shot (MS)

(e) Medium Close Up (CU) (f) Close Up (CU) (g) Big Close Up (BCU) (h) Extreme Close Up (ECU)

Figure 2.1: Examples of different Frame Size

Although there are no absolute rules in using the different types of shot sizes in movies, each one of them conveys a different emotion. For example, close-up shots are mostly used in dialogue scenes or to convey strong emotion where as long shots are used to establish a scene or show action sequences.

2.1.2.2 Camera Angle

A variation in camera angles are used for many reasons such as establishing a location, develop a mood, to establish a scene or to show the relation between subject and the camera point of view *etc.*

In most of the shots, the camera is at the eye-level of the subject. However, there are other variations such as low angle and high angle shots. All these are variations in vertical camera angle. The horizontal camera angles include frontal (which shows the face of a character), profile (which shows the side view of a character) and read (which shows the back of a character). All these combinations are used to create new perspectives and compositions that are relevant to the narrative aspects of the story.

2.1.2.3 Camera Movement

Camera movement is an important aspect of shot composition as it gives the flexibility of shooting a scene in less number of shots. The following are popular techniques used to achieve camera movement:

- Pan and Tilt: In this type of shot, the camera rotates around its fixed axis either horizontally or vertically. Hence, it is a static shot. They create the effect of fast horizontal or vertical movement without physically moving the camera to track actors or objects. A good pan/tilt movement should be smooth and steady, ”leading” the movement of the subject [75].

- Track and Crane: Here, the camera follows or tracks the subject, usually with camera mounted on a track. Hence these are moving shots.

2.1.2.4 Aspect Ratio

Aspect Ratio (AR) is the ratio of the width to the height of a frame. Although aspect ratio is a subtle and unnoticed feature, narrative aspects play an important role in the choice of aspect ratio. For example, The wider aspect ratio makes the audience mode aware of the shot environment where as smaller aspect ratios create a feeling of claustrophobic feel or ancient time.

Sometimes the viewing devices and the targeted audience plays a major role in deciding the aspect ratio. For example all of the current feature films are shot at 2.35:1 AR because of the wider theater screen whereas TV shows have an aspect ratio of 16:9 to match the TV or phone aspect ratio.

2.1.2.5 Shot-Reaction Shot

This framing technique is used as a part of continuity editing. During the dialogue shots, the character interaction can become monotonous after some point. This can be avoided by changing the shot to show the reaction shot intermittently. For the purpose of the thesis, if a shot shows the speaker we call it an Act-Shot(AS) and if a shot shows the reaction of a certain character in relation to the dialogue, we call it a Reaction Shot(RS)

2.1.2.6 Shot Type based on Camera Position

Different compositions can be achieved by varying the camera position. A few widely used categories are given below:

- Over the Shoulder (OTS): In this case, the camera takes the role-objective observer and shows the relation between the speaker and listener in conversation scenes.

- Point of View(POV): The camera takes the subjective role and gives the point of view of a character.

- Regular(Reg): The camera role can be subjective or objective. For our work, a shot is called Regular if it is not OTS and POV.

2.1.2.7 Shot Type based on Number of Characters in the Frame

Shots can be categorized based on the number of characters in the frame:

- Solo-shot: Its a shot with a single character performing some act in the frame.
- Multiple Character shot: There are many sub-classes for Multiple Character Shot: two-shot (shows two characters), three-shot (showing three stationary characters), four-shot (four characters in a frame) and more character shots.

2.1.3 Editing

Editing is a process of controlling the order in which the shots are organized to convey the story information to the viewer. The shots are rarely filmed in the same order as they were presented in the script. The editing process of rearrangement, selection, cutting shots and combining them into sequences with a goal of storytelling.

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. As each shot or view of the action is supposed to show new information it also makes sense that one should edit shots with a significant difference in their visual composition. This will avoid what is known as a jump cut two shots with a similar framing of the same subject, causing a visual jump in either space or time.

Pacing is the rate at which the change in the shots occurs. The pace of the cut should go along with the story. There should always be a balance between fast and slow-paced scenes. For example, action sequences or dramatic sequences are fast-paced whereas slow-paced sequences are used to establish a scene or develop a character.

2.2 Related Works on Automatic Cinematography

Film making is a creative and collaborative process. And automating any aspect of this is very difficult because there are only guidelines for film making not rules. However there are 3 aspect that can be automated and they are (a) Composition (b) Filming or camera control and (c) Editing. We give a brief overview of previous works on automating the above aspects of cinematography in the following sections.

2.2.1 Shot Composition

Composition deals with the placement of visual elements on the frame. Millerson and Owens [61] stress 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). Many of the previous works have analyzed the

framing techniques and gave interpretations [19, 67, 83]. Many other works such as [12, 74] tried to predict the quality of videos in terms of style and aesthetics from low-level computational features and conducted user study to understand the user experience. [82, 2] addressed the aspects like pacing and saliency in movies for shot classification tasks.

2.2.2 Filming: Camera Control

Camera Control involves specifying camera viewpoints such as camera position and camera path *etc.*Virtual camera control involves specifying these parameters for a virtual camera in 3D graphics whereas the other type involves manipulating real camera in actual shooting.

There are a few guidelines in cinematography on camera motion. A few of them are given below: Importance of a steady camera has been highlighted in Thomson and Christopher's 'Grammar of the shot' [75]. They suggest that

a. The camera should not move without a sufficient motivation (as it may appear puzzling to the viewer) and brief erratic pans should be avoided

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.

c. The fast panning movements should be avoided as much as possible, as it may lead to breakup of the movement [61].

d. Apart from panning, another crucial parameter which needs to be controlled is the amount of zoom, which is often correlated with the localization and intensity of the scenes [61].

In virtual cinematography, there are 2 ways to achieve camera control. The first one is manual camera control where the user decides the camera parameters. [80, 53, 17, 24] are a few works in this type of setting where the users have full control over camera using some metaphors or controllers. The other technique is automating the virtual camera control. These type of methods take some specification from users in terms of requirements and uses some form optimization methods to generate the camera position and path [72, 7, 55]

There are many recent works which focuses on aerial cinematography where the drones are used to film scenes. [62, 29, 26, 41] are a few relevant works in this direction. These techniques use path planner algorithms for camera control in real environment that satisfies the user specification.

2.2.3 Editing

Many techniques have been proposed in literature to address the problem of partially (or interactively) or fully automating the editing process.

[37, 5, 56, 51] are a few examples that use a model, whose parameters have been specified by user, to edit the video footage. [38, 58, 70] propose methods for editing lectures and social gathering. All these techniques produce a single editing sequence as output as there is not user intervention. The other type of methods [15, 71, 10] uses semi-automatic techniques to remove bad segments, annotate clips or rearrange clips and gives the control over editing process to the user.

2.3 Conclusion

In this section, we have outlined the general framework of filmmaking and discussed the common terminology of cinematography. We have also discussed some of works on automating the cinematography in three aspects: (a) Composition (b) Filming and (c) Editing. In this thesis we focus on automating the composition and editing aspects of cinematography and the following sections give a detailed account of this.

Chapter 3

Video Retargeting using Gaze

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 [50] 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 [69]. 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 [57] 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 [77]. 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 [45, 73]. Recently, Jain *et al.* [43] 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.

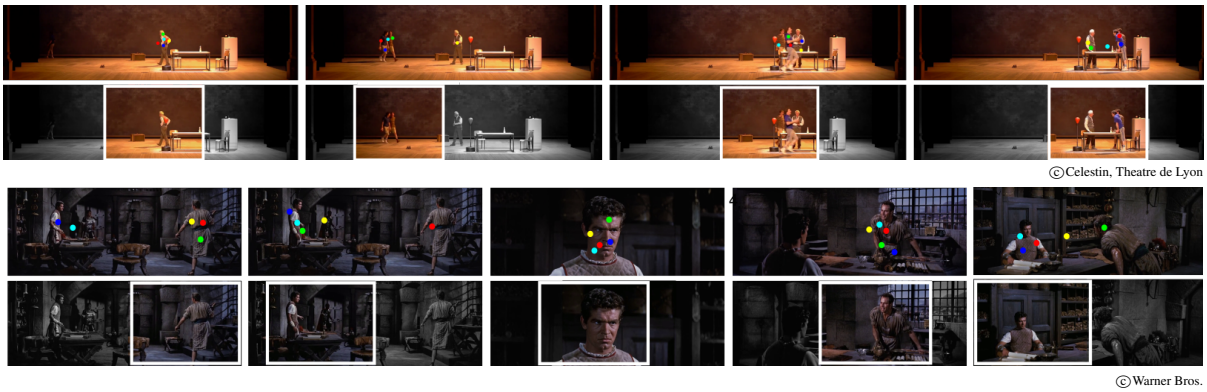


Figure 3.1: We present an algorithm to retarget a widescreen recording to smaller aspect ratios. The original recording with overlaid eye gaze data from multiple users (each viewer is a unique color) and the results computed by our algorithm (white cropping window) are shown. Our algorithm is content agnostic and can be used to edit a theatre recording from a static camera (top) or re-edit a movie scene (bottom).

To address these problems, we propose a novel retargeting methodology via gaze based re-editing employing convex optimization. Our work is inspired from GDR [43] and also estimates a cropping window path traversing through the original video introducing pan, cut and zoom operations in the process (Figure 3.1). 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.

3.1 Related Work

Several media retargeting solutions have been proposed, and they can be broadly categorized into three different categories: discrete, continuous and cropping-based approaches. Discrete approaches [6, 66, 63] judiciously remove and/or shift pixels to preserve salient media content. Examples include the relative shifting of pixels [63], or adding/removing connected seams (paths of low importance) from

the image [6]. Continuous approaches [28] 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 [66, 79, 48]. 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 [57, 23, 81] have been made to automate the 'pan and scan' or re-editing process. 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 [57] define the virtual camera path as a combination of piecewise linear and parabolic segments. Grundmann *et al.* [36] employ an $L(1)$ regularized optimization framework to produce stable camera movements thereby removing undesirable motion. Gleicher and colleagues [34, 57, 39, 31] 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.

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 [73, 32]. Leake *et al.* [52] 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.* [30] 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 [45, 73], to infer salient scene content.

Santella *et al.* [68] employ eye tracking for photo cropping so as to satisfy any target size or aspect ratio. More recently, Jain *et al.* [43] 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.

3.2 Method

In this section, we formalize the problem statement to pose it as an optimization and also discuss the data collection process.

3.2.1 Problem Statement

The proposed algorithm takes as input

(a) a sequence of frames $t = [1 : N]$, where N is the total number of frames;

(b) the raw gaze points over multiple users, g_t^i, for each frame t and subject i

(c) the desired output aspect ratio.

The output of the algorithm is 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 first explain the data collection process and then describe the two stages of our algorithm.

3.2.2 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). Table 3.1 gives the corresponding details.

