Propriétés du discours de la caméra virtuelle

Hui-Yin Wu

ANNÉE 2016

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Bretagne Loire

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

Ecole doctorale Matisse

présentée par

Hui-Yin Wu

préparée à l'unité de recherche UMR 6074 IRISA
et au centre INRIA - Rennes Bretagne Atlantique
ISTIC

Cinematic Discourse for Interactive 3D Storytelling

Thèse soutenue à Rennes
le 7 octobre 2016

devant le jury composé de :

Ulrike SPIERLING
Professeur, Hochschule RheinMain / rapporteur

Michael YOUNG
Professeur, University of Utah / rapporteur

Tim SMITH
Senior lecturer, Birbeck University of London / examinateur

Bruno ARNALDI
Professeur, INSA / examinateur

Stéphane DONIKIAN
Directeur de recherche, INRIA / directeur de thèse

Marc CHRISTIE
Maître de conférence, Université de Rennes 1 / co-directeur de thèse

Acknowledgements

Many thanks to my thesis supervisors, Stéphane Donikian and Marc Christie for offering me this thesis on developing and understanding technologies for interactive storytelling, topics that are very dear to me. I would like to thank Marc in particular, who apart from offering advice and guidance, never hesitated to send me anywhere in the world, if only to capture new ideas, present our work, or ignite possible future collaborations.

During the thesis, I was lucky to continue working with Prof. Tsai-Yen Li from IMLab (National Chengchi University) on many aspects following my master's work. I am also deeply grateful to Prof. Michael Young for hosting me in the Liquid Narrative Group and our continued collaboration and publication of a paper. It was also a great pleasure to work alongside colleagues in the MimeTIC and Hybrid teams in INRIA, members of the Liquid Narrative Group, and members of IMLab, who have been nothing but enthusiastic and encouraging in work and happy to share ideas about languages, cultures, and various lifestyles.

Finally, this thesis is for my family, for Mom, Dad and Yoyo, for Ola and Tata, and for David who was there to support me, and there as we set out on our next journey. I love you all.

Contents

List of Figures

List of Tables

Resumé en français

1 Présentation

Je me souviens avoir suivi une classe de cinéma élémentaire, le professeur a invité le réalisateur et directeur de la photographie Hsiang Chienne au cours pour nous aider à comprendre la différence entre les théories et les pratiques du tournage d'un film. Hsiang nous a demandé de décrire une scène immobile, à laquelle nous avons fourni avec enthousiasme les détails d'une femme assassinée, couchée sur un lit, dans une chambre en désordre, regardant le paysage urbain sombre. Le realisateur nous a demandé comment nous pourrions les tourner ces plans. « Qu'est-ce que vous filmez d'abord? Quelle est la taille de la tête dans le plan ? D'où vient la lumière ? Comment tout cela converge ? Quelle est *l'histoire* que vous voulez raconter? Quelle est *l'histoire* que vous voulez raconter ? » Quelle est *l'histoire*? Une question qu'il a nous posé à plusieurs reprises.

Figure 1 – Lors du placement d'une caméra, il faut veiller au cadrage du plan: quelle est la taille de la tête dans le plan? D'où vient la lumière? (Capture d'écran de *V pour Vandetta*, James McTeigue 2006)

Cette thèse porte sur les propriétés du discours à travers l'œil de la caméra: la narration cinématographique – « comment » raconter une histoire qui s'oppose à « ce que » l'intrigue contient. Les conteurs utilisent la perspective et l'enchaînement des événements pour exprimer l'atmosphère, les émotions, et même les morales qu'ils souhaitent communiquer au spectateur. Un bon réalisateur peut créer des liens de cause à effet, les relations entre les personnages, et l'atmosphère grâce à des décisions apparemment aussi simples que la distance entre l'acteur et la caméra ou le changement de plan. Cette intuition qui indique au directeur comment placer tous les acteurs et les caméras de la scène provient de l'expérience de leurs instructeurs, de leurs propres observations accumulées, de la créativité, et du tâtonnement.

Le choix des événements et l'ordre temporel dans lequel ces événements sont présentés sont au cœur du discours de la caméra. Un roman policier peut choisir d'afficher l'enquête mêlée avec des morceaux de flashbacks et des reconstructions de la mémoire de la scène du crime. Ils créent la surprise ou le suspense en cachant les détails à l'auditoire et en les révélant à des moments appropriés. Pendant ce temps une histoire d'amour pourrait commencer juste avant que les deux amants se rencontrent, et progresser chronologiquement pour suivre le développement de leurs relations dans un montage parallèle afin de favoriser des émotions fortes dans le public.

Figure 2 – Le **montage** implique l'assemblage temporel et spatial des plans. Bien que les deux scènes – l'assassinat dans les deux premiers plans, et la scène de baptême dans les deux derniers plans – se produisent séparément, la façon dont ils sont agencés fait paraître que l'homme dans le plan 3 regarde l'assassinat, et ce grâce au montage qui relie l'espace et le temps. (Capture d'écran de *The Godfather*, Francis Ford Coppola 1972)

Aujourd'hui, la narration cinématographique ne se limite plus aux rails et grues de caméra autour des personnes et des accessoires réels. Dans les jeux vidéo, la caméra devenue virtuelle peut couper, manoeuvrer, changer de perspective, et accélérer ou ralentir pour suivre les personnages et les environnements 3D. Dans les animations, les choix sur la façon de placer une caméra au mieux pour raconter l'histoire du jeu sont également fortement influencés par la pratique réelle du film. Étonnamment, la façon dont les vrais films sont faits a changé en utilisant des environnements 3D au travers par exemple de la pré-visualisation[1], et l'assemblage de différentes caméras pour créer des mouvements de caméra autrement impossibles dans le monde réel, tel que la technique Panocam utilisée dans le film *Jupiter Ascending*.

Figure 3 – Panocam filmant Chicago Cityscape partir d'un hélicoptère pour la scène de poursuite dans *Jupiter Ascending* (Lilly et Lana Wachowski 2015).

La progression des techniques dans les environmments 3D virtuels change le rapport du spectateur avec la narration. Il est désormais possible pour celui-ci de faire des

[1]Les outils de pré-visualisation permettent d'explorer virtuellement les possibilités d'une scène, de modifier le placement des acteurs, de changer la position de la caméra ou de sa focale. La préparation du tournage ainsi optimisée facilite la recherche de financements et réduit fortement le coût de production.

choix concernant le scénario, et ces choix entraînent des fins différentes. L'histoire et le contenu tous les deux sont fonction des préférences du spectateur. Cette progression est intéressante non seulement pour le spectateur, mais également pour les conteurs. Désormais un réalisateur ou un développeur de jeu possède un ensemble d'outils puissants pour créer des contenus réalistes, émouvants, et interactifs. Alors que le joueur bénéficie des avantages de l'autonomie, le conteur contrôle également ce qui est présenté et comment apporter le message du conteur au spectateur. C'est une relation réciproque, mais aussi contradictoire.

Cette thèse décline les nouveaux outils de la narration dans le numérique autour de trois axes: le temps, l'histoire, et la présentation visuelle. Nous nous répondrons principalement à la question sur la façon d'analyser, d'exploiter des données, et de générer automatiquement des arrangements temporels de l'histoire et des contenus visuels pour raconter des histoires. Nous nous référerons à la théorie du film et utiliserons des données de films réels pour comprendre les bonnes pratiques de la narration visuelle. Les techniques proposées dans cette thèse peuvent être appliquées aux problèmes de planification automatique de la caméra dans des environnements 3D, et ouvrent des perspectives pour l'analyse cognitive du cinéma et de la narration visuelle.

2 Problématique

Avec une demande importante envers des expériences de narration plus immersives et engageantes s'ouvrent des défis nouveaux pour inventer les technologies associées à leur présentation. Dynamicité, interaction, et autonomie combinée avec les connaissances professionnelles de la construction d'environnements virtuels, les entreprises exigent une foule d'artistes et de programmeurs pour réaliser un jeu ou un film. Obtenir les outils professionnels et les logiciels de modélisation et d'animation 3D est aussi cher que recruter quelqu'un avec les compétences nécessaires pour les utiliser.

Mais pourquoi la narration, devenue interactive, est aussi difficile? Au moins au début, les concepteurs/conteurs doivent pouvoir bénéficier d'outils facilement accessibles pour construire, tester, et réaliser un prototype d'une histoire interactive complexe. Ce prototypage des histoires interactives doit permettre au conteur d'observer des lacunes ou des conflits possibles dans l'intrigue. Les outils cinématographiques automatisés doivent permettre aux créateurs de se concentrer sur les aspects de la constructions de l'histoire du film sans avoir à traiter avec les mathématiques et la géométrie de positionnement de la caméra pour le cadrage ou le montage. La connaissance complexe de la façon dont la caméra narration virtuelle fonctionne implique cependant un certain nombre de difficultés techniques – difficultés dont nous souhaitons protéger le conteur quand il est encore dans la phase initiale de construction des histoires, et ce sont les défis liés à ces difficultés que nous allons aborder dans cette thèse.

Contrôle L'un des aspects les plus difficiles de la narration interactive est le contrôle sur l'histoire pour l'auteur – la logique, le flux de l'histoire, le message que l'auteur souhaite transmettre – tout en permettant plus d'autonomie pour le joueur. Dans les narratives interactives, créer des histoires logiques, cohérentes et contrôlables est un

problème difficile et ouvert. Le travail dans cette thèse aborde spécifiquement le problème de contrôle sur le discours: les éléments de présentation d'histoires dynamiques et 3D, impliquant des éléments tels que la perspective et le séquencement des évènements.

Psychologie cognitive Le contrôle de la narration permet à l'auteur de choisir les éléments de l'histoire et du discours afin de guider la perception du public. Cependant, validater que l'intention de l'auteur est bien celle qui est communiquée par la narration représente une étape importante. Le lien entre l'intention du auteur et la perception du public est mystérieux en raison de la complexité de la perception et du raisonnement humain. La difficulté consiste tout d'abord à établir un modèle simplifié, mais représentatif de la perception des événements de l'histoire qui intègre des aspects tels que la mémoire, et ensuite d'évaluer la pertinence de ce modèle auprès des utilisateurs.

Exploitation des connaissances Un outil qui permet une traduction directe entre le langage cinématographique, et le placement et le montage de la caméra virtuelle requiert une expérience certaine que de nombreux animateurs ou ingénieurs n'ont pas. Puisque nous nous intéressons à la narration cinématographique, il est tout d'abord essentiel d'avoir des outils d'analyse pour comprendre et exploiter les éléments de narration à partir de données de films réels. Pour appliquer ensuite cette connaissance à des environnements virtuels, ces pratiques cinématographiques de style et de placement de la caméra ont alors besoin d'être formalisées avec un langage afin qu'ils permettent rapidement et précisément de sélectionner et de placer des caméras virtuelles.

Dynamicité Dans les jeux 3D interactifs, le joueur peut prendre des décisions à certains moments de l'intrigue. Dans la plupart des jeux vidéo, la camera est fixée à un point de vue – par example, 1ère personne ou 3ème personne par-dessus l'épaule – et permet suivre le joueur en étant attaché à lui. Cependant, cela reste insuffisant pour exprimer des situations de jeu très filmiques. Pour un conteur, il est important que les intentions émotionnelles et cognitives sont correctement transmises. Bien que de proposer des systèmes caméras entièrement dynamiques et automatisées pour des histoires émergentes soit une tâche impossible, la caméra doit être en mesure réagir à l'histoire avec des transitions appropriées et des positions pour engager d'avantage le joueur. Cela reste difficile en raison de la complexité des caméras et des scènes, et de l'imprévisibilité des interactions du joueur.

Accessibilités des outils Fournir des outils accessibles pour les concepteurs/créatifs est un défi important, avec des médias devenus de plus en plus interactifs et multimodaux. Les techniques de pré-visualisation, par exemple, deviennent populaires en raison de leur moindre coût dans le cinéma, l'animation 3D et les logiciels de conception de jeu sont de plus en plus accessibles, par exemple, au travers de l'utilisation de données de capture de mouvement, pour créer des effets réalistes, et pour délivrer un rendu de haute qualité. Pourtant, du coté de la cinématographie virtuelle, le travail de placement et de montage dans des environnements 3D dynamiques nécessite encore une grande quantité d'édition manuelle et d'essai-erreur pour produire des séquences satisfaisantes. Il n'y a pas de d'outils accessibles aujourd'hui pour faciliter ce travail de

cadrage et de montage en raison de la complexité du placement et des mouvements de la caméra et de la difficulté d'intégrer les connaissances des films dans de possibles algorithmes.

Le but ultime est de fournir aux créatifs un ensemble d'outils qui rend plus facile (1) la création d'histoires interactives contrôlables, (2) le recueil et l'analyse des techniques cinématographiques de narration à partir des pratiques cinématographiques existantes et (3) la pré-visualisation des histoires impliquant des interactions de l'utilisateur et des intrigues dynamiques dans des environnements 3D.

3 Contributions

Dans cette thèse, nous présentons trois contributions.

Dans la première contribution, nous adressons le problème du **contrôle** par l'auteur de la narration interactive et des histoires générés dynamiquement. Nous abordons spécifiquement le problème en deux parties: la première fournit un contrôle par l'auteur sur la logique d'histoires dynamiques et non-linéaires; et la seconde est de prédire l'effet que ces histoires ont sur les spectateurs pour assurer la bonne communication des intentions de l'auteur. Pour aborder la question de l'interactivité et de la logique, nous proposons une méthode pour contrôler le flot d'évènements dans des histoires temporellement réarrangées afin de garantir des intentions de narration, sans sacrifier l'autonomie des joueur. Nous proposons ensuite un modèle de perception pour les intrigues temporellement réarrangées, et procédons à une évaluation de celui-ci.

Les deux contributions restantes se concentrent sur la cinématographie virtuelle. La première est motivée par la quantité de connaissances que nous pouvons **extraire de la pratique du film**. Nous proposons ici un langage de requête couplé avec un outil de visualisation pour l'analyse de motifs de techniques cinématographiques dans des séquences de films. Ce langage permet aux utilisateurs de rechercher et de récupérer des séquences qui remplissent un certain nombre de contraintes stylistiques et visuels liés aux cadrages et aux montages. La deuxième contribution est un outil d'annotation de films, une extension de l'outil d'annotation *Insight* qui permet d'intégrer le vocabulaire de notre langage de requête.

Dans notre contribution finale, nous abordons le problème de la **dynamicité** et la **créativité** dans des environnements 3D interactifs. Nous transformons le langage de requête filmique de la contribution précédente en un approche générative. Deux sous-contributions sont présentées. La première aborde spécifiquement le problème de placement de caméra intelligente pour des histoires où l'intrigue change de façon dynamique en fonction des interactions de l'utilisateur. Le second est un outil auteur qui agit à la fois comme un directeur de la photographie virtuelle – aider les cadrages de l'utilisateur, en appliquant des styles de caméra existants à une séquence – et un monteur – en effectuant des montages et réarrangements de plans dans une séquence cinématographique grâce à des motifs filmiques.

Introduction 1

1 Overview

I remember taking an elementary film class, the teacher invited director and cinematographer Hsiang Chienne to the course to help us understand the difference between the theory and practice of film shooting. Hsiang asked us to describe a single, still, scene, to which we enthusiastically provided details of: a murdered woman lying on a bed in an untidy room looking out into the dusky urban cityscape. The director then proceeded to ask us how we would shoot it. "What do you shoot first? How large is the head on the screen? Where does the light come from? How do all these come together? What is the $Story$ you are trying to tell?" What is the $story$ you want to tell? What is the $story$? He repeatedly asked.

Figure 1.1 – When placing a camera, we must think about questions such as how large is the head on the screen? Where does the light come from? (Screenshot from *V is for Vandetta*, James McTeigue 2006)

This thesis concerns the discourse–"how" a story is told, as opposed to "what" the content is–properties of visual storytelling through a camera: cinematographic storytelling. Storytellers use perspective and sequencing of events to express the atmosphere, emotions, and even morales they wish the viewer to perceive. A good film director knows how to imply cause-effect, character relations, and atmosphere through decisions as seemingly simple as how far the camera should be away from the actor. This intuition that tells the director how to place all actors and cameras in the scene comes from their instructors' experience, their own accumulated observations, creativity, and trial error.

At the core of camera discourse is the choice of, and temporal order in which the events are presented. A detective story may choose to show the investigation interwoven with bits and pieces of flashbacks and memory reconstructions of the crime scene to create surprise or suspense by hiding details from the audience and revealing them

at suitable moments; meanwhile a romance might want to start just before the two lovers meet each other, and chronologically progressing through the development of their relationship in order to foster strong emotions in the audience.

Figure 1.2 – Editing involves temporal and spatial arrangement of shots. Even though the two scenes–the assassination in the first two shots, and the baptism scene in the last two shots–occur separately, the way they are arranged makes it seem the man in shot 3 is watching the assassination through the space-time barrier. (Screenshot from *The Godfather*, Francis Ford Coppola 1972)

Now, cinematographic storytelling is no longer confined to the rails and cranes surrounding actual people and props. In video games, the virtual camera can cut, manoeuvre, change perspective, or slow down along with the choreographed characters and 3D environments. In animations, choices on how to place a camera to best tell the story are also strongly influenced by actual film practice. Most surprisingly, even the way real films are made has changed, using 3D environments to conduct pre-visualisation, and camera stitching to create otherwise impossible camera moves in our real world, such as the Panocam technique used in the film *Jupiter Ascending*.

Figure 1.3 – Panocam filming Chicago cityscape from a helicopter for the chase scene in *Jupiter Ascending* (Lilly and Lana Wachowski 2015).

The progression of interactive digital storytelling techniques brings even strong benefits to the viewer. They get to make choices on the storyline, and are offered the opportunity to discover multiple outcomes. Both story and content can be catered to the viewer's preferences. Yet this progression is not only for the audience, but is even more so for storytellers. Now a film director or game developer has a powerful set of tools to create realistic, moving, and interactive content. While the gamer reaps the benefits of autonomy, the storyteller can equally control what to present, and how, in order to achieve storytelling goals. It is mutual, but also conflicting.

The three elements: time, story, and visual presentation form the basis for this thesis. We address primarily the question of how to analyse and gain knowledge from data, and automatically generate temporal arrangements of story and visual content for storytelling. We work with both actual film data and film theory to understand the

good practices of visual storytelling. The techniques in this thesis target applications to automatic camera planning problems in 3D environments, and also open perspectives for cognitive analysis of film and visual storytelling.

2 Problems

With the demands for more immersive and engaging storytelling experiences comes the challenges of inventing the technologies to present them. With dynamicity, player interaction, and autonomy combined with the professional knowledge of building virtual environments, companies now require armies of digital artists and programmers to realise a game or a film. 3D modelling software and physical cameras are just as expensive to obtain, as to hire someone with the skills to use them.

But why should storytelling be such a hassle? At least in the beginning, storytellers can greatly benefit from easily accessible tools to build, test, and prototype complex stories. Prototyping interactive stories can allow the storyteller to observe possible shortcomings or conflicts in the plot line. Automated cinematography tools can allow editors to focus on the story editing aspects of the film without having to deal with the math and geometry of camera placement and framing. The complex knowledge of how virtual camera storytelling operates involves a number of technical challenges–challenges that we would like to shield the storyteller from when they are still in the initial stage of building stories, and challenges that we will address in this thesis.

Control One of the most difficult aspects of interactive storytelling is allowing the author to have control over the story–the logic, its flow, the message the author wishes to convey–while allowing higher player autonomy. In interactive narratives, creating logically coherent and controllable stories is a challenging and open problem. The work in this thesis specifically addresses the problem of control over the discourse, the presentation elements of dynamic and 3D stories, involving perspective, sequence, and timing.

Cognition Control allows the author to decide on elements of the story and discourse in order to guide the audience into a specific mindset. However, validating that the author's intention is achieved is the second step to the problem. The link between authorial intent and viewer perception is still a mysterious one due to the complexity of human thought. The first difficulty in studying this relationship between the author and the viewer is establishing a model of the human thought for story events that is simplified but representative of specific aspects of the human perception such as memory, beliefs, etc. The second challenge is designing user evaluations that can validate the model against the viewer's feedback.

Extracting Knowledge from Film Data A tool that makes a direct translation between film language and virtual camera placement/editing requires experience in filmmaking that many animators or engineers may not have. Since we are primarily dealing with cinematography storytelling, first and importantly is to have tools that can analyse and

extract elements of camera storytelling from actual film data. For applications to virtual environments, these existing film practices of styling and camera placement would then have to be formalise-able in language familiar to filmmakers so that they can quickly and accurately select and place virtual cameras without worrying about the details of precise camera placement.

Dynamicity In interactive 3D games, the player can make decisions at certain plot points. In most video games, the camera just chooses a perspective–1st person or 3rd person over the shoulder–and follow the player like a dog on the leash. However, this may no longer be enough to express heavily filmic and narrative gaming situations. As storytellers, it is important that the proper emotional and cognitive intentions are conveyed. Though fully dynamic cameras for emergent stories is a near impossible task, the camera should be able, to some degree, react to the story with suitable cuts, transitions, and positions to further engage the player. Yet this is challenging due to the complexity of placing cameras, the unpredictability of story outcomes, and the difficulty to map intentions with filmic techniques.

User-Friendly Tools Providing tools for storytellers without strong technical background has been an existing challenge, especially as media becomes more interactive and multimodal. With pre-visualisation techniques becoming more and more popular due to their lower cost in filmmaking, 3D animation and game design software have become progressively more automated in terms of, for example, applying motion capture data, creating realistic effects, and outputting high quality rendering. Yet, virtual cinematography still requires a large amount of manual editing and trial-error to produce satisfactory shots. There are currently no user-friendly tools for direct pre-visualisation and editing due to the complexity of camera planning and the difficulty of integrating film knowledge into the camera placement algorithms.

The ultimate goal is to provide the storyteller with a set of tools that makes it easier to (1) create interactive but controllable stories, (2) analyse and collect cinematographic storytelling techniques from existing film practice that can be used directly in 3D environments, and (3) pre-visualise stories involving user interactions and dynamic plot lines in 3D environments.

3 Contributions

In this thesis, we present three contributions.

In the first contribution, we address authorial **control** and **cognition** in interactive storytelling and dynamically generated narratives. We specifically tackle the problem in two parts: the first is to provide authorial control over the logic of dynamic, non-chronological stories; and the second is to predict viewer perceptions in these stories. To address the issue of interactivity and logic, a method for storytellers to control the story flow of temporally rearranged story events in order to guarantee desired outcome, without sacrificing player autonomy is presented. We then model viewer memory and

perceptions for temporally rearranged story lines, and conduct an evaluation to show how our computational model fits viewer perceptions.

The remaining two contributions then move towards visual storytelling, focusing on virtual cinematography. The first of these is motivated by the amount of knowledge we can **extract knowledge from film practice**. We design a query language coupled with a visualisation tool for the analysis of camera storytelling. The query language allows users to search and retrieve film sequences that fulfil a number of stylistic and visual constraints. Alongside this contribution is a film annotation tool extended from the existing *Insight* annotation tool to incorporate the vocabulary in our query language.

Our final contribution addresses the problem of **dynamicity** and **creativity** in interactive 3D environments. We transform the query language from the previous contribution into a generative one for stories. Two sub-contributions are presented in this part. The first specifically addresses the problem of smart camera placement for stories where the plot changes dynamically according to user decisions. The second is a prototype interface that acts both as a virtual cinematographer–helping the user select shot compositions, apply existing camera styles to a sequence, and place virtual cameras–and an editor–making cuts and rearranging shots in the sequence.

State of the Art

2

> "The time has come," the walrus said,
> "to talk of many things: of shoes and
> ships–and sealing wax–of cabbages
> and kings."
>
> Lewis Carroll, Alice in Wonderland

Contents

The wide expanse of literature covering interactive narrative discourse, film cinematography, data-driven cinematography, and virtual camera control can be overwhelming to cover in just a few pages. Moreover, each field is so different from the other, that to forcefully group them in this single chapter does not seem to do each field justice. Yet in this thesis, they come together in contributions to understanding cinema discourse in 3D environments, and thus, we must try to view them as such: independently important, but mutually intersecting.

The purpose of this chapter is to review work that has core contributions in film cinematography for virtual storytelling applications. We begin with a general background on film studies–the theory, practice, and cognitive studies–on which all three contributions are based. Each of the following sections reviewing work on interactive

storytelling (Section 2), film analysis(Section 3), and virtual cinematographic for storytelling (Section 4) loosely correspond to the three contributions of this thesis, but overlap other contributions in aspects of collecting cinematographic data, building of virtual environments, and analysis/generation of discourse elements in story.

1 Film Theory and Cognition

Close observation and study of film cinematography can help us understand the camera as a communicative tool for logic, story, and emotion. According to [Bra92], the audience's understanding of a film can be described via two processes: top-down and bottom-up. The top-down process corresponds to the long term memory, engaging the audience in inductive thinking to identify problems, overall plot flow, and story goals. On the other hand, the bottom-up process is a discrete, short-term knowledge process, allowing the audience to observe emotions and actions in the current frame.

1.1 Visual Storytelling

Structuralist theory of film has proposed that film narrative is composed of two parts: story and discourse [Bor85] [Cha80]. The story refers to the events, actions, setting, and characters that closely concern the plot. The discourse answers the question of "How is the story told?", involving aspects of style, mode(s) of presentation, the arrangement of story events temporally. Though we primarily focus on the discourse aspect of film storytelling–specifically cinematography–it is important to note that story is often inseparable from decisions authors make in the discourse.

Analyses of visual narratives such as comic strips suggest that viewers construct logical and emotional links between frames as well as structural narrative arcs over the entire sequence [Coh13] [FO12] [BW14]. In well-structured sequences, the viewer is found to have the capacity to identify large sections of plot elements such as establishing, initial, prolongation, peak, and release. Moreover, some sections can also be recursively broken down to smaller units, creating complex, embedded narratives, or even left out, without interfering with the viewer's comprehension. On the other hand, viewers can also be misled or confused when elements are switched around or if the narrative is not well-structured.

Film is also a visual story where shots are sequenced in order to show events concerning the character and scene, and to convey a progression of the plot. As a visual narrative, film cinematography also uses sequence, timing, and arrangement of various media content in order to convey logic and emotional intentions of the directors and storytellers. However, there is the additional element of movement, which includes actor movements and camera movements. In connection to the language-like properties of films, [Tse13] further categorises the types of events present in films and analyses how film shots are selected corresponding to these actions–dialogue, actor-actor or actor-object interactions, and actor reaction to events. The book proposes a comprehensive model that links audio, sight, and other cognitive reactions to the event types in the story. However, these models still face challenges in justifying that the audience's emo-

tional response are the ones the author wants to convey. We can even further indicate that the film camera's rules, not unlike actual languages, has its own set of syntax and semantics. This syntax is used when deciding what to show in a shot, how to compose the shot, and how to rearrange the shots during the editing process.

1.2 Framing and Cognition

Film theory has made connections between cinematography techniques and their effects on the viewer. Bordwell's pioneering book on film narration [Bor85] uses specific examples in films to discuss the effects of timing, pacing, framing, movement, many of them closely concerning camera control. Importantly, but also controversially, the author points in the direction of categorising the space, time, framing, and montaging techniques of film storytelling in relation to their effect on the viewer, suggesting that like a word and its meaning, some visual elements in film have links to communicative acts that can be observed and formalised. Other approaches apply film statistics, cognitive experimentation, tracking devices, and other methods to understand viewer perception. Cutting [Cut15] analyses number of actors per frame as well as the positions in which characters appear on the screen. He has found a consistency in character positions on-screen over the evolution of popular films, which shows filmmakers do have preferences for arranging the visual elements in a shot. Smith [Smi12] uses eye tracking devices to draw gaze maps of the audience, which allows them to identify the important regions on the screen and understand what the audience's attention is drawn to in a film clip. Neuroscience has gradually moved towards the concept of "embodied simulation", proposing the idea that there is a continuity between what we see, and the sensory motors of our body, such as the limbic system [GS11] [Smi97]. These studies show that there is a connection between technique and audience perception, even if this connection is still insufficiently understood.

In practice, the artistic choice directors make for placing the camera can be seen as a metaphor of the observer: directors want to draw the attention of the audience to important elements of the scene while hiding others to create suspense. Textbooks and references such as [TB09] [Zet07] are eager to provide guidelines to framing such as size, distance, balance, color...etc from film. These have become popular guidelines to create models for placing and moving cameras in virtual cinematography systems. There are a number of on-screen features that are a direct result of camera placement and movement in a scene. These involve size (or distance), on-screen position, angle, and movement. We do not discuss other elements such as light or color, even though camera placement in the scene does affect color and light, they are more of the result of scene arrangement and post-production rather than as a direct result of the camera.

Size The distance of actors to the camera will affect the size of actors and objects on the screen. An actor's size tells us its importance–closer cameras create bigger actors, increasing their importance; conversely, longer camera distance makes the actor smaller and the less important. Changes in size of frames in a sequence can also indicate the emotional distance the audience has to the actors and the intensity of the scene [Zet07].

Angle Corresponding to the x, y, and z axes in 3D, the camera can be rotated horizontally, vertically, and rolling.

Vertical angle assumes the heights of the camera in relation to the target it is filming. Angle can imply the personality of actors [Zet07], with low angle shots (looking up at the actor) expressing courage, confidence, or importance, and higher angle shots (looking down on the actor) showing weakness. Horizontal angle strongly implies the audience's involvement in the story, including strong involvement (front view), observer (side-view), vulnerability (back view). Roll angle is usually not used in the film, but it can be adopted for the purpose of showing disorientation of the actor, or distortion of the environment.

Position Framing region refers to how actors are arranged in the screen space. Where actors appear on screen has much to do with aesthetics, but also the inner state of the actors. For example, appearing in lower part of the frame or long distance from the centre could indicate isolation, while higher positions in the frame can indicate dominance. A common principle in film practice is known as the rule of thirds, where actor heads are usually placed one-third from the top of the screen to achieve overall balance of the frame [Zet07] [Bro12]. When violated, the object on the screen may appear isolated or insecure, which can also be a storytelling device.

Movement Finally, movement is a straightforward metaphor of guidance to the viewer [Bro12]. The major types of movement involve $still$ camera (no movement, allowing the viewer to observe other movements on screen), pan (horizontal rotating to explore a scene), $zoom$ (in and out, either to focus or reveal), $track/dolly$ (to follow), $tilt$(vertical angle tilting to show height), $pedestal$ (vertical movement), and $rolling$ (to create a feeling of disorientation or creepiness).

1.3 Editing

There are two properties considering relationship between shots. The more observable one is transitions, referring to the techniques that link two shots: a cut, fade, screen wipe, etc. Another is the relation between two shots in terms of on-screen properties: distance, angle, and framing regions. For example, we may want a sequence of shots that move gradually closer to actors; or a sequence of shots in which the actors always appear in the same region on the screen. Here we focus on the on-screen properties of editing.

One way to view frame sequences, as proposed by film theorists, is to view film as a language with its syntactical and semantic properties. Analysis of how all media elements each carry individual meaning is often closely tied to the syntactic analysis–the linguistic analysis of structure and rules–notably Christian Metz, who is one of the forerunners of this branch of film analysis, proposed the grande syntagmatique [Met66]. He argues that the most basic unit in film is the shot and there are rules on how shots transition according to rules of narrative structure. Following the grande syntagmatique, [BS12] argues for semiotic film theory to analyzing film, modeling the semantics (i.e. the context, meaning) of the shot, as a complementary to Metz's model, which

models the syntax (i.e. the grammar or structure). The grande paradigmatique is able to accommodate much more complex levels of meaning not only in the shot, but also story context, such as temporal and spatial relations, and perspective or projection. [FO13] also uses semiotics to describe emotions of audience towards film, proposing the three stages of eliciting condition, expression, and feeling. It derives from understanding what kinds of shots are suitable to convey a certain meaning.