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

ID	Clip Name	Type	Clip Length (sec)	No of Cuts	Aspect Ratio
1	Cat on a Hot Tin Roof	Theater Recording	190	0	4:1
2	Death of a Salesman - 1	Theater Recording	320	0	4:1
3	Death of a Salesman - 2	Theater Recording	75	0	4:1
4	Gravity	Movie	198	0	2.4:1
5	Ben-Hur: Chariot Race	Movie	300	121	2.76:1
6	Rush: Racing	Movie	318	194	2.4:1
7	Ben-Hur: Rowing	Movie	175	13	2.76:1
8	It's a Mad World: The shareout	Movie	340	26	2.76:1
9	Death of a Salesman - 3	Theater Recording	45	0	4:1
10	Ben-Hur -1	Movie	150	3	2.76:1
11	Ben-Hur -2	Movie	210	20	2.76:1
12	Bajirao Mastani	Movie	221	63	2.35:1

Table 3.1: Description of the videos used for creating Dataset

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 [46] 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.

3.2.3 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 problem aims to minimize the following cost function:

$$E(\epsilon) = \sum_{t=1}^{N} E_s(r_t) + \lambda \sum_{t=2}^{N} E_t(r_{t-1}, r_t, d) \tag{3.1}$$

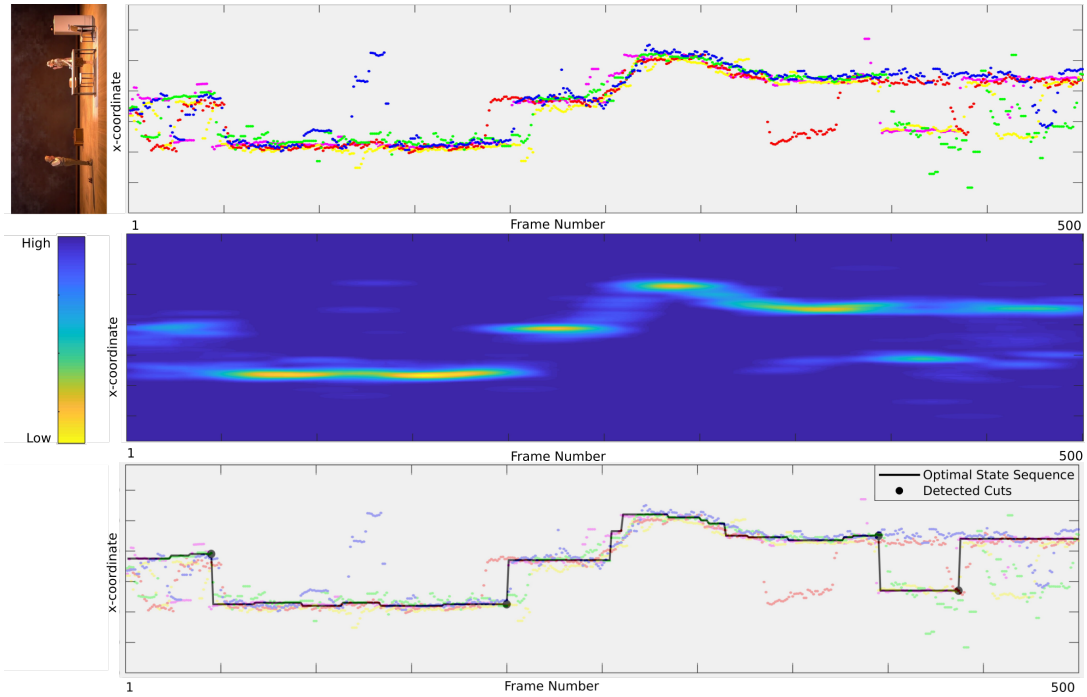


Figure 3.2: The x-coordinate of the recorded gaze data of 5 users for the sequence "Death of a Salesman" (top row). Gaussian filtered gaze matrix over all users, used as an input for optimization algorithm (middle row). The output of optimization: The optimal state sequence (black line) along with the detected cuts (black circles) for an output aspect ratio of 1:1(bottom row).

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 3.2 (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 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:

$$E_t(r_{t-1}, r_t, d) = \begin{cases} \left(1 - e^{\frac{-4|r_t - r_{t-1}|}{W}}\right) & |r_t - r_{t-1}| \leq W \\ \left(1 + e^{\frac{-|d|}{D}}\right) & |r_t - r_{t-1}| > W, \end{cases} \tag{3.2}$$

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 [19]. The first case in Equation 3.2 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 3.1 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 3.2 (bottom row).

3.2.4 Optimization for cropping window sequence

The output of the dynamic programming optimization is 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 3.2.1. 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 [3]); 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:

3.2.4.1 Data term

The data term should penalize the virtual camera path (cropping window path) from the initial estimates (which eventually is capturing the gaze behavior). Hence, the term can be defined as follows:

$$D(\xi) = \sum_{t=1}^{N} (x_t^* - r_t)^2 \tag{3.3}$$

The above function penalizes if the optimized sequence x_t^* deviates from the initial estimate of the camera position r_t. This forces the optimized sequence x_t^* to be as close the gaze data as possible. However, simply following gaze data will add unnecessary pan action. This can be avoided by using the following equation:

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

Here, the deviation 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.

3.2.4.2 Movement regularization

As discussed in Section 3.2.1, smooth and steady camera movement is necessary for pleasant viewing experience [75]. 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 [33], however recent work by Grundmann *et al.* [36] 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:

$$M_1(\xi) = \sum_{t=1}^{N-1} (|x_{t+1}^* - x_t^*|). \tag{3.4}$$

The second term induces constant velocity segments by minimizing accelerations:

$$M_2(\xi) = \sum_{t=1}^{N-2} (|x^*_{t+2} - 2x^*_{t+1} + x^*_t|). \tag{3.5}$$

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

$$M_3(\xi) = \sum_{t=1}^{N-3} (|x^*_{t+3} - 3x^*_{t+2} + 3x^*_{t+1} - 3x^*_t|). \tag{3.6}$$

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

3.2.4.3 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 [43]. However, we use gaze fixations instead of the raw gaze data for computing the standard deviation. A fixation is when the eyes remain relatively stationary in a position for a certain amount of time. We observed that using fixation gives added robustness over the outliers/momentary noise. We use the EyeMMV toolbox [49] for computing the fixation, with a duration of 200 ms.

Let f_t^i be a variable which indicates if g_t^i is a fixation or not. It is given by

$$f_t^i = \begin{cases} 1 & \text{if } g_t^i \text{ is a fixation point} \\ 0 & else \end{cases}$$

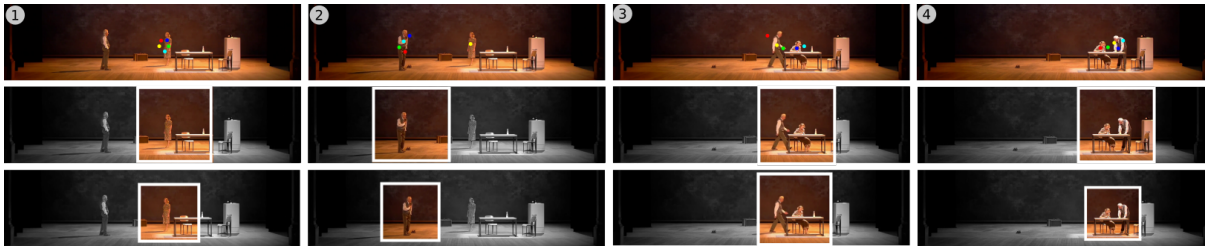
Let σ_t be the standard deviation of the gaze data at the fixation points at each frame and it is defined as $\sigma_t = \underset{\forall i}{std}\left(g_t^i\right)$. And the ratio

$$\rho_t = \begin{cases} 1 - 0.3 * \left(1 - \frac{\sigma_t}{\underset{k}{max}(\sigma_k)}\right) & \text{if } f_t^i = 1 \text{ for any i} \\ 1 & else \end{cases} \tag{3.7}$$

$\rho_t \in [0.7, 1]$ 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:

$$Z(\xi) = \sum_{t=1}^{N} \left(z_t - \rho_t\right)^2, \tag{3.8}$$

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.



©Celestin, Theatre de Lyon

Figure 3.3: 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).

3.2.4.4 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 [61] 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: $|x_t^* - x_{t-1}^*| \leq 6$.

3.2.4.5 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 [43], 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 [11];

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.

3.2.4.6 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:

$$\begin{aligned} \underset{x^*,z}{\text{minimize}} \quad & D(\xi) + \lambda_1 M_1(\xi) + \lambda_2 M_2(\xi) + \lambda_3 M_3(\xi) + \\ & Z(\xi) + \lambda_1 M_1^z(\xi) + \lambda_2 M_2^z(\xi) + \lambda_3 M_3^z(\xi) \\ \text{subject to} \quad & \frac{W_r}{2} \leq x_t^* \leq W_o - \frac{W_r}{2} \\ & |x_t^* - x_{t-1}^*| \leq 6, \\ & 0.7 \leq z_t \leq 1, \ t = 1, \ldots, N-1. \end{aligned} \tag{3.9}$$

As discussed in Section 3.2.1, 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.

3.3 Results

The results are computed on all the 12 clips (8 movie sequences & 4 theatre sequences) mentioned in Section 3.2.2. 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) [43] 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 3.3, Figure 3.5 and Figure 3.6. An explicit comparison on output with and without zoom is shown in Figure 3.3. All the original and retargeted sequences are provided in the supplementary material.

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 3.2: Parameters for path optimization algorithm and cut detection algorithm.

We have implemented the Dynamic Programming on MATLAB. The convex optimization is solved with MATLAB using CVX, a package for specifying and solving convex programs [35], with MOSEK [4] as the solver. The parameters used for the algorithm are given in Table 3.2. Same set of parameters are used for all theatre and movie sequences.

3.3.1 Runtime

The proposed algorithm optimizes the cropped window for any length of video sequence with arbitrary number of cuts. Table 3.3 shows the run time of all videos in the increasing order of time. 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 [43] 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 and it can be observed from Figure 3.4.

Video ID	No of Frames	Run Time for DP (sec)	Run Time for CVX	Total Run Time
9	1094	0.52	5	5.52
3	1788	0.84	6.5	7.34
10	3600	1.33	11.93	13.26
7	4198	2.54	11.42	13.96
4	4758	2.7	15.18	17.88
11	5040	2.7	15.21	17.91
12	5304	2.6	17.09	19.69
1	5703	1.95	16.36	18.31
5	7195	4.58	25	29.58
2	7674	3.34	24	27.34
6	7625	4.75	25.42	30.17
8	10210	5.73	34.57	40.30

Table 3.3: Run-time of the proposed algorithm for all sequences (arranged in the increasing order of time)

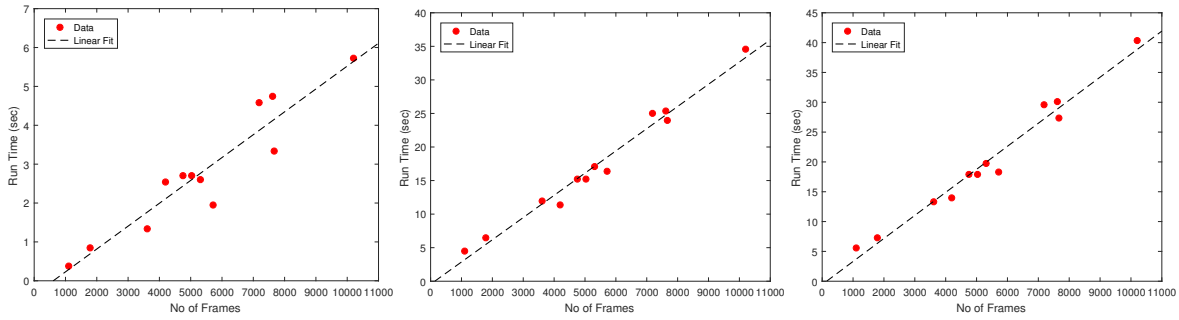


Figure 3.4: Run time of the proposed algorithm on various videos

Dynamic programming has complexity of $O(W_o N)$, which grows linearly with the number of frames N. The convex optimization solved using CVX-MOSEK [4] 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 210 frames per second.

3.3.2 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 [14] and [43]. Table 3.4 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 all sequences also confirms content agnosticism of our algorithm. From this, it is evident that our method is significantly better that GDR at retaining the most significant regions of each frame indicated by gaze data.

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

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.

3.3.3 Qualitative evaluation

One factor that should be considered while evaluating a re-targeting system is the performance at extreme shot lengths (very small shots like action sequences and very long shots like theater sequences). We compare our results with GDR in Figures 3.5 and 3.6.

The hard assumptions of at most two pans and a single cut in a shot limits the applicability of GDR on longer sequences, as seen in Figure 3.5. The cropping window becomes constant after frame 3 and misses the action taking place in the scene. In contrast, our method consistently follows the action by using smooth pans (with smooth interpolation between static segments) and introducing new cuts.

GDR can also lead to sudden unmotivated camera movements as it is applied on individual shots. An example is shown in Figure 3.6, where gaze is primarily fixated on the main character and the GDR algorithm produces an unnecessary pan. Conversely, our method produces a perfectly static camera behavior via retargeting considering the *entire video*. Results over all sequences achieved using GDR and our method are provided in the supplementary material for further qualitative comparisons.

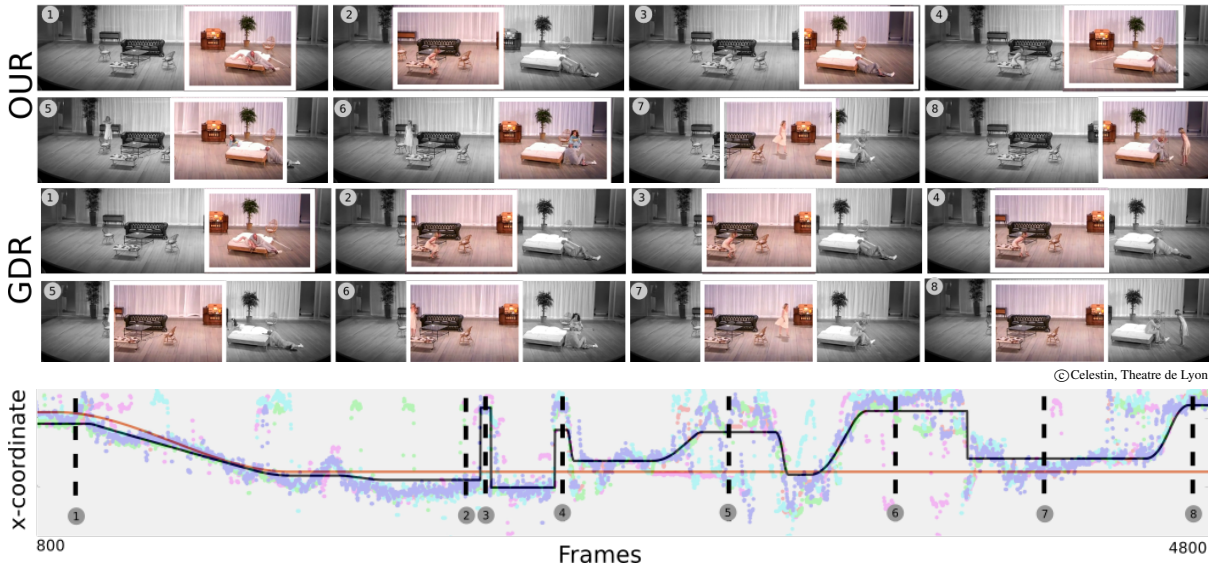


Figure 3.5: The figure shows original frames and overlaid outputs from our method and GDR (coloured, within white rectangles) on a long sequence. Plot shows the x-position of the center of the cropping windows for our method (black curve) and GDR (red curve) over time. Gaze data of 5 users for the sequence 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).

3.4 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 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.* [43] 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.

3.4.1 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.

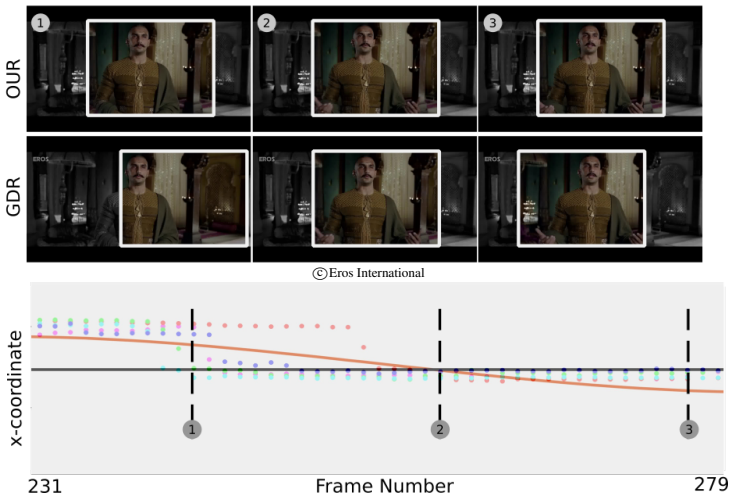


Figure 3.6: Frames from the original sequence cropped by the output of our algorithm and GDR (white rectangles). Corresponding gaze data (below) reveals that gaze is broadly cluttered around the shown character. Our algorithm produces a perfectly static virtual camera segment (black curve), while GDR results in unmotivated camera movement (red curve).

We slightly modified the evaluation protocol of Jain *et al.* [43], 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.

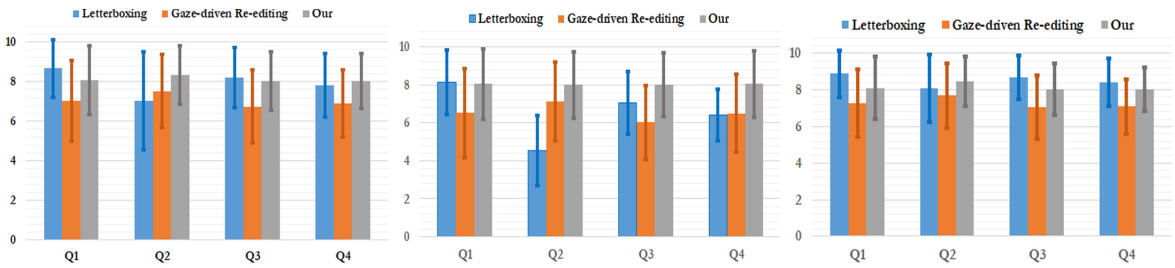


Figure 3.7: Bar plots showing scores provided by users in response to the four questions (Q1-Q4) posed for (left) *all* clips, (middle) *theatre* clips and (right) *movie* clips. Error bars denote unit standard deviation.

3.4.2 User data analysis

Figure 4.10 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 4.10 (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***.

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

Scores for $Q2$ in Figure 4.10 (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 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 4.10 (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 3.5 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***.

3.5 Summary

We present an algorithm to retarget video sequences to smaller aspect ratios based on the gaze tracking data collected over multiple users. The algorithm uses a two stage optimization procedure to preserve the gaze data as much a possible while adhering to cinematic principles concerning pans, zooms and cuts. In contrast to previous approaches [43], we employ a heuristic-free motion modeling and our algorithm does not make any assumptions on the input sequences (both in terms of type and length). This makes our algorithm applicable for re-editing existing movie sequences as well as edit a raw sequence.

The robustness of our algorithm to noise (i.e., spurious gaze data) allows us to employ gaze recordings obtained from a low cost Tobii Eyex (100 euro) eye tracker for generating the retargeted video. Such eye trackers can be easily connected to any computer and in fact may come integrated with laptops in future (https://imotions.com/blog/smi-apple-eye-tracking/), which generates 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 performed user study further confirms that our approach enables users to obtain a better view of scene emotions and actions, and in particular enhances viewing experience for static theatre recordings. Alternatively, our algorithm can also be effective for safety and security applications as it would enable a detailed view of events that capture the attention of a surveillance operator

One limitation of our approach is that it only optimizes over the x-position and zoom. The y-position is not altered and that limits the algorithm from (a) retargeting to 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.

Chapter 4

A Data Driven Approach for Automating Cinematography for Facilitating Video Content Creation

The advent of digital cameras opened up the possibility of making a feature-film/short-film for novice filmmakers. The process of filmmaking involves pre-production (planning and getting ready), production (filming) and post-production (editing etc) with people from different crafts involving in each stage. The most important departments are direction, editing and, screenwriting. Direction and editing processes are closely related stages of movie making process whereas the process of screenwriting is still seen as a separate discipline and there is a gap among these cinematic forms [47].

In most cases, the films were shot out of continuity. And the only guide in this non-linear order of shooting is the film script. However, the script tells nothing about the way the shots to be photographed. In such cases, creating a director's shooting script which has shot by shot instructions about shot specification plays a major role. The other alternatives of working with the script before the actual shooting include storyboard, shot list, aerial view *etc.*The way of turning the script into sequences may vary from person to person.

The Shot Specification or Shot List details about location, framing and composition, action, dialogue, and a general shot description In general, the director and cinematographer brainstorm and decide the best way to tell the story visually with the camera. Initially the script is broken down into chunks which can be visualized as a shot. And shot specifications are assigned to each of these chunks interpreting the story/scene with the camera. For example, a dialogue-line such as *"Wake Up!"* or *"Shut Up!"* are visualized as Close-Up shots as we want to show the emotion of the speaker whereas dialogue-lines such as *"Hello, How are you?"* or two-person conversations are likely to be a medium shot.

In this work, we propose a way of suggesting the shot specification for a given script which helps in creating the director's shooting script. This can be later used by a film-maker or a virtual cinematography agent to produce shots as output. The proposed method presents a way of visualizing the film from a script before the actual shooting which helps in refining the shooting ideas, designing sketches or storyboards with composition and staging details. The shot specification can be used in many other ways. For example, it can be converted into a formal description such as [64] which can be used as an input to a virtual camera control method to generate videos automatically.

4.1 Related Work

Two classes of methods were proposed to address the problem of automating the virtual cinematography. One category of these methods focuses on explicitly defining the rules based on cinematic principles to create edited sequences in a controlled setting (such as limited actors and limited possible camera configurations). And the other category of methods focuses on learning the cinematic style from the edited sequences and try to reproduce them.