In practice, filmmakers have established good practices to compose and sequence frames, sometimes called montage or mis-en-scene. Properties of editing include the selection and compiling of a number of camera rushes (a single recording of the scene) from various angles, movements, and perspectives. There is the selection of the take, as well as the question of timing and rhythm: when to switch rushes, and how frequently.

A large category of rules on how frames should be sequenced are called continuity rules. Continuity rules govern how the camera can be placed in a sequence of frames relative to the actors and events in the scene so that it does not spacially confuse the viewer [Zet07] [TB09]. Continuity rules target the actors (and objects) in the scene, and their actions. The most commonly known continuity rule is the 180 rule. The rule maintains that the relative position of any two actors on screen must be maintained–which is achieved by always placing the camera on the same side of the 180 degrees line formed by connecting the two characters. Figure 2.1 shows how violating the 180 degree rule could result in the audience's confusion. Another category of spacial continuity rules targets action: gaze, direction of movement, or other relevant actions. These continuity rules extend the 180 degree rule by establishing the line from an actor along the direction of the movement. Therefore, it would gaurantee that the movement would continue in the same direction from frame to frame.

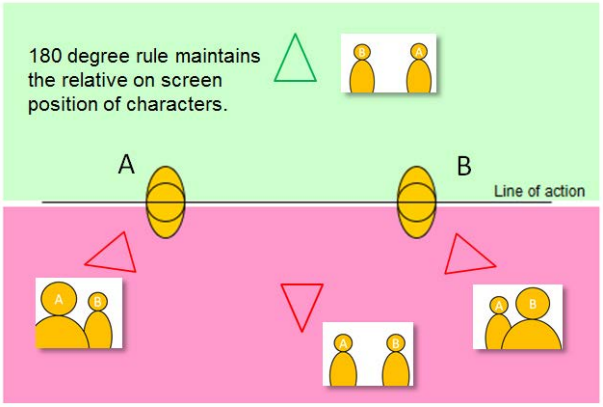


Figure 2.1 – The 180 rule states that the relative position of any two actors on screen must be maintained–which is achieved by always placing the camera on the same side (either green or red) of the 180 degrees line formed by connecting the two characters.

Continuity, apart from logical reasons, also carries semantic meanings by maintaining the connection between different actors, objects, or scenes in two consecutive shots. This can mean gaze matching Character A is looking at something, then CUT to the person/object in the line of the gaze (Figure 2.2-(a)); it can mean matching objects on screen (Figure 2.2-(b)).

Concerning timing properties of film editing practice, Pearlman [Pea09] believes in its broad sense, rhythm is not just the speed and frequency in which the editor performs

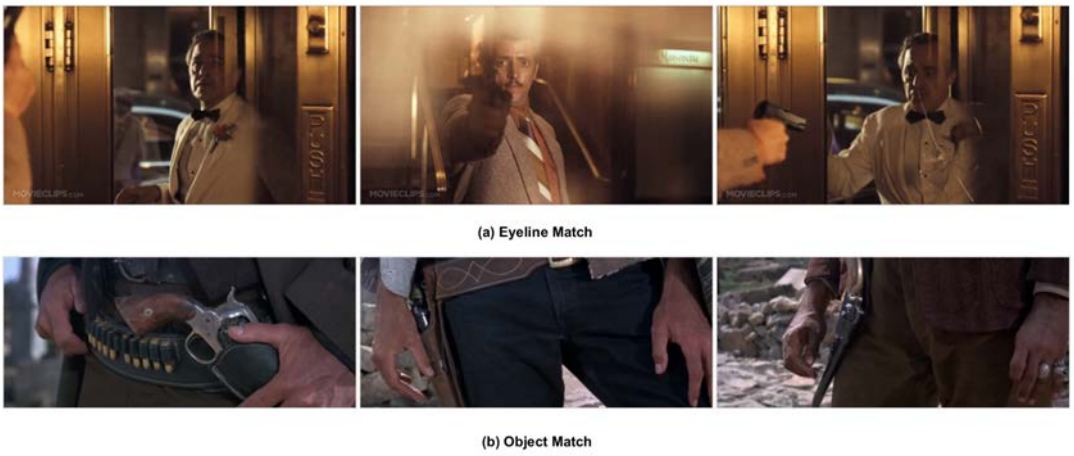


Figure 2.2 – Example of matching between shots using common characteristics such as (a) eyeline and (b) objects. (Screenshots from (a)*The Godfather* (b)*The Good, The Bad, and the Ugly*)

cuts, but a choreography between the camera and the actors in the scene. Though in actual filmmaking the duration of the shot and the frequency of cuts can indicate the pace of the story, when to cut and what to cut to still relies heavily on what filmed content is available. Continuity rules also limit the possible cuts that can be made. Setting aside continuity limits, as just mentioned, cutting speed affects the pace of the story. When shots are short and frequent, the tempo of the story increases, whereas longer shots can indicate a slower, less exciting event. Action is also an important indicator of cuts, as compiled by [Bro12]. That is, a shot usually cuts when an action is taking place because the viewer would be distracted by the action and will notice less the change in camera position.

1.4 Summary

This is a brief and incomplete overview to film theory and practice, but covers the main principles of how cinematographic storytelling works, of cognitive studies on audience perception, and how cinematographers and editors select and sequence shots in actual filmmaking practice. Integrating elements of theory and practice, we observe a number of research topics that rise from the study of virtual cinematography.

Due to the complex problem of placing the camera, as well as limited understanding of how viewers react to camera movements, cuts, and transitions, many animations and games choose static or minimal camera movement, with smooth (if not monotonous) cutting and transitioning techniques that draws as little of the viewer's attention to the camera as possible. Effective understanding of what works, and what doesn't, from abundant examples in film literature by use of data analysis tools is something that could in turn benefit virtual cinematography by enriching the placement and movement techniques available. The processing and analysis of film data by extracting visual features from shots and shot sequences can help us create models of smart cameras for 3D environments. This we explore in the second contribution in Chapter 4.

Second is the relationship between cinematography and viewership. The key to

a good link between the two is a thorough understanding of what viewers perceive. Currently, viewer cognitive models for films and for storytelling remain complex and incomplete, which makes them challenging to bring into virtual camera systems. Yet, as we have noted from this review of film practice, cinematography is directly linked to the way we perceive and interpret the story events, actor emotions, time, and space in a film. We explore the question of understanding viewer experience in storytelling in Chapter 3 and the relation between cinematography and story in Chapters 4 and 5.

Finally, the way film cinematography follows certain linguistic properties enables a scientific approach to designing virtual cinematography. Specifically, by analysing, categorising, and finally formalising cinematographic rules, rule-based systems can be developed that may not be able to produce artistic cinematography, but would be useful in both film pre-visualisation for quick prototyping, or for dynamic and interactive environments where efficiency and clarity are priority. This is explored in a number of applications for virtual camera editing in Chapter 5.

In the contributions, we will frequently return to the actual filmmaking practices of framing and editing when talking about the design of our techniques, but less to the theoretical principles of film cinematography and cognition. Especially in Chapter 4 where we introduce the Patterns language for film style, the vocabulary in Patterns is targeted towards the on-screen specifications of visual features discussed in Sections 1.2 and 1.3.

2 Interactive Storytelling

When we discuss storytelling, newer forms of stories on mobile and home entertainment are designed to be more and more interactive in order to engage the audience. Interactive storytelling research encompasses interaction design, algorithms, authoring, player experience, etc. Here we provide a broad overview of research in interactive storytelling before focusing on two main aspects that closely concern this thesis: authorial control and temporal structures.

2.1 Interactive Storytelling Systems

Interactive storytelling encompasses a category of digital entertainment that rose with technologies for more participatory storytelling experiences for the receiving end of a story, whether it is the audience of a film, reader of a text story, or player of a game. This is not just in the aspect of enhanced graphics, animations, and visual effects, but also how the audience and/or player immerses themselves in the world of the story and characters, participates in the story by making choices or interacting with the virtual characters, and directly change the outcomes of the story.

There are countless interactive storytelling systems used for visual entertainment, research, gaming, VR simulation systems, etc. and far too many to enumerate here. But for a comprehensive overview of various dichotomies of these systems, we provide a short definition and examples of the features that define interactive storytelling systems. We refer to the well-organised categorisations provided by Barot [Bar14] on

VR environments, but from the perspective of storytelling, and with some more recent examples from the AI storytelling community.

2.1.1 Character vs. Plot Driven

Currently, there are two main approaches to the *story* in interactive narrative systems, namely character-driven and plot-driven. In **character-driven** approaches, the story emerges from the interactions between the player(s) and the virtual characters. This approach focuses on developing believable virtual characters, and gives the player much more autonomy in deciding when and how one would like to interact with the scene and characters. A well-known system in this category is *EmoEmma* [CPC+09], where the player interacts directly with a virtual character, and the virtual character's actions and responses evolve as the story progresses. Similar works on synchronising character actions to create coherent plot lines [PTCC11], or projecting the story through the perspective of one character [PCC10] also benefit from the character-driven approach to make the virtual characters more believable.

The **plot-driven** system, on the other hand, builds the plot from bits of pre-authored story events. This approach has the purpose of constructing a sound and well-structured plotline. An early example of plot-driven systems is *Façade* [MS02], which is a game that allows the player to intervene in a couple's dialogue whenever the player wants, and keywords trigger beats–fragments of pre-authored events–to be played. An overall narrative arc is designed to increase the emotional density of the story. Most interactive fiction such as *Anchorhead* [NM05], which take the form of text-adventures, are also plot-driven.

2.1.2 Planning vs. Graph Algorithms

Interactive storytelling systems usually have drama managers that makes decisions on what happens next in the story. In a character-driven story, that would be what a virtual character should do next and how it should react, or in a plot-driven one, it would be which event to show next to create the best narrative arc. The way the drama manager operates also defines how the story is represented in the system. The two approaches used predominantly in AI storytelling techniques to representing the story are planning and the story graph, as shown in Figure 2.3. **Planning** systems represent the story as a collection of actions that (both real and virtual) characters can carry out, or events that can occur in the story, and preconditions to these actions or events. A planner then takes the role of the drama manager and builds up the story from these pieces of actions and events, decides the order in which they take place in order to achieve authorial, character, or player goals. The planning approach has greater dynamicity, resulting in a wider range of stories, mainly because actions and events can be mixed, reordered, duplicated, or removed to create multiple variations of stories. However, this also means the author has less control over the output of the story. Most character-driven systems such as *Thespian* [SMP09], *EmoEmma* [CPC+09], and *Glaive* [WY14] are planning systems. Examples of more plot-driven systems includes case-based reasoning systems [GDAPH05], vignettes (a variation of a case-based planner) [RL08], and event-centric planners [SGKB13] are also exploring more plot-driven stories.

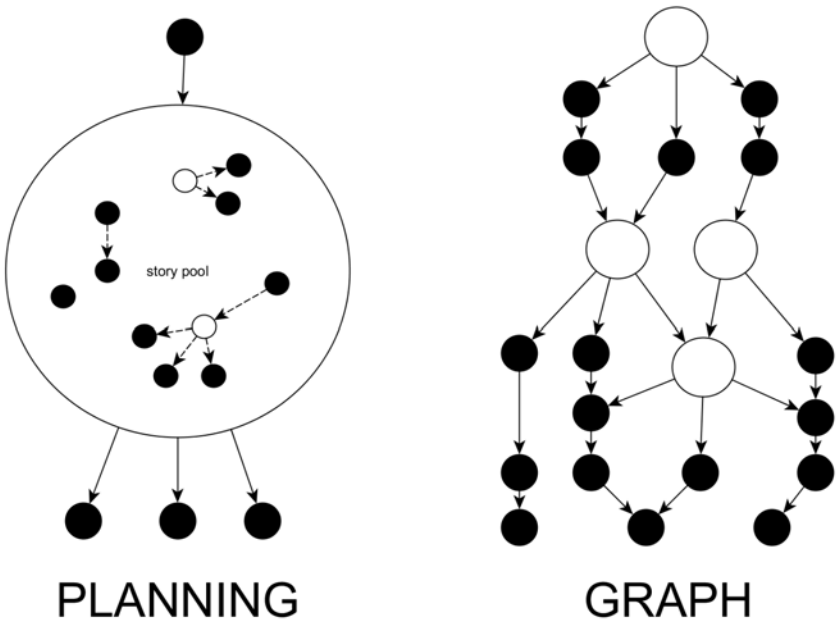


Figure 2.3 – Panning v. Graph-based approach to representing interactive stories. A planning based approach takes the author's desired constraints on the story, such as a starting point, an outcome, or constraints on intermediate events, and searches the story pool for events and actions to compose into one or more plots that fulfil all constraints. In contrast, the story graph has pre-authored or computed hard constraints on the transitions between actions/events.

The **graph** algorithm, also known as the story graph or plotpoint graph, is inspired from the idea of choose-your-own adventure stories. The plot point graph was first defined by Weyhrauch [Wey97] to represent the story as a directed-acyclic graph where the nodes represented story events and the edges denoted direct precedence constraints between these events. Story graphs offer much more authorial control over the story, since the author decides how events can be linked during the authoring process. However, compared to planning approaches, the variety of stories is limited by the edges the author creates in the story graph, and thus events cannot be ordered freely. Graph-based algorithms are used prominently in plot-driven stories, which seems like a natural choice since stories can be more structured and the author can drive the overall plot toward a more desired outcome. Interactive fiction systems such as *Anchorhead* [NM05] represent stories as graphs, and authoring tool for *Scenejo* [SWM06] (Figure 2.4) also uses graph-like structures to better visualise the overall story flow.

Some works do not fall completely on one side or the other of the planning vs. graph spectrum, and instead use a hybrid approach. One prominent example is *Façade* [MS02] in which beats are not all connected by precedence edges, but the drama manager uses search algorithms to find best matches to the overall emotional arc.

2.1.3 Pre-computed vs. Dynamic

Finally, it is necessary to mention interactivity: how much flexibility does the player have while experiencing the interactive story? In some systems, plot lines are **pre-**

computed, that is, the main events and all possible plotlines and outcomes are decided and shown in order to the viewer. In this kind of system, players follow the plotline decided by the system with limited interaction, and limited autonomy over the outcome of the story. The author may make the choice to have the story pre-computed with considerations such as (1) authorial control: ensuring that user interactions only take place as pre-planned; (2) efficiency: online planning algorithms can be expensive for stories with complex characters, scenes, and plots; and (3) story experience: ensuring the main story flow is not disturbed by user interactions. Narrative planners such as *Glaive* [WY14] and *Thespian* [SMP09] are generative systems that generate full plots that fulfil authorial constraints (such as conflict) and virtual character goals.

The opposite, dynamic systems compute the story on the fly as player interactions take place, and react directly to the player's decisions. Though they tend to be slower due to online computing, some efficient planning algorithms, stories of limited scope, or story graphs that are pre-authored can be played dynamically much less overhead in speed. Dynamic systems have the benefits of (1) believability: the drama manager for the virtual characters and scenes will likely have a wider choice of actions and events that can be carried out that respond more authentically to the user's interaction; (2) player autonomy: players have much more freedom in the choices they can make. *EmoEmma* [CPC+09] falls in this category: though it is a planning algorithm, the scenario is short involving at most 4 player interaction points that can affect the story outcome, and the interactions mainly switch between 4 possible narrative arcs.

Of course, systems can be a hybrid of the two categories: pre-computing a story graph and then allowing the story to be experienced dynamically, or pre-calculating parts of the story in order to increase runtime efficiency while allowing player autonomy in others.

2.1.4 Summary of Interactive Storytelling Systems

From the range of interactive storytelling systems introduced above, we derive two prominent research topics. The first is the dichotomy between authorial control and player autonomy. As seen from the comparison between planning and graph-based algorithms, one difference between these two approaches is the amount of control the author has over the stories, versus the player. The more precedence constraints the author places between story events, the fewer decisions or outcomes of the storyline are controllable to the player. Techniques to allow the author strong control over the story without sacrificing player autonomy are still much under development.

The second is believability of the storytelling experience. This involves the dichotomy between the plot line and the characters. It is more difficult to guide a story in a certain direction if character actions, when trying to appear credible, end up contradicting the direction of the story. Likewise, if the focus is to make the story more logical, characters may have to adjust their behaviour in order to stick to the plot, even if it means they are acting against their own beliefs.

In the next section, we focus specifically on the question of authorial control in interactive stories for the purpose of maintaining the believability of the story (i.e. logic), while maintaining the player's choice.

2.2 Authorial Control for Logic

There are many types of authorial control in interactive storytelling that can affect the pace, aesthetics, interactivity, etc. in the story, Here we specifically look at logic, which mainly concerns the causal relations between events in terms of story events in order to maintain the story's believability. The authoring of complex and unpredictable narratives can be a intimidating task for the author, and tools for visualising and evaluating the causality in stories can help authors detect possible conflicts and inconsistencies. Ciarlini [CPFF05] first proposed a method of generating plans for stories, and allowing the user to alter the story by changing an event in the plan as long as it remains consistent with the rest of the plot. However, though the authoring phase is interactive, constantly updating the story as the user makes changes, the generating story and dramatisation are linear. Spierling [SWM06] first introduces intuitive editing tools for authors of interactive stories to control the flow of the narrative through stimulus response elements, and offers an interface to visualize the complex story graphs (Figure 2.4). [Rie09] discusses the importance of authorial intent and the setting of authorial goals for dynamic narratives: because the author cannot be there to guide and stop the player from making unexpected decisions in real-time. With this purpose in mind, the work uses constraints to limit the search space of possible story events. In a later work [RLAR11], graph-like structures are constructed for multiplayer storytelling, and introduces a logical control mechanism that can detect exceptions on player actions, and readjust the story graph to guide the player back to the authored paths. [KFF+15] introduces interactive behaviour trees as a more author-friendly method for free-form interactive narratives. The approach also automatically detects possible interaction conflicts, and also mechanisms to accommodate user inaction.

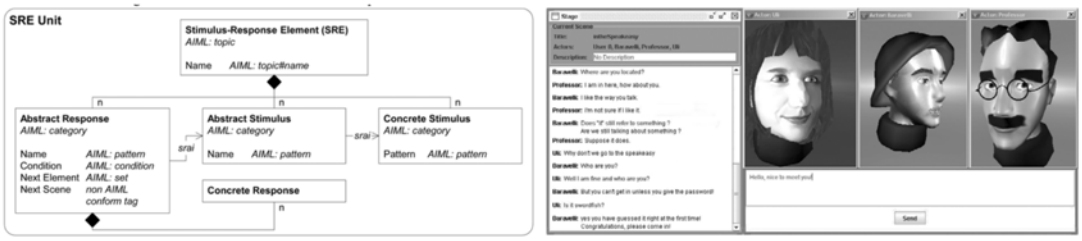


Figure 2.4 – Scenejo editing interface for story graphs. Editing interfaces can use graph like structures (left) for representing story logic to the author, for interactive applications like *Scenejo* (right) [SWM06].

In direct application to games where players can freely explore the story world, Carmichael and Mould [CM14] adapts the story model of Chatman [Cha80], using the concept of kernels and satellites to represent core events (kernels) and anticipated storylines (satellites) in open-world exploration games. This structure fits its original purpose to loosely-structured open-world exploration narratives by ensuring key events are played in order. Plot-point graphs Weyhrauch [Wey97], and further explored by Nelson and Mateas [NM05] propose the authoring of a number of important moments (i.e. plot points) into a directed-acyclic graph, and search algorithms can prune and readjust this graph as constraints are added or player decisions are taken into account.

2.3 Temporal Structures

When we talk about causality, there is also the implication that events happen in a certain temporal sequence. We have just discussed the importance of maintaining this causality by showing the relevant events; but on the discourse end–how these events are sequenced–has not been addressed. For some stories, telling the story chronologically may seem like a natural decision, whereas for others, such as detective stories, readers may enjoy deciphering what happened. In film, the flashback device is broadly defined as a technique in which the showing of an earlier event is inserted into an otherwise chronological sequence [Gen80] [Tur89] [Bor85]. Turim [Tur89] analyses the use of flashbacks throughout film history, and finds that flashback is essentially a visual device, and has much to do with memory and interpretation.

The temporal arrangement of events has been a study of AI storytelling systems, both in cognitive and generative aspects. Winer [WABBY15] proposes a framework to analyse of structural properties of timing in discourse while Eger [EBY15a] [EBY15b] proposes a similar framework for story events. The two works allow the temporal representation and reasoning of events in the story and discourse, but they have yet to be applied in actual storytelling applications. A number of works target flashback generation. Montfort [Mon07] proposes an algorithm that reorders events in text stories while respecting correct grammatical tense of the output order of events, which is later used in [Mon11] to generate flashbacks focalised from a specific character's viewpoint. Flashbacks for character focalisation is also addressed by [HTL14] specifically to recreate backstories of actors. Similarly, Bae and Young [BY08] use planning methods to create temporal rearrangements, mainly to invoke surprise arousal in viewers by hiding certain information and revealing it at emotionally intense moments. So far, these approaches focus on finding a sequence of events that satisfies inter-event causal constraints. Lönneker [Lön05] introduces an architecture for tense in "levels" of narrative, which is a basis for designing embedded story structures. Concerning logicality of temporal rearrangements, [PTCC11] and [SGKB13] tackle problems of timing in character or agent-based planning.

2.4 Summary

Previous work on narrative formalisms come in the form of either planning or graph-based, leaning towards planning methods in generative systems. This is largely due to the popularity of character-driven storytelling approaches for video games in which the overall flow of the plot is fixed, and the believability of the interactions between virtual characters in the story is more important. However, video games are not the only genres of entertainment that involve interactive storytelling techniques. Well-developed genres such as interactive fiction, interactive e-books, and genres still under exploration, such as interactive film also use interactive storytelling techniques, but have more emphasis on coherent and engaging stories than character believability. Our work targets such genres where the player or viewer's choice should have a strong impact on the story dynamically, where efficiency is important for the player experience, and where the story the author wants to tell is prioritised over character believability.

If we continue to weigh in the aspect of temporal control, the question becomes complex. While the majority of character-based systems have to deal with the temporal aspects of planning and sequencing virtual character actions, the aspect of temporality is discussed only on the story level, and not the discourse, which is the focus of this thesis. Generative systems have long been exploring the temporal aspect of discourse in flashback generative systems, but not for interactive stories.

How would one choose an approach that has high story variability, can deal with the problem of temporal discourse, and is non-linear (not a generative system)? Story graph structures can be seen as the intermediary output of a planning algorithm, in which all or a subset of story lines returned by a planner are constructed into an interactive graph structure. The graph structure has been shown to efficiently manage user decisions in real-time [Rya06], and has been used to represent complex stories with multiple states [SWM06] [RLAR11]. In terms of authoring, both planning and graph structures require laborious authoring for all the pre-post conditions, characteristic labels, and links for and between events, but graph structures offer more authorial control.

In our first contribution in the thesis, we propose an authorial control mechanism over interactive story lines with atemporal events (i.e. flashbacks). This method is targeted towards the secondary output–the story graph–stage of authoring. Positioned within previous work, our approach aims to be:

1. interactive and constantly updating story graph to reflect player interactions, as opposed to generating flashbacks in a linear narrative [BY08] [HTL14] [Mon07] [Mon11] [Lön05].

2. discourse-based, that is, we do not change character actions in the story events as in [PTCC11] [SGKB13], only the sequence in which it is presented

3. intermediary authorial goals on the story graph, as opposed to either incorporating authorial goals into the authoring phase [SWM06] [Wey97] [NM05] or replanning during realtime [Rie09] [RLAR11] [KFF+15], authorial goals can be defined in the intermediary, graph-filtering phase

4. for structured stories with multiple plot lines in a graph structure, as opposed to open-world exploration as in [CM14]

A side extension of our approach, an evaluation method on the quality of inserted events as flashbacks on viewer perception, is also proposed. The state of the art concerning the evaluation method is not included in this chapter, but detailed in the background of Chapter 3 along with the contribution.

3 Film Analysis

A large amount of work on understanding film cinematography has come from the field of image or video processing techniques, which involves applications to scene segmentation, content analysis, and event detection, all of which have strong implications in film storytelling. The availability of tools to annotate and extract cinematographic data

has changed the way we learn from films. Video and film analysis algorithms can improve accuracy for viewer recommendations in online streaming services to appeal to the viewer [CBL13]. Autonomous virtual camera systems learn from films by replicating existing movie shot compositions [LC15], as well as reanimate camera trajectories that are extracted from tools like voodoo camera [SDM+14]. Visualization tools such as Cinemetrics [Bro11] use color distribution to "summarise" the colour palette and evolution of the film.

3.1 Film Annotation

Annotation tools often have the purpose to link to contextual elements in the film in a way that extends previous contributions [RBLA13]. Due to the multitude of story, visual, contextual, and emotional information in films, annotation tools and annotation formats for the analysis of audiovisual documents are usually designed with maximum flexibility for user-defined annotation categories to manage a wide range of tasks. Some well known ones include *Elan* [SW08] and *Anvil* [Kip10] allowing annotation of features such as actor appearance, dialogue, metadata, and other user-defined elements.

Merabti [MWS+15] designed the *Insight* film annotation tool specifically for the collection of cinematographic data, based on the practical filmmaking principles described in Section 1. *Insight* does not provide as flexible tools for content-analysis as *Elan* or *Anvil* in terms of dialogues or metadata, but the tool provides three main benefits: (1) the labels are designed around actual filmmaking terms used for cinematography, (2) it has a much more flexible on-screen annotation tool for actors and objects, and (3) the data can be used directly by constraint-based camera placement algorithms to recreate actual film compositions.

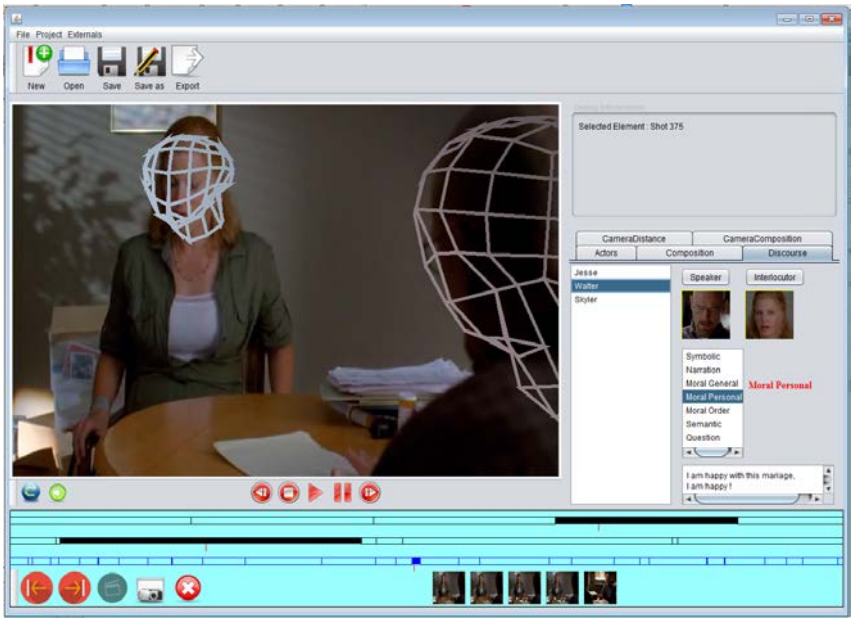


Figure 2.5 – The *Insight* annotation tool allows the annotating of on-screen and semantic features of frames and shots in a film clip. This screenshot shows the head position tool in *Insight*.

3.2 Automated Analysis

Analysis of film cinematography has much to do with understanding film as a communicative tool for logic, story, and emotion. A dominant field of video and image processing converges on algorithms for shot boundary detection [CB98] and genre analysis [RSS05]. Scene and event boundaries provide information on how a film is structured, while genre identifies crucial emotional or story-archetype features. This has strong applications to film summarisation for streaming services that would appeal to the viewer or provide insight to the film's content. Another aspect of cinematographic analysis focuses on aesthetics of framing composition such as detection of the usage of shot types [SBA+15].

To bridge the gap between the viewer's emotion in relation to cinematography style, Markov Chains have been applied to understanding how multiple film features contribute to how films communicate with the audience. Both Canini [CBL11] and Merabti [MBC14] use the markov chain to represent state transitions of film parameters between shots. Canini [CBL11] observes the correlation between shot sizes and audience emotions; [MBC14] focuses on learning transition parameters to replicate director style.

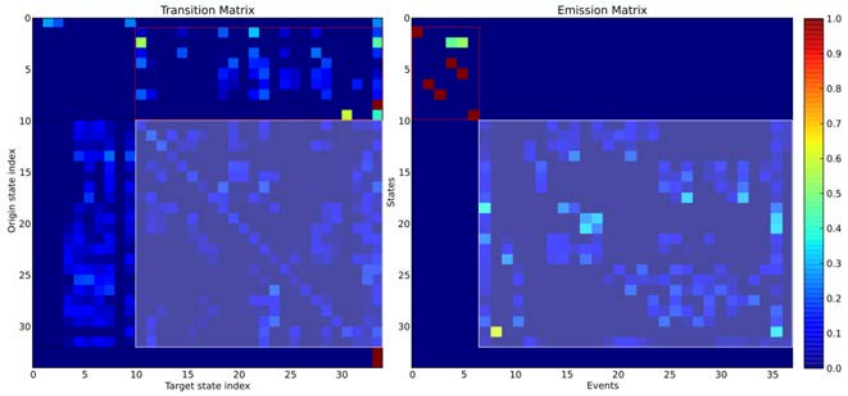


Figure 2.6 – Using Hidden Markov Models to construct transition and emission matrixes that can calculate the transition probabilities of each on-screen feature of consecutive shots [MBC14].

The use of cinematography rules to detect meaningful sequences has also gained interest. Early on, Wang [WC03] and Tavanapong [TZ04] found that continuity rules are very good indicators of scene and event boundaries, and for clustering shots that belonged to the same story scene respectively. By using image comparison techniques on certain regions of the screen, even interweaving and parallel events can be separated and detected.

3.3 Summary

One main challenge of film analysis is addressing the multimodality of film (i.e. the combination of multiple elements of sound, colour, composition, movement, light, etc. may result in different meanings) over long sequences of shots. The difficulty as such

is incorporating cinematographic knowledge into learning or pattern recognition algorithms. The use of Markov Chains [CBL11] [MBC14] to analyse the director's preference of transitions between shots helps us observe elements of style from on-screen features such as size, angle, actor position, etc. However, style itself is so complex that from such an analysis, we still cannot observe long stylistic sequences or continuity rules that are common practice by film directors, such as the 180 degree rule we introduced earlier.

Image and video processing techniques have strengths in using colour, lighting, sound to provide us information about the genre, about how scenes are segmented, and insights into frame composition. However, these approaches are still limited when we want to understand film storytelling over a sequence of shots, understand what kind of events are taking place. The use of cinematographic idioms or continuity rules as features has also been limited to techniques for scene and event boundaries, but can be further used to draw information on the story.

This is mainly addressed in our second contribution (Chapter 4) through our definition of the *Patterns* language for describing cinematography editing style.