One of the earliest works on virtual cinematography is by He *et al.*[37], which proposed a method for camera control and directing using hierarchical Finite State Machine (FSM). In FSM, each possible shot (such as CU, MS etc) is a state and transition between state is a cut. A set of pre-defined camera configurations are encoded as weights of FSM. However, due to the complexity of encoding a variety of scenes limits the applicability. A more principled approach was proposed by Jhala *et al.*[44] to create a link between story and the discourse of the video. However, his method is not suitable for real-time application as it requires similar types of pattern to be given as input in advance. Lima *et al.*[22] proposed a real-time editing method for interactive storytelling systems, which automatically generates the most adequate shot transitions, swaps video segments to avoid jump cuts. Assa *et al.*[5] poses the problem of automatic camera controls as selection of one active camera at a given time from the multitude of cameras. The selection is based on the correlation using modified Canonical Correlation Analysis. Lino *et al.*[55] proposed an optimization based method for creating edited sequences in 3D environment. The optimization is posed as a dynamic programming technique where various types of costs are included based on cinematic principles. [25, 54] uses film-tree to create a video sequence from script. HMM based technique for editing dialogue driven scenes was proposed by Mackenzie *et al.*[51] based on user specified idioms. The film-editing idioms are encoded into the Hidden Markov Model (HMM) model by manually defining the suitable probabilities. All of these techniques require the users to translate the cinematic principles/constraints into model parameters/rules/costs in one form or other which restricts the applicability of each of those techniques to real-time scenarios in terms of scalability to new scenes and effort to hand code rules.

A few of the recent methods focused on the learning the cinematic style. Lima *et al.*[20] posed the problem of shot selection as a classification problem using Support Vector Machines (SVM) specific to two-shot scenes (shot with 2 actors in a frame) . However, the work doesn't include the sequential information which is inherent to the script. Merbati *et al.*[60] proposed an HMM based model to encode and decode the elements of cinematic style. The problem is posed as learning the HMM model parameters from movies and reproducing the style on new scenes. The drawbacks include the applicability of the method is limited to 2-actor scenes while considering the first order dependency on dialogue sequences. Lema *et al.*[21] records a scene using multiple angles and use the geometric and emotional features to classify each dialogue into a possible shot. This approach too doesn't consider sequential nature of data.

In all of the above mentioned methods, there is a major limitation interms of the capacity of the model to learn/mimic the cinematic principles. This can be because of the limitation of the learning model or because of the limitation of the problem formulation itself. To overcome these limitations,

we propose a deep learning based framework for predicting shot specification from a raw script. The predicted labels are generic as they specify the shot specification in a broader sense and leave the choice of the number of actors or number of possible camera configurations to the film-maker.

4.2 Problem Statement

The problem of predicting shot specification from the script is posed as a sequence classification problem using LSTM. The LSTM model takes as input the word embeddings and a few other high level structural features (such as sentiment, dialogue acts, genre etc.) corresponding a line of dialogue and predicts the shot specification for the corresponding line of dialogue in terms of Shot-Size, Act-React, and Shot-Type categories.

4.2.1 LSTM in action

Recurrent Neural Network (RNN) is a variant of neural network that invovlves self loops. Figure 4.1 shows a simple example of a RNN network with self-loop and it's equivalence when unrolled in time.

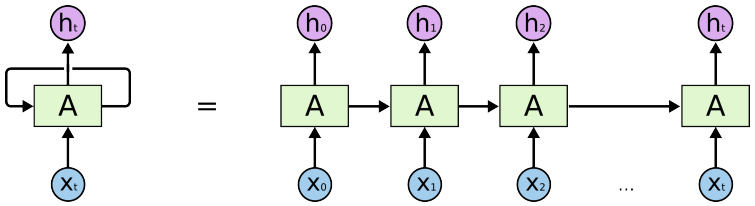


Figure 4.1: RNN

Long-Short Term Memory (LSTM)[40] is a variant of RNN architecture. Figure 4.2 shows as example block of LSTM cell. LSTMs are proposed to overcome the vanishing gradient problems of RNNs and have the ability to capture the long term dependencies present in a sequential data.

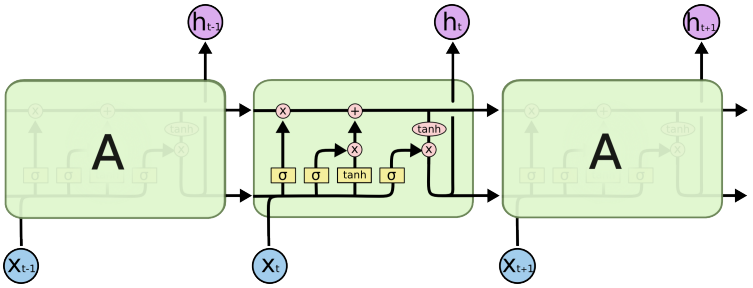


Figure 4.2: LSTM Cell

A fundamental block in LSTM is referred to as cell. Each cell uses gating mechanism which controls the information flow. A cell in LSTM consists of following elements: an input get i_t, a forget gate f_t, an output gate o_t, a memore cell c_t and a hidden state h_t. Given any variable length input sequence

$X = \{x_1, ..., x_T\}$ and corresponding variable length output sequence $Y = \{y_1, ..., y_T\}$, the following equations represent the way any intermediate hidden representation h_t at time t is computed:

$$
\begin{aligned}
i_t &= \sigma\left(W_i x_t + U_i h_{t-1} + V_i c_{c-1}\right) \\
f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + V_f c_{c-1}\right) \\
o_t &= \sigma\left(W_o x_t + U_o h_{t-1} + V_o c_{c-1}\right) \\
\widehat{c}_t &= \sigma\left(W_c x_t + U_c h_{t-1}\right) \\
c_t &= f_t^i \odot c_{t-1} + i_t \odot \widehat{c}_t \\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
\tag{4.1}
$$

Here, W_i, U_i, V_i are parameters for input gate, W_f, U_f, V_f are parameters for forget gate, W_o, U_o, V_o are parameters for output gate, σ is the activation function and $\odot$ denotes the element-wise product.

Intuitively, the input gate controls which part of the input values to be updated, forget gate controls which part of input values to be discarded and output gate controls the exposure of the internal memory state.

4.2.2 LSTM for Sequence Classification Task

Let d_i be the i^{th} dialogue line in the script and x_i , y_i be the corresponding feature representation and ground truth label respectively. Then $D = \{d_1, d_2, ..., d_N\}$ will be the set of dialogue lines, $X = \{x_1, x_2, ..., x_N\}$ with the corresponding set of features and $Y = \{y_1, y_2, ..., y_N\}$ will be the set of ground truth labels. Here, N denotes the total number of dialogue lines in the script.

For our task, we use a fixed length sequential data as input to the LSTM and let L be the maximum input sequence length. Hence, the input sequence for k^{th} dialogue line is given by $I_k = \{x_{k-L}, ..., x_k\}$ and the corresponding label will be y_k. This implies that the network uses the information from the past $L - 1$ dialogue lines to output the shot specification for the the k^{th} dialogue d_k.

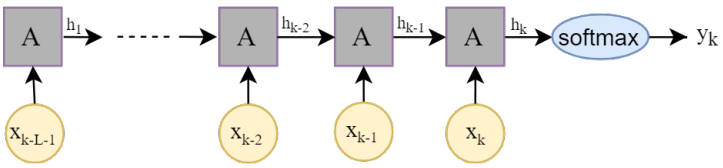


Figure 4.3: RNN for sequence classification

To predict the label for the k^{th} dialogue line d_k, the input sequence I_k is fed to the LSTM. The LSTM outputs h_k and it is regarded as the representation for the whole sequence. We pass this to a softmax layer which outputs $\widehat{y}_k$, predicted probability distribution over class. Figure 4.3 shows an example forward pass for this case. The network parameters are updated by computing categorical cross entropy loss given by Equation 4.2 and back propagated using any of the variants of stochastic gradient descent

method.

$$L(y, \widehat{y}) = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_i^j log(\widehat{y_i^j}) \tag{4.2}$$

Here, y_i^j is the ground truth label, $\widehat{y_i^j}$ is the predicated probaility, N is the number of samples in training set and C denotes the total number of output classes.

4.3 Dataset

4.3.1 Movies used for Dataset Creation

Our dataset consists of 6 movies with more than 16000 shots approximately. These movies cover various styles, composition and genres such as action, comedy, drama, thriller *etc.*Cutting *et al.*used a set of 24 movies to investigate scene structure and various other shot attributes in their paper [18]. We have selected a total of 6 movies from those set of 24 movies for which the movie scripts are available online. These movies were selected to span release years from 2000 to 2015 at 5-year interval and to cover various genres such as action, comedy, drama, thriller etc. All the movies have a native aspect ratio of 2.35:1. These movies are divided into shots using [3] which can detect various types of shot transitions such as fade in/out, dissolve, wipe. Table 4.1 shows the details of the dataset along with shot information.

ID	Movie Name	Rel. Year	Genre	Run Length	No of Shots	Avg Shot Length
1	Mission Impossible 2	2000	Action, Adventure, Thriller	2:03:35	2911	$2.332(\sigma = 2.093)$
2	Erin Brockovich	2000	Biography, Drama	2:11:11	1333	$5.109(\sigma = 3.389)$
3	Wedding Crashers	2005	Comedy, Romance	2:07:00	2144	$3.203(\sigma = 2.560)$
4	The Social Network	2010	Biography, Drama	2:00:27	2246	$2.928(\sigma = 2.477)$
5	Inception	2010	Action, Adventure, Sci-fi, Thriller	2:28:08	2703	$3.010(\sigma = 2.601)$
6	Avengers Age of Ultron	2015	Action, Adventure, Sci-fi	2:21:18	3344	$2.288(\sigma = 2.10)$

Table 4.1: Details of the movies used in creating Dataset

4.3.2 Shot Specification

The design of shots for cinematography consists in considering a number of visual features such as shot type, visual composition, focus, lighting, and color [78]. Many of the previous works [64, 9, 67, 75, 11] tried to formalize the language used for specifying the shots in movies. All of these approaches provide a rich description of a shot in terms of composition, camera movement and angle, actions, cues *etc.*Inspired by those approaches, we consider the following three categories:

1. **Shot-Size**: Close Up, Medium, Full.
   Sec 2.1.2.1 gives a list of 8 possible configurations for Shot-Size (SS) category, we consider only 3 of those classes. Table 4.2 gives the information about the way the regrouping of classes is

performed. This regrouping is necessary as the data for classes like ECU, BCU or ELS is very less which if not addressed causes problem during the training process.

New Shot Size	Original Shot Size
Close Up	ECU, BCU, CU
Medium Shot	MCU, MS, MLS
Long Shot	LS, ELS

Table 4.2: Details of Shot Size regrouping

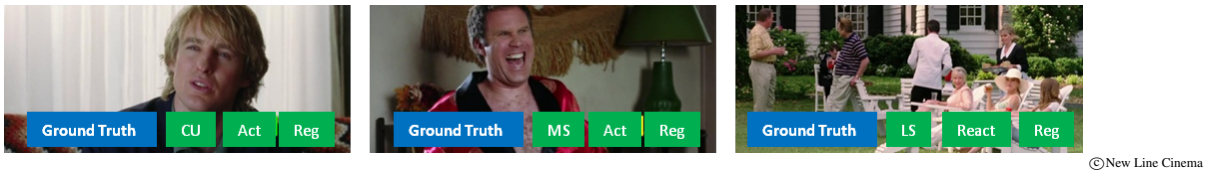
2. **Act-React:** Act shot, React shot
   As explained in Sec 2.1.2.5, this category specifies whether the speaker is visible in the shot or not.

3. **Shot-Type:** Regular, Over-The-Shoulder
   Sec 2.1.2.6 gives a list of 3 possible classes for this category. However, we consider only 2 of them for our work. As the POV shots are very few the dataset, they are considered as Regular shots.

These categories are the minimal set to broadly describe any shot or to place cameras. They include the composition information in terms of Shot-Size and framing in an abstract way. They do not explicitly specify which actor to show. But specifies whether to show the speaker of a dialogue or not. Hence, they can be used for solo-shot or multiple character shot (such as two-shot, three-shot *etc.*). These categories lack the information regarding the camera motion. However, they provide necessary information which can be used by director/cinematographer to decide the camera position, orientation, zoom, type of lens and depth of field *etc.*

4.3.3 Annotation of Movie Shots

Due to the difficulty in automating the annotation process, we have manually annotated all the movie shots. Two users who were familiar with the cinematography terminology were asked to label the movie shots under 3 different categories given in sec 4.3.2. Fig 4.4 shows a few examples of annotated shots.



©New Line Cinema

Figure 4.4: Data annotation examples

During the annotation task, the labels can be ambiguous sometimes. For examples, if there is a movement in either camera or subject, the Shot-Size changes within the same shot. In few cases, some

people may perceive a shot as CU and others may think of it as MCU shot which might induces users personal bias into the labels. We let the annotators decide the best fitted label whenever there is an ambiguity during the annotation process.

4.3.4 Script and Subtitles

For all the movies, *scripts* and *subtitles* were obtained from publicly available sources. These scripts and subtitles have different formatting as they were written/provided by various different people. So, the first task is to convert all of them into a common formatting so that it can be easily parsed to create a dataset.

4.3.4.1 Script

A script is a document that describes every oral, visual, behavioral, and lingual elements required to tell a story [65]. Script elements are a type of paragraphs with a standard format with unique margins and styling. This provides an order and consistency to the script. Table 4.3 shows a list of script elements with description and examples. A typical script provides the setting of the scene (given by scene header) along with some scene information (given by Action). It follows with a speaker information (Character) followed by his dialogue (dialogue) and/or tone of dialogue (parenthetical). Fig 4.5 shows an example of a formatted script with the script elements. Given a script, we use Trelby, a free screenwriting program, to convert it into a standard screenplay format with different element, each with a specific element style.

S.No	Element Type	Description	Example
1	Scene	Scene header	”INT. MOTEL ROOM - NIGHT”
2	Action	Describes action	”John grabs the gun from the desk.”
3	Character	A speaking character’s name	”HARRY”
4	Dialogue	Speech	”You could do that, sure.”
5	Parenthetical	Describes how the actor should say the following dialogue	”(serious)”
6	Transition	Describes a non-standard transition between scenes	”DISSOLVE TO:”

Table 4.3: Script Elements

4.3.4.2 Subtitles

In general, a subtitle starts with a blank line indicating the start of a new subtitle. Then, a number indicating the sequence number in the subtitle followed by a time that subtitle should appear and disappear on screen. It follows the subtitle. Fig 4.5 shows an example of an excerpt from a subtle file.

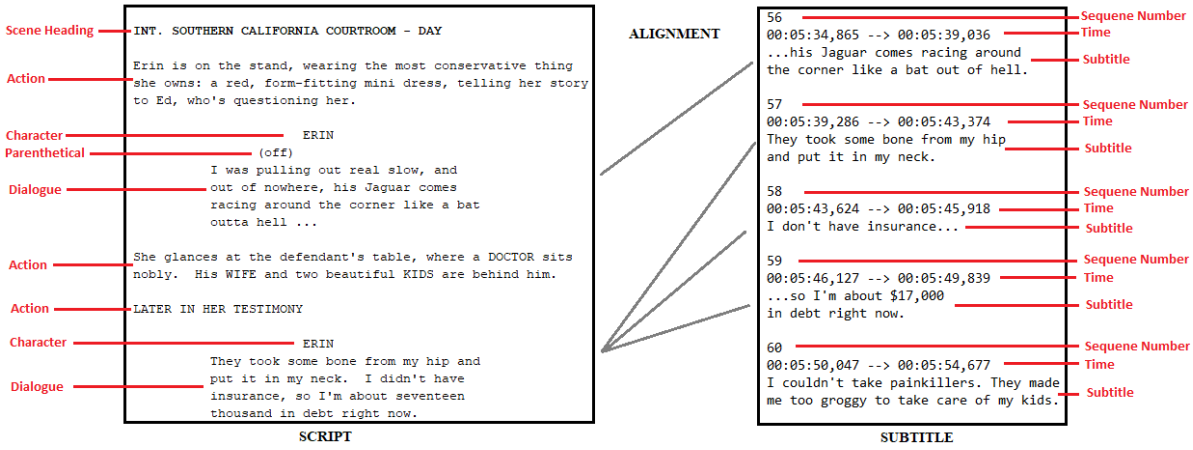


Figure 4.5: An excerpt from script and subtitles for the movie Erin Brockovich

4.3.5 Alignment

In general, scripts contain the information regarding the progression of the story. Hence, rich information related to the movie can be extracted from the script. However, a script doesn't contain time information. So, additional processing is required to synchronize script with the movie. The subtitles usually contain dialogue from script and time information related to movie. Therefore, synchronization between the script and the movie can be accomplished through an alignment process between the script's dialogue and it's subtitle.

4.3.5.1 Dynamic Time Warping for Script-Subtitle Alignment

We use Dynamic Time Warping (DTW) for aligning script and subtitle. DTW find the optimal match between two temporal sequences. The sentence (dis)similarity is used as a distance measure. DTW outputs the globally optimal alignment between the subtitle and script. However, there will be a few mismatches in this alignment for various reasons such as the available script may not be the working script of a movie etc. If two sentences have a matching score of 60% or above, they are marked as a good match. For the rest of the sequences, we manually find the alignment.

With this global alignment, we have a unified information for a given movie. For example, given a dialogue, we can say who spoke the dialogue (from script), the setting of the scene (from script) and the corresponding shot in the movie (from the subtitle and movie shots).

4.3.6 Statistics of the Dataset

Of the 16000 shots, a total of 10250 shots contain some form of dialogues. Fig 4.6 shows the percentage of these shots in each category and Fig 4.7 shows the transition probabilities in each of the three categories.

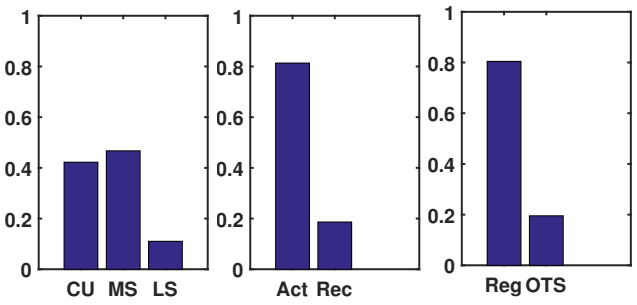


Figure 4.6: Class distribution for category (left)*Shot-Size*, (middle) *Act-React* and (right) *Shot-Type*

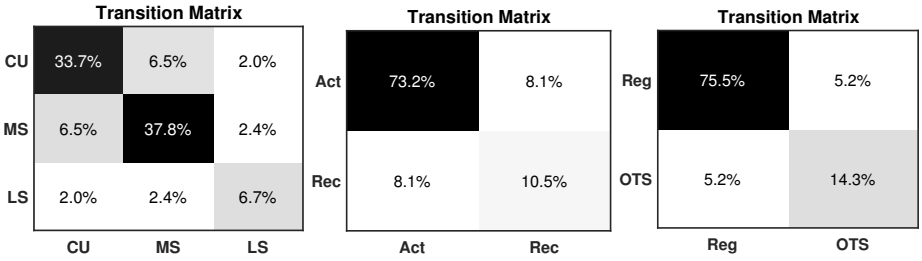


Figure 4.7: Transition Probabilities for category (left)*Shot-Size*, (middle) *Act-React* and (right) *Shot-Type*

It is evident from these figures that the task at hand has a severe class imbalance. For example, in shot-size category, the total number of close up and medium shots are 4 times the number of long shots and few transitions are more favored than others. The learning algorithm should take care of this and shouldn't generalize too much so that it captures the necessary subtle information in the dataset.

4.4 Method

In this section, we discuss the LSTM architectures and various features used for the task of predicting shot specification from script.

4.4.1 Features Used

The given script is aligned with the corresponding subtitle using DTW Section 4.3.5. Later the dialogues are tokenized into sentences using NLTK [8] sentence tokenizer. If a dialogue is very long, this splits them into multiple sentences and we propagate the same labels of the original dialogue to all of these dialogue lines. Later, we pre-process the data by removing all the special characters and converting it into lower case format. We obtain various features on this and the details are given below:

4.4.1.1 Sentence Embeddings using DSSM

Deep Structured Semantic Model (DSSM) [42] is a latent semantic model with a deep structure that project queries into (sentences, queries, predicates, entity mentions, etc.) in a continuous semantic space where the semantic similarity between two text strings can be readily computed. DSSM uses a fully connected feed forward network to map the text into the low dimensional semantic space. DSSM embeddings are suitable for this task because:

- Any given sentence, irrespective of its length, is represented as a 300-dimensional vector.
- It uses grams as input to generate the embeddings which makes it robust to spelling mistakes and other variations

For our task, we use the pre-trained DSSM model to obtain a 300-dimensional representation for any given dialogue line.

4.4.1.2 Sentiment Analysis

Sentiment analysis is contextual mining of text which extracts the subjective opinion of the sentence. Although there are many advanced techniques, we apply the standard NLTK [8] sentiment analysis algorithm based on naive Bayes classification to each dialogue to obtain the probability of positive, negative and neutral. This 3-dimensional vector explicitly specifies the intensity of the dialogues which might be useful for shot specification.

4.4.1.3 Dialogue Act Tags

A dialogue act is an utterance, in the context of a conversational dialogue, that serves a function in the dialogue. Types of dialog acts include a question, a statement, or a request for action [59]. They provide additional information regarding the understanding of the conversations. However, computing dialogue acts is a challenging task as there is not much agreement on its definition.

We train a naive Bayes classifier on NPS chat corpus [27] which has 10000 posts and each post is classified into one of 15 classes. These 15 classes are: *'Accept', 'Bye', 'Clarify', 'Continuer', 'Emotion', 'Emphasis', 'Greet', 'nAnswer', 'Other', 'Reject', 'Statement', 'System', 'whQuestion', 'yAnswer', 'ynQuestion'*. For a given dialogue line, we classify it into one of these 15 dialogue acts using the trained naive Bayes classifier and encode them as one-hot vectors.