The main features that set our approach apart from previous work include:

1. multmodality, as opposed to analysis of a single visual element
2. a mix of automated and manually annotated data, as opposed to solely image processing techniques to obtain data
3. full search techniques on well-defined style parameters, as opposed to machine learning techniques (where weights of features are less defined) [CBL11] and [MBC14]

In the second and third contributions in Chapters 4 and 5, we collect large quantities of data using a revised version of the *Insight* annotation tool designed and developed by Merabti [MWS+15], and the revisions to the tool are described in Chapter 4.

4 Virtual Cinematographic Storytelling

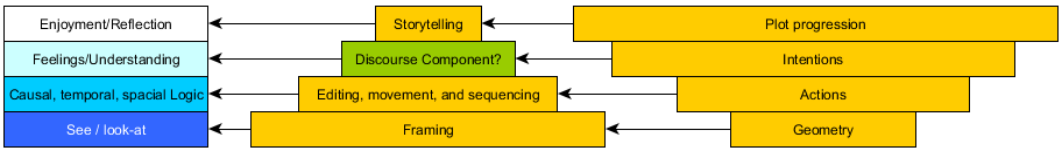


Figure 2.7 – An overview of virtual camera techniques, their inputs, and relation to their roles in visual storytelling. The central column is the virtual cinematography techniques that build on each other. The column on the right indicates the input of each technique. The left-most column indicates the cognitive metaphor of the audience in the visual story.

This section is structured following Section 1, and each part in this section corresponds to one level in Figure 2.7, starting from the lowest geometric camera level, then editing, and finally AI systems. The central column is the list of virtual cinematography techniques that build on each other from the low level framing to the story. The

column on the right indicates the input of each technique from simplest and easiest to process (geometry) to more complex and abstract (plot progression). The left-most column of the figure maps the technique to its cognitive metaphor of the audience or the side-observer in visual stories like film.

We begin with the framing techniques of matching framing specifications in a virtual environment, which can be understood as the look-at metaphor for the invisible observer in a film. Next we review techniques of automated editing between multiple cameras in the scene in order to create sequences that match a certain style, and convey elements of storytelling such as causation, timing, and space. The final part of this section reviews developments on discourse components and AI smart cameras that work with the drama managers of interactive storytelling systems to put together the story and its visualisation.

4.1 Camera Positioning and Movement

Currently, in computer graphics (and robotics) there is an abundance of automated camera control techniques that address diverse issues such as viewpoint computation, target tracking, path-planning, visibility computation, or editing. Figure 2.14 is a detailed overview of the first level, the framing level, of Figure 2.7. The input of viewpoint computation and tracking algorithms is the 3D geometry of the virtual environment, including elements such as actors, objects, the navigation map of the scene, and etc. This information can be obtained directly from the game engine or 3D modelling software in which the props are placed. This information is processed, and the output would be one or more selected camera positions and trajectories in relation to the 3D environment that fulfil the constraints that a director may set, such as avoiding occlusion and conforming to specific aesthetic styles.

A detailed presentation of the techniques involving virtual cameras, categorised by the type of system and approach is provided by Christie et al. [CON08] and updated by Lino [Lin13]. Here, we focus on approaches that build on such technical solutions to encode cinematographic theories and practices observed in film literature, and find potential in storytelling and virtual cinematography contexts. This includes aspects such as:

- calculating a camera position from framing constraints of elements on-screen
- movement and navigation of cameras in a semantic environment
- arrangement of elements in a scene in relation to the camera to produce choreographed camera positions

4.1.1 Framing

A camera's function is to go where the action, characters, objects, and events are located, and show them. News reporting or certain documentaries are close examples of this in non-virtual environments. Cameras need to deal with the challenge of framing: selecting from infinite possible camera positions, the one that can accurately position

Figure 2.8 – In this section, we explore virtual camera techniques that have the purpose of either placement to satisfy on-screen constraints, or navigation through a 3D environment. While we focus on problems concerning individual cameras in fixed environments, problems of placing multiple cameras and posing/choreographing scenes are also reviewed.

targets on the screen, follow framing guidelines (such as the rule of the thirds, diagonal dominance,...) while expressing complex relations between targets (such as a character dominating another one) through appropriate layout of scene elements.

Blinn [Bli88] first proposed the challenge to solve camera placement problems of situating two objects in the 3D scene to result in exact 2D positions on screen. Responding to the call for a more general and expressive approach, [RU14] proposes a method for calculating camera positions and evaluating frame quality with multiple on-screen parameters including size, actor relative positions, occlusion, and angles. The computational complexity is addressed using general purpose solving techniques (genetic algorithms) with dedicated intialisations and heuristics. A less general approach, however more efficient has been proposed by Lino et al. [LC12] [LC15]. Given two target objects, the authors propose a new representation of the camera's degrees of freedom, and propose efficient means to compute viewpoints given visual properties to be satisfied (Figure 2.9). The method has its strengths in real-time camera planning in that the calculation time is fixed and can react to moving targets.

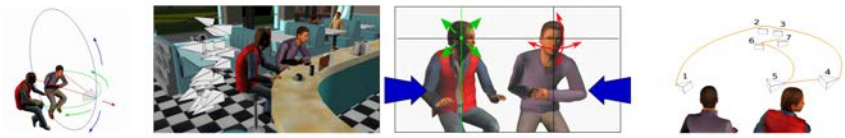


Figure 2.9 – Camera placement using the Toric Space allows quick calculation of framing specifications, and constructing long tracking sequences from keyframes [LC15].

Both approaches easily adapt to dynamic environments, given simple information on actor and object locations, and can easily reuse framings for one scene to another without any pre-computation on the scene. On the other hand, though it does address aesthetic important properties of on-screen actor distribution and smooth transitions between framings while maintaining the on-screen specifications, the output result varies greatly depending on actor positions. The authors demonstrate applications to storytelling, but the method, being purely geometric, has very limited context (only taking into account actor positions, perspectives, and orientations).

4.1.2 Movement and Navigation

Navigation through with a smooth path and meaningful (i.e. to capture the most important elements in the scenario, and in the right order) in a complex 3D environment

is one of the most basic types of camera problems. One such environment is the 3D museum environment. Drucker et al. [DZ94] uses the A* algorithm to plan the camera paths through multiple rooms and within rooms. While the framework is useful for a 3D exploration context, further constraints could enrich the framework, for example, navigation between art pieces to create some narrative sequence. Galvane et al. [GCLR15] developed the method of virtual camera rails, which was inspired by pre-constructed rails for cameras to slide on in actual film sets, calculating smooth camera trajectories around standing or moving actors using the Toric Space [LC15].

Whether it is the solution of a single camera that is well-placed, or a selection process among multiple prospective cameras, methods for camera placement are currently targeted towards capturing action in a precise and efficient manner. In terms of accuracy and efficiency, existing placement techniques have been shown to be sufficient for simple storytelling scenarios with clearly defined and limited number of targets. To automatically recreate complex movements, such as long shots and tracking scenes with changing targets is still a great challenge.

4.1.3 Actor Choreography

Another way to describe the cinematographer's job is to find the best relative positions between the actors, the 3D scene, and the camera to produce the desired framing. [ER07] adopted the idea of blockings from actual film practice, which involves the absolute placement of a number of characters in a non-occluded environment (i.e. "stage"). Then the camera is placed in pre-calculated relative positions in relation to the actors and stage. The database of blockings, stages, and shots can be expanded.

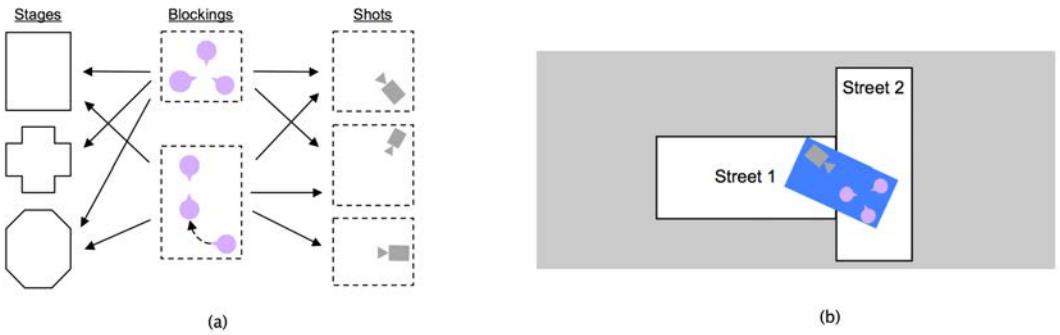


Figure 2.10 – The Cambot system proposed using a database of blockings to ensure that actors are not occluded in the environment [ER07].

This is much closer to the reality of a film set, without constraining the camera's possibilities due to awkward actor positions, and thus can strongly control the quality of output shots. Yet only very few works approach virtual cinematography from this aspect. Since the camera adapts well to actor actions, the scene, and relative positions, the author has better control over the camera in relation to story context. However, with large possibilities of scenarios, the calculation of possible positions must be done offline. Blockings must also be designed to match the 3D environment and actors. It is unclear whether any cognitive aspects of this approach are taken into account from existing literature, however, we believe it would be simple to add framing information

to the blockings in order to determine shot types, and have some level of control over viewer emotions.

4.2 Editing

In this section, we specifically review one main area of virtual camera control: virtual camera editing.

As summarised in Section 1 there are a specific set of rules of how to cut and sequence, and then ways to break the rules. These editing rules can carry meanings on the causal, temporal, and spacial aspects of visual storytelling. In actual film editing, the cinematographer may film a number of takes–the same scene shot from various perspectives, angles, and distances–which are turned over to the editor to cut and paste for the final arrangement. Yet in the virtual environment, infinite number of cameras can be placed and moved anywhere, and the editor's job is to select among all these possibilities the best position or plan the best movement for the story. Thus, placement, movement, and editing become a single problem of sequencing virtual camera positions and movements in a meaningful and fluid way.

Figure 2.11 is a detailed view of the second level of Figure 2.7, editing. Editing techniques must have not only the geometry of the 3D scene, but also have the events that are taking place, and the causal, temporal, and spacial relation between events in order to make meaningful sequences. The output would be where cuts should take place, how shots are ordered, and the duration of each shot (i.e. timing and rhythm). The output sequence would convey the same things that are the required input, namely, the spacial relation between actors and scenes, and the temporal relation between events.

Figure 2.11 – In this section, we explore virtual camera editing techniques. Many virtual camera planning systems and algorithms address sequencing and timing problems in the same system or approach. We found that two main approaches that were adopted for such systems: constraint-based systems and learning (through data annotation).

4.2.1 Rule and idiom-based systems

With very clear and implementable criteria, constraint and rule-based methods have strong applications to virtual systems that wish to replicate film language.

The first system to use cinematographic idioms in order to decide the sequence of camera positions was He et al. [HCS96], who implemented the selection algorithm as a finite state machine, a transition graph in which the states are the possible shot types, and the edges between the states are possible transitions between the shots. Christianson et al. [CAH+96] was the first to view virtual cinematography as a language constructed from film idioms, designing the DCCL language for representing cinematic knowledge that could direct camera movements for autonomous camera control

in game applications. The language takes in a number of fragments–basic camera positions and motions, such as pan, track, static...– combined to create camera shots and idioms that would be used in cinematography. Example fragments are shown in Figure 2.12 The work features 16 idioms designed by the authors that are used in a sample game. The approach is a big step in creating a film-like language for virtual environments. The system's pipeline is also very typical of the workflow of actual filmmaking. However, as the authors described, the purpose of the language is only to focus on the geometrical context. Yet in actual films, the selection of camera idioms also has much to do with contextual information in the scene: actors, objects, actions, emotions, and events.

Setting out from rule-based systems, [BGL98] introduces a real-time system for camera placement using constraint-based approaches, gathering information about the occurring story events and making decisions on how to best show the action. The system provides a nice solution to real-time context aware scenarios. Similarly, Bares et al. [BTM00] developed a constraint-based approach to framing and camera positioning for virtual cinematography, where the user can specify a number of constraints on the depth, angle, distance, region, occlusion...etc, and the camera will find shots that satisfy the decided constraints. It is more contextually aware of the scene and characters. A constraint-based approach was used by the same author to design a shot editing tool for virtual environments [BMBT00] [BL99]. However, resolving conflicts between constraints, maintaining consistency of camera style, and creating smooth transitions between shots remains open challenges for this model.

[FF06] provides a rule-based method of generating automatic camera shots, and carries out a user evaluation. From a more dynamic storytelling point of view, the system provides camera specifications where the shot-actor relative positions are fixed, and thus could result in problems such as occlusion. Also, the specification of rules is more limited in the number of techniques that can be used.

Film idioms not only instruct cameras how to frame, but also how to edit: transitioning from one shot to another and sequencing shots and frames. [HHS01] proposes a method of enhancing frame coherence in gaming environments that fulfils constraints, makes character action predictions, checks for visibility, and tries to preserve smoothness of camera motion. The method provides a different camera trajectory than the typical first person perspective in games. Though the work mentions the consideration on story parameters, it seems the algorithm mainly only solves for geometrical smoothness and visibility, and therefore the camera is not really context aware and selecting shots that best convey a certain scenario or event.

The implementation of camera idioms is also widely used in storytelling scenarios, such as [HCS96]. The shot sequences represented as finite state machines begin to reason story contextual and emotional layers. The work points out the importance of encoding cinematic expertise into computer generated content. It also demonstrates how story and discourse work together by making slight adjustments to the actors so that the presentation is good. However, the camera models are few and the shot positions are relative to the actors. Also the scope incorporated into the state machine makes the number of framings and possible transitions more limited. Editing of such idiom state transitions is also difficult for people of non-technical background. Amerson et al. [AK05] proposes a system for real-time camera control called FILM which is devel-

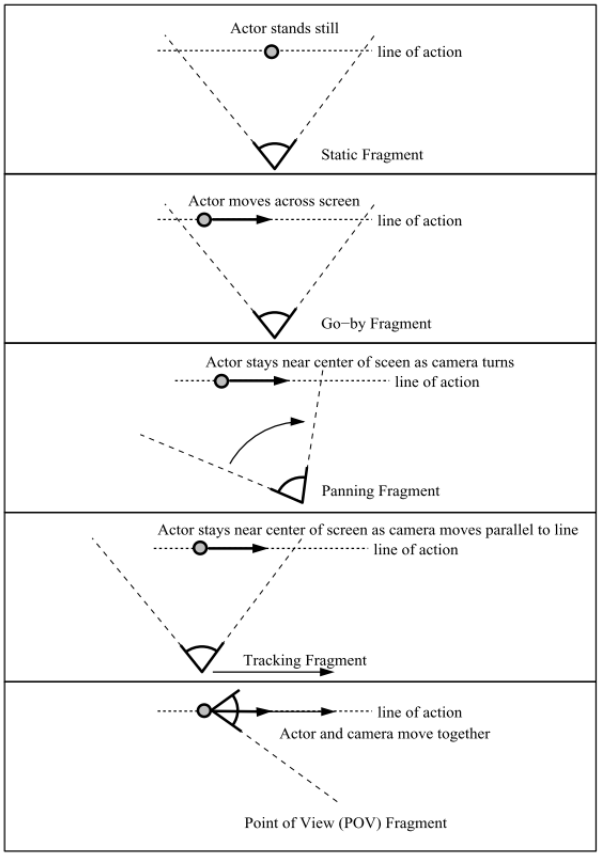


Figure 2.12 – Original image from [CAH+96] showing how fragments in the DCCL language that embody basic cinematic idioms. It is the first idiomatic approach to virtual camera systems.

oped from well-established film idioms. The authors create the structure of a scene tree which goes from a generic shot to shots with very specific emotions and contexts. Each shot has cut cues, shot position, and constraints. The system has the benefit of being in real time and aware of film expressions and story context. We see how future work in this direction could develop scene trees that address the problem of structuring shot sequences to express these emotions and contexts, and not just individual shots. Also, the model is still quite simple in terms of the constraints and characteristics it takes into account.

In a more recent work, the authors design a cinematography system for interactive environments that are real-time, can enforce certain framing rules, and provides variation over director styles [LCL+10]. The work proposes the use of director volumes to compute camera positions that would result in certain framings while avoiding occlusion. The method also respects continuity rules and responds to narrative dimensions such as affinity, dominance, and isolation. The proposed method departs from traditional constraint-based systems by not only taking into camera rules, but also narrative context. It is also a bit step towards automated camera control for highly-contextual, interactive, and dynamic environments. However, these systems seems to need more flexible means to extend the character behaviour and reaction options. Also as the camera is not an active teller of the story, it may not be able to provide visual structure to a

sequence of actions or obey cinematic constraints and rules.

Related vocabularies for describing shot sequences in films. Most notably, the Prose Storyboard Language (PSL) [RBLA13] was designed based on actual film practice on framing, including vocabulary for elements like size, region, or movement. In the implementation in [GCR+13], PSL targets autonomous camera steering and positioning, conducting a search for suitable camera positions based on PSL constraints.

Constraint-based systems have a strong similarity to actual film practice, to implement good film aesthetics, and follow guidelines of viewer cognition. Moreover, even with higher regard to the story context and character actions in coordination with good filming practice, the general idioms and rules can be reused in different scenarios. They do not yet replicate good cinematic style, but following general constraints on framing, movement, and cutting greatly improves the aesthetics of film experience. That being said, these systems can be slower depending on the number of constraints that need to be fulfilled and parameters optimised. Also, they require much more information on the scene in order to be adapted.

4.2.2 Multiple Camera Approach

Variant of the above idea is to allow multiple cameras to be placed in the scene, then select the one with the best on-screen output. [AWCO10] proposes a method in which by performing a canonical correlation analysis between the given scene animation and a camera output, the algorithm can quickly select best camera views from a large number (tens) of cameras. The method has a number of benefits, being very fast (computing online at 15-20 FPS for less rapid motions), simple, and can take into account of some camera rules (such as the 180 rule and jump cut). Though the algorithm is sufficient for simple scenes with singular motions and few characters, it will have limited ability to adapt to environments with complex intention-drive motions, such as baseball or basketball games, and further, storytelling scenarios.

In an application scenario for crowds, [GCR+13] uses steering behaviours to control multiple cameras to either search, and when found, track interesting events in a crowd. The approach takes into account various perspectives and smooth movement of cameras in the scene. Virtual cameras also take on smart behaviours, like coordinated journalists that find and follow subjects of interest in complex and non-predefined environments. The silent observer approach to cinematography has its basis in embodying the viewer as an invisible and observant bystander. However, takes limited context into account, only identifying events with two characters face-to-face or groups moving in the same direction.

3D gaming has also seen an increasing importance in cinematography through multiple cameras. In [MK06], the author proposes to replace the first-person camera for shooter games with a virtual cinematographic system that selects film-like shots to create more engaging storytelling experiences. The cinematography engine takes in information concerning the current state of the game world, positions and selects active camerabots, and makes decisions on framing or cuts. Even though the user evaluation shows that cuts provide confusion to the user, the game was more engaging. However, the application is specifically targeted towards first person shooter scenarios.

4.2.3 Annotating and Learning Style

From a machine learning approach, [dLPD+09] uses SVM to train the virtual director on making shot selections based on the environment, scene, and actor features in the scene. The system trains on a shot database extracted from real films, and then makes prediction on shot selections. The authors then evaluate the performance in terms of speed and accuracy. Though this approach can maintain a certain amount of accuracy based on the quality of the features extracted, it has limited ability to address story structure over a sequence of events. Similarly using learning techniques to replicate director style decisions, [GRLC15] uses semi-Markov chains to optimise a number of parameters in camera shot and cut decisions. These decisions are mainly motivated by film practice of evaluating narrative importance (of actors), observing continuity, and creating rhythm in the edits. [MBC14] uses Hidden Markov Models to imitate a director's decision process when deciding when/how to cut between shots, what shot types to use, and which actors to focus on.

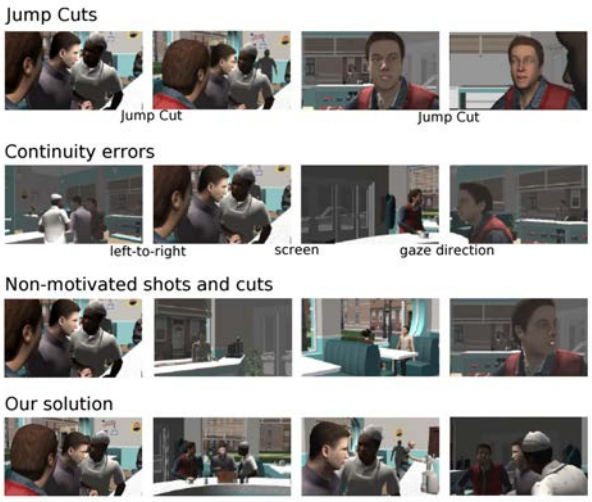


Figure 2.13 – Motivated by film editing practices, editing algorithms try to optimise cutting, shot type, and tempo in entire sequences [GRLC15].

4.3 AI Cinematography Systems

We are now moving away from techniques to look at storytelling systems that have smart AI-camera systems. Figure 2.14 combines the final two levels of our overview figure: the discourse component, and the storytelling. In cinematography systems, the discourse component often use planning and optimisation approaches, taking director intentions and story emotions as input, and translating them into a visual story that has plot progression, and brings the audience reflection and enjoyment.

Current cinematography systems and architectures developed for storytelling applications have an awareness to the necessity of linking the story generation/director to cinematography architectures. Starting from film editing techniques, [En04] proposes an architecture for interactive narratives with camera placement, lighting, and character control. The approach attempts to bind narrative elements to discourse elements.

Figure 2.14 – In this section, we explore AI cinematography for storytelling. We focus on the how these components communicate between the editing and camera components, and their decision-making process in relation to story, context, emotions, and intentions.

The system is composed of a story engine, director, film system (lighting, camera, and character/actor), and the final rendering. The system proposed seems rather typical of interactive storytelling systems, while the approach from film perspective is innovating. [KM02] proposes the design of a cinematographic system that is able to convey scenic moods and character themes in a 3D animated scene. The system takes as input a series of characters, objects, and scene specifications, and outputs a series of shots that best conveys the scene. The decisions are then sent to the renderer to produce. The cinematographic aspect of system architecture seems comprehensive in that it takes into consideration frame, light, color, and shot structure to convey a certain mood. However, from the example, it seems the system only responds to very simple story scenarios. It also cannot produce complex cinematographic shots or relate on context between consecutive shots. The implementation of the above camera parameters are also very basic implementations, and cannot produce very specific framings or focus on secondary objects in the scene.

The camera can be implemented as an agent that reasons on story context and select shots at run-time that best convey the story [HLT03]. The idea comes from modelling the camera with its own decision process. It takes into account parameters such as emotion, actions, object, priorities, excitement...etc. and selects a best shot from the database. The workflow of the camera agent is typical of a database lookup for shots that fit the parameters of an event. In this way the quality of the output may depend highly on the quality of shots in the database. In contrast to the implementation complex film idioms and sequencing rules, it cannot reason on style over multiple narrative events apart from immediately preceding events.

[CLDM03] introduces a framework to interactive storytelling scenarios composed of a camera control unit to position the camera using placement techniques, and a decision-making component that evaluates cinematic components as well as observes stylistic factors in the frame. The framework grounds the design principles to a cinematographic system in terms of storytelling applications. However, its reactive approach to storytelling may require extension on the communication both back and forth between story and camera for interactive scenarios. The efficiency and effectiveness of the framework would also need to be evaluated in an actual implementation.

A challenge that is still to be met by these systems involves how to design cinematography components that take account of the structural and/or semantic aspects of narrative in its result. Using the fundamental blocks of camera techniques, our goal is to build a cinematographic system that reasons on and responds to context related information. We have observed how placement algorithms and rule and constraint-based systems can implement film idioms and replicate real film practice in virtual environment, and we see the applications to complex scenarios. In this section, we review the

AI techniques involved in smart camera systems for AI storytelling systems.

Planning techniques, which are central to emergent stories, have also been expanded for cameras in virtual environments. Similar to the concept of a scene tree in FILM [AK05], Darshak is a discourse planner that can recursively find suitable camera shots for a given story scene [JY10]. The algorithm starts from top level story events (e.g. a "bank robbery"), gradually decomposing the event into sub-scenes or individual actions (e.g. "opening the safe"), and finally individual commands for the camera to follow (Figure 2.15). The decision process then selects camera shots suited to each action. However, the planner has to specify a cost function for every scenario or action being shown, and it would be much desirable to have a more general model of expressing emotions for similar scenes. [CL12] presents a method of automatic camera planning for interactive storytelling scenarios in 3D environments using motion planning.

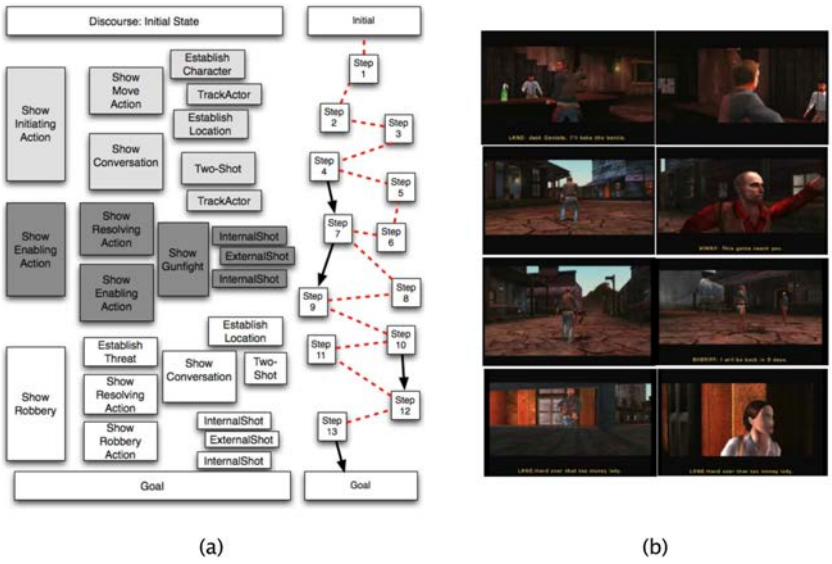


Figure 2.15 – The Darshak discourse planner (a) generates discourse plans that can be mapped to specific shot types, which are then (b) realised in a 3D environment [JY10].

From film to virtual film, many works target at replicating good camera practices in storytelling scenarios. [MKSB11] uses behaviour trees to represent relationships between the camera and its events. Camera idioms such as shot reverse shot, tracking, can be linked respectively to dialogue, or movement events. [TBN00] develop a reactive autonomous cinematographer that takes emotion and motivation as the main properties to camera decisions. The camera reacts by selecting shots according to the camera agent's mood or desire to show some part of the scene in a certain way. Then the camera shoots according to factors such as relationship of actors, motions, angle, transition, and occlusion. The agent also manipulates the lighting to give atmosphere to the scene. The system naturally reacts to user responses in interactive scenarios. [GRCS14] uses an optimisation-based implementation of PSL [RBLA13] combined with placement techniques of [LC12] to create cinematic replays of game sequences.

These intelligent camera behaviors for AI storytelling applications play the important role of linking camera techniques to context due to the decision processes always strongly tied (or even involved) in the current story scenario. One characteristics of

these systems such as [MKSB11] [JY10] is that they can conveniently code cinematographic knowledge. However, these systems are usually more somputationally expensive, and in many cases, are designed to work offline. They are also usually limited to the scenarios they were designed for.

4.4 Summary

Existing methods take a leaf from classical framing and editing techniques through implementation of film idioms and existing practices. Virtual cinematography problems have identified a number of factors such as visibility, distance, and movement that carry meaning in actual film practice. However, these works–focusing on aspects of camera placement, navigation, editing, and discourse planning–have been, at most, parts and components to a cinematography engine that may or may not have had storytelling applications in mind. Though many of these works propose video games as a field of application, it should be noted that in most video games–especially in genres that contain elements of action, racing, platform, sport, and simulation–one smooth, and sometimes user-controlled, camera from a fixed perspective is preferred over having complex cuts and transitions when the player is interacting with the environment.

The lack of film-like cinematography in games is due to three reasons: first and primarily, cuts and transitions can confuse the player when ease of control, ease of navigating a 3D scene is vital in fast-paced and intense gameplay. The second reason, and the one that concerns our work most, is why we rarely see good cinematography for other genres that are strongly narrative driven, is that it is a hard problem. Camera placement, movement, and editing techniques are still an insufficiently explored field in computer graphics and animations. Currently, cut-scenes, with hard-coded camera shots, are the major vehicle to storytelling in action-filled interactive experiences. In other interactive film or story strong games, such as *Heavy Rain*, the camera moves are beautifully choreographed. As we have seen from our review of film practice, virtual cinematography can be no less an art than actual film cinematography. Finding a good shot for a story scenario requires the coordination between the actors in the scene, the story, the discourse manager, along with the camera techniques reviewed in first two parts of section. Finally, understanding on how film cinematography techniques can be applied to game scenarios in order to progress the story is also lacking. In order to have good cinematography, we must first observe and learn from films, and evaluate what can be borrowed, and what new visual storytelling idioms need to be developed.

The challenge of designing dynamic and film-like cinematography for interactive storytelling is addressed our third contribution. We position this work as a discourse-based and automated camera component for interactive storytelling with the following characteristics in comparison to the literature:

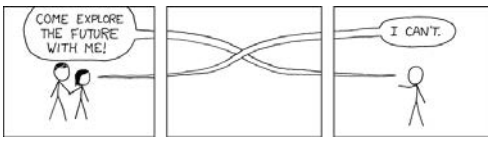
1. **(partially) dynamic**, as opposed to completely pre-computed camera movements
2. **data-driven** from observances in actual film practice, as opposed to only constraint fulfilling and occlusion avoidance
3. **interactive**, as opposed to pre-calculating a series of camera shots in the story

4. targeted towards **storytelling**, as opposed to other virtual camera applications

Within the same contribution, we also introduce context-aware editing tools for the camera for pre-visualisation of 3D stories.

Temporal Properties of Interactive Stories

3



XKCD

Contents

Russian formalisms have described narratives as being constructed from the *fabula*–which is the actual raw material of the story–and the *syuzhet*–which how the events in the story are organised temporally. The study of temporality in narratives has shown the importance time plays in the telling of the story, and by simply rearranging the seqeunce an event is presented, one can change the whole interpretation on the same set of events.

We begin the journey by understanding these temporal properties of stories. This is essential before delving into the visual cinematography component of film and game storytelling. Film editing, which involves selecting and arranging camera takes into a single sequence, can be viewed as the syuzhet of a visual narrative, and has much to do with the story, too. Questions such as "Which events happen first? Second? Last?", "What is the cause-effect relation between events?", "When do we want to reveal

the-most-important-plot-point to the audience?" are frequently posed by directors and editors throughout all phases of the filmmaking.

This chapter presents two contributions related to temporal properties of interactive storytelling. The first contribution mainly concerns authorial control in interactive stories: the capacity of the author to guide the story towards the morals and outcomes that the author would like to communicate to the audience. We design a logical mechanism that allows authors to ensure that stories are logical–that is, cause-effect relations between story events can be observed and are upheld–even when the story is not chronologically presented. The second contribution establishes a model of memory in order to understand how viewers would perceive and interpret stories that are generated with temporally rearranged events (such as flashbacks).

1 Controlling Logic in Temporally Rearranged Stories

After watching a blockbuster film like *Star Wars VII: The Force Awakens*, fans start to discuss and speculate what will happen, and argue over controversial points of the plot, demanding "[spoiler[1]]". Youtube channels such as *CinemaSins*[2] and *How It Should Have Ended*[3] begin listing impractical and illogical plot events in movies, while providing a more reasonable (albeit sarcastic) ending to the story. Despite the common myth that viewers are goldfish[4], the general public is still very keen to spot errors in a plot and raise debates about everything in the story.

With the growing complexity of the story and managing outcomes of interaction, the task of authoring with regards to story logic and content becomes a big challenge for the story author. Computational algorithms and the mature understanding of narrative structures can provide authors with better tools to create more personalized, engaging, and well-controlled narrative content.

1.1 Background and Contributions

In Chapter 2 we have provided a general overview of interactive storytelling systems. Between planning (i.e. the plots are composed of a selection and ordering of actions/events from a pool of authored actions/events) and graph (i.e. the plots are composed of events represented as nodes, linked into a graph-structure by edges that represent absolute precedence constraints) approaches to representing interactive stories, story graphs have been popularly adopted by existing games as game trees or story graphs [Wey97] [NM05] [RLAR11] (an visualisation of planning and graph representations of the story can be seen in Figure 2.3). The story graph, whether hand-authored or as an output of a planning-based algorothm, has benefits of being suited to represent the story structure, and can efficiently process runtime user decisions. Yet maintaining logic over complex story graphs (and maybe even temporally non-linear) at runtime is also a difficult task that requires laborious authoring to ensure that no illogicalities

[1] Why did R2D2 just suddenly wake up at the right time?
[2] https://www.youtube.com/channel/UCYUQQgogVeQY8cMQamhHJcg
[3] https://www.youtube.com/user/HISHEdotcom
[4] http://tvtropes.org/pmwiki/pmwiki.php/Main/ViewersAreGoldfish

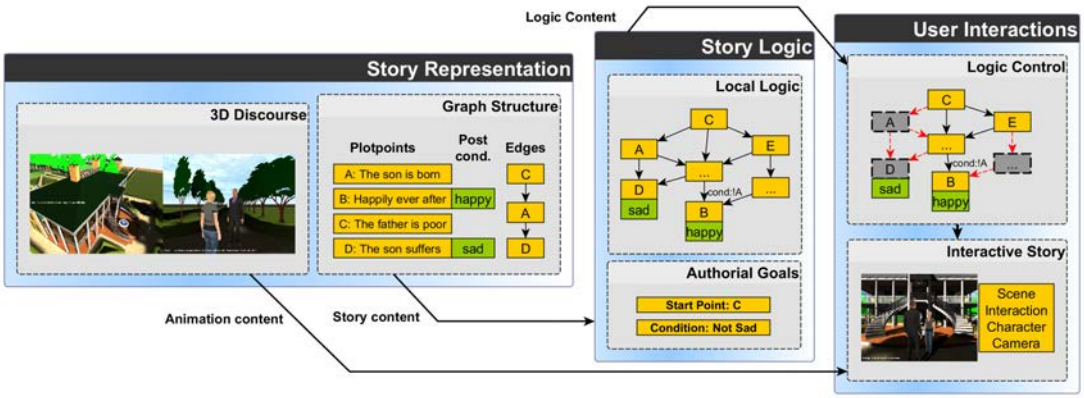


Figure 3.1 – Interactive storytelling system overview. In our system, the author can design story and animation content, and set authorial goals. The logic control uses a double traversal algorithm that checks and filters story nodes and edges so that no dead ends would result from user decisions.

exist in the story graph. Also, user interactions may be greatly restricted or have low impact over the story due to the rigid structure of such graphs. Therefore, it is much desired to have a set of logic controls that allows the author freedom of authoring, and then adjusts the tree to fulfil story logic, user decisions, temporal arrangements, and authorial goals in one mechanism.

We propose a logic control algorithm specifically suited to maintain the logic control of the story graph structure. Our algorithm is suited to story graph structures that are interactive, require stronger control of logic, and possibly involve non-chronological story segments. The method uses the plot point representation of stories [Wey97]. It takes as input a pre-authored (or generated) story graph of plotpoints connected by edges that represents a range of possible stories and a set of user-defined authorial goals (e.g. "a murder and a happy ending"). Our method then outputs a subgraph of the original story graph, which guarantees that whatever the user interactions are, the story will achieve the given authorial goals. At each interactive plotpoint, the algorithm subsequently prunes the story graph to maintain logic at runtime. Moreover, when the temporality of the story is non-linear, involving embedded scenarios, the algorithm ensures user interactions within and outside of the embedded scenario extend to other parts of the story, ensuring that no matter in what sequence the story events play out, the logic remains consistent. The algorithm is linked to an authoring interface and scene in Unity 3D, which we call the *Theater* to demonstrate the effect of logic control on the discourse in a real-time 3D environment.

The main contributions of this work is the design of a logic control mechanism for narratives with embedded plot lines that (1) enforces local story logic (such as pre-conditions, and post-effects of story events) and authorial goals to be upheld without running into unresolvable plot points, (2) manages story logic over user interactions of non-linear stories at runtime, and (3) performs dynamically in 3D environments.

1.2 Overview

There are three main components to the system (1) authoring component presenting the story as a graph structure and 3D content (characters, animations, scene...etc.), (2) logic control algorithm over temporality and user interactions, and (3) the link from the story graph to the 3D presentation. The relation between these components of the system and the presentation platform are shown in Figure 3.1.

The authoring component allows authoring of the events to create a pool of possible plotpoints, and linking them up into a story graph. The logic control algorithm then performs a double depth first search traversal for two purposes: to ensure that pre- and post- conditions on each event is upheld by removing any illogicalities in the story graph, and to add additional constraints that would ensure that authorial goals are achieved. Finally, the method is linked to the 3D content and allows users to experience and interact with the story in real time. The story can begin at various major plot points in the graph, and flashbacks are used to fill up missing details. The next section begins to outline our method by explaining the story representation: the story units, the story graph, and how the story graph represents complex structures such as embedded plot lines.

1.3 Story Representation

In our branching story graph structure, the story events are represented as nodes, connected by edges that represent temporal precedence constraints. The logic control algorithm places constraints on the edges of this story presentation in order to control and restrict the flow of the story. We explain here how the story is represented using these nodes and edges.

Note that, in this section though we coin the term “authoring" to describe the process of building the story graph, the story graph does not necessarily have to be manually authored. We envision the capability of planning or search algorithms that could generate such a story graph. The story graph is therefore intended to be seen as a formalisation or a generalisation of graph-based representations of branching narratives that are widely used in current games.

1.3.1 Story Units: Plotpoints and Postconditions

Interactive narratives for games are often comprised of basic units of events (such as a dialogue, a fight, a travel, etc.). We refer to these units as plotpoints in our story graph. This differs from character-based systems where the basic unit is not an event but an action that a certain character in the story can take, and from beats or vignettes, which are plan-based units derived from film terminology to describe basic story units that compose one overall story arc. The logic control evaluate the properties of these plotpoints, making sure the the story goes through plotpoints that have logical cause-effect relations to each other.

To help the logic control identify plotpoints that have cause-effect relations with later plotpoints, postconditions can be attached to each plotpoint to serve as markers.

Postconditions have either boolean values (e.g. the plotpoint concerns $Sara$, $murder$, and $mansion$) or integer values (e.g. the plotpoint has the effects of $happy+ = 1$). Using the above postconditions as an example, if the story goes through a plotpoint that has the postconditions $murder; happy+ = 1$ then the global parameter $murder$ is assigned the value of $true$ while the value of the global parameter $happy$ is incremented by 1.

The plotpoints do not need to be authored in any specific order, and any postconditions can be attached to the plotpoints, which act like postconditions that can be evaluated later on in the story. We do not restrict the size, content, or scope of a plotpoint. A story, or even multiple varying stories can be composed out of the plotpoints simply by linking them in a specified order. The linking and maintaining of logic between plotpoints is described below.

1.3.2 Story Logic: Edges and Embedding

On their own, each story plotpoint represents a single event unit within the story. When a number of plotpoints are grouped together, we call the grouped plotpoints a “move”, adopting the terminology from [Pro68], to describe a complete sequence within the story.

We then need to specify how each plotpoint within a move relate to other plotpoints. Arrangements of the plotpoints within the move are achieved by linking plotpoints to each other with directed edges that specify a total order between plotpoints. Plotpoints can exist in moves without any edges, but then signifies that the plotpoint has no temporal continuation with any other plotpoint in the move. Every move has an empty entering and exiting plotpoint (Figure 3.3). The first plotpoint in the move is the entering plotpoint, which directs to all the plotpoints that can serve as the first event or action in the move. All plotpoints that are the last plotpoint in the move point to an exiting plotpoint, which is either a concluding point in the story (thus a “The End" of the story), or the entrance point of another move.

The story representation can also allow embedding, which refers to the process of jumping to an external move B from after a plotpoint in move A. The idea is on finishing the current plotpoint in move A, the story then goes to and plays the embedded move B (suspending the current plot in move A) and when move B is finished, the story returns to where the plot was suspended in move A, and continues the story in move A. A small example of am embedded move can be seen is Figure 3.2.

Apart from the order of the events, edges are also a way to control logic. Preconditions can be placed on the edges, such as the boolean precondition $murder =$false (meaning that a plotpoint with the postcondition tag murder must not precede this plotpoint) or $happy \geq 2$ (meaning that the integer “happy" must have a value of 2 or more at this point). These preconditions can be grouped with boolean operators (**AND**, **OR**, and **NOT**) to form evolved preconditions to control the story. As edges can be placed between any two plotpoints that can logically follow each other, we can envision many circumstances when one node can have two or more exiting edges pointing to different story nodes respectively. Note that preconditions are placed on edges linking plotpoints, and not on plotpoints themselves, keeping the plotpoint as minimal as possible so that it may be reused in multiple stories flexibly. This is an important property in the story graph, where the plot line actually branches: the separation of the plot leading from a

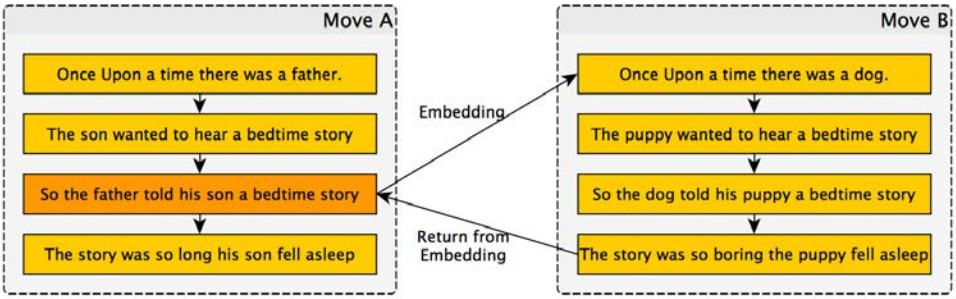


Figure 3.2 – Embedding is the method we have designed to allow atemporal rearrangements of story events inside a given plot line where events from a past or future move can be embedded and played out of order before returning to the main storyline. The orange plotpoint "So the father tells his son a story." in move A embeds move B, which happens to be the story that the father tells.

plotpoint is where user intervention can change the outcome of the story (by leading to different end nodes within the story graph).

1.4 Logic Control

Once the story graph has been either manually designed or generated through learning or planning approaches as in [RLAR11], we then demonstrate our method to perform logic control on this representation. Story logic comes into light as the author wants to achieve storytelling goals while managing user interaction in possibly non-chronological story lines. Like matching keys to their locks, the process of establishing story logic requires a series of pre- and postconditions; the more elaborate the plot design, the more complicated this web of preconditions, and the higher likelihood of dead ends, unresolvable user interactions, or unachievable story goals.

In this section, we introduce our logic control method over the generic story graph. The method is designed to assist the author in specifying global authorial goals, evaluate the sequence of events, and automatically manage story logic during runtime.

1.4.1 Authorial Goals

Authorial goals are defined as the storytelling goals that the author wants to achieve. The design of authorial goals corresponds with Chatman's theory of modes of plot [Cha80], where certain combinations of postconditions will result in specific emotions in the audience. For example, Chatman defines that a good hero that fails will arouse sympathy, while an evil hero that succeeds invokes a feeling of disgust. Such statements can be easily expressed as constraints on the postconditions on the plotpoints in the story graph. Taking an example, "**AND**($HeroFail = \text{true}$; $HeroKindness \geq 2$)" and "**AND**($HeroFail = \text{false}$; $HeroKindness \leq -2$)" respectively, where in the story graph, there are ending nodes with boolean postconditions of $HeroFail$ and nodes that accumulate or decrement the value of the integer parameter $HeroKindness$. Authorial goals are to be placed on the whole story graph and seen as preconditions that must be fulfilled before the end of the story. If we have an authorial goal of "$Magic = \text{true}$" it

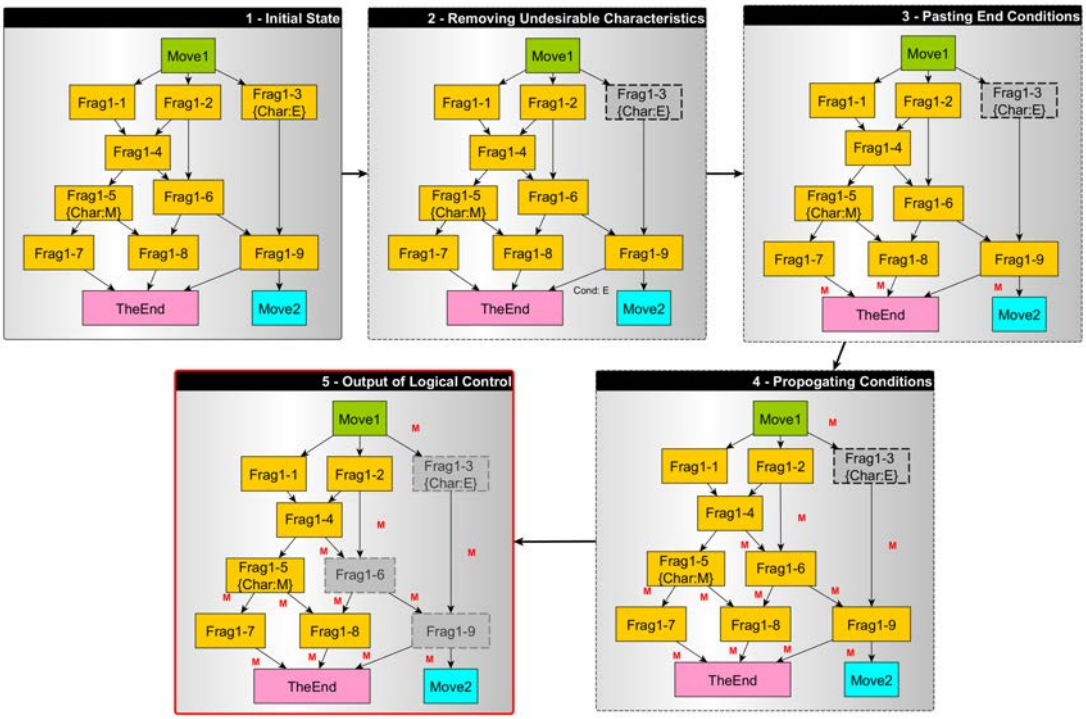


Figure 3.3 – Graph traversal algorithm for logic control. Given an authored move and authorial goal ¬ E ∧ M ("(**NOT** Evil) **AND** Magic')', the algorithm (1) removes plotpoints contradicting the goal (E), (2) pastes the goal (M) as an end precondition, (3) reverses and (4) propagates the preconditions on all edges, and (5) removes dead ends. As a result, remaining paths ensure the story logic as well as authorial goals.

would mean that somewhere in the story, there must exist a plotpoint tagged with the postcondition $Magic$.

We aim to ensure (1) that the local logic in the previous section did not resolve in any dead ends or unresolvable plot lines, and (2) that the authorial goals are ensured to be achieved (if a solution exists). We design the logic control algorithm for the purpose of upholding both the local logic and authorial goals (see Algorithm 1). The algorithm uses a double depth-first traversal to prune the story graph and output a subset of validated paths with reinforced preconditions on the edges to ensure that authorial goals are met. There are four stages in the algorithm to carry out authorial control: removal of contradicting plotpoints, paste end preconditions, reversal and propagation of preconditions, and the final validation and dead end removal.

Removal of contradictory plotpoints: From the story representation, plotpoints can contain a number of boolean or integer postconditions. The removal of contradictory plotpoints excludes plotpoints containing undesirable boolean postconditions. The algorithm takes as input the boolean authorial goals, performs a DFS (depth-first search) traversal on the story graph to remove any plotpoints with postconditions that contradict with the goal, where an authorial goal requires a boolean postcondition to be false. For example, for the authorial goal "**AND**($HeroFail$ = false; $HeroKindness \leq -2$)" all plotpoints with the postcondition $HeroFail$ are automatically removed, since by definition, they automatically contradict the goal.

Algorithm 1 LogicControl (node N, goals G)

```
1: if G violates descriptors of node N then
2:     remove node N, and node's incoming and outgoing edges
3: for all sons s of node N do
4:     if s has not been visited then
5:         tag s as visited
6:         LogicControl(s, G)
7: if all outgoing edges of N have preconditions then
8:     for all outgoing edges e in node N do
9:         cond ← cond ∨ preconditions in edge e
10:    for all descriptors d of node N do
11:        cond = negateDescription (d, cond)
12:    for all incoming edges e of node N do
13:        add preconditions cond to edge e
```

This ensures that no plotline will go through undesirable plotpoints. Boolean goals that are evaluated as true (i.e. "$SomeChar = \text{true}$") as well as integer goals are not evaluated for this step.

Pasting end preconditions: The algorithm then pastes the authorial goals as end preconditions on all the incoming edges of the end nodes. The reason for doing this is to ensure that, before the story concludes, these goals will be upheld. However, this step would result in possible unresolvable plot lines if not all of the goals are fulfilled before the story comes up to this point. The next step, the reversal and propagation addresses this problem.

Reversal and propagation of preconditions: As mentioned previously, pasting goals alone cannot ensure that a story will conclude. It is possible to choose a path in the story graph that reaches an end node, but cannot find any possible path that achieves the goal, thus causing the story to fail at runtime. To prevent this, our algorithm does a second traversal through the story graph bottom-up. For every plotpoint, it concatenates the preconditions from the outgoing edges, and pastes them to the incoming edges. We refer to this step as the propagation. However, this task is not just a copy-paste of preconditions from one edge to another.

Before preconditions are propagated, they are first reversed; the plotpoint checks its own postconditions against the preconditions. If a boolean precondition is fulfilled (for example, a plotpoint with the postcondition $HeroFail$ is propagating the precondition "$HeroFail = \text{true}$") it will see the precondition as fulfilled, and will remove it. Integer preconditions such as "$HeroKindness \leq -2$" are reversed by reversing the calculation done by the plotpoint (for example, the propagated precondition "$HeroKindness \leq -2$" will be translated as a "$- = 1$" precondition when encountering the integer postcondition "$HeroKindness \leq -1$"). This allows the integer value to increase and decrease freely throughout the story, thus creating highs and lows in the plotline. Finally, repeated boolean preconditions and redundant integer preconditions

can be eliminated. Though not required, this step ensures that the preconditions are concise and do not expand much throughout the propagation.

Validation and removal of dead ends Since preconditions are propagated up to the top of the story graph, it is fast to identify which edges have preconditions that cannot be fulfilled. The remaining step traverses the graph once more removing any dead ends (i.e. a plotpoint with no feasible outgoing edges), then outputting the sub-graph with all the feasible plot lines.

The algorithm we have described above not only enforces the achievement of authorial goals over a story graph, but also reinforces local logic on the edges by propagating them until they are fulfilled. By definition of this logic control algorithm, it is guaranteed that all paths that the user may choose in the graph must have at least one feasible path (chronologically) that (1) can terminate the story, (2) achieves all the authorial goals, and (3) does not contain any illogicalities. Figure 3.3 illustrates this process.

1.4.2 Embedding

Our method can also control logic in embedded moves that may represent temporal rearrangements, such as parallel events, flashbacks, or foreshadowing. As described previously, an embedded move is an internal representation within the plotpoint allowing it to embed another existing move like a sub-plot.

Given a start point in the story, the story progresses sequentially and chronologically down a feasible story path. When an embedding plotpoint is reached, if the embedded move has not been played before, the embedded move id played (otherwise, the story just continues to the next plotpoints). The algorithm automatically determines whether an embedded move should be played, records the progression of the story, and returns the story to the original point after the embedded move is finished.

However, when embedding and user interaction happen simultaneously, we need to ensure that user decisions extend to the embedded move at runtime. And vice versa, we need to ensure that decisions made within the embedded move extend to the the rest of the story. For example, if at the story entry point, an event such as the murder of Actor C is assumed to have occurred, then in a flashback of the crime taking place, only story lines that lead up to the murder of Actor C should be feasible.

1.4.3 Managing of User Interactions for Embedded Storylines

We designed a second algorithm for user interaction, Algorithm 2, to solve the problem of logic control for user decisions with embedded moves. It extends the previous algorithm by propagating, for each decision the user takes, the preconditions on the outgoing edges of the plotpoint throughout the story graph, as if they were new authorial goals. In this way, the plotpoint is ensured to be reachable and to be reached for all the remaining paths. The graph is then re-filtered for each decision. This algorithm also maintains a set of flags to record the current level of embedding, and the embedding point, ensuring that the story returns to the correct plotpoint after finishing embedding.

Our mechanism for logic control in the stories is consistent with Branigan's mode of comprehension of narrative [Bra92]. Each story plotpoint is understood as a bottom-

Algorithm 2 UserInteraction (Decision D)

1: **if** current node N has $embedding$ **then**
2: $embedLevel$ += 1
3: push node N into levels stack L
4: next node N' = embedding entrance point N_e
5: $embedFlag = true$
6: **if** $embedFlag$ is $false$ and $embedLevel > 0$ **then**
7: current node N = pop L
8: get next adjacent node N' from current node N and decision D
9: **for all** incoming edges e of node N' **do**
10: $cond \leftarrow cond \vee$ preconditions in edge e
11: LogicControl(story entrance point s,$cond$) **return** next node N'

up independent unit of action with certain postconditions in relation to its content, and the story graph is a top-down logic control unit, with a filtering algorithm that (i) preserves preconditions between narrative plotpoints, and (ii) assists the author in meeting authorial goals in the form of end preconditions.

1.5 The *Theater*

The simulation of our interactive story is performed by a 3D animation engine built on *Unity 3D*, featuring smart control of characters, which we refer to as the *Theater*. We designed the *Theater* stage manager for the purpose of communicating the story events to the presentation–the 3D animations, scene, and sound–and the user interactions back to the story. The *Theater* decouples the story and presentation elements of the system such that the two must communicate through a virtual director.

In the *Theater*, we first semantically label all the actors, objects, and the regions in the scene (e.g. tagging the regions of a 3D model of a house into rooms, such as the kitchen, living room, corridor, etc.). Other semantical attributes of the actors and objects are also annotated, such as emotions of the actors (which would affect the facial animations of the actors), or for objects, whether they allow interactions such as grasping or sitting on.

Each plotpoint (i.e. each node in the story graph) is linked to an authored XML play script, which describes the high-level actions and character behaviours occurring in the fragment, together with abstract locations. The author specifies behaviours at a semantic level (e.g. meet someone in a given room, find a specific object or exchange utterances with a character). Our 3D character animation engine then computes the exact locations, paths and 3D animations for the characters, given the geometry of our 3D environment and the behaviours to simulate. Here are multiple examples of the XML play-script for various character actions.

An actor called $Sara$ walking to any unoccupied, free position in the region labelled $EntryGraden$:

```
<IrisAnimCharacterGoto actor="Sara">
  <Target type="FreePosition" surface="EntryGarden"/>
```

Figure 3.4 – Sematical annotation of a scene in *Theater*. The 3D scene can be split into various semantical regions as shown in this example of a house. Each region contains a number of position labels that the actors can go to.

```
</IrisAnimCharacterGoto>
```

An actor Jack and Grady having a dialogue while Jack does a stretching animation.

```
<IrisAnimSequentialActivity>
   <CallBacks>
      <CallBack type="IrisAnim">
         <IrisAnimMotion actor="Jack" loop="true" motion="stretch_arm"/>
      </CallBack>
   <CallBacks>
   <IrisAnimDialog speaker="Jack" lookat="true" listener="Grady" text="Eh,
       Mr. Grady, haven't I seen you somewhere before?" audiofile="audio1"/>
   <IrisAnimDialog speaker="Grady" lookat="false" listener="Jack" text="Why
        no, sir. I don't believe so." audiofile="audio2"/>
<IrisAnimSequentialActivity>
```

The input of our simulation engine is an XML-based interactive script generated by our double-traversal graph algorithm. This script encodes the plotpoints, the links to the high-level description of behaviours in each plotpoint, the user interactions which link possible fragments and a first order logic formulation of constraints which trigger transitions from one plotpoint to another. While designed in mind for interactive stories, this script language is very similar to the Q language [Ish02] featuring synchronous and asynchronous commands as well as guarded commands and is a superset of MPML3D [NPAI06]. Though it shares character behaviour modelling features with the BML language (Behavior Modeling Language [KKM+06]), it is not restricted to reactive conversational agents, and can be compared with the level of abstraction proposed in storyboarding using X3D [DB07], with extensions to handle multiple branches.

Since the purpose is to demonstrate the result of logical control on the discourse, we currently use a simple implementation of the discourse system where the drama director communicates its plot decision to the stage manager, who in turn executes a pre-authored animation scenario corresponding to the particular fragment.

Running the simulation therefore consists in executing the current plotpoints and then evaluating the constraints which trigger the execution of next fragments with or without user interaction. The evaluation of the constraints guarantees both the completeness of the story (whatever branch is selected, the player will reach the end), and the correctness of the story (whatever edge is selected, the resulting story will fulfil the selected constraints).

User interactions can also take place when the user is given options by the drama director. When this takes place, the stage manager records and returns the user's decision to the drama director so as to assist the drama director in its decision-making. To enable a proper separation between the semantic description of behaviours and their geometric application, our character behavioural engine implements the following features. First, the user can author complex behaviours as a hierarchical and temporal organisation of primitive or complex behaviours with possible synchronisation points. This extends BML by providing means to structure and reuse complex behaviours and to specify complex trigger constraints (in a way similar to Q). Then, characters have the ability to automatically plan navigation paths in the environment, towards labeled regions of the environment or towards other (eventually moving) characters. Trajectories include static obstacle avoidance as well as dynamic obstacle avoidance (other characters). Besides, characters can interact with a basic number of everyday objects (the limitation is a matter of available 3D animations more than a limitation in the language). The engine includes automated blending between animations to avoid the burden of authoring complex static motion graphs. Finally autonomous behaviours can be modelled by assigning a range of behaviours to a character and using a rule-based expert system to implement basic reasoning (e.g. for background characters).

A full demonstration is described in the following section.

1.6 Evaluation

We demonstrate the logic control on an example scenario involving a dissatisfied woman, her millionaire husband, her secret lover, and a local investigator.

1.6.1 Example

Our story describes a woman who is unhappy with her current life and wants to run away with her lover. She makes decisions on whether to confront her husband, which results in some conflicts, and the husband is killed. An investigator arrives and questions the woman. Based on the previous decision, the woman will have options to lie, to confess, to escape, or to protect her lover, each leading to different consequences.

The story has two possible initiating points: one from the chronological start, when the women decides to take action; and one is when the millionaire is already dead and the investigator arrives. The second initiating point will invoke a flashback on the woman when she recalls what had happened prior to the investigator's arrival. In both

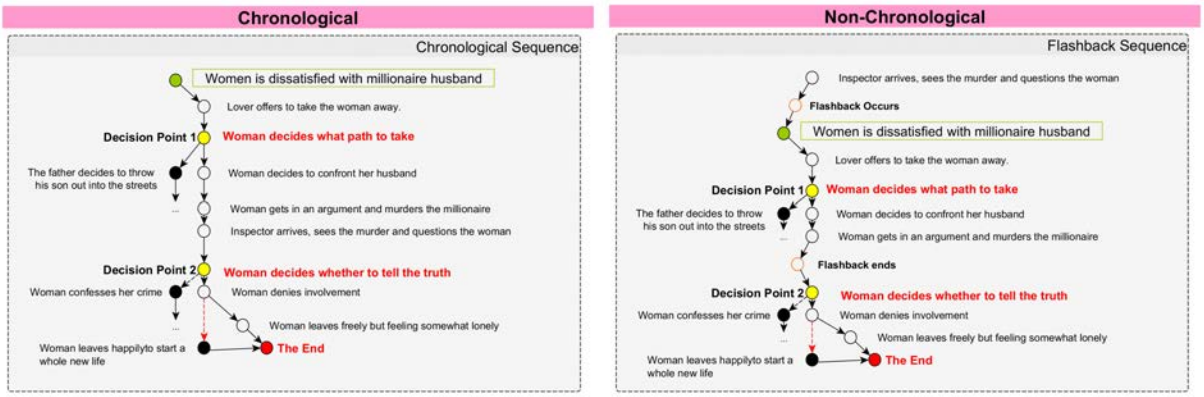


Figure 3.5 – Example generated plotlines. The example story with two plot lines that involve non-chronological events.

cases, the user is offered the same amount of decision, and the logic control ensures that all stories generated are plausible. The two plausible plot lines are shown in Figure 3.5.

The accompanying video shows the flashback version of this story, demonstrating how the logic control achieves story goals while ensuring all preconditions are met over user decisions and the non-linear storyline at runtime.

Link to video: https://vimeo.com/129289640

An addressing of all three aspects–temporality, interactivity, and authorial goals–for complex storyliens has not been displayed before in existing contributions.

1.6.2 Pilot Study

We conducted a qualitative pilot study on 5 users to gain feedback on the logic control of the algorithm.

All five users were university students from different backgrounds. Each were asked to experience the story in four modes: (1) text, (2) non-interactive animated, (3) interactive animated, and (4) interactive animated with story goal (they were given the choice to select a story goal of “happy" or “sad").

In the post survey, users mentioned that while Mode 3 was more enjoyable, but Mode 4 had a higher capacity for creativity as an author. The selection of story goals offered them control over how they, as an author, would like the story to unfold. When asked what they would like to use the system for, all five users noted the control they had on the story in Mode 4, and its potential for creativity as an author. One user particularly noted from the perspective of narrative creation, he would be interested in using the system of Mode 4 by adding his own fragment to create a new storyline with a reinforced goal. Another user stated that Mode 4 would be helpful in creative writing with its branching narrative as compared to traditional linear narratives.

1.7 Summary and Future Work

Our current approach does not rank or evaluate the quality of a story line as planning algorithms do with well-designed heuristics. Every feasible path in the story graph

is considered equally probable. This maintains the simplicity of the algorithm, but limits its tolerance to what it would consider a good plot line. This could be achieved as a second stage, after pruning the inconsistent parts of the graph. We also discuss briefly the evaluations of dynamic stories with temporal rearrangements in the following contribution, which may serve as suitable heuristics for future extensions of this work.

In this work, we have proposed a method for logic control of interactive narratives with regards to authorial goals, user interaction, and temporal rearrangements. The algorithm we designed performs a traversal on the story graph, thereby re-shaping interactive story graphs to fulfil authorial goals and maintain story logic. We demonstrated the output of the logic control in a 3D virtual environment.

As an extension of evaluating story postconditions and understanding of story structure and content, we believe our approach can be further developed to enhance real-time context-aware storytelling techniques. The logic control could provide story level information such as emotion, timing, perspective, and genre to the discourse level such that the virtual camera can make decisions on viewpoint and compute suitable sequences of shots. The mechanism for story filtering and authorial goal design also provides an exciting move towards even more tailored and personalised storytelling experiences for interactive digital storytelling.