4.4.1.4 Genre

Film genre is a category or set of categories the specifies the narrative elements or emotional response of a film. Each genre has a specific characteristic element such as setting, style, theme and narrative etc. For example, consider a case of Action and Adventure films where the film maker aims to generate

suspense and excitement. In such cases, wide shots might be preferred over others which captures as much action as possible and smaller shots might be used more to create a sense of excitement with fast paced editing.

For all the movies, we gathered genre information from IMDB and these details are given in Table 4.1. This information is encoded in one-hot vectors.

4.4.1.5 Same Actor

We identify whether the speaker corresponding to the current dialogue is same as the speaker from the previous dialogue. This feature provides additional information regarding the character and help in maintaining same shot size for a speaker which prevents jump cuts.

4.4.2 LSTM Architecture

We experiment with 2 variants of LSTM for the task of predicting shot specification. The details of these models are given below.

4.4.2.1 LSTM-STL : LSTM for Separate Task Learning

As mentioned in Sec 4.3.2, the output shot specification has 3 different categories (Shot-Size, Act-React, Shot-Type). In this current formulation, we treat each of these output categories as a separate learning task (STL) and train three different models one for each of these categories. From here on, we refer to these models as LSTM-STL.

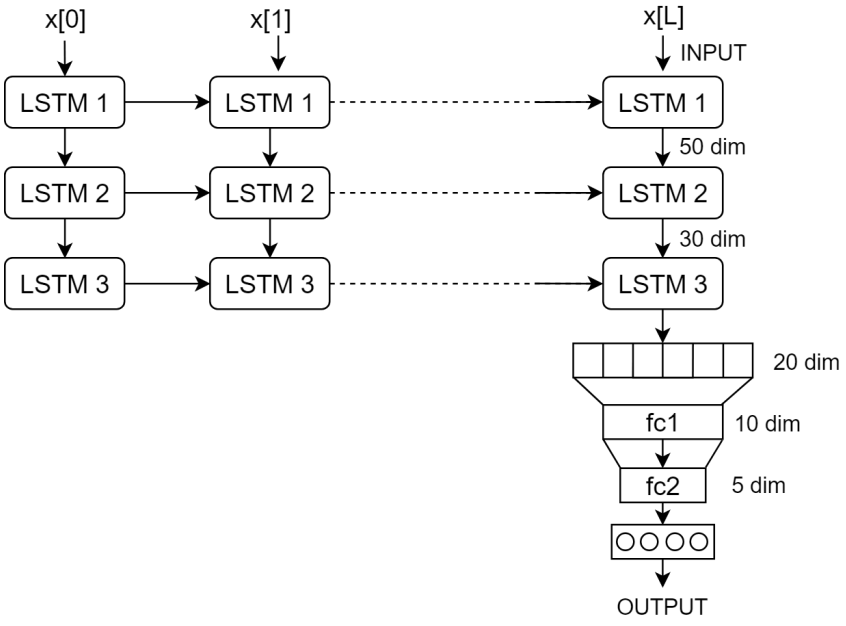


Figure 4.8: Architecture of LSTM-STL

The architecture for LSTM-STL is shown in Figure 4.8. The sequential features corresponding to the dialogue lines are fed to LSTM-1 layer as input. LSTM-1, LSTM-2, and LSTM-3 layers are connected in a hierarchical manner. They capture the sequential patterns in the input data and outputs them as a 20 dimensional feature vector. The output of LSTM-3 layer (20-dimensional feature vector) is fed to FC-1 layer with Rectified Linear Unit (ReLU) activation and FC-2 layer to further model the interaction. The final softmax activation outputs the probability of belonging to a particular class.

4.4.2.2 LSTM-MTL: LSTM for Multi-Task Learning

In this formulation, as opposed to LSTM-STL where we train a separate model for output category, we pose it as a Multi-Task Learning (MTL) problem.

In Multi-task learning, multiple learning tasks are solved at the same time, while exploiting commonalities and differences across tasks. This can result in improved learning efficiency and prediction accuracy for the task-specific models, when compared to training the models separately [13]. The current task of predicting shot specification from the script can be posed as an MTL problem.

The architecture used for LSTM-MTL problem is given in figure 4.9. Here, the LSTM-1 layer takes as input the feature vector corresponding to the dialogue sequences and LSTM-2 and LSTM-3 layers learns the generalized common representation which is useful for the later sub-tasks. Each of the sub-tasks (Shot Size prediction, Act-React prediction, Shot-Type prediction) takes the common representation from LSTM-3 as input and learns a specific **sub-model** for further modelling the interactions.

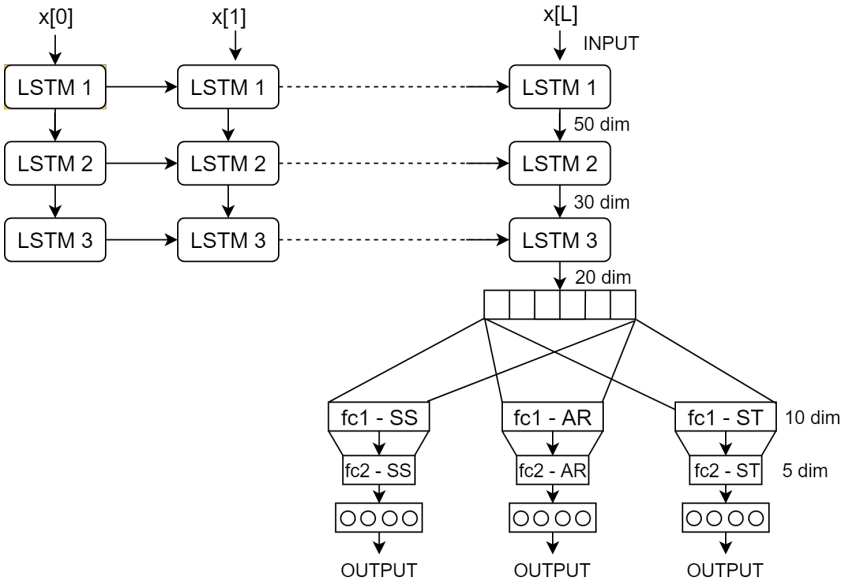


Figure 4.9: LSTM-MTL Architecture

Loss function : Weighted

4.5 Experiments and Results

4.5.1 Data Preparation

For all the scripts, we align them with the corresponding subtitles using DTW. Here, we assume that each line of dialogue can be visualized as a unique shot and we tokenize the dialogues using NLTK Tokenizer. If a dialogue is split into multiple sentences, the corresponding original labels were propagated to them. All the special character were removed and they were converted into lower case as a pre-processing step. On this data, we obtain various features mentioned in sec 4.4.1 and concatenate them.

As mentioned in Section 4.2.2, the input to the LSTM is a fixed length sequence. So, we convert these features into a set of sequences of length L, where L represents the maximum length of input sequences. If a dialogue line doesn't have a history of L samples, we pad them with zeros to ensure that the length of each input sequence is L.

4.5.1.1 Data Split

The remaining data is randomly split into training, testing and validation samples in $75:15:10$ ratio. Here, the test data is selected such that all the samples from randomly selected scenes are added until the size of test data is 15% of total samples. This type of sampling helps us to evaluate the performance of the trained network on visualizing scenes.

4.5.1.2 Data Normalization and Augmentation

The data is normalized using standard normalizer which removes the mean and scale each feature dimension to unit variance. The number of samples in the original data is around 10000 which is relatively less for training an LSTM network. In such cases, data augmentation is very useful to avoid over-fitting. So, we augmented the training data by adding Gaussian noise with $\mathcal{N}(\mu = 0, \sigma^2 = 0.01)$ to each feature dimension. The test and validation data were unaltered.

4.5.2 Experimentation with LSTM-STL

The network architecture for LSTM-STL is shown in Figure 4.8. In this setting we train three different models, one for each of the shot specification categories.

4.5.2.1 Model Training Parameters

The model is implemented in Keras [16] with Tensorflow [1] background. During the training, the validation data is used for tuning the hyper-parameter of the architecture. The optimal choice for the

hyper-parameters are given in Table 4.4. We use RMSprop [76] optimization routine for minimizing the categorical cross entropy loss.

Parameter	LSTM-STL
Batch Size	64
Epochs	30
Optimizer	RMSprop
Loss	Categorical cross entropy
Input Sequence Length	5
LSTM Cells	NA
Drop out	0.4
FC-activation	ReLU

Table 4.4: Optimal hyper-parameters for LSTM-STL architecture

4.5.2.2 Effect of Input Sequence length on Classification Accuracy

As a part of architecture study, we examined the effect of look back size on classification accuracy to decide the value for optimal sequence length. We trained 3 different LSTM models. The input to these models is the combination of all the features mentioned in Section 4.4.1.

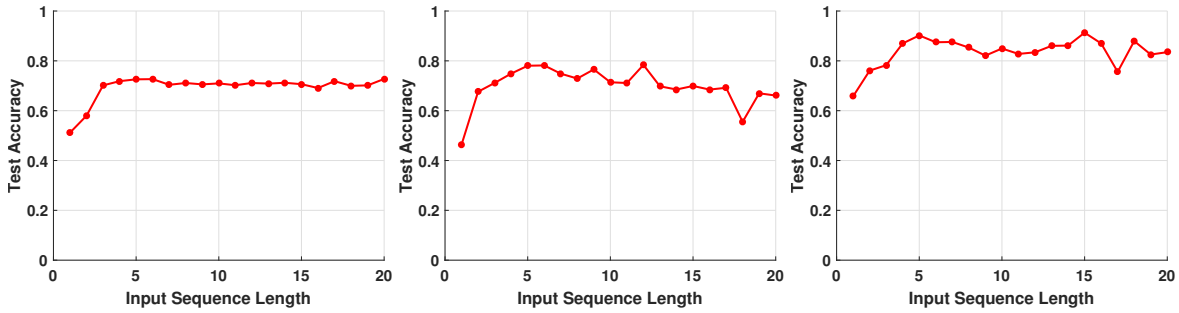


Figure 4.10: Effect of the input sequence length on the classification accuracy for category (left) *Shot-Size*, (middle) *Act-React* clips and (right) *Shot-Type*

All these three models were trained by varying the sequence length from 1 to 20. The classification accuracies are reported in Figure 4.10. From this figure, we can observe that the classification accuracy reaches the maximum for the sequence length of 5. After this, even with the increase in the sequence length, there is no significant improvement in classification accuracy.

4.5.2.3 Effect of various features on Classification Accuracy

Here, we try to evaluate the effectiveness of each of the features mentioned in Section 4.4.1. The architecture given in Figure 4.8 is trained for all possible combinations of features and output categories. We trained a total of 21 networks (7 possible feature combination X three output categories). The features *genre, same-actor* are used in all of these experiments. Table 4.5 summarizes the results.