2 Evaluating Viewer Perceptions of Temporally Rearranged Storylines

The possibility to play with temporal variations of stories in interactive 3D environments is a great way to allow the author to create stories where they can hide and reveal events at the correct time, increasing the number of possible forms of story. On the other hand, how can we evaluate the effect of such temporal rearrangements in the audience, the viewers of this story? In this scope, the flashback is a complex example of a temporal rearrangement for which the effects should be better understood in the context of interactive storytelling.

The flashback is a well-known storytelling device used to invoke surprise, suspense, or fill in missing details in a story. Film literature provides a deeper and more complex grounding of flashbacks by explaining their role to stimulate the viewer's memory in order to guide and change viewer comprehension. Yet, in adapting flashback mechanisms to AI storytelling systems, existing approaches have not fully modelled the roles of a flashback event on the viewer's comprehension and memory.

To expand the scope of AI generated stories, we propose a formal definition of flashbacks based on the identification of four different impacts on the viewer's beliefs. We then establish a cognitive model that can predict how viewers would perceive a flashback event. We finally design a user-evaluation to demonstrate that our model correctly predicts the effects of different flashbacks. This opens great opportunities for creating compelling and temporally complex interactive narratives grounded on cognitive models.

This work was carried out in collaboration with Professor Michael Young from North Carolina State University in the United States, with discussions with the Liquid Narrative

Group led by him.

2.1 Background

Genette [Gen80] says flashback was a feature of *memory* before it was of narrative; Bordwell [Bor85] says it's there to *remind* us; Turim [Tur89] further characterizes flashbacks with the role of enlightening, haunting, surprising, and *changing our beliefs* towards story events. Having much to do with memory, inserting events as flashbacks throughout the story can guide the viewer to establish causal relations between events that are chronologically distant, and to remember (or forget) them at a specific pace.

Flashbacks have a role of guiding the viewer's comprehension of events. Current AI storytelling systems can generate flashbacks that invoke surprise and suspense [BY08], and provide focalization [HTL14] [Mon11]. These works address flashbacks as an event sequencing problem: if every event appears once, in what order should we place them? Flashbacks however have a much wider range of cognitive effects on the viewer and have not been fully addressed in the storytelling literature. The film theorist Maureen Turim [Tur89] finds that flashbacks have the cognitive memory function of guiding the viewer's perception: establishing new beliefs, changing or reinforcing them to invoke an emotional response, and reminding through repetition. As an inserted event, flashbacks can help the viewer establish causal relations to close and salient events. Yet, how does an AI system select an event that provides the author's intended effect to the audience? How do we know that it does indeed guide the viewer's understanding of the story? These aspects have been insufficiently explored in existing storytelling systems.

Our approach sets out from a cognitive perspective. Like in previous work focusing on suspense in stories [OR14], we use a memory model for calculating saliency of events (and associated beliefs) and their duration in viewer memory. However our approach is targeted towards not the generation of flashbacks, but the exploration of authorial creative choices, identifying a broader and more general range of cognitive effects of inserted events: establishing, reinforcing, and changing viewer beliefs.

We establish a model of calculating memory retention based on research on the temporal theory of memory decay. It is found that working memory has an exponential decay in proportion to the strength of the memory, or salience, which is the state and quality that makes an item stand out from its neighbors [Ebb85] [BM06] [AH11]. The Event Indexing Model [MM09]–an empirically verified model of how viewers interpret story events along the indices of time, space, character, intention, and causality between events in the story–models the saliency of events in film narrative. This model was further developed into a computational EISM model by [CrCWY12], using the above indices to calculate the saliency of an event in a sequence of events. Using this saliency calculation combined with the memory retention model, we evaluate how viewer's select salient events to establish causal links at various points of the story.

Thus, the purpose of this work is to answer two fundamental questions concerning flashback generation: (A) How can we evaluate the range of choices an author can make when inserting an event as a flashback to achieve discourse goals? (B) What is the effect of a flashback on the audience's memory state? We design an algorithm which provides a more general formulation of flashbacks than done in previous work, recommends

flashbacks with a wider range of cognitive functions, and its output is validated through an extensive user evaluation.

2.2 Overview

The main function of this work is to evaluate using a saliency-memory model, how atemporal inserted events such as flashbacks change a viewer's perception of the story. We design an algorithm that takes as input a pre-authored set of events (with a number of events forming default chronological sequence) and a list of viewer beliefs that are to be evaluated in relation to a specified event in the story. The algorithm (a) selects all the events that establish or change this belief; (b) for each selected event, and each possible insertion point, calculates the saliency of the candidate event to other events, thus ranking the relevance of the insertion point; (d) determines which type of memory model the flashback corresponds to, and (e) returns the list of all the evaluated event-insertion point pairs with the score calculated from step (c).

The rest of our contribution is structured as follows: we first present our story representation as a collection of events with annotations, our model of viewer memory, and our definition of beliefs and goals. We then follow with the algorithm for exploring possible flashback constructions and their varying viewer memory states. The output of the algorithm and user evaluations are presented in the results section. Finally, we discuss the limitations and future work.

2.3 Story Representation

We propose the following representations and definitions on which we build our contribution.

2.3.1 Story and Temporal Representation

Two timelines are distinguished: story time and discourse time. Story time represents what actually happened in the world the actors live in, their time, their events. Discourse time represents what the viewer perceives, in the viewer's time, the sequence in which story events are shown in a movie.

We view the story as a collection of pre-authored events represented by propositions (e.g. P="Jerry goes to movie A."). Due to the importance of temporal relations between events in flashbacks, we rely on the format of temporal representation proposed in [EBY15a] [EBY15b], which annotates a start and end timestamp in story time during which the event spans (e.g. "Jerry goes to movie A."; start:18h00 16 November 2015; end:20h30 16 November 2015). The notation P_{st} refers to a proposition P which occurs during the story time interval st. The temporal information helps to establish relations between events in story time by reasoning on Allen interval relations [All84], and to ensure the flashback event being prior in story time.

2.3.2 Beliefs

When an event is shown to a viewer, the event invokes certain beliefs in the viewer about the story world for a certain story time. For example, the event P_{st} =“Jerry goes to movie A.” at time interval st could make viewers believe “Jerry knows movie A.” is $positive$, starting from the end time of st (after Jerry viewed the film). Our story event representation associates to each event a number of beliefs.

Definition 1 (Belief) *A belief $b(P_st)$ is the viewer's perception of a proposition P occurring during st.*

A belief is composed of three properties:

Value of the belief of the viewer is either positive (“believe that”), negative (“believe not that”), or ignorant.

Proposition of the belief, such as "Jerry knows movie A."

Story Time interval of the belief (the story time for which the viewer believes the proposition, see Figure 3.6).

The viewer has a set of beliefs that are changed and upadated as the story progresses. The memory state of the viewer at any given discourse time is therefore defined as:

Definition 2 (Memory state) *The memory state of a viewer v at time t is represented as the set of beliefs $B_{v,t}$ of the viewer v at time t. A belief $b_v(S_{st})$ amongst beliefs B_{v,dt_n} for a viewer v at discourse time dt_n corresponds to the belief that a proposition P_{st} that occurred at story time st is positive or negative at time dt_n in the viewer's memory*

$$(P_{st} \vee \neg P_{st}) \in B_{v,dt_n} \tag{3.1}$$

If the proposition P_{st} is ignorant (as are all beliefs at the beginning of the story), P_{st}, neither positive or negative, is not in B_{v,dt_n}:

$$(P_{st} \vee \neg P_{st}) \notin B_{v,dt_n} \tag{3.2}$$

IInitially, we consider the status of all beliefs in the viewer to be ignorant. As the story unfolds, beliefs can be established (from ignorant to positive or negative), changed (from positive to negative, or conversely), or forgotten (from positive or negative to ignorant). Beliefs are effective or ignorant at specific story times based on their timestamp (e.g. “Jerry knows movie A.” is effective for the rest of the story after “Jerry goes to the movie A.”).

2.3.3 Memory decay and salience

Not only are beliefs associated with story events (propositions) in story time, but they can also be forgotten (i.e. become ignorant again) as discourse time increases. Events and their associated beliefs in the viewer's memory are modelled to calculate this decay.

According to research on the temporal theory of memory decay, it is found that working memory has an exponential decay in proportion to the strength of that memory. The strength is defined as its salience–the state and quality that makes an item stand out from its neighbours [Ebb85] [BM06] [AH11]. The probability that a person will recall an event a certain amount of time after it is introduced is termed *memory retention*. The function for memory retention (R) in relation to time (T) and memory strength-/saliency (S), also referred to as *forgetting curve*, is defined by Ebbinghaus [Ebb85]:

$$R = e^{-\frac{T}{S}} \tag{3.3}$$

To estimate the saliency, we rely on the Event Indexing Model [MM09], an empirically verified model of how viewers interpret story events along the indices of time, space, character, intention, and causality between events in the story. This representation models the saliency of events in film narrative, and was further developed into a computational EISM model by Cardona-rivera et al. [CrCWY12], using the above indices in order to calculate the saliency of an event in a sequence of events. In our story representation, we propose to add EISM links between two events E_i and E_n in the story, with indices of time (t), location (l), character (ch), intention (in), and causality (ca), which we assume are authored or can be generated. We calculate the salience between two events E_i and E_n as the sum of these indices multiplied by evenly distributed weights as proposed in [CrCWY12].

$$\begin{aligned} S(E_i, E_n) = 0.2 * t_{E_iE_n} + 0.2 * l_{E_iE_n} + \\ 0.2 * in_{E_iE_n} + 0.2 * ch_{E_iE_n} + 0.2 * ca_{E_iE_n} \end{aligned} \tag{3.4}$$

For example, the indice $t_{E_iE_n}$ represents the degree to which E_i and E_n are temporally connected (defined between 0=not connected and 1=connected). At a given discourse time of the story, knowing the saliency S between events and the discourse time T, we can then compute the memory retention value R that represents the decaying strength of a belief to a viewer. The model is simple, but has a basis in cognitive theory, and fulfills our purpose to (1) continuously evaluate the viewer's memory state along the discourse, and (2) guide viewer comprehension and interpretation of causality between events.

Up to this point, we have differentiated between story time (time in story world) and discourse time (time of the telling of the story), discussed how our story representation models events, how events can have associated beliefs, and how the saliency of these events and associated beliefs are calculated using the EISM model. The way these events and beliefs are modelled in our system is summarised in Figure 3.6.

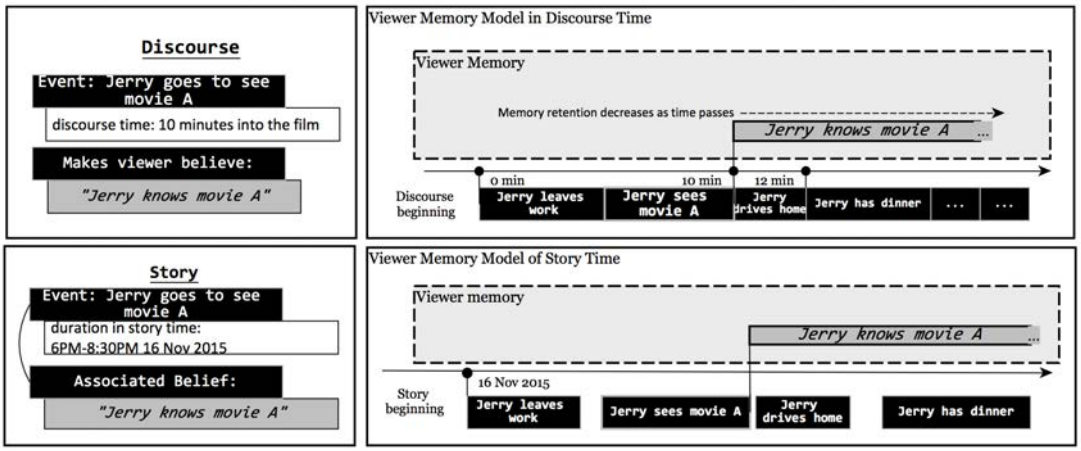


Figure 3.6 – Events (black) have associated beliefs (dark gray). When an event is shown to the viewer, our model of the viewer's memory takes beliefs into account on two timelines: interval in story time, and memory retention in discourse time.

2.4 Model for Evaluating Flashbacks

Given our story representation, we now explore the issue of inserting a story event as a flashback with two questions: (1) How relevant is an inserted event to its neighbouring events? (2) What is the effect of inserting the event on the viewer's beliefs? The first question of relevance to neighbouring events concerns the salience of the event to the viewer. If the event is more salient, it would be easier for viewers to establish cause-effect relation between the inserted event, and other neighbouring or close events.

2.4.1 Insertion points

Events can be inserted as a flashback at insertion points between two events. Possible insertion points can be numerous. We rank the score of the insertion point by calculating the salience between the inserted event and the neighbouring events. The higher the score, the better the quality of the insertion point. The score for an insertion point i_{E_r,E_s} neighboured by events E_r and E_s is calculated using the salience value in Equation (3):

Definition 3 $InsertPointScore(E_1, i_{E_r,E_s}) = 0.5 * (S(E_1, E_r) + S(E_1, E_s))$.

2.4.2 Four States of Memory

The relation between the inserted flashback event with the rest of the story can result in varying interpretations by the viewer. From [Tur89], we simplified the types of flashback events in terms of the ways flashbacks influence the viewer's beliefs. Four types of changes on the viewer's beliefs are identified, namely:

establishing a new belief previously ignorant to the viewer;

reinforcing of an event/belief to refresh a belief that is no longer salient;

changing the value of a salient belief from positive to negative (or vice versa);

salient : the inserted flashback does not result in a change of memory state.

For simplicity, we refer to a flashback by the type change it results in (e.g. establishing flashback, reinforcing flashback...etc.). In the following descriptions, we provide examples of each kind of flashback from scenarios in the film *Lord of the Rings: The Fellowship of the Ring*.

The Salient state means that inserting the event at the specified insertion point adds a belief that already exists with higher than 50% retention. A salient flashback is easily defined as:

Definition 4 (Salient flashback) *If E_1, inserted at dt_m, has a retention of $E_1.R(t = dt_n - dt_m) \geq 0.5$, the belief is still in the viewer's memory, and the flashback has no effect on the viewer.*

Throughout the *Fellowship*, frequent flashbacks of Sauron's eye do not introduce any new beliefs, and are more for dramatic effect than for the need to remind the viewer of its link to the ring.

Establishing means that the viewer establishes a belief that some proposition is true where the viewer was previously ignorant of that proposition, thus increasing both the memory retention and the saliency of the event's associated beliefs at the target discourse time.

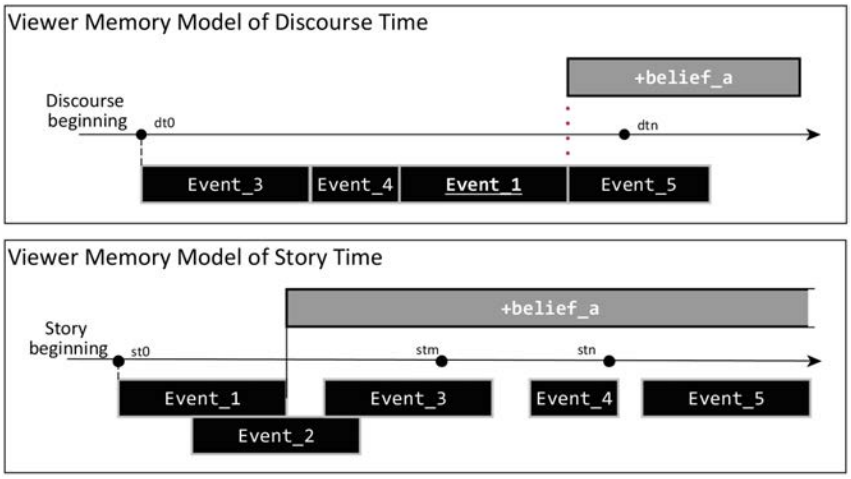


Figure 3.7 – $Establishing$ **flashback**: if $Event\_1$ is inserted as a flashback between $Event\_4$ and $Event\_5$. This establishes a new $belief\_a$ in the viewer's memory at discourse time.

Definition 5 (Establishing flashback) *An $establishing$ flashback is where some proposition P about story time interval st is inserted at discourse time dt_n, and that P_{st} was not in the set of beliefs B_v of the viewer v at any discourse time $k < n$.*

$$(\forall k < n : P_{st} \notin B_{v,dt_k}) \wedge (P_{st} \in B_{v,dt_n}) \tag{3.5}$$

The visualization of the establishing flashback can be found in Figure 3.7. In the *Fellowship*, Elrond narrates Isildur's fall in a flashback, establishing the belief that Elrond distrusts men.

Changing indicates that a belief already held by the viewer towards some proposition is negated. In the memory model, the belief itself changes value (negated), but the memory retention and saliency is unchanged.

Definition 6 (Changing flashback) *A $changing$ flashback is when some proposition P about story time interval st is inserted at discourse time dt_n, and P_{st} replaces the salient belief $\neg P_{st}$ in the set of the viewer beliefs B_v at dt_n.*

$$(\neg P_{st} \in B_{v,dt_{n-1}}) \wedge (P_{st} \in B_{v,dt_n}) \tag{3.6}$$

The visualization of the changing flashback can be seen in Figure 3.8. In the *Fellowship*, when Gandalf narrates the danger of the One Ring, Frodo verbalises the belief that Sauron does not know where the ring is. A flashback of Gollum being tortured by Sauron soon negates this belief.

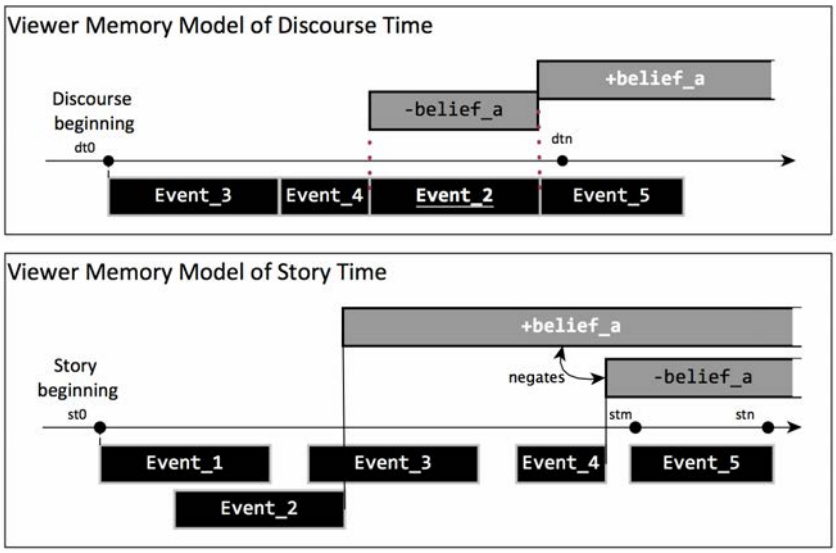


Figure 3.8 – $Changing$ **flashback**: $Event\_2$ is inserted as a flashback between events 4 and 5, changing the value of $belief_a$.

Reinforcing is where the viewer believes some proposition that was shown previously but forgotten, which means that the memory retention of an event is renewed.

Definition 7 (Reinforcing flashback) *A $reinforcing$ flashback is when some proposition P that is about story time interval st is inserted at discourse time dt_n, and that belief*

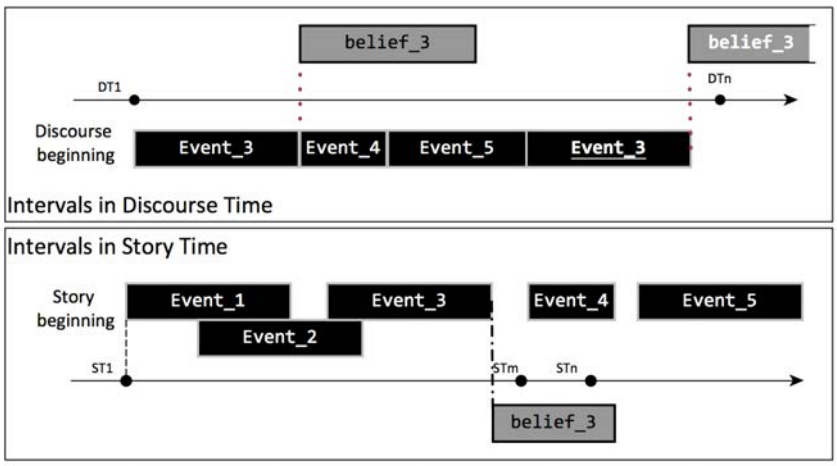


Figure 3.9 – $Reinforcing$ **flashback**:$Event\_3$ is repeated after $Event\_5$ in order to remind viewers of $belief\_3$.

P_{st} is in the set of beliefs B_v of the viewer at some discourse time $k \leq n$ but not in B_v at discourse time dt_n.

$$(P_{st} \notin B_{v,dt_{n-1}}) \wedge (\exists k < n : P_{st} \in B_{v,dt_k}) \wedge (P_{st} \in B_{v,dt_n}) \tag{3.7}$$

A visualization of a reinforcing flashback is similar to the establishing flashback, as shown in Figure 3.9, except the inserted event would repeat a non-salient event (such as $Event\_3$ in Figure 3.7). In the *Fellowship*, Isildur cutting the ring off Sauron's finger appears twice in the film: once in the beginning of the film, and again in the middle of the film just before Elrond provides new information on Isildur's fall.

2.4.3 Algorithm

We design Algorithm 3 from our formal definition of flashbacks, involving three main steps: find events that are relevant to a given belief at a given discourse time (e.g. events related to the belief "Jerry knows movie A" that should be positive at discourse time dt), rank salient insertion points for these related events, and calculate the type of change on the viewer's beliefs.

The algorithm takes as input a story $mathcalS$ (a collection of events), the belief we would like to observe, and a target discourse time at which the belief should occur. First, the algorithm searches for all candidate events that possibly alter the belief by first testing whether the candidate event impacts the belief b, and if the story time of the belief b is during (in terms of Allen time intervals) the story time of the belief invoked by the candidate event. It then evaluates the insertion point score of each candidate event/insertion point pair. The retention R of this event is computed at time $T = dt - E_{dt}$ representing the amount of time that has passed between showing the event E and the target discourse time dt. From Definitions 5∼8 (Algorithm 1. Line 12), we deduce the type of change on the viewer's memory the flashback has (function $flashbackType()$. Finally, we return a ranked list event-insertion point pairs (by insertion point score), and the flashback type.

Algorithm 3 RecommendFlashback (Story **S**, Belief $b(P_{st})$, Discourse Time dt)

for all events E **do**
  if E contains a belief $P'_{st'}$ where $P' == P$ **then**
    if $st\ during\ st'$ **then**
      Candidate Events $C_e = C_e \cup \{E\}$
for all events $E \in C_e$ **do**
  for all insertion points $i \in \mathcal{S}\ after\ E$ **do**
    $i.score = InsertPointScore(E, i)$
    $fType = flashbackType(compute(E.R), i, dt)$
    $Ranking.add(E, i, fType)$
return $Ranking$

2.5 Evaluation

To evaluate the output of our algorithm, we annotated events from the film synopses of Constant Gardener (CG), Lord of the Rings (LotR), and one hand-created story "The Interview". We chose the same scenario in LotR as in [EBY15a] to demonstrate how our algorithm evaluates the effect of the inserted flashback event on the viewer. Temporal information was collected from the original chronology provided by the author, J.R.R. Tolkien, but names were changed to reduce association with the original material.

SCENE #1. Jerry is at home drinking and playing video games.
SCENE #2. Jerry's father calls his son useless.
SCENE #3. Jerry says he will prove himself.
SCENE #4. Jerry runs out of house.
SCENE #5. Jerry stays in a hotel.
SCENE #6. Jerry calls for a hooker*.
SCENE #7. Jerry pays the hooker without having sex.
SCENE #8. Jerry gets ready for a job interview.
SCENE #9. Jerry leaves the hotel.
SCENE #10. FLASHBACK: Jerry's father calls his son useless.
SCENE #11. Jerry bumps into a tree because he is not watching his way.
SCENE #12. Jerry falls on the ground and gets his suit dirty.
SCENE #13. Jerry arrives at the interview.
SCENE #14. FLASHBACK: Jerry says he will prove himself.
SCENE #15. Jerry finishes the interview.
SCENE #16. Jerry leaves the interview.

Figure 3.10 – A computed variation of "The Interview" presented to participants. Here, the viewer's perception at discourse time S15 (Scene #15) is altered due to the belief $P_{S13-15} \in B_{v,dt:15}$ where P="Jerry is confident for his interview." being associated to the inserted flashback.

Figure 3.10 shows a sample variation of "The Interview" computed by our algorithm with viewer beliefs $\neg P_{S13-15} \in B_{v,dt:10}$ and $P_{S13-15} \in B_{v,dt:15}$ where S13-15 means the story time of scenes 13-15, dt:15 means discourse time at Scene 15, P="Jerry is confident for his interview."

For each story, we presented participants with three variations: the one recommended with highest event-insertion point score from Algorithm 1, the film version,

and the linear version. We tested two hypotheses. **H1**: does our algorithm correctly evaluate how changing the order in which events are presented reflects the viewer's interpretation? **H2**: can the cognitive model correctly predict the type of flashback the viewer perceives?

We recruited 41 participants: 22 males and 19 females, ages ranging 19-37. Three variations of each story–recommended, film, and linear–were presented to participants in random order. For our hand-created story, we presented two generated versions with different flashbacks. Participants were given a list of 1 sentence scene descriptions as in Figure 3.10. Since we refer to the flashback technique in terms of film, participants were asked to imagine to be the viewer at a movie. All participants read at least one version of each story, and we ensured no two variations of the same story appeared consecutively. In the end, each variation was evaluated by 24-30 subjects.

For **H1**, we asked participants to identify the scene that they felt was the salient cause of a specific Scene X. In each story variation, multiple Scene Xs were demanded. By asking this question, our purpose is to see if participants would identify the same scenes as salient causes, when they appeared as a flashback, or when they appeared chronologically. Figure 3.11 shows the relation between the memory retention (normalized for each story variation such that the sum of all scenes is 1.0) for the scenes that were identified by participants as salient, and the number of participants that identified the scene as a salient cause. Each dot represents a scene in the story, on the x-axis, its memory retention percentage (from Definition 2), and on the y-axis, the percentage of participants that selected the scene as being the salient cause. In each graph for the stories, the flashback events are coloured in red.

On the whole, our findings are as follows: First, there is a positive correlation between the memory retention and the participant responses, the highest r-square value up to 0.65 in our hand-crafted story, and 0.32 over the whole dataset. For an experiment involving human participants, an R-squared value lower than 50% is expected, due to humans being harder to predict [Fro13]. Thus our data is sufficient to show that the calculation of memory retention percentages of events does reflect whether participants found the events salient or non-salient at different points of the story. Second, we found that the difference between inserting an event as a flashback and presenting it in chronological order changes the viewer's interpretation of the overall plot line. In the top part of Figure 3.11, the green circles we have annotated indicate the same scene–Scene#2 in Figure 3.10–that was chosen by participants as the salient cause of the same target scene in the three variations of our hand-crafted story. It shows that when Scene#2 is inserted as a flashback closer to our target event, a higher percentage of participants identified it as a salient cause. We found the same results with other target events, and for the two other stories, *Lord of the Rings* and *Constant Gardener*. This shows our algorithm can correctly evaluate the change in the viewer's belief when the order of events is changed in the story, and **H1** is validated.

For **H2**, three statements corresponding to the $Establishing$ ((“it tells me something new about the story”), $Reinforcing/Salient$ (“it reminds me of something I may have forgotten”/“it is redundant: it does not tell me anything new”), and $Changing$ (“it changes my view towards the events/characters in the story”) effects on viewer beliefs were presented to the participants. They could select any number of statements that they thought appropriate to describe the flashback. Figure 3.12 shows the distribution

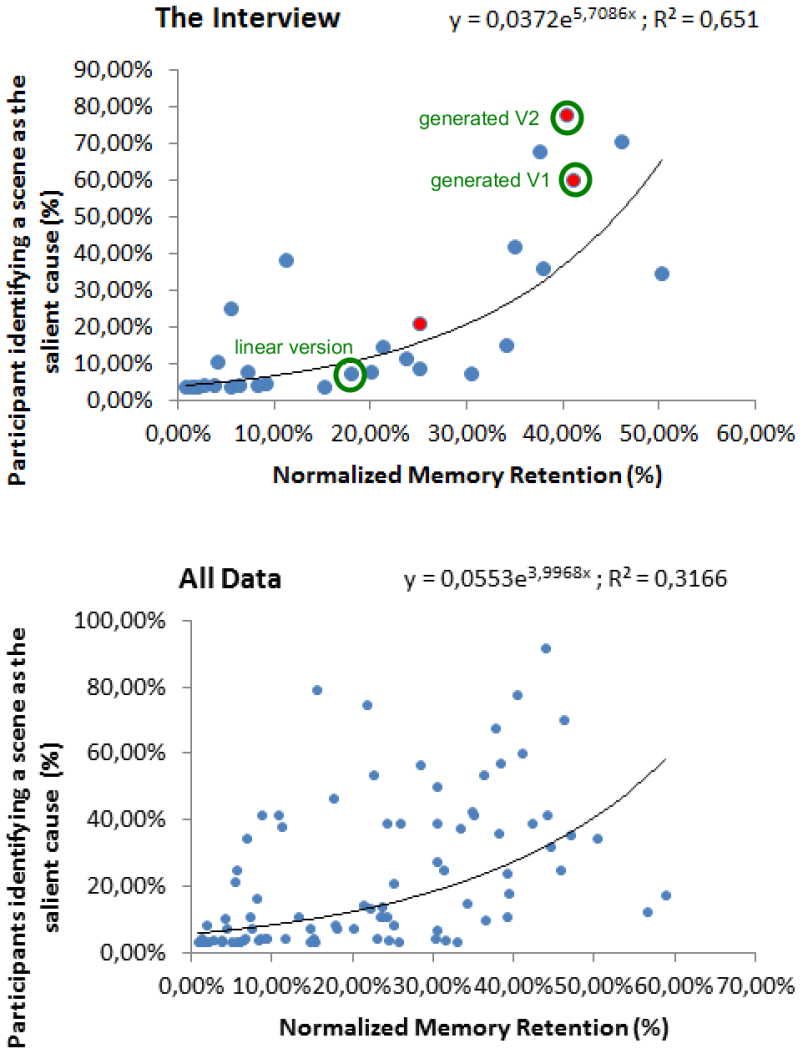


Figure 3.11 – Relation between the normalised memory retention for a scene (normalized for each story variation such that the sum of all scenes is 1.0), and percentage of participants who chose a scene as the salient cause. Each dot represents a scene selected as a salient cause to a target scene by the participant, and the red dots in the top graph represent those that are flashbacks.

of participant responses for each flashback type to the type identified by the algorithm (as indicated by the label). We found that the participant responses strongly correspond to the flashback type identified by our algorithm: Especially, all establishing and reinforcing flashbacks show the majority of participant responses fall in the correct category, and 2 of 3 changing flashbacks were also significant. Thus, **H2** is validated.

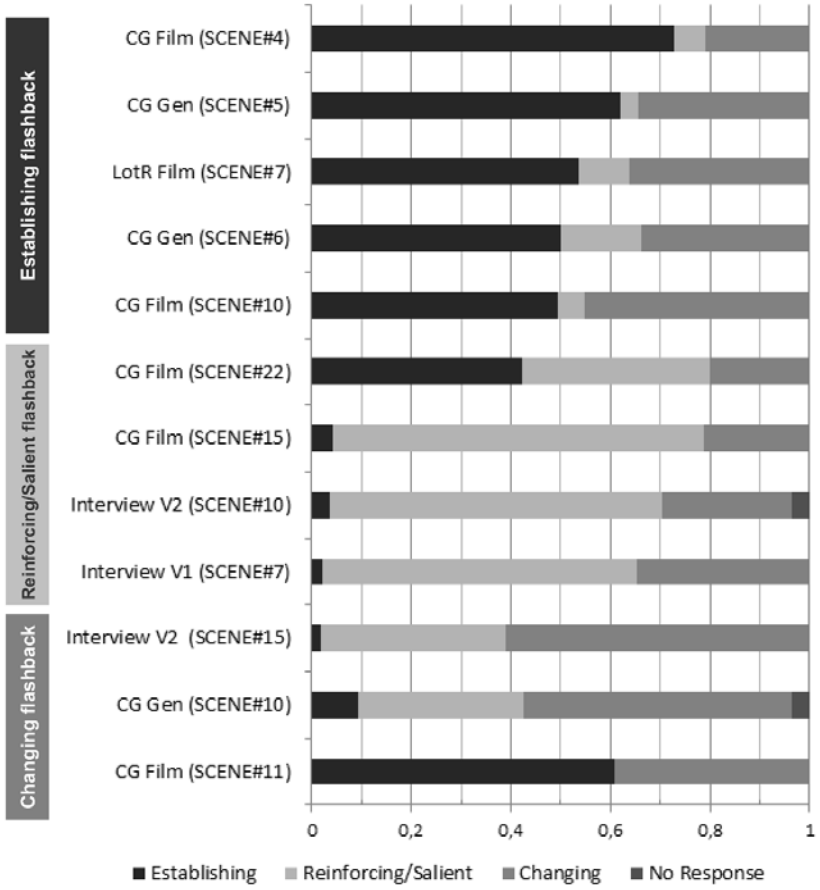


Figure 3.12 – Distribution of participant responses compared against four types of flashbacks calculated by our algorithm. Our evaluation finds distinct participant responses between all types of flashbacks (i.e. types of changes on viewer's beliefs). This shows that our algorithm appropriately characterized the types of inserted flashbacks: establish, reinforce, change, salient (**H2**)

2.6 Summary and Future Work

We have presented a cognitive model to explore the flashback choices an author can make (inserting an event as a flashback at a certain point) to achieve a desired belief, and how these choices affect the viewer's perception through different flashbacks (establish, change, reinforce, or salient). While our approach is computationally simple, we show that our cognitive-based modelling of flashbacks correctly evaluates and guides viewer's interpretations.

There are a number of limitations to address. The model of saliency on was empirically verified, yet memory in storytelling remains not well-understood, and thus, it is still far from predicting exactly how a viewer would interpret flashbacks. One work in this direction is by Kives et al. [KWB15], who conducted a user evaluation the memory retention for event pairs in each of the five indices in the saliency model. Secondly, our evaluation is designed to understand how sequence and proximity of events alone affects interpretation. To limit control factors, our evaluation was in text form which cannot simulate temporal features of flashbacks in film such as duration.

3 Chapter Conclusion

In this chapter we have introduced a logic control mechanism for interactive stories with multiple possible story lines and outcomes, and temporally arranged events. We then proceed to evaluate how atemporal events in stories affect the viewer's perception. These two works are a basis of understanding how temporal qualities of discourse can guide and change the player's interpretation of a story in computer-generated narrative.

AI storytelling systems have been a subject of extensive study in the recent 10 to 20 years due to their applications in the game and entertainment industry, and also to the development of intelligent AI robots and software. This has opened the possibility to more personalised storytelling experiences that can be tailored to the player's preferences. A next step in this work would be to join these two pieces of work–logic control for interactive stories and cognitive model for evaluation viewer perceptions–for interactive and dynamic games. This could create game narratives that can predict, and then guide the viewer's perception to match the author's intention. Another line of work is evaluating the quality of a computer generated story using viewer models of what is a good story, allowing systems to generate personalised stories appealing to each player's viewership model.

The limits of human creativity are boundless, even in the way story events can be rearranged to reflect the author's thought process. We hope this interesting topic will be revisited to refine models of viewership, as well as to adapt to newer forms of games and entertainment.

Patterns: Analysis of Cinematographic Storytelling

4

"Here's Johnny!"

The Shining

Contents

In a horror movie, you can almost always sense something scary is about to happen. How? When the jump scare occurs, it still gets you. Why? Apart from the haunting lighting and eerie sound effects, how the camera is placed and how it moves also plays the role of guiding your emotions, hiding and revealing details at the right moments. This is cinematographic storytelling: using the camera discourse in order to convey meaning and guide emotions of the story.

In film theory, Jakob Lothe [Lot00] describes the mechanisms of discourse as how "people are presented through characterisation, and the transmitted content filtered

through narrative voices and perspective." Discourse, such as that in film, is how the director guides and awes the audience to understand the multiple emotions, feelings, and other hidden messages in each story scenario. These ways of cinematographic storytelling have been gradually developing into rules and semantics of film shooting that filmmakers use to guide the audience's understanding and interpretation of the story. The experience of which shot to use and how to transition between shots for a specific event is not inspiration born out of the director's own brilliance, but knowledge accumulated through the study of film theory, watching and analysing hundreds of films, and intuition cultivated through many years of practice, trial error, and observation on film sets.

Of all of these techniques to gain experience, analysing existing successful films is the most easily accessible, and many observations of the links between camera discourse and story can be drawn directly. Up to this day, film students still painstakingly go through movies shot by shot–sometimes even frame by frame–to identify basic visual properties such as shot boundaries, transition types, shot sizes, angles, etc. because these visual features draw insight into the cinematographic storytelling devices that are used in films to convey emotion and meaning. However, when one wants to generalise a trend over tens or even hundreds of films in a genre, by the same director, or in a specific time period, this task becomes daunting, not only in the time that it takes to annotate individual shots, but even more, on defining, then finding long sequences of style that convey discourse elements (such as temporal and spacial progression) otherwise difficult to observe in a single shot, and finally validating these findings.

This chapter addresses the challenge of creating the query language *Patterns* that can assist film analysts to define patterns of visual style in long sequences of films, and a solver that find all instances of these patterns in a database of annotated film clips. We present the analytical properties of *Patterns*, which we use as a link between film language and constraint based systems for cinematography. The concept behind *Patterns* is what we term *embedded constraint patterns (ECPs)*, which are multiple framing, sequencing, target selection, and transitioning constraints that are applied to one or multiple shots.

After a brief introduction to the background and position of this work, we detail the vocabulary in *Patterns*, with respect to actual film practice, and how ECPs are constructed. For the purpose of collecting cinematographic data from actual films, we introduce our redesign of the *Insight* annotation tool specifically for *Patterns*. The design of ECPs has been shown to have applications in both analytical and generative aspects of content-based film analysis, and virtual cinematography systems respectively. This chapter focuses on the former–the analytical–properties of ECPs. An ECP solver was developed to search annotated film sequences for editing and transitioning styles that fulfilled the ECP.

1 Background

In-depth film analysis in cognitive film studies [Cut15] [Bor85] [Pea09] have shown that arranging visual elements in a shot and editing is a strong choreography between the actors, scene, story, and the camera. As the story is tightly linked to the discourse,

the visual features resulting from good camera placement often provides insight into elements of storytelling such as emotion, character personalities, relationships, and event types.

When searching for examples in actual film data, we are limited by our capacity to observe, record, and generalise cinematography style and practice. We are distracted by the multitude of color, movement, sound on the flashing screen. Yet, to harness the power of computers, there are currently no general purpose languages or tools for automated analysis of cinematography in film. This is due to three issues. First, though there are popular practices and continuity rules, film cinematography has no fixed implementation from director to director. As shown in Figure 4.1, the same technique of intensify– a camera moving closer and closer to an actor–can be implemented in two completely different stories, with different shot sizes, angles, and character orientations, but both reflect intensifying emotions. A language to define these cinematographic sequences must be as flexible and as extensible as possible. Second, formalising cinematographic knowledge requires understanding of actual film practices which, though general principles exist, may again have varying implementations from director to director. Finally, image processing tools are able to detect occurrences of actors across a sequence, but their accuracy can be strongly influenced by the color, lighting, orientation, and occlusion in the scene.

Figure 4.1 – These two sequences in *Lord of the Rings* and *Godfather* both show the intensify ECP (moving gradually closer to the actor).

The concept of designing film idioms for applications to computers was the DCCL language proposed by [CAH+96]. More recently, the Prose Storyboard Language [RBLA13] was designed specifically to address encoding of visual features of a framing. However, these are techniques for camera control problems in virtual environments, to find camera positions and movements that fulfil a number of constraints. They were designed to annotate and direct, but not designed to conduct film analysis. The more relevant field of video and image processing has seen a number of works that analyses low-level on-screen features combined with continuity rules to detect event types and boundaries [WC03] [TZ04], yet neither of these approaches has analysed full sequences of camera style. Moreover, most video processing algorithms are still limited to scene boundary detection, and scene type classification, and much less on the storytelling aspect of cinematography.

This work proposes *Patterns* as a solution to analysing film. *Patterns* has a emphasis

on vocabulary to describe the *relations* between individual shots, and recurring styles over a sequence of shot. The solver for *Patterns* is an Embedded Constraint Pattern (ECP) solver that conducts a pattern matching algorithm over annotated (or extracted) shot data. We use examples and case studies to show the potential of ECPs in film analysis scenarios.

2 Visual Features of Film

From film textbooks, there are a number of visual features to describe the on-screen specifications of actors in relation to the camera: actor position features, movement features, and size and angle features. Our selection of features closely follows the editing chapters of filmmaking textbooks [Zet07]. We first exclude elements of sound, light, and color since they have more to do with scene and actor arrangement than with camera placement. Color, light, and audio have received much attention in literature of film analysis, but are less discussed in film grammar books.

The definition of an ECP sets a number of constraints on the visual features of a sequence of shots. There are 3 types of constraints: (1) framing, (2) shot relations, and (3) sub-sequence constraints. Below, we describe each category in detail.

2.1 Framing Constraints

Framing constraints restrict how actors are arranged on-screen, mainly including four visual features: $size$, $angle$, $region$, and $movement$. Each of these parameters has strong definitions in literature. The constraints are applied to actors. We adopt a very broad definition of actors that incorporates humans, animate creatures, and objects.

2.1.1 Size

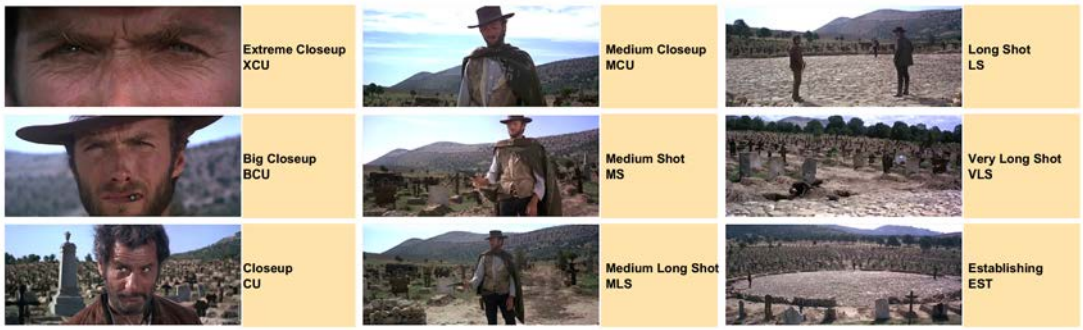


Figure 4.2 – *The Good, the Bad, and the Ugly* uses all 9 scales of shot size within a single scenario, with the establishing shot EST as the longest shot (smallest size) and extreme closeup XCU as the shortest (largest size).

The distance of actors to the camera will affect the size of actors and objects on the screen. The well-known "Hitchcock Rule" stated by the film director Alfred Hitchcock states that an actor's size is proportional to its importance in the current story. Closer

cameras create bigger actors, increasing their importance; conversely, longer camera distance makes the actor smaller and the less important. We categorise 9 sizes, as defined by [TB09] with the upper half body filling the screen as the median Medium Shot. A diagram of all shot sizes on a human actor is shown in Figure 4.2.

2.1.2 Angle

Corresponding to the three axes in 3D, the camera can be rotated horizontally, vertically, and rolling. Examples of each type of angle can be found in Figure 4.3.

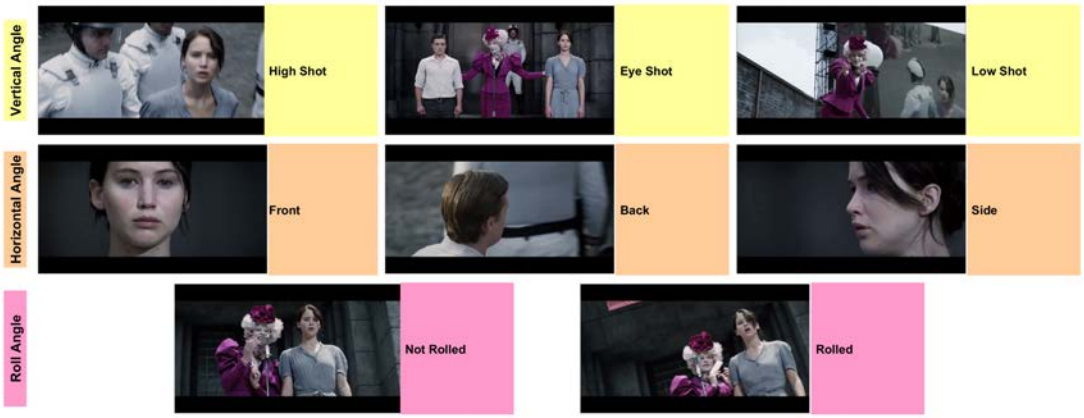


Figure 4.3 – Angles of characters on the screen can be categorised into vertical, horizontal, and roll angle. (Screenshots from *The Hunger Games*.)

Vertical angle assumes the heights of the camera in relation to the target it is filming. Angle can imply the personality of actors [Zet07], with low angle shots (looking up at the actor) expressing courage, confidence, or importance, and higher angle shots (looking down on the actor) showing weakness.

Horizontal angle strongly implies the audience's involvement in the story, including strong involvement (front view), observer (side-view), vulnerability (back view).

Roll angle is usually not used in the film, but it can be adopted for the purpose of showing disorientation of the actor, or distortion of the environment.

2.1.3 Regions

Framing region refers to how actors are arranged in the screen space. Where actors appear on screen has much to do with aesthetics, but also the inner state of the actors. For example, appearing in lower part of the frame or long distance from the centre could indicate isolation, while higher positions in the frame can indicate dominance.

Framing region is most complex to describe since actors often move in and out of the frame. *Patterns* provides a simple set of vocabulary for roughly constraining the regions either by a $4\text{-}split$ or $9\text{-}split$ regions, or simply $top/bottom$ and $left/right$.

In our annotation, we focus on the most important aspects of actors. For inanimate actors, it would be the body of the object. For animate actors, it would mean the the head and eye positions, and possibly limb positions in scenes involving actions and interactions.

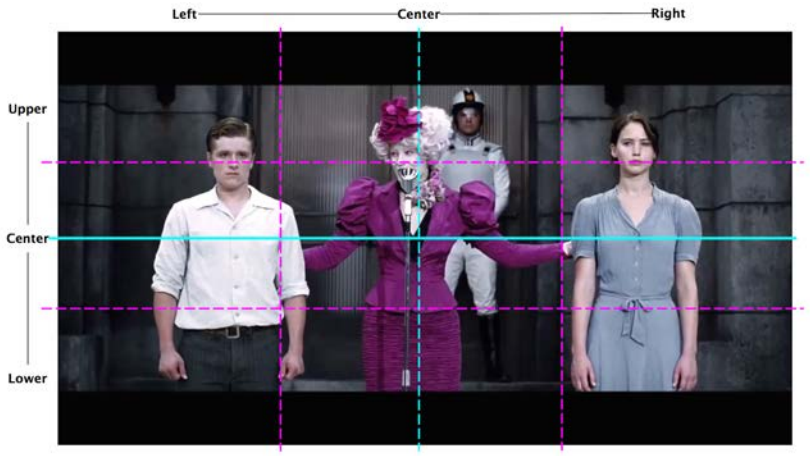


Figure 4.4 – There are many ways to describe on-screen positions of actors. In film literature, the most well-known one is the 9-split standard (purple dotted lines; also referred to as the rule of thirds), where the screen is split into three regions horizontally and vertically. *Patterns* provides vocabulary for $9\text{-}split$, $4\text{-}split$ (blue lines in the figure; 2 regions both horizontally and vertically), and also $left/right$ (dotted blue line) and $upper/lower$ (solid blue line) regions.

2.1.4 Target number

The number of targets–actors or props that are significant to the current story event–on the screen also have important implications in terms of storytelling. If only one actor is visible on screen, we could assume that that actor plays an important role in the shot, or that the director is drawing attention to the visual or emotional details of the actor. If a few actors are on the screen, the director may be drawing attention to how these actors are interacting. If large crowds of actors appear, the director may be establishing the story background, or protraying a general sentiment among the crowd.

The number of targets is automatically determined by the number of actors annotated in each framing.

In our annotation,

2.1.5 Camera Movement

Finally, movement is a straightforward metaphor of guidance to the viewer. The major types of movement involve $still$ camera (no movement, allowing the viewer to observe other movements on screen), pan (horizontal rotating to explore a scene), $zoom$ (in and out, either to focus or reveal), $track/dolly$ (to follow), $tilt$(vertical angle tilting to show height), $pedestal$ (vertical movement), and $rolling$ (to create a feeling of disorientation or creepiness).

2.2 Shot Relation

Constraints can be placed not only on single framing specifications, but also between shots. There are two properties considering relationship between two shots. The more

observable one is transitions, referring to the techniques that links two shots: a cut, fade, screen wipe, etc.; however, this is not our focus here.

The other is the relation between two shots in terms of on-screen properties: distance, angle, and framing regions. For example, we may want to search for a sequence of shots that move gradually closer to actors; or a sequence of shots in which the actors always appear in the same region in the frame. Relation constraints provides ECPs with the ability to detect full sequences following certain constraints, giving a strong advantage over other computational cinematography languages.

2.2.1 Size Relations

The distance of the camera to actors in consecutive framings can either be closer, further, or remain the same. Changing the distance from one shot to another can thus show the difference in importance of actors in the scene, as well as to intensify or relax the atmosphere by moving closer and further to targets respectively. Maintaining the same distance can imply that the overall emotion is unchanging, or that actors are equally important from shot to shot. *Patterns* includes the size relation constraints of $same\text{-}size$, $closer$, and $further$ between two consecutive shots.

An example of changing distance relations is the one we have been referring to frequently–intensify in Figure 4.1. In the example of intensify, the camera becomes $closer$ to the actors from one shot to another.

2.2.2 Angle Relations

Similar to size, angles can be either higher, lower, or the same angle. Since angles carry the meaning of confidence or importance of actors, change of angles between shots can imply the relative strength of different actors, or change of emotional state for the same actor. Yet unlike size, shot angles usually do not change so frequently, mainly as not to confuse the viewer. In *Patterns*, we provide the vocabulary mainly for changes in vertical angle, namely $higher$, $lower$, and $same\text{-}angle$ to describe angle relations between two consecutive shots.

In Figure 4.3, the high shot and low shot are consecutive frames in the original movie, which clearly depict the disparity in status between the two actors.

2.2.3 Framing Region Relations

When actors appear in the same regions on the screen across shots, it often means an agreement, compassion, or mutual recognition between the actors. It can also be a director's way of expressing the equality between the actors. If actors are distributed on different sides of the horizontal axis (i.e. left and right), it often carries the meaning of opposition. When there is a upper-lower disparity, appearing higher in the shot often symbolises power. Since framing can be complex, we provide the basic values $same$ and its negation $!same$ that can be assigned to the $region$ (the same $9\text{-}split$ or $4\text{-}split$ region), the $horizontalRegion$, and the $verticalRegion$.

Figures 4.10 and 4.11 show of how framing regions could be an interpretation of relations between actors. Figure 4.10 places actors in the same region to show agreement

and compassion, while Figure 4.11 places two actors in opposite horizontal regions, reflecting a disagreement or enmity.

2.2.4 Continuity

An important category of shot sequencing rules comes from continuity editing practices. The purpose of continuity rules is primarily to not confuse the viewer on the spatial relation between actors and the actions they carry out. There are three categories of continuity constraints: the 180 degree rule, gaze matching, and action matching.

The 180 degree rule states that when the camera is filming two actors in a scene, the camera should remain on the same side of the index line established between the two actors. This is in order to maintain the horizontal relative position of the two actors on the screen, and not confuse the viewer. Crossing the line, and thus violating the 180 degree rule, results in the two actor positions on the screen to be switched, which could confuse the viewer, or when done intentionally, create a feeling of disorientation and spacial discontinuity.

Gaze and action matching are both variations of the 180 degree rule, except the line is not established between two actors, but by one actor, and the direction of the gaze or the movement.

Gaze matching refers to a technique in which when we begin with a shot of an actor looking at something or someone off screen, the next shot should portray what the actor was looking at, thus matching the gaze direction of the actor in the previous shot, and creating a cause effect relation between the two shots, as well as spacial continuity along a line of interest.

Action matching is very similar to gaze matching in that instead of matching the gaze of an actor, the rule matches the direction in which an actor is moving. This ensures that if the actor is moving, for example, off the screen from the right, the actor will reappear from the right of the the screen in the next shot, thus continuing the movement from the previous shot.

In our vocabulary, we allow the user to specify continuity constraints such as $180\text{-}rule$ which observes the 180 degree rule, $match\text{-}gaze$ which matches the line of gaze between 2 shots, and $match\text{-}action$ which matches the direction of action. The user can also ask the camera to deliberately violate the continuity rules by negating the above constraints.

3 ECP Definition

An ECP can contain multiple framing and relation constraints. Further constraints such as length and embedded sub-sequences can also be added to ECPs.

We will use the incremental construction of the intensify sequence to illustrate how an ECP is structured. We show how the combination of different vocabulary in *Patterns* can provide multiple definitions of the intensify ECP. Examples can be seen in Figure 4.1.

3.1 Sub-Sequences

Embedded sub-sequences are continuous sequences of shots that follow the constraints of some other ECP, such that the sequence of shots can be grouped together in the parent ECP. Individual shots in the sub-sequence are not evaluated by the relation constraints set by the parent ECP. Suppose we defined a very simplified definition of intensify as:

```
intensify{
    relation
        constraint: closer
    sub-sequence
        constraint: 'shot'
}
```

meaning that all shots must have a shot distance relatively closer than the previous shot, and intensify should be solely comprised of single shots. Then intensify would be able to match shots 3-4 in Figure 4.8, but it would not consider 3-7 as intensify since shots 5 and 6 have the same shot distance, not fulfilling the $closer$ constraint. Yet, as a film analyst, keyframes 3-7 may still be considered an intensify, since the shot size gradually increases over the whole sequence. To overcome the limitation, one solution is to allow same-sized shot sequences embedded in the larger intensify sequence. That is, we allow sequences of 1 or more same-sized shots that ignore the relation constraint of $closer$.

```
intensify{
    relation
        constraint: closer
    sub-sequence
        constraint: ECP{
            relation
                constraint: same-size
        }
}
```

Only the first and last shot in the sub-sequence are restricted by the relation constraint. In Figure 4.8, Shot 5 is constrained by the "closer" constraint with Frame 4, and Frame 6 with Frame 7, but Frame 5 and 6 are grouped together, and thus ignore the closer constraint.

3.2 Ranges

If the user would like a constraint to only be evaluated once, such as at the first or last shot of the sequence, or interweave shot sizes where shots 1, 3, and 5 are CU, and 2, 4, and 6 are LS, it can be achieved through setting ranges for constraints. A range parameter that can either be a continuous range expressed as $[x\text{-}y]$, a discrete list $< x, y, z... >$, or it can be one of the keywords of $initial$ (the first container in the

sequence, equal to $< 1 >$ in the discrete list representation), all (all containers in the sequence), $none$ (a strong negating constraint on all containers in the sequence), or end (the last container of the sequence). In the intensify ECP, we can add range restrictions to each constraint. By default, range is set to all.

```
intensify{
    framing
        constraint: size>=MCU
        range: initial
    relation
        constraint: closer
        range: all
    sub-sequence
        constraint: ECP{
            relation
                constraint: same-size
                range: all
        }
        range: all
}
```

3.2.1 Length

We can also restrict the length of an ECP, which indirectly affects the number of shots an ECP can match. The length parameter is targeted towards the number of sub-sequences, and not the number of shots. Here we add a length constraint to intensify:

```
intensify{
    length
        constraint: length>=3
    relation
        constraint: closer
        range: all
    sub-sequence
        constraint: ECP{
            relation
                constraint: same-size
                range: all
        }
        range: all
}
```

In Figure 4.8 even though $4 \rightarrow 5$ fulfils the relation constraint $closer$, and both $[4]$ and $[5\text{-}6]$ form a sub-sequence of same-distance shots ($[4]$ has only one frame, and thus is a same size, too), there are only 2 sub-sequences, and thus does not fulfil the length requirement. The reason for setting a length requirement on sub-sequences instead of number of shots is central to our goal of using *Patterns* to define ECPs that capture

meaningful changes over sequences. Since the evaluation of relational constraints allows observations over long sequences, but only between sub-sequences, the number of sub-sequences on which a relation constraint is evaluated is much more meaningful than the actual number of shots in the ECP. By default, this constraint is set to $length \geq 1$.

4 Insight Annotation Tool

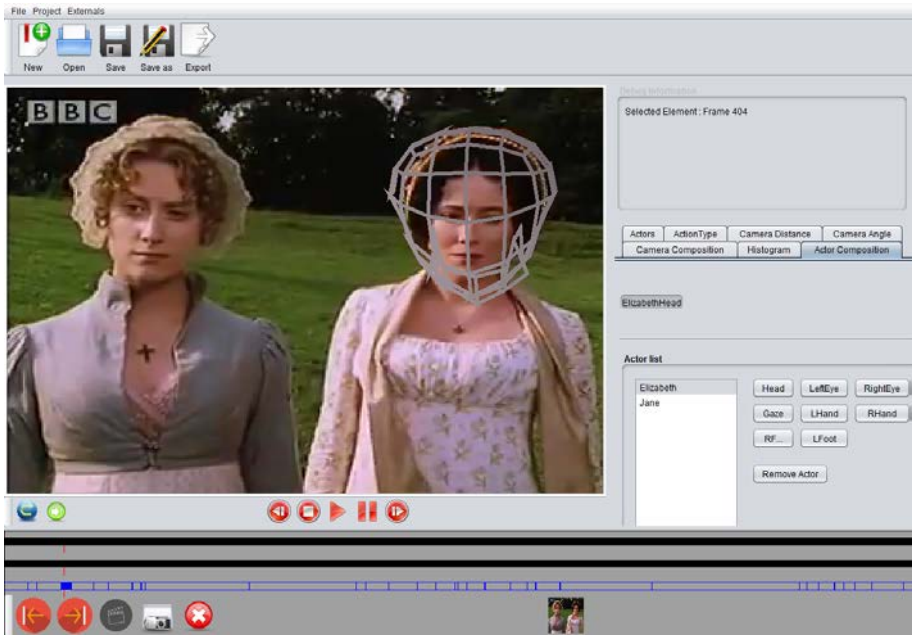


Figure 4.5 – A screen capture of the Insight annotation tool showing the on-screen head tool to estimate actor head positions, orientations, and sizes. Screenshot from *Pride and Prejudice*

The *Insight* annotation tool developed by [MWS+15] was edited to allow the annotations of the visual features used in the $Patterns$ language. Each shot specification in the database provides information that links visual features in cinematography (such as character positions, shot angle, size, distance...etc.). Insight allows easy capturing of such features through on-screen interaction features, as can be shown from a screenshot (Figure 4.5).

There are two levels of annotation in Insight: the frame and the shot. The frame refers to a single still frame in the film clip, without any movement of the camera nor actors. Actor head, eye positions, sizes, and angles are annotated at this level. A shot is the collection of frames between two transitions (which can be a cut, fade, or other visual transition effect), and contains the annotation for camera movement over the frames in the shot. Only a number of representative frames are annotated in each shot, which we call keyframes.

Insight then outputs the annotated clips in the form of an XML file. A sample XML entry for a frame in the output file that corresponds to the frame in Figure 4.6 would be annotated as follows:

```
<frame id="2" refVideo="1" timeId="33">
    <actor refId="Danny">
        <onScreenPart part="Head">
            <position x="0.52205896" y="0.27715048"/>
            <orientation azimuth="182.00002" elevation="4.0"/>
            <scale relHeight="0.73791665"/>
        </onScreenPart>
        <onScreenPart part="RightEye">
            <position x="0.42034313" y="0.32437277"/>
        </onScreenPart>
        <onScreenPart part="LeftEye">
            <position x="0.6213235" y="0.38530466"/>
        </onScreenPart>
    </actor>
    <Properties>
        <property name="distance" type="string" value="BCU"/>
        <property name="angle" type="string" value="Eye"/>
    </Properties>
</frame>
```

Figure 4.6 – An example framing extracted from the film database. Screenshot from *American History X*

The shot entry includes the frames in the shot, the camera movement, as well as the interval and duration of the shot (in terms of number of frames between the start and end frame). The frame from the example above is one of the keyframes of the shot, with frame $id = 2$. The XML format for a shot entry is:

```
<shot endFrame="362" id="2" startFrame="10">
    <frame id="2"/>
    <frame id="300"/>
    <Properties>
        <property name="movement" type="string" value="Static"/>
    </Properties>
</shot>
```

This XML output is then the input of *Patterns* in order to detect occurrences of ECPs in the film clip. The solver for ECPs is described in the next section.

5 Solver

With a set of *Patterns* visual features, a way to define a number of constraints over a sequence using ECPs, and a tool for annotating film clips, the next question is how to use the ECPs we defined for analysing the annotated data.

As defined before, each ECP has (i) a set of framing constraints, e.g. framing=MCU, (ii) a set of relation constraints, e.g. relation=closer, and (iii) a set of subsequence constraints, e.g. subsequence='shot', and represents a query that a user can perform over an annotated film clip.

The solving process is an extension of the Knuth-Morris-Pratt algorithm [KMP77], expressed as a search over a sequence $\mathbf{S}$ of annotated frames. Since in our annotated data, multiple keyframes are annotated in each shot, the first keyframe is selected to rerpesent the shot in $\mathbf{S}$. The search iterates through $\mathbf{S}$ and tries at each iteration to match the given ECP starting from frame i (where $1 < i < Card(S)$). The solver returns a set $\mathbf{R} = \{\mathbf{r}_1, .., \mathbf{r}_n\}$ of subsequences such that $\mathbf{r}_n = [f_I, f_F]$ where f_I and f_F represent respectively the starting and ending frames of the sequence that match the ECP.

For the sake of performance (ie avoiding re-evaluation of the satisfaction of framing, relation and sequence constraints), the search is ran in two stages. The first stage builds a cache of valid solutions as three sets of frame sequences $\mathbf{FC}$, $\mathbf{RC}$ and $\mathbf{SC}$. The first represents the sets of frames that satisfy each of the framing constraint mentionned in the ECP. The second represents the sets of successive frames $[f_i, f_{i+1}]$ that satisfy the relation constraints, and the last represents the set of frame sequences $\mathbf{SC} = \{\mathbf{s}_1, .., \mathbf{s}_m\}$ where $\mathbf{s}_i = [f_I, f_F]$ and where f_I and f_F represent respectively the starting and ending frames that satisfy the specified subsequence.