	Frame Size		Act-React		Shot Type	
Features	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**
Sentiment	20.19	16.50	42.03	29.59	24.98	24.58
DAT	23.35	20.64	43.55	30.33	27.77	22.06
Sentiment + DAT	28.25	24.95	42.17	29.65	24.26	23.97
DSSM	62.52	62.43	71.70	70.67	81.60	77.27
DSSM + Sentiment	64.61	64.71	**77.63**	**76.67**	84.86	76.08
DSSM + DAT	68.69	67.74	75.43	75.20	**88.34**	**84.97**
DSSM + Sentiment + DAT	**71.45**	**69.37**	**77.13**	**76.67**	**88.54**	**89.93**

Table 4.5: Effect of various features on classification accuracy

- **Shot Size:** The precision and F1 scores suggest that sentiment and DAT features are useless for the current task, when they are used alone. On the other hand, the DSSM features alone perform reasonably well. However, sentiment and DAT features when used with DSSM features achieves a marginal improvement in accuracy and F1-Score. The best results are achieved when all of these features are used together.

- **Act-React:** The precision and F1 scores suggest a similar trend in the usefulness of sentiment and DAT features for Act-React category. Even here, The DSSM features alone perform well. Even though, DAT features adds a marginal improvement when used with DSSM features, the best results are achieved when DSSM and sentiment features are used together. This suggests the DAT features are redundant or doesn't add any new information than what sentiment features already captures for the category Act-React.

- **Shot Type:** For this category, the best results are achieved when DSSM features are used with DAT features. Although sentiment features add marginal improvement, in this case they don't add any additional information than what has already been captured in DAT features.

In summary, each of these features captures some specific information which is more useful for a particular category. DSSM features captures the general semantic structure, sentiment features adds more information to the Act-React category and the dialogue acts are more useful for Shot-Type category. Hence, all of these features should be used in combination to achieve best results.

4.5.3 Experimentation with LSTM-MTL

4.5.3.1 Model Training Parameters

The model is implemented in Keras [16] with Tensorflow [1] background. During the training the validation data is used for tuning the hyper-parameter of the architecture. The optimal choice for the hyper-parameters are given in Table 4.6. We use RMSprop [76] optimization routine for minimizing the weighted multi-task learning loss.

Parameter	LSTM-MTL
Batch Size	32
Epochs	90
Optimizer	RMSprop
Loss	Weighted Categorical cross entropy
Input Sequence Length	5
LSTM Cells	NA
Drop out	0.4
FC-activation	ReLU

Table 4.6: Optimal hyper-parameters for LSTM-MTL architecture

4.5.3.2 Results: LSTM-MTL vs LSTM-STL for Shot Specification Task

The LSTM-MTL architecture is trained with all the features, given in Section 4.4.1 and all the shot specifications, given in Section 4.3.2 as output. Table 4.7 summarizes these results and compares with the results of LSTM-STL case.

	Frame Size		**Act-React**		**Shot Type**	
Features	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**
LSTM-MTL	69.93	63.70	74.13	74.46	84.48	84.10
LSTM-STL	**71.45**	**69.37**	**77.13**	**76.67**	**88.54**	**89.93**

Table 4.7: Classification accuracy using LSTM-MTL architecture

From this table we can see that LSTM-MTL performs well on all three shot categories. Even though the difference is marginal, the better results are obtained when we train a separate network for each output category (LSTM-STL architecture). But if we consider the training process of these two cases, LSTM-STL is more prone to overfitting unless explicitly taken care (by drop out, regularization *etc.*). On the otherhand LSTM-MTL better regularizes for the data without additional external input.

4.5.4 Qualitative Results

4.5.4.1 Qualitative Analysis with example predictions

We show example sequences from test data corresponding to 2 different scenes in Figure 4.11.

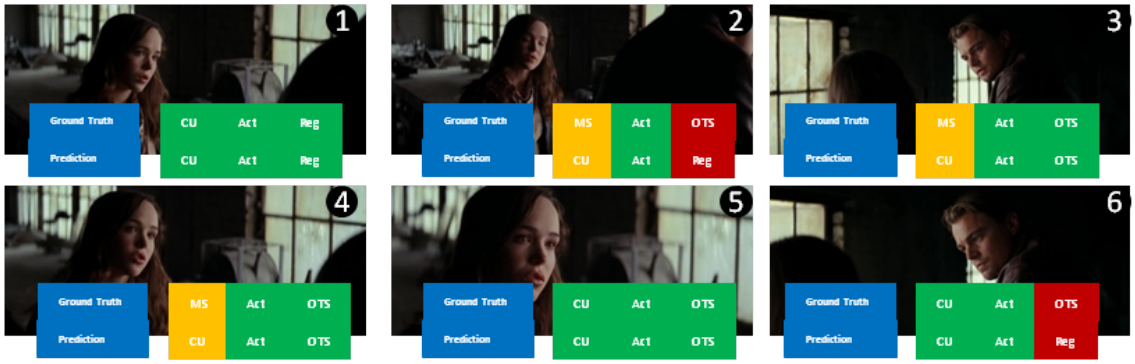
Figure 4.11(a) shows 6 shots in Inception movie. The Act-React and Shot-Type categories predictions are comparable to the ground truth prediction of the same. On the other hand there is a small error for the Shot Size category. Even though shots 1 and 2 involves same speaker (ARIADNE), the director visualized them as 2 different shots (CU-Reg followed by MS-OTS). However, the network predicts them to be into a single Shot-Size class (CU-Reg).

Figure 4.11 (b) shows another example sequence of 6 shots from movie Wedding Crashers. In this case, there is a severe mismatch in the Shot-Size category (only one out of six is correctly classified).

But if we look at the predicted labels carefully, the director visualizes all of them as Medium shots whereas network predicts them to be close up shots.

Although these type of mistakes are considered as misclassification, when visualizing the shots with the predicted category of labels, they seem to be good and tend to follow a pattern/cinematic style. This raises a need to look into better loss functions where we can incorporate cinematic constraints instead of standard cross-entropy kind of losses.

To better understand the quality of the results in terms of viewing experience, we need to perform user study evaluation. However, we are currently limited by the dataset size to do such an extensive study.



©Warner Bros. Pictures

(a) Example scene with a good percentage of match between predicted labels and ground truth labels

©New Line Cinema

(b) Example scene where the Shot-Size category is misclassified in 5 of the 6 shots

Figure 4.11: Qualitative Results

4.5.4.2 Qualitative Analysis with Heat Maps

To give a better understanding of the results, we compare the network predictions with the actual ground truth labels for multiple scenes. Fig 4.12 shows the difference between the prediction and ground-truth as a heat map. In each of those figures, rows indicate the three categories (Shot Size,

Act-React, Shot Type) and the columns correspond to the shot numbers. The intensity (yellow for good match and red for mismatch) shows the relative match between the predictions and ground truth.

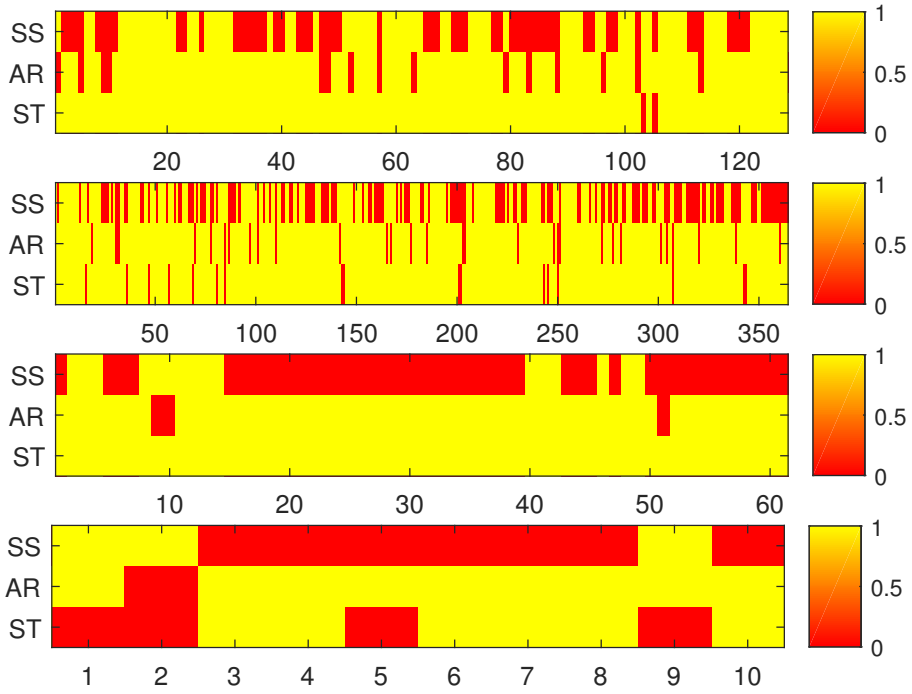


Figure 4.12: Heat map representation of the relative match between predicted labels and ground truth labels (Yellow/light-color indicates a perfect match and Red/dark color indicated a mismatch in predicted labels

From Fig 4.12(a),(b), we can see that the predictions are reasonably good for long sequences. Most of the mismatches are with the Shot-Size category. This can be attributed to the fact that a single dialogue (such as ”yes” or ”okay”) can appear in many scripts and can be shot in differently each time, which depends on the director’s vision. Such cases are hard to generalize and are misclassified most of the time.

There is a significant mismatch for scenes with less number of shots as it can be seen in. It can be seen in all the examples in Fig 4.12. This severely limits the ability of the network to learn the associations for the dialogues for the shots at the beginning of a scene. One possible explanation for this behaviour is that while preparing the data, we pad the data with zeros if there is no sufficient history which may not be a good practice. The only possible solution to this problem is to present the network with a wide variety of samples. But currently, we don’t have such a dataset.

4.6 Summary

In this chapter, we studied the possibility of using deep learning techniques for automating cinematography. To this end, we posed the problem of learning shot specification from a script as a sequence

classification problem using LSTM. We have shown that after learning the patterns/associations between shot specification and dialogue script from real movies, the LSTM model predicts the shot specifications with reasonable accuracy.

We extensively studied the effect of input features other parameters on the performance. The experiments suggest that to get best results the input sequence length should be at least 5. This implies that to predict the shot specification for the current dialogue, the network takes as input the past 4 dialogue lines history. On the effect of features, there are no individual features which works better, only the combination of them gives the best result.

We proposed 2 variants of LSTM architecture for this task. While learning a separate network gives best results, it is more prone to over-fitting. On the other hand the multi-task learning formulation better regularizes with a little compromise in the accuracy. The performance gap between these two networks is very less.

We created a dataset of which consists of 16000 shots which consist of 10000 dialogue sequences and manually labelled them for three categories: Shot size, Act-React and Shot-Type. The proposed dataset is relatively small to train an LSTM model and there is a severe class imbalance in the class label for of all categories. All of these are reasons for using smaller LSTM architecture so that the network does not over-fit.

Although the results are promising in terms of quantitative measures (such as classification accuracy and F1-score), there is a need to perform baseline and user study to fully understand the usability of proposed method.

A few limitations of the proposed approach are as follows:

- One of the limitations of the proposed approach is that it assumes that a line of dialogue should be visualized as a single shot. However this is not a realistic assumption. Hence, there is need to create a pre-processing tool which does the job of segmenting dialogue lines into sub-parts which can be framed in a single shot.

- The proposed method uses just dialogue and speaker information to predict the shot specification. There is need to look for better and efficient ways to process the script to get more useful features. For example, a film-maker subdivides the script into shots after reading the entire scene. So, adding information regarding scene summary or about type of scene (such as dialogue, action, confrontation *etc.*) on top of the dialogue features might significantly improve the accuracy.