Then, in a second stage, the process iterates over all frames f_i of the sequence $\mathbf{S}$ (see 5). At each iteration a double recursive depth search is performed from the current frame with a simple idea: the next frame is retrieved from the sequence S, and if valid, is either considered as part of a subsequence (see line 4) or part of the sequence (see line 6).

5.1 Verifying constraints

From a practical view, we cast the problem of verifying frame, relation and subsequence constraints as a string search problem. Annotated film data can be viewed as text articles with multimodal properties such as size, position, angle...etc. Therefore, the problem of finding framings that fulfil certain constraints becomes straightforward: on one side, extract the needed data from the film annotations to form the source text; on the other hand, construct the query from the constraints, that can easily find matches in strings. For this purpose, we choose a regular expression interpreter for the vocabulary in *Patterns*. Each word of our vocabulary in *Patterns* can be formalised using regular expressions.

For example, framing constraints are expressed as an alphabet code followed by a number which expresses either different sizes, angles, or regions based on what alphabet code is given. For example, 'S4' would match a medium closeup (MCU) on the size,

while 'A4' would match an eye-angle shot on the angle. Figure 4.7 shows how the two constraints would be matched to a sequence of keyframes.

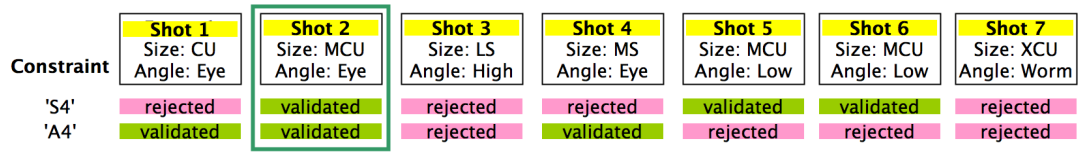


Figure 4.7 – Constraints on single framings can be matched individually and validated or rejected directly. In this example, Frame 2 fulfils both constraints. We build a set that contains, for each constraint, the set of valid frames.

Shot relations are much more difficult to express due to the limitation that regular expressions cannot compare mathematical values. However, since the parameters for features in *Patterns* are discrete, it is easy to exhaust all possibilities. Relations are evaluated between each keyframe and its preceding keyframe, which means the first keyframe would always validate.

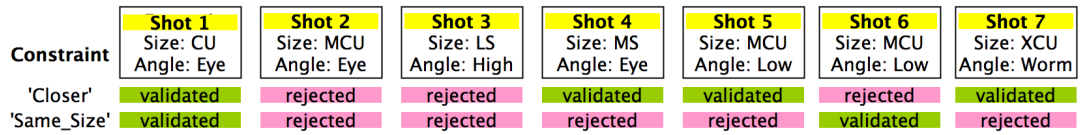


Figure 4.8 – In this example, both closer and same-size sets constraints on the framing size relation. Naturally, since their definitions contradict each other, there is no overlap.

Finally, embedded ECPs are treated just like any other groups of constraints. However the computation requires to run a specific search algorithm in order to return all sequences of keyframes that can be grouped into a subsequence. We describe this search algorithm in detail in the next section (5.2).

5.2 Search Algorithm

The search algorithm (ECPRecurse) relies on functions isValidFrame() to evaluate whether the frame f_i is within the set of valid frames $\mathbf{FC}$ (similar for isValidRelation() and isValidSubsequence()). The isValidSequence() simply checks that the given length of the sequence is valid (since all frames, relations and subsequences are valid).

Algorithm 4 ECPSearch (ECP p, Sequence S)

1: ResultSet $\mathbf{R} = \emptyset$
2: **while** S not empty **do**
3: $f_i = first(\mathbf{S})$
4: ECPRecursive(p, $\emptyset, \mathbf{S}, i, \mathbf{R}$)
5: $\mathbf{S} = \mathbf{S} \setminus f_i$
return R

Algorithm 5 ECPRecurse (ECP p, CurrentSequence $\mathbf{C}$, Sequence $\mathbf{S}$, Index s, ResultSet $\mathbf{R}$)

```
1: if S not empty then
2:     f_i = first(S)
3:     if isValidFrame(f_i) AND isValidRelation(f_i) then
4:         if isValidSubsequence(p, C ∪ {f_i}) then
5:             ECPRecurse(p, C ∪ {f_i}, S \ {f_i}, s, R )
6:         if isValidSequence(p, C ∪ {f_i}) then
7:             ECPRecurse(p, {f_i}, S \ {f_i} s, R ∪ {[s, i]} )
```

6 Potential for Film Analysis

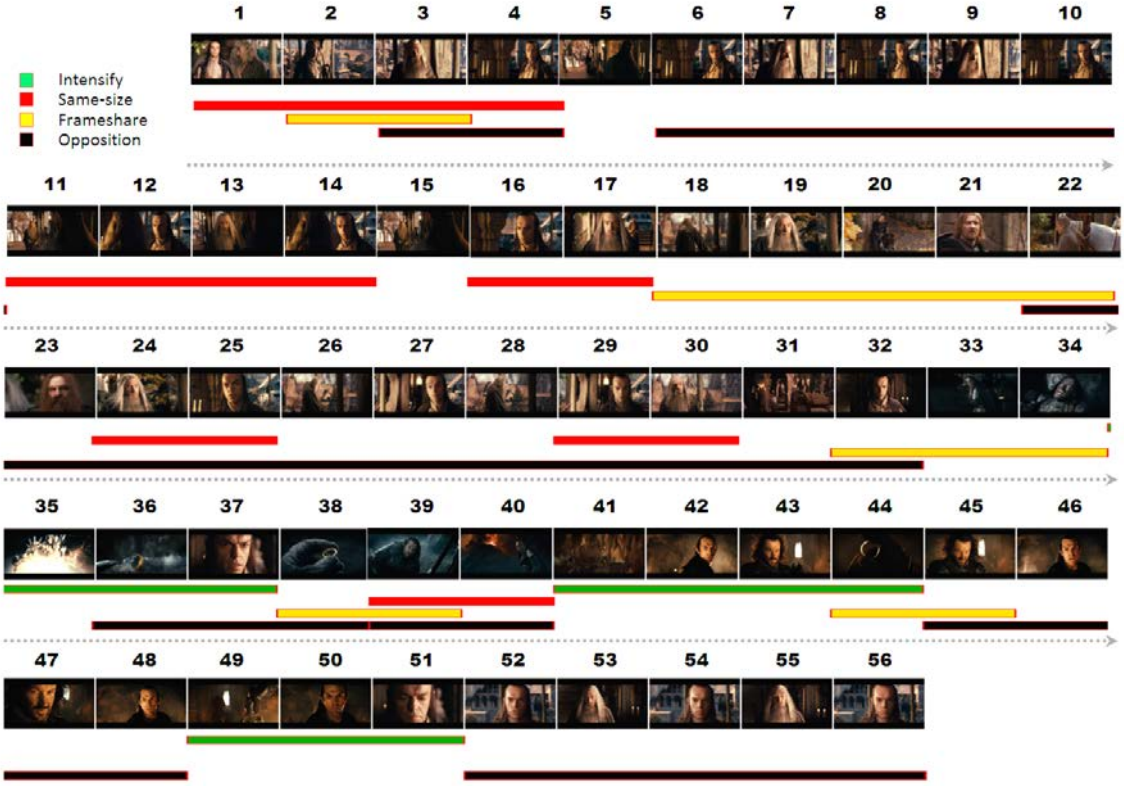


Figure 4.9 – Example ECPs over *Lord of the Rings* sequence. Here we present the result of a search for 4 ECPs: intensify, same-size, frameshare, and opposition.

The main purpose of developing this solver was for film analysis. We know that on-screen properties such as size, angle, and position are visual metaphors to elements of story and emotion, but what of film sequences? And how do we analyse the output of the solver in relation to cinematographic storytelling?

Here we try to answer the question of mapping film techniques to story by first providing an extensive case study on a film clip annotated from *The Lord of the Rings: Fellowship of the Ring* on four ECPs based only on on-screen size and region. On our

way to explore the possibilities of ECPs in film analysis, we annotated more than 20 film clips containing over 1000 shots from popular and classic films. We share some of the results in the second part of this section.

6.1 Extensive Case Study: *The Lord of the Rings*

Our solver was used to analyse user-defined ECPs using the video clip of *The Lord of the Rings: Fellowship of the Ring* as an example[1]

The basic storyline follows the elvish lord Elrond who disagrees with the wizard Gandalf, believing that men are untrustworthy. A flashback occurs in the middle of their dialogue, showing Elrond trying and failing to convince the man Isildur to destroy the one ring, an evil artefact.

In this example, we analyse two sets of ECPs concerning size and framing regions respectively, each set containing two ECPs. The two ECPs for size are $intensify$ and $same\text{-}size\text{-}chain$. The two ECPs for framing regions are frameshare and opposition. Figure 4.9 presents all instances of found ECPs throughout the 56 shots.

6.1.1 Study of Shot-Size Evolution

The size of a shot implies the importance of the actors as well as the emotional density of the scene. Using the *Patterns*, we define two ECPs: $intensify$ and $same\text{-}size\text{-}chain$. The definition of intensify is the same as the final definition in Section 3.2.1.

Our definition of $same\text{-}size\text{-}chain$ relies on the relation parameter $same\text{-}size\text{-}chain$ defined in *Patterns*. We add a restriction of $length >= 2$ to ensure chains of same-sized shots, and restrict the length of each sub-sequence to only contain one shot.

```
same-size-chain{
    length
        constraint: length>=2
    relation
        constraint: same-size
        range: all
    sub-sequence
        constraint: 'shot'
        range: all
}
```

As we can observe from Figure 4.9, the sequence begins with a number of same-size shots of various sizes. As mainly a dialogue scene there is much less action taking place. However, when the flashback occurs, we can clearly feel the increase in the density of emotion from the frequent changes of shot sizes. Moreover, three intensify sequences are detected within the flashback sequence between shots 32 and 51.

Figure 4.10 – A frameshare sequence extracted by the query.

6.1.2 Study of Framing Region Evolutions

In the film clip, can observe two types of actor interaction scenarios: actors opposing each other (e.g. Elrond against the ring), and actors are in coordination with each other (e.g. Isildur protecting the ring). To imply actor relationships, cinematographers often use $frameshare$–a technique where a number of actors appear separately in two or more consecutive shots ($length >= 2$), but the target in both shots must appear in the same horizontal region of the frame ($horizontalRegion == same$) such as in Figure 4.10–to show empathy. Each frame has the restriction of having only one prominent target ($targetNum == 1$). We use *Patterns* to define $frameshare$ as:

```
frameshare{
    length
        constraint: length>=2
    relation
        constraint: horizontalRegion==same
        range: all
    framing
     constraint: targetNum==1
    sub-sequence
        constraint: 'shot'
        range: all
}
```

Figure 4.11 – An opposition sequence extracted by the query.

We define a second ECP called $opposition$ where a number of actors appear separately in different horizontal regions of the frame such as in Figure 4.11. This technique often used to show distance or conflict between actors. Using *Patterns*, $opposition$ is defined as:

```
opposition{
    length
        constraint: length>=2
```

[1]The clip can be found at: https://www.youtube.com/watch?v=O7X1BCCH9a8

relation
constraint: horizontalRegion==!same
range: all
framing
constraint: targetNum==1
sub-sequence
constraint: 'shot'
range: all
}

By defining the two ECPs and running our solver on the annotated film clip, we easily extract uses of *frameshare* and *opposition* sequences throughout the clip. Comparing two specific examples in Figures 4.10 and 4.11, the result strongly corresponds with plot events: Isuldur in coordination with the evil ring, and Elrond in disagreement with Isuldur. In the rest of the sequence in Figure 4.9, we also see that that for the three actors, Elrond, Isuldur, and the Ring, the director indeed uses the techniques opposition and frameshare to express character relationships. In other parts of the film clip, the director also uses opposition during the heated discussion between Gandalf and Elrond, but frameshare when portraying the arrival of surrounding allies to support their cause.

6.2 Selected Findings

Here we provide a brief summary of what we found in the full dataset of annotated film clips.

6.2.1 Dataset

Our dataset is comprises of 22 clips from 18 films. The period in which these films were released spans from1966 (*The Good, the Bad, and the Ugly*) to 2012 (*The Hunger Games*). Table 4.1 gives an overview of the films that we extracted our clips from, as well as the duration of the clip in minutes and seconds, number of shots, average shot length (in seconds), and a rough scene type (dialogue, action, or mixed). The films are ordered by the year they were released.

The clips were selected primarily for either being a classical scene from the film, or for its temporal properties such as containing flashbacks. The second criterion is mainly in connection to understanding temporal properties of films in the previous chapter.

6.2.2 Pattern Occurrences

Table 4.2 gives an overview of the percentage of shots each ECP occurs in the clip, followed by the average length (in shots) of the ECP. Each cell is formatted with the two values as "Percentage%(average length)". A dash (-) indicates that no occurrence of the ECP was found in the clip.

From the information on the usage of ECPs in film clips, we can make quantitative observations such as on the evolution of average ECP length over these film clips such as in Figure 4.12 or the evolution of the usage of these techniques over a long period

Table 4.1 – Overview of annotated datasets. The average shot duration is represented in seconds.

Title	Year	Duration	No. Shots	Avg. Shot Duration	Scene Type
The Good, the Bad, and the Ugly	1955	6m16s	98	3.83	Action
The Godfather	1972	1m19s	30	2.63	Action
The Shining	1980	3m09s	5	37.8	Dialogue
American History X	1988	3m57s	57	4.16	Dialogue
Forrest Gump	1994	2m20s	17	7.72	Mixed
Pulp Fiction	1994	2m07s	28	4.34	Dialogue
Shawshank Redemption (clip 1)	1994	3m32	48	4.41	Action
Shawshank Redemption (clip 2)	1994	3m09s	26	7.26	Action
Gattaca (clip 1)	1997	1m52s	19	5.89	Dialogue
Gattaca (clip 2)	1997	1m59s	19	6.21	Dialogue
Armageddon	1998	3m10s	100	1.88	Action
Lord of the Rings	2001	3m33s	56	3.67	Mixed
Infernal Affairs	2002	3m05s	47	3.91	Dialogue
Big Fish	2003	2m10s	47	2.77	Mixed
Constant Gardener	2005	3m33s	64	3.33	Mixed
Curious Case of Benjamin Button	2008	3m10s	78	2.44	Action
Departures	2008	5m28s	62	5.29	Action
Invictus	2009	1m29s	22	4.04	Mixed
The Help (clip 1)	2010	2m50s	28	6.07	Dialogue
The Help (clip 2)	2010	3m39	55	3.96	Dialogue
The Hunger Games (clip 1)	2012	3m59s	101	2.37	Mixed
The Hunger Games (clip 2)	2012	22s	11	2	Action

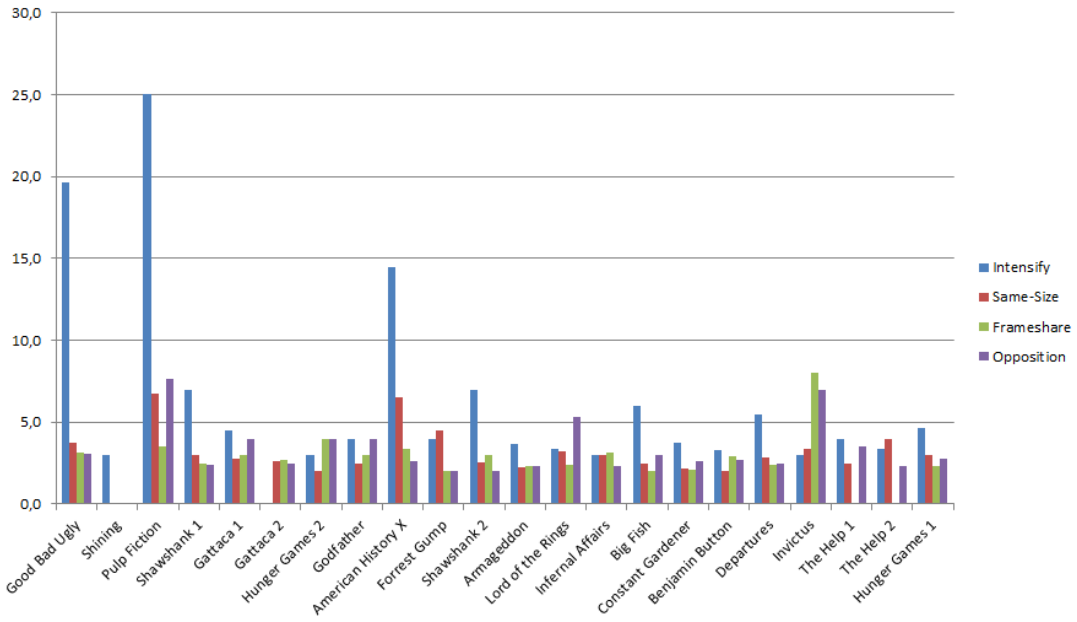


Figure 4.12 – ECPs used by films in our dataset by the average ECP length in number of shots.

Table 4.2 – Percentage (%) of occurrence of ECPs over all shots and average length in shots

% of shots in the clip that belong to ECP (avg. ECP length in no. of shots)				
Title	Intensify	Same-Size	Frameshare	Opposition
The Good, the Bad, and the Ugly	60 (19.7)	50 (3.8)	45 (3.1)	44 (3.1)
The Godfather	40 (4.0)	33 (2.5)	30 (3.0)	67 (4.0)
The Shining	60 (3.0)	0 (-)	0 (-)	0 (-)
American History X	51 (14.5)	68 (6.5)	53 (3.3)	23 (2.6)
Forrest Gump	22 (4.0)	50 (4.5)	11 (2.0)	11 (2.0)
Pulp Fiction	86 (25.0)	93 (6.8)	24 (3.5)	79 (7.7)
Shawshank Redemption - 1	44 (7.0)	25 (3.0)	21 (2.5)	40 (2.4)
Shawshank Redemption - 2	27 (7.0)	69 (2.6)	12 (3.0)	23 (2.0)
Gattaca - 1	47 (4.5)	58 (2.8)	32 (3.0)	63 (4.0)
Gattaca - 2	0 (-)	68 (2.6)	42 (2.7)	26 (2.5)
Armageddon	40 (3.64)	31 (2.2)	35 (2.3)	37 (2.3)
Lord of the Rings	17 (3.3)	28 (3.2)	21 (2.4)	55 (5.3)
Infernal Affairs	19 (3.0)	38 (3.0)	53 (3.1)	34 (2.3)
Big Fish	51 (6.0)	43 (2.5)	9 (2.0)	19 (3.0)
Constant Gardener	46 (3.75)	20 (2.17)	39 (2.1)	45 (2.6)
Curious Case of Benjamin Button	30 (3.3)	33 (2.0)	33 (2.9)	39 (2.7)
Departures	18 (5.5)	50 (2.8)	19 (2.4)	16 (2.5)
Invictus	14 (3.0)	46 (3.3)	36 (8.0)	32 (7.0)
The Help - 1	29 (4.0)	61 (2.43)	0 (-)	25 (3.5)
The Help - 2	31 (3.4)	36 (4.0)	0 (-)	13 (2.3)
Hunger Games - 1	14 (4.7)	51 (3.0)	32 (2.3)	47 (2.8)
Hunger Games - 2	27 (3.0)	18 (2.0)	36 (4.0)	36 (4.0)

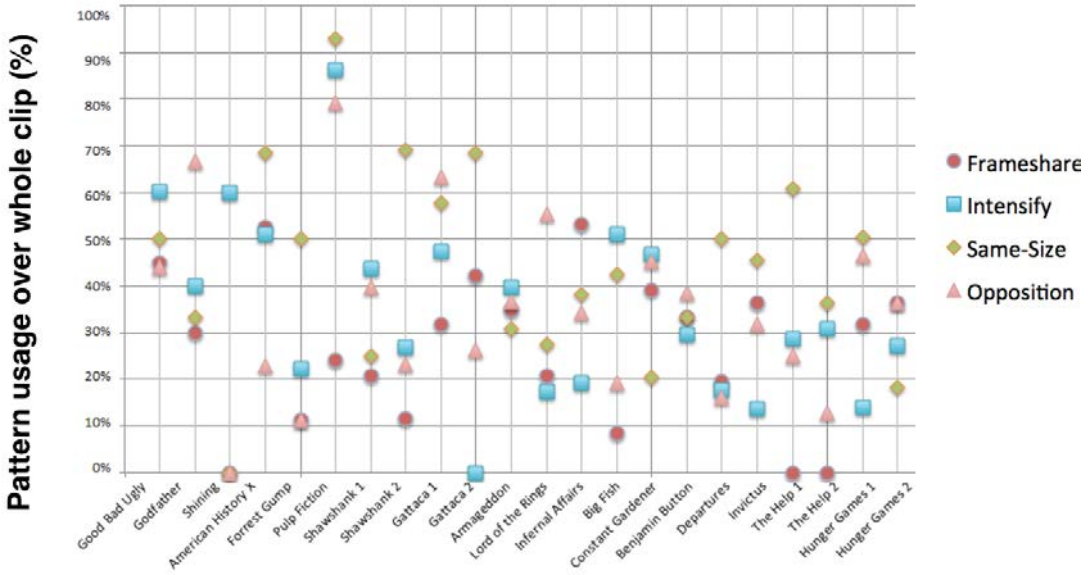


Figure 4.13 – Evolution of ECPs used by films throughout the period of our dataset. Each dot on the graph represents the percentage of shots that a certain pattern is used throughout the clip. For example, in the clip from *Pulp Fiction*, almost the whole clip is composed of the three types of sequences: $intesify$, $same-size-chain$, and $opposition$. We can thus deduce that the techniques are overlapped throughout the clip.

of time Figure 4.13. This is different from the case study of *Lord of the Rings* from the previous section in that we can compare how ECPs are used (for example, on average, intensify sequences are longer than any other ECPs), or immediately point out special cases (such as long intensify chains in *The Good, the Bad, and the Ugly*, *American History X*, and *Pulp Fiction*) among the datasets.

Further data such as the number of specific shot sizes, regions, or angles could also be extracted from the data to identify differences of style between different genres, scene types, directors, or various periods of time.

7 Chapter Conclusion

This work is the first attempt at linking cinematography to storytelling for the purpose of computational film analysis. There are a number of limitations to this work that we found through the development of *Patterns* and the ECP solver, and also through discussions and exchanges with professionals in the field. From our experience, the greatest limitation to the work is the need for accurate data. This requires either very refined image processing tools, or, as is in our case, many hours of annotations for each dataset. Through the development of *Insight*, we are able to extract shot sizes, regions, and angles with much higher accuracy, but it still requires the annotations to be carried out by a human.

The second limitation, which is also an ongoing challenge in fields of computer science, film study, and cognitive analysis, is the link between visual features of cinematography, and the story. In this work we have used very classical examples of manipulating shot size for emotion, and framing regions to express relationships between characters. Yet even with these classical techniques, exceptions occur. Even more difficult is to observe new and non-documented sequencing techniques, and use ECPs to find occurrences in other films.

However, we do believe in the potential of *Patterns* and ECPs to play roles in 1) validating hypotheses on recurring film styles in film analysis, 2) educational scenarios for learn by example or as an assisted creativity tool, and 3) in data collection for applications to virtual cinematography or editing tools. The third point especially is the focus of the next chapter.

Data-Driven Cameras for Interactive Storytelling

5

Contents

Film language comes from the accumulated experience of filmmakers as well as the observations of film analysts and critiques. Initially, this use of a dedicated terminology may seem discouraging to the outsider; however, over time, directors can use language they are familiar with for the purpose of shielding excessive technical details when, for example, efficiency of communication is the goal. Moreover, as we have seen from Chapter 4, such terminology often corresponds to well-known visual styles over sequences that create emotional and meaningful depth to storytelling.

With the advances of multimedia and virtual storytelling comes the benefit of quick and realistic cinematography in 3D gaming environments. Existing machinima tools can automate camera placement to capture action in well-defined story scenarios based on certain elements of *context* such as number of characters, action, and scene type. However, the power of cinematography lies in its capacity to communicate the filmmaker's intentions–plot progression, emotions, and morals–which we term as *communicative goals*. From film literature, we observe a number of well-established camera techniques–which we define as ECPs in our system–used to convey these communicative goals. For example, the intensify sequence can build emotion in the current event. As in the making of film, creators of interactive stories and games for 3D environments would need cinematographic tools that can achieve multiple communicative goals in each scene, but also respond to dynamically changing story lines. Thus, a simple but expressive method of defining and applying camera techniques to construct elaborate levels of discourse is still highly desired.

This chapter explores the integration of *Patterns* into 3D interactive storytelling environments in order to automatically generate the cinematography for the story. Two main contributions are introduced. The first is a real-time cinematography component for interactive stories which uses ECPs to find camera positions that take into account the story context and pertain to a specified style offer the sequence while players are interacting with the story scenario. The second is a smart camera tool that condenses the animation pipeline of camera-placement, rendering, and editing to a single interface. The interface and its smart solver provides the user with a database of camera framings collected from real films, and allows the user to experiment various ways to cut and reorder shots, and apply layers of ECPs over the sequence.

1 ECP-Based Camera for Interactive Stories

In virtual environments, such as for 3D animations, the job of the cinematographer is transferred to armies of digital artists and 3D animators in order to position characters and cameras to match framing specifications. Unlike in the real-world, virtual cameras do not have cranes and rails to guide the camera's positions or movement, making virtual camera editing a challenging and time-consuming task. Camera systems rely on visual constraints of the desired output–where actors appear in the shot, how large they are, from what angle, what is the start/end framing of a shot movement–to make placement and movement decisions. This is where constraint and rule-based systems such as [CAH+96] [DZ94] [BTM00] come in with the idea of encoding film idioms that are understandable to virtual camera systems. More recently, the Prose Storyboard Language (PSL) [RBLA13] was designed based on film practices outlined in Walter Murch's film textbook [Mur01] on fulfilling certain framing constraints on each shot in a sequence.

One challenge that remains to be addressed is automated cinematography in dynamic stories (where the events in the story are not pre-determined), takes into account the story context, and pertains to a certain visual style. This challenge mainly emerges from issues that there are infinite camera placement possibilities in the virtual environment. For a visual style to be consistent over long sequences, the camera must be able to plan ahead to take into account the upcoming actors and events in the story. Yet in a dynamic storytelling scenario, information on upcoming scenes may not be available, and to calculate all possible next events would lower the efficiency of the camera placement.

From observations of previous work, our objective is to develop smart tools for creators of interactive stories to search and experiment with various camera techniques that can fulfil multiple communicative goals in a given scenario, find shots that fit camera patterns for a sequence of actions, and observe their effects in a real-time 3D environment. This objective is achieved in three steps: (1) we use an established shot database, which are on-screen framing specifications annotated from real films, (2) we interpret communicative goals on the story events or the emotions as ECPs, which translates as constraints over a sequence, and (3) we propose a decision-making process by the virtual cinematographer that finds suitable shots and transitions based on the current story context and specified communicative goals, and performs camera placement and edits

in a real-time 3D environment.

This work was done in collaboration with master's student Pei-Chun Lai and Professor Tsai-Yen Li from the Intelligent Media Lab in National Chengchi University, Taiwan on the INRIA/MOST Formosa project. On their side was the main design of the project including the communication goals to be achieved, the linking of the camera placement algorithm to the 3D environment, occlusion detection, and various smart tools for fine-tuning camera positions. On our side, the main contribution was the design of a simplified version of ECPs for the generative stories, establishing of the shot database, and the link of the solver to the 3D environment.

1.1 Overview

In Chapter 2, we identified four main stages of a virtual cinematography component in interactive storytelling systems: the *framing*, which takes as input the 3D geometry of the scene and produces aesthetic and relevant camera positions and framings; the *editing*, which takes into account the context –namely, actions, events, and temporal and spacial elements–of the story; the *discourse component*, which conveys the director's intention, the communicative goal; and the *storytelling system*, which decides story events to progress the plot.

The components of previous chapters are designed to fit into this workflow shown in Figure 5.1. The first part is the virtual director, composed of the *Theater* platform and the logical control mechanism that was introduced in Chapter 3 Section 1.5. The second part is the virtual cinematographer, which includes the shot database, ECP solver, and camera placement algorithm.

In the next sections we introduce the roles of the virtual director and the virtual cinematographer. We then show a demonstration scenario using a scene from the movie *The Shining*.

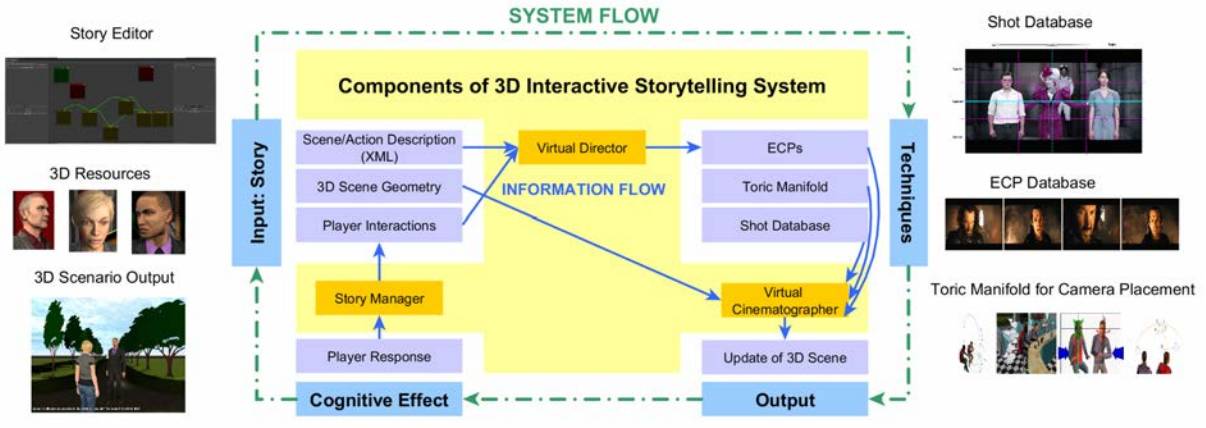


Figure 5.1 – This workflow shows how our system follows the cinematography system for storytelling we outlined in Chapter 2. The green arrows present the overall flow of the system, and the blue ones represent the flow of information on the geometry, story, and communicative goals.

1.2 Virtual Director

The virtual director takes the role of the real director in a filmmaking process: reading the story script, deciding what to film, and directing the cinematographer to match a desired style or feeling in the visual output. This part of our system is composed of the Unity *Theater* and the story logic control mechanism.

1.2.1 Virtual Director Workflow

The work flow of the virtual director is as follows: First, it takes as input the story graph, and using our logic control algorithm (Chapter 3) filters the story graph based on the most recent user interaction and temporal rearrangements. It can then select the next event that is to be shown to the player.

When an event is selected to be shown, the event is mapped to the pre-designed 3D animation and content with all the animations in the format of the *Theater* XML script. From the *Theater* XML script, the virtual director can then extract the context of the event, which includes information on the actors involved in the event, the event type (e.g. dialogue, fight, travel, etc.), and the emotion, which serves as the communicative goal. The virtual director then sends this contextual information to the virtual cinematographer, and waits for the virtual cinematographer to return a suitable camera position for the event. It then shoots the event, and then repeats the previous steps until the end of the story.

1.2.2 Communicative Goals

We have defined communicative goals such as $dialogue$, $buildEmotion$, $conflict$,and $compassion$ which can be realised by a combination of ECPs such as $same-size-chain$, $intensify$, $opposition$, and $frameshare$ defined in Chapter 4. The mapping between the communicative goal and the ECP is direct, but a communicative goal could require more than one ECP to fulfil, or can have an option of multiple ECPs that can all fulfil the goal. It is then up to the virtual cinematographer to select which ECP is most feasible based on the availability of shots in the shot database, the position of actors in the scene, and etc.

1.2.3 Context

The first input of the *Theater* is the 3D animated scenario comprised of a scene, virtual actors, communicative goals for each scene, and the event type (such as $escape\ danger$, $tell\ secret$, or $discuss\ murder$). This gives the virtual director an idea of what kind of event is taking place, which actors are involved, and their physical locations in the 3D environment. The combined information about the event is what we call $context$.

Our story is represented by an XML scenario which specifies the scene, character position and actions, and story context (such as emotion parameters) in high level commands. The specification of the story context also defines the point at which the camera can make a cut. A sample of the context description is:

<AnimContext Scene="discontent" Action="tellSecret" Goal="buildEmotion" Actor1="millionaire" Actor2="woman" />

Where Scene represents the name of the scene. Action represents the event type that is carried out by the characters, such as walking, talking, fighting, etc. Goal is the communicative goal that the story should convey, such as emotional intensify, horror, affinity, isolation, etc. Actor + Number represents the IDs of the character involved in the specific action. In the above example, the XML represents the statement "*millionaire* is *telling a secret* to *woman* with communicative goal of *buildEmotion*." Each scene may have any number of actions and communicative goals, and the virtual director parses and extracts all the necessary elements. A communicative goal can then be achieved by fulfilling the constraints in an ECP by the virtual cinematographer.

The next section describes in detail the process of the virtual cinematographer.

1.3 Virtual Cinematographer

The virtual cinematographer is the component in our system that makes decisions on the placement of cameras and sequencing of shots. It is composed of the shot database, an efficient camera placement algorithm, and the ECP solver.

The shot database is made up of all the framing specifications we have annotated from real films, introduced in Chapter 4. Each specification provides the on-screen 2D coordinates of important targets, such as actors' head and eye positions, or objects that are being interacted with. This provides enough information for a shot to be recreated using a algorithm to place and move cameras. Figure 5.2 shows how the shots in the shot database can be rearranged and applied to a new sequence.

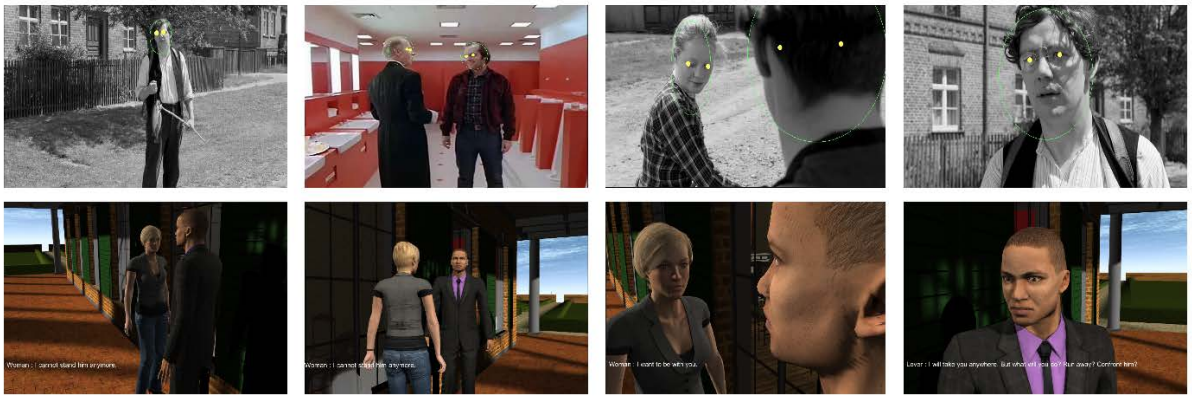


Figure 5.2 – How shots in the database are reused and recombined to generate the intensification sequence. The top row shows shots from the database The bottom row is how they are reapplied to a generated sequence.

The communicative goals that the virtual director sends to the cinematographer and shot specifications in the shot database provide the virtual cinematographer with the necessary information to decide on the suitable camera techniques over a sequence of story actions. Each communicative goal can be translated into an ECP that would fulfil that goal.

The next step is finding shots in the shot database that have the same event type as the current event. This is designed so that the virtual cinematographer is actually consulting the shot database on previous film examples of how to film a specific event. Since there is no direct mapping between each event type to shot specification, the virtual cinematographer engages in a decision process that involves searching all the shots in the database that match both the event type, and the constraints in the ECP.

Then the virtual cinematographer needs to evaluate the quality of the shots, and select the best one. In the shot database, there may be many shots that fit the event type in the story and the ECP constraints. However, the shot context does not account for camera style nor for 3D scene components, which may result in problems such as occlusion of characters, or inconsistency of camera styles such as the line of continuity (otherwise known as the 180 degrees rule). Therefore, it is necessary to check for shots to ensure critical parts of the scene–such as the character's face, critical scene components, or objects of interest–are not occluded from the camera.

After a number of framing specifications have been selected, they are sent to the camera placement along with the list of target actors. The role of the camera placement algorithm is to calculate the camera position for a framing specification, place the camera, and shoot. We use the computing method proposed by [LC12] to find the camera opsition that would output the exact on-screen positioning of two or three subjects, where the solution space for the exact specification of two subjects is represented as a 2D manifold surface. The manifold is defined by two parameters φ (the vertical angle of the camera) and θ (the horizontal angle). The search algorithm finds a 3D camera position P such that the two key subjects A and B $(A \neq B)$ project respectively at normalized screen positions $p_A(x_A; y_A), p_B(x_B; y_B) \in [-1; +1]^2$, and the camera orientation is algebraically determined from the camera position. With the camera placed in the environment, we can then calculate amount of occlusion and camera distance to subjects using ray casting from the camera to the target actor. Finally, the cinematographer ranks the frame specifications based on the amount of occlusion, and the best framing is selected and executed using the same camera positioning algorithm.

1.4 Demonstration

In our sample scenario we use a dialogue scene from the movie *The Shining* where the antagonist Jack Torrence is talking to a ghost Delbert Grady about past events in the Overlook Hotel. Jack is the winter caretaker at the Overlook Hotel. While they chat, Jack interrogates Grady about Grady's horrifying past (Jack: "You eh... chopped your wife and daughters up into little bits...and you blew your brains out"). The director of the original film chose to violate the 180 degree rule, crossing the index line between shots to create a feeling of split personality in Jack, giving the viewer a sense of confusion and horror. We call this technique $crossline$, and define the $crossline$ ECP as:

```
crossline{
    length
        constraint: length==2
    relation
        constraint: continuity!=180-rule
```

```
        range: all
    framing
     constraint: targetNum==2
}
```

In our example, we replicate the technique by using *crossline*, which places the camera on opposite sides of the index line between Jack and Grady for the purpose of creating the feeling of disorientation and horror. Suppose apart from horror, one would also like to build emotion as the dialogue progresses. The virtual director receives the communicative goal of “Horror" and “BuildEmotion" for the dialogue scene. The two goals are looked up in the ECP library and are translated into two ECPs: *crossline* and *intensify*. The virtual director then selects shots that fulfil both the distance requirement (i.e. starting from a long shot and moving closer) and the crossline requirement (i.e. switching between the two sides of the axis). The output is shown in Figure 5.3.

From the demonstration we can see that the tool we developed can provide a number of possibilities to interpret and shoot communicative goals, and also how multiple communicative goals can be specified to experiment multiple levels of discourse in a single scene. The context–including the story, location, character positions, and actions taking place–are provided by the system in real time while the virtual director makes decisions on the shot specification to use based on the communicative goals, translated into the camera techniques that are described in terms of shot patterns.

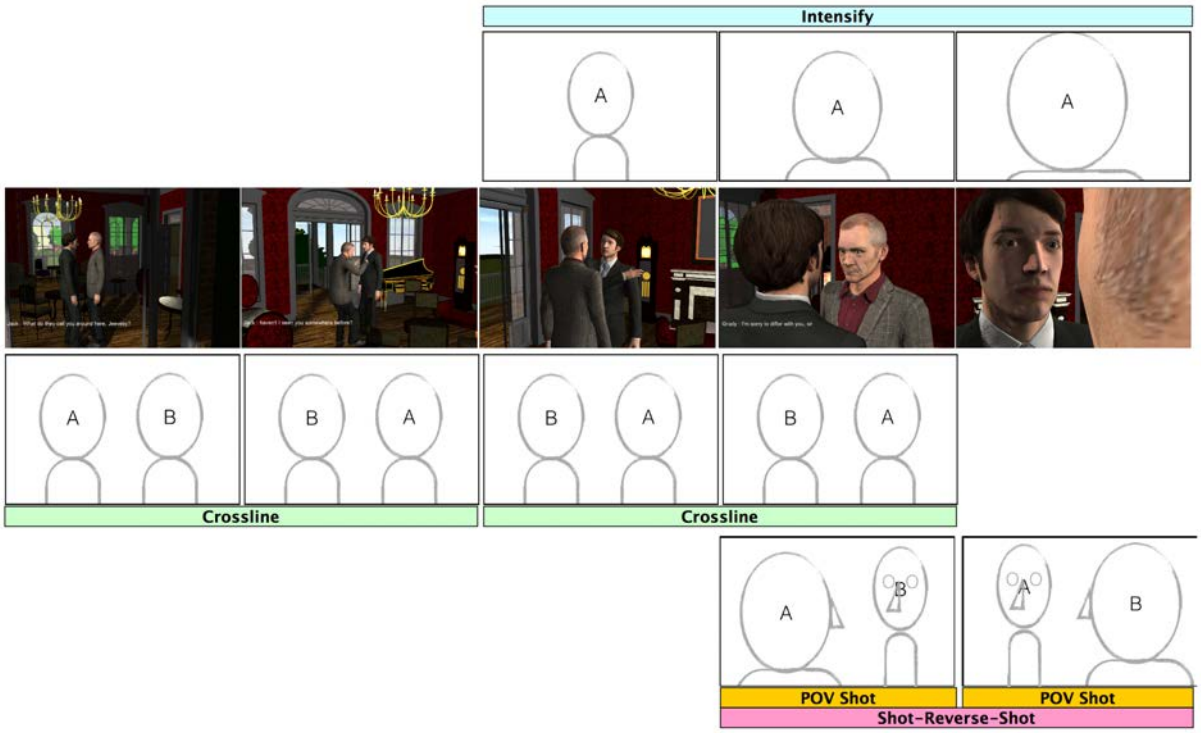


Figure 5.3 – Three ECPs are applied in this sample scene from *The Shining*, and the shots are selected dynamically to fulfil these ECPs dynamically.

1.5 Summary and Future Work

Our work provides creators of interactive stories a simple and efficient tool for defining and experimenting with cinematographic styles in 3D environments. As a first step to creating the link between story context, communicative goals, to shot specifications and camera placement in the 3D environment, though we believe a more complex model of linking story context could greatly improve the final results. For example, though we currently use occlusion as the primary factor for ranking shot specifications, selecting shots with least occlusion may not always be the most desirable when occlusion is used as a framing technique for creating emotions such as mystery or suspense. By allowing users to specify elements of story context, style, and importance of objects in the frame that should be taken into account for the ranking, the decision process of the virtual director can be further enriched.

When we view the tool as a potential for assisting creativity, our next step is to evaluate the use of the tool in real scenarios: game design, film production, and machinima. From understanding the needs of application scenarios, we also envision easy-to-use interfaces and authoring tools for storytellers and cinematographers to quickly create story scenarios, and select, edit, and apply camera shots to scenarios, while maintaining conformity to story logic and camera style. In this work we have introduced a tool to assist creators of interactive stories in enriching machinima discourse and achieving their communicative goals through a simple and expressive pattern language developed from well-established camera techniques in film. We enforce the link between discourse and story context through the building of a shot database that provides rich context information from existing shots in film literature. Our animation system further allows users of the system to immediately adjust and observe the effects of the chosen communicative goals. Through a demonstration of replicating and enhancing a dialogue scene from the movie *The Shining*, we show how multiple effects can be combined to allow the user to experiment various cinematographic representations and build elaborate levels of discourse.

2 Using ECPs for Smart Cinematography Editing

Camera editing for virtual environments can be a difficult task due to the technical skill required to use animation software and place the virtual camera either through detailed calculations or trial and error to get the desired results. Moreover, the animation pipeline can be long, requiring, modelling, animating, then rendering, editing, and if all is not satisfactory, all the way back to the beginning. This is makes using virtual environments costly and time-consuming for filmmakers. Virtual environments have the benefit of being much lower in cost (requiring no physical props) and convenient (can be used anywhere with a good computer), and can allow students of film to learn and experiment, and for directors to rehearse and pre-visualise at a greatly reduced cost. Yet for the virtual environment to be beneficial to these film artists and those in training, the interface, functionalities, and tools must be more accessible. Reducing the duration and stages of the pipeline can also make it easier to test multiple ideas and concepts.

Yet, are filmmakers ready for automated tools for virtual cinematography? Can creativity in virtual environments be assisted by the computer? Previous evaluations in computational interface design have found that film students and artists are prepared for interfaces that make it easier to edit cinematography in virtual environments [DZO+13]. The research took place on an interface for virtual camera editing by allowing the user to select camera positions. A dialogue box on the interface connected the user to 5 professionals and experts of film editing, who would, under the disguise of an automated system, provide real-time feedback on obvious camera placement errors. The study found that the feedback improved the accuracy of film editing by the users, compared to those who did not receive the feedback. Though the design of the interface does not in itself contain smart components for detecting camera editing errors and nor an automated feedback mechanism, this study shows users are open to real-time feedback from computational interfaces as they conduct creative activities. It is time one really implemented these smart tools for virtual camera control and editing interfaces that can benefit a much wider creative group.

As reviewed in the previous work in virtual cinematography for storytelling (Chapter 2), existing approaches in virtual cinematography have begun focusing on continuity rules and framing properties into constraint-based or optimisation systems, introducing film-like virtual camera control for storytelling scenarios [LC15] [GRCS14] [GRLC15]. Yet the use of film idioms for automated or assisted virtual cinematography editing has not been sufficiently explored, especially when it comes to stylistic choices–and not just continuity rules–directors make over a sequence (like changes of shot size, angle, or framing specification).

In this work, we design a virtual cinematography editing interface that allows the user to conduct virtual cinematography editing directly in the virtual environment and see the results in real-time. One of the most important features is the capacity to apply user-defined ECPs to the sequence that the user has cut, edited, and temporally adjusted. It is an interactive design where the constraints in each ECP is maintained. When the user makes changes to the camera sequence, the solver automatically filters framings in the database to provide the user a refined set of framings that fulfil the constraints of the ECPs, and the constraints are propagated.

This work was conducted in collaboration with master's student Francesca Palu and Professor Roberto Ranon from the University of Udine, Italy. On their side, the work included all the visual design, interaction design, display of debug logs, and implementation of story time and discourse time concepts. Initial interface designs and user project save/load features were made by master's student Jouwen Wang from National Chengchi University, Taiwan, under the INRIA/MOST Formosa Associate Team project. On our side, the main contribution is the design of the underlying data structures, coding the $Patterns - ECP$ solver tailored to the interface, linking the solver to the interface, and building the framing database.

2.1 Overview

The system is composed of two parts: the ECP solver and the interface. The interface is composed of a set of editing tools that the user can use to film a 3D scene. The

sequence begins with a single shot shot from a default camera position that is equal to the entire duration of the animation in the 3D scene. The user can then cut the sequence into multiple shots, reorder the shots, delete or duplicate shots, and also adjust the duration of shots. The camera positions of each shot can also be changed by selecting a different framing specification from the shot database, which is a visual library of framing specifications that are captured according to the positions of the actors at the beginning of each shot. When one or more shots are selected, an ECP can also be selected and applied to the selected shots as a stylistic constraint over the sequence. This calls the ECP solver.

If a shot has an ECP applied to it, the role of the ECP solver is to filter the shot database so that the user can only select those framing specifications for the shot that can result in the style defined by the ECP. An ECP can be a constraint on a single shot, or on the relation between multiple shots.

The next section details the algorithm for the ECP solver. We then introduce the design of the interface and how the solver and the editing tools are linked to it.

2.2 Pattern Interface Solver

The main smart component of the interface is the mechanism that allows users to apply ECPs to a number of selected shots in the sequence. By applying an instance of an ECP to the selected shots, the user is setting a constraint to fulfil the ECP on all the shots selected, as a sequence. The job of the solver component is to remove the framing specifications (from the shot database of each selected shot) that violate or cannot fulfil the ECP that the user applies. Thus, the idea is to limit the framing specifications that the user can choose from in each selected shot to only those that can fulfil the ECP instance.

Algorithm 6 ConstraintPropagate (ECPInstanceList P)

```
1:  ECPListCopy P'=P
2:  while P'.count!=0 do
3:      ECPInstance e=P'.pop()
4:      ECP p = e.ECP
5:      ShotSequence S = e.S_[x,y]
6:      ECPFilter(p,EmptySet,0,S)
7:      for all ECPInstance ei ∈ P do
8:          for all Shots s ∈ S do
9:              if s∈ei.S_[x,y] and ei∉P' then
10:                 P'.add(ei)
11:                 break
```

The interface solver extends the ECP solver in Chapter 4. The solver initialises each shot with a list of candidate framings (from the shot database). The main body of the algorithm (Algorithm 6) is a constraint propagation method that maintains and updates an active list of ECP instances that need to be solved. Each ECP instance contains an ECP and an ordered list of shots $S_{[x,y]}$ to which the ECP is applied. Algorithm 6 iterates on

the list of ECP instances, and incrementally filters the framing candidates for each shot in the ECP instance by calling Algorithm 7. Thus at each iteration, Algorithm 6 produces the subset of framing candidates that fulfils this ECP instance and all the preceding ECP instances. If the ECP instance removes framings in shots that overlap with other solved ECP instances, the constraints must be propagated by adding those affected instances back to the active list of ECP instances so that they can be re-solved.

Algorithm 7 ECPFilter (ECP p, FramingSeqeunce F, Position pos, ShotSequence **S**)

```
1: if pos<=S.count then
2:     for all Framings f ∈ S_pos.candidateFramings do
3:         F.add(f)
4:         ECPFilter(p,F,pos+1,S)
5:         F.remove(f)
6: else
7:     if IsValidECP(p, EmptySet, F) then
8:         for all Framings f in F do
9:             f.validate()
```

Algorithm 8 IsValidECP (ECP p, CurrentSequence **C**, Sequence **F**)

```
1: if F not empty then
2:     f_i = first(F)
3:     if isValidFrame(f_i) AND isValidRelation(f_i) then
4:         if isValidSubsequence(p, C ∪ {f_i}) then
5:             return IsValidECP(p, C ∪ {f_i}, F \ {f_i})
6:         else if isValidSequence(p, C ∪ {f_i}) then
7:             return IsValidECP(p, {f_i}, F \ {f_i} )
8:         else
9:             return False
10: else if ValidateLength(C) then return True
    return False
```

Algorithm 7 filters the framings among the candidate framings for all shots $s_j \in S_{[x,y]}$. Each possible framing f_i from the candidate framings must be either validated or rejected as a candidate for s_j in this ECP instance based on the following condition: if there exists a sequence of framings $f_x, f_{x+1}, ...f_i, ...f_y$ for each shot $S_{[x,y]}$ that fulfils the constraints set by the ECP (Algorithm 8), then f_i is validated, which means the framing should be available for the next ECP instance. If no combination containing f_i can be validated, then f_i is rejected. At the end of the process, the remaining framings for each shot are made available to the user for interactive selection.

The solver is called under all circumstances when a new ECP instance is added by the user, when shots are added or removed from one or more ECP instances, or when a framing is selected for a shot in one ore more ECP instances. The solver reacts to these three types of actions as following: (1) A new ECP instance is added: the new ECP instance is solved, and the constraints are propogated to overlapping ECP instances. (2) Adding/Removing of a cut or a shot: The shot is added/removed from all overlapping

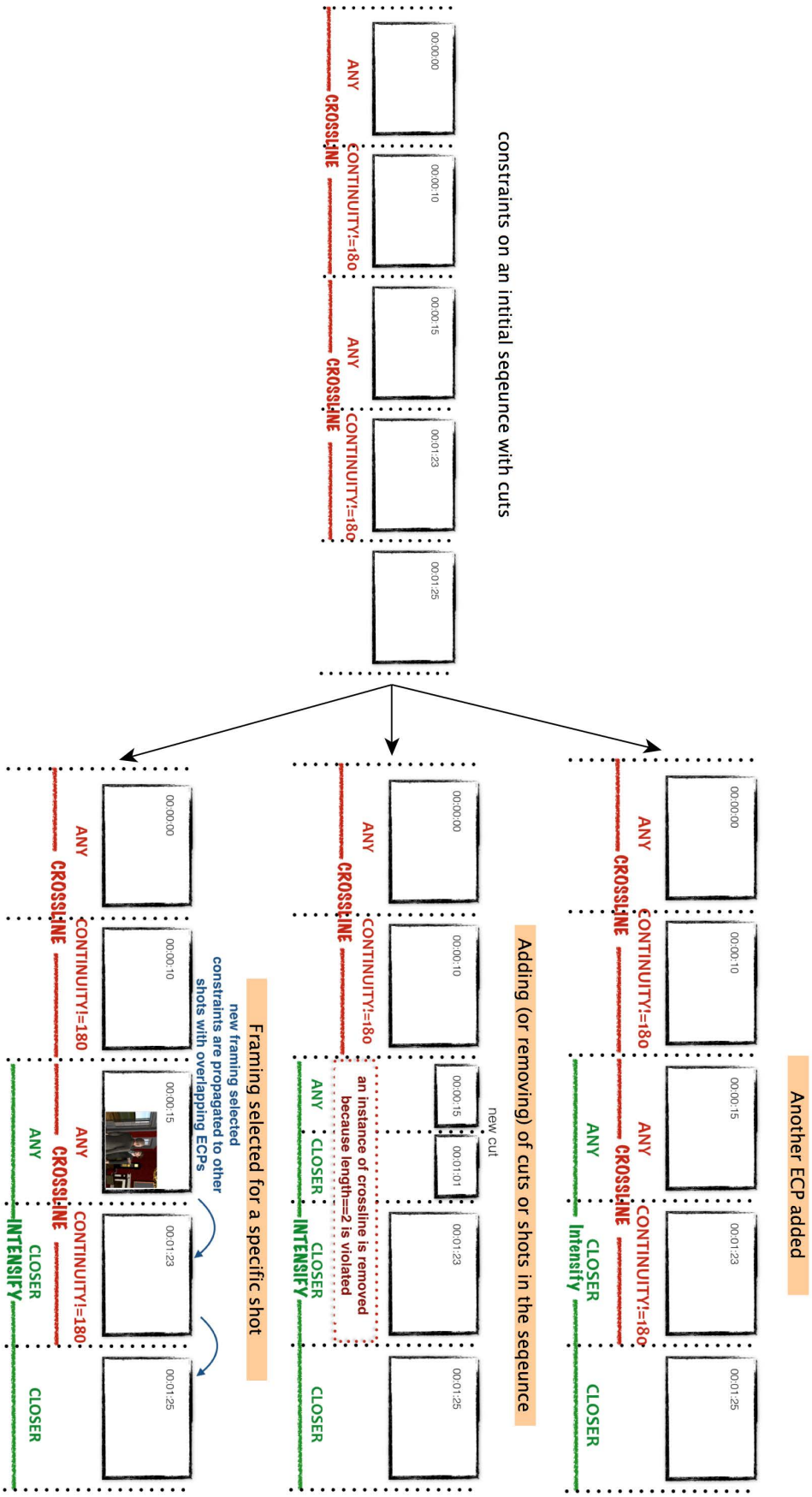


Figure 5.4 – The solver for *Patterns* interface is called when a user's action affects a new of existing ECP. This figure shows the types of actions that would trigger the solver to filter and update the shot database according to new constraints: (1) when a new ECP is added (2) adding/removing of a cut or a shot, and (3) a framing is selected for a shot.

ECP instances, and the ECP instances are re-solved. (3) A framing is selected for a shot: the framing is set as a new constraint and propagated to overlapping ECP instances. Figure 5.4 shows these three actions that would trigger the solver, and what the solver would do to uphold the ECPs on the new sequence.

Suppose there is a framing database of n framing specifications over a sequence of m shots, the complexity of the algorithm would be at the worst case n^m, which makes the algorithm quite slow, since it is a full search over all possibilities.

2.3 Interface Concept Design

The interface as shown from Figure 5.5 is composed of 5 main parts. The visualisation window on the upper-left corner provides a realtime replay of the current edit, including the cuts, temporal rearrangements, and user-selected framings. The current edit is shown as two timelines on the bottom of the interface, and each box in the timeline is a shot, and the shots are separated by cuts. The window on the upper-right shows the framing database: all the possible framings for a current selected shot. The middle section contains two main types of editing tools: the first are the simpler cut, delete, and duplicate shot options, which operate on the point where the cursor is on the timeline. The second type of editing tool is the ECP window to apply ECPs to a number of selected shots in the sequence.

Figure 5.5 – Our interface design for directly placing and shooting the scene. A framing database provides a variety of camera position selections. Cuts and shot lengths can be directly edited on the timeline through handlers. Playback is in real-time.

2.3.1 Visualisation Window

The visualisation window allows the user to view the realtime playback of the current edit. As the playback progresses, the playback also updates the cursor on the timeline to

the precise frame that is being played. This allows the user to directly correct problems when they appear without having to search the timeline for the precise position.

2.3.2 Discourse and Story Timeline

In the timeline, we introduce the separation of the concepts of story time and discourse time, and these two temporal concepts are represented by the top and bottom timelines respectively.

The top timeline is the discourse time. The discourse timeline shows one box for each shot. The order and size of the boxes directly correspond to the relative position of the shot compared to the other shots and the length of the shot (in seconds) respectively. Each shot contains the screenshots of the shot, updated when the user selects a different framing. The shots can be lengthened or shortened by dragging handlers, which changes the position of the cut between previous or following shots. The shots themselves can be dragged and rearranged. They can also be selected individually or as a group to be further manipulated by the editing tools. The playback in the visualisation window corresponds to the shots selected on the discourse timeline–that is, for example, if only shots 1, 2, and 4 are selected, the playback in the visualisation window will only play shots 1, 2, and for, leaving out shot 3 and anything after shot 4.

The bottom timeline show the relative story time the shot covers. As introduced before, the story time refers to the time in the world the story takes place. This allows users to observe flashbacks, overlaps in the story, or gaps in the story where some ellipses may occur. When the shots are rearranged temporally, the story timeline remains unchanged, since the shots still correspond to the same time period in the story time. The story time covered by each shot can be manipulated using the handlers.

2.3.3 Framing Database

The framing database contains the framing specifications.

When only one shot is selected on the discourse timeline, the framing database will show all the options of framing specifications for the shot in the window. Each framing is a specification of the on-screen position of the actors, and the horizontal and vertical angles of the camera. We currently assume none of the shots are tilted. All the framings are currently stable shots, containing no camera movements or zooms. Each framing is subsequently tailored to the character positions for the selected shot. In this way, it is guaranteed that the framing will be realised for each shot, and would not be affected if an actor walks around or moves from one shot to another. When a framing is selected by the user, it is immediately applied to the shot, and the result can be seen in the playback. This allows the user to quickly experiment various framings for a shot to see what works out best.

A search function is envisioned in the future to allow the user to quickly filter through a much larger database containing all the framings we annotated in Chapter 4.

2.3.4 Cut, Delete, and Duplicate Editing Tools

The first set of editing tools is cut, delete, and duplicate. These serve as the basic manipulation tools one can find in most video editing software, but retargeted for realtime virtual cameras.

These are tools that would manipulate number of shots, and sometimes their durations. The cut tool cuts a currently selected shot where the cursor is on discourse time. This results in separating one shot into two independent shots at the point of the cursor. Thus the user can choose a new framing specification for either shot without changing the framing of the other one.

The delete function removes all the selected shots completely from the sequence, while the duplicate function duplicates all the shots and appends them to the end of the sequence. This allows quick creating of ellipses or overlapping sequences to adjust the pace or highlight details in the scene.

2.3.5 ECPs

The final editing tool links ECPs to the interface. When one or more shots is selected, the user can select ECPs from a drop-down menu to apply to those selected shots. For example, the user could select shots 1, 2, 3, and 5, and then apply the intensify ECP, and then select shots 2 and 3, and apply shot reverse shot. There can be 2 results of applying the ECP to the shots. When successful, the framing database is filtered to show only those framings that have framings that fit into the constraints set in the ECP. Those who violate the ECP (either by violating a framing constraint, or because there is no framing selections for the preceding or following shots that allow it to fulfil relational and sub-sequence constraints) are hidden front he user. The ECP can fail to be applied, either if the number of shots selected do not correspond to the length constraint of the ECP, or if there are simply no available framing selections in the framing database for one or more shots.

2.4 Summary and Future Work

The most important next step for this work is conducting user evaluations. Though previous research may show that users could be ready for smart interfaces in assisted creativity, tailoring the user experience is still a big challenge. How much information is enough? What kind of feedback should the solver provide? Currently, we can only provide simple information on whether an ECP can be applied to a sequence, or whether their are shots in the database to support the ECP, but we cannot accurately pinpoint the source of the user's error (for example, pointing out the exact shot causing a conflict) or provide suggestions on how to improve the sequence (for example, suggesting the user to select a different framing for a certain shot). We also need to understand what kinds of feedback could restrict the user's creative inspiration, against our original purpose.

An eventual next step is providing tools for users to add self-created patterns without having prior programming experience, as well as creating self-defined shots directly from the interface into the shot database. These tools can be separate from the interface design, but would be directly beneficial to this work.

In conclusion, we have introduced an interface editing tool for virtual cinematography in storytelling. The interface enables the rapid design of 3D film sequences, which is a key to creativity, allowing users to explore a range of creative choices for placing, editing, and sequencing virtual camera positions and being able to see the result in real-time. Moreover, we provide a solver for ECPs directly linked to the interface that allows users to apply ECPs on a sequence, which filters framings in the database, which would imitate style decisions directors and editors make in actual film cinematography.

3 Chapter Conclusion

These two contributions for automated cinematography in virtual environments link together our logic control mechanism for interactive storytelling, and the patterns language for describing cinematographic style. This contribution focuses on applying interactive storytelling and real film techniques to virtual cinematography. Yet, we believe that this work is but the first step into pulling together the fields of film cognition, AI storytelling systems, and virtual environments together. Through the implementation of this work, we have found a number of broad directions this research could carry us towards creating immersive and engaging storytelling experiences as well as creativity assistance tools. Our future goal is in two steps: on the aspect of interactive storytelling, we would like to develop the *Theater* into an all-encompassing platform for virtual storytelling, providing storytellers with tools to edit the story graph, manage and create 3D content, add shot specifications to the database, and define their own ECPs for virtual camera planning; on the aspect of editing film sequences, we hope to conduct user evaluations on the editing interface for educational applications where students can learn and experiment various film style, and for potential use as a rapid prototyping tool for filmmaking .

Conclusion

6

This thesis has aimed to explore varying topics in interactive storytelling, cinematography, and virtual environments. Here, at the end, it is time to reflect on our findings and the further directions these contributions point towards.

1 Contributions

As the gap between the fields and ideas covered by each contribution was quite wide, here is a quick rundown of the work covered in the thesis, and how they have connected to each other.

Chapter 3 began with contribution to AI interactive storytelling systems, specifically looking at the **temporal properties of interactive stories**. We focused