- The current formulation doesn't involve any explicit cinematic constraints in loss function. Although we hope that the LSTM finds such patterns in the data, adding additional constraints may help in improving the learning capabilities of the network.

- The current formulation works only on dialogue sequences. Extending it to the new type of scenes (such as action scenes) require step-by-step instructions which are difficult to create.

Most of these limitations are because of the lack of an extensive dataset in this field. So, these limitations can be seen as possible directions for future works.

Chapter 5

Conclusions

In this thesis, we address two problems: Video Retargeting, and investigate the possibility of using Deep learning for automating cinematography.

We proposed a novel, optimization-based framework for video re-editing utilizing minimal user gaze data. To this end, it can work with any type of video (professionally created movies or wide-angle theatre recordings) of arbitrary length to produce a re-edited video of any given size or aspect ratio. Also, our optimization is $L(1)$ regularized, which economizes and smoothens virtual camera motion to mimic professional camera capture behavior, and ensures a smooth viewing experience. Also, since our methodology only requires approximate rather than accurate information regarding salient scene regions, we utilize eye-tracking data recorded with a low-end and affordable ($\approx$ 100 euro) eye tracker.

The applicability to edit a raw sequence adds another dimension to research in video retargeting. For instance, a user can simply record the scene from a static/moving camera covering the entire scene and can later use a retargeting algorithm to edit the recording based on gaze data and the user study confirms that the retargeted version better conveys the important details and significantly improves the overall viewing experience.

With the success of deep learning in Computer Vision areas, we also looked into the possibility of using Deep learning for automating Cinematography. Here, we proposed two LSTM model variants for predicting shot specification for dialogue sequences in a script by learning this association from real movies. The results are satisfactory in term of accuracy and precision. However, there is a need to perform extensive user study to understand the quality of results in terms of user experience.

As there is a lack of a standard dataset, we have created a small dataset which consists of 16000 shots which consist 10000 dialogue sequences and manually labelled them for three categories: Shot Size, Act-React and Shot-Type. We limited ourselves to these 3 categories of labels because the goal is to test whether or not deep learning techniques work for such task. Although the results are promising, there is need to create an extensive dataset in this field to explore this further. Our current work makes an effort in that direction.

The promising results open up new ways of content creation in virtual cinematography. For example, the proposed technique can be used to learn a specific style/genre patterns and reproduce on a similar

type of data. This has a huge potential gaming and VR industry where cinematically reproducing style is important for complete immersion of users. This can also be used by novice/inexperienced film-makers in creating videos that are cinematically good. On the otherhand these shot specifications can be converted into formal instructions which can be fed to the existing virtual camera control methods to generate video sequences.

Related Publications

- Rachavarapu Kranthi Kumar, Kumar Moneish, Gandhi Vineeth, Subramanian Ramanathan. **Watch to Edit: Video Retargeting using Gaze.** In Computer Graphics Forum 2018 May (Vol. 37, No. 2, pp. 205-215).

Bibliography

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[2] B. Adams, C. Dorai, and S. Venkatesh. Toward automatic extraction of expressive elements from motion pictures: Tempo. *IEEE Transactions on Multimedia*, 4(4):472–481, 2002.

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

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

[5] J. Assa, L. Wolf, and D. Cohen-Or. The virtual director: a correlation-based online viewing of human motion. In *Computer Graphics Forum*, volume 29, pages 595–604. Wiley Online Library, 2010.

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

[7] W. Bares, S. McDermott, C. Boudreaux, and S. Thainimit. Virtual 3d camera composition from frame constraints. In *Proceedings of the eighth ACM international conference on Multimedia*, pages 177–186. ACM, 2000.

[8] S. Bird and E. Loper. Nltk: the natural language toolkit. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 31. Association for Computational Linguistics, 2004.

[9] L. Blackwell, B. von Konsky, and M. Robey. Petri net script: a visual language for describing action, behaviour and plot. In *Australian Computer Science Communications*, volume 23, pages 29–37. IEEE Computer Society, 2001.

[10] N. Bonneel, K. Sunkavalli, J. Tompkin, D. Sun, S. Paris, and H. Pfister. Interactive intrinsic video editing. *ACM Transactions on Graphics (TOG)*, 33(6):197, 2014.

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

[12] L. Canini, S. Benini, and R. Leonardi. Affective analysis on patterns of shot types in movies. In *Image and Signal Processing and Analysis (ISPA), 2011 7th International Symposium on*, pages 253–258. IEEE, 2011.

[13] R. Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

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

[15] P.-Y. Chi, J. Liu, J. Linder, M. Dontcheva, W. Li, and B. Hartmann. Democut: generating concise instructional videos for physical demonstrations. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*, pages 141–150. ACM, 2013.

[16] F. Chollet et al. Keras, 2015.

[17] M. Christie and J.-M. Normand. A semantic space partitioning approach to virtual camera composition. In *Computer Graphics Forum*, volume 24, pages 247–256. Wiley Online Library, 2005.

[18] J. E. Cutting. The framing of characters in popular movies. *Art & Perception*, 3(2):191–212, 2015.

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

[20] E. E. de Lima, C. T. Pozzer, M. C. d'Ornellas, A. E. Ciarlini, B. Feijó, and A. L. Furtado. Virtual cinematography director for interactive storytelling. In *Proceedings of the International Conference on Advances in Computer Enterntainment Technology*, pages 263–270. ACM, 2009.

[21] E. S. de Lima, B. Feijó, and A. L. Furtado. Video-based interactive storytelling using real-time video compositing techniques. *Multimedia Tools and Applications*, 77(2):2333–2357, 2018.

[22] E. S. De Lima, B. Feijó, A. L. Furtado, A. Ciarlini, and C. Pozzer. Automatic video editing for video-based interactive storytelling. In *Multimedia and Expo (ICME), 2012 IEEE International Conference on*, pages 806–811. IEEE, 2012.

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

[24] S. M. Drucker, T. A. Galyean, and D. Zeltzer. Cinema: A system for procedural camera movements. In *Proceedings of the 1992 symposium on Interactive 3D graphics*, pages 67–70. ACM, 1992.

[25] D. K. Elson and M. O. Riedl. A lightweight intelligent virtual cinematography system for machinima production. 2007.

[26] J. Fleureau, Q. Galvane, F.-L. Tariolle, and P. Guillotel. Generic drone control platform for autonomous capture of cinema scenes. In *Proceedings of the 2nd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*, pages 35–40. ACM, 2016.

[27] E. N. Forsythand and C. H. Martell. Lexical and discourse analysis of online chat dialog. In *Semantic Computing, 2007. ICSC 2007. International Conference on*, pages 19–26. IEEE, 2007.

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

[29] Q. Galvane, J. Fleureau, F.-L. Tariolle, and P. Guillotel. Automated cinematography with unmanned aerial vehicles. *arXiv preprint arXiv:1712.04353*, 2017.

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

[31] 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.

[32] 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.

[33] 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.

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

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

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

[37] L.-w. He, M. F. Cohen, and D. H. Salesin. The virtual cinematographer: a paradigm for automatic real-time camera control and directing. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 217–224. ACM, 1996.

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

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

[40] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[41] C. Huang, F. Gao, J. Pan, Z. Yang, W. Qiu, P. Chen, X. Yang, S. Shen, and K.-T. T. Cheng. Act: An autonomous drone cinematography system for action scenes.

[42] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2333–2338. ACM, 2013.

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

[44] A. Jhala and R. M. Young. A discourse planning approach to cinematic camera control for narratives in virtual environments. 2005.

[45] 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.

[46] 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.

[47] E. Knudsen. The total filmmaker: thinking of screenwriting, directing and editing as one role. *New Writing*, 13(1):109–129, 2016.

[48] 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.

[49] 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.

[50] 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.

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

[52] 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.

[53] C. Lino. *Virtual camera control using dynamic spatial partitions*. PhD thesis, Université Rennes 1, 2013.

[54] C. Lino, M. Chollet, M. Christie, and R. Ronfard. Computational model of film editing for interactive storytelling. In *International Conference on Interactive Digital Storytelling*, pages 305–308. Springer, 2011.

[55] C. Lino and M. Christie. Intuitive and efficient camera control with the toric space. *ACM Transactions on Graphics (TOG)*, 34(4):82, 2015.

[56] C. Lino, M. Christie, F. Lamarche, G. Schofield, and P. Olivier. A real-time cinematography system for interactive 3d environments. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 139–148. Eurographics Association, 2010.

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

[58] Q. Liu, Y. Rui, A. Gupta, and J. J. Cadiz. Automating camera management for lecture room environments. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 442–449. ACM, 2001.

[59] M. McTear, Z. Callejas, and D. Griol. *The conversational interface: Talking to smart devices*. Springer, 2016.

[60] B. Merabti, M. Christie, and K. Bouatouch. A virtual director using hidden markov models. In *Computer Graphics Forum*, volume 35, pages 51–67. Wiley Online Library, 2016.

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

[62] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges. Real-time planning for automated multi-view drone cinematography. *ACM Transactions on Graphics (TOG)*, 36(4):132, 2017.

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

[64] R. Ronfard, V. Gandhi, and L. Boiron. The prose storyboard language: A tool for annotating and directing movies. In *2nd Workshop on Intelligent Cinematography and Editing part of Foundations of Digital Games-FDG 2013*, 2013.

[65] R. Ronfard and T. T. Thuong. A framework for aligning and indexing movies with their script. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 1, pages I–21. IEEE, 2003.

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

[67] B. Salt. *Film style and technology: History and analysis*. Starword, 1992.

[68] 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.

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

[70] H. V. Shin, F. Berthouzoz, W. Li, and F. Durand. Visual transcripts: lecture notes from blackboard-style lecture videos. *ACM Transactions on Graphics (TOG)*, 34(6):240, 2015.

[71] H. V. Shin, W. Li, and F. Durand. Dynamic authoring of audio with linked scripts. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 509–516. ACM, 2016.

[72] J. M. Snyder. Interval analysis for computer graphics. *ACM SIGGRAPH Computer Graphics*, 26(2):121–130, 1992.

[73] 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.

[74] J. Tarvainen, M. Sjöberg, S. Westman, J. Laaksonen, and P. Oittinen. Content-based prediction of movie style, aesthetics, and affect: Data set and baseline experiments. *IEEE Transactions on Multimedia*, 16(8):2085–2098, 2014.

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

[76] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

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

[78] P. Ward. *Picture Composition*. Focal Press, 2002.

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

[80] W. Y. Wong and P. Reimann. Web based educational video teaching and learning platform with collaborative annotation. In *Advanced Learning Technologies, 2009. ICALT 2009. Ninth IEEE International Conference on*, pages 696–700. IEEE, 2009.

[81] 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.

[82] M. Xu, J. Wang, M. A. Hasan, X. He, C. Xu, H. Lu, and J. S. Jin. Using context saliency for movie shot classification. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 3653–3656. IEEE, 2011.

[83] H. Zettl. *Sight, sound, motion: Applied media aesthetics*. Cengage Learning, 2013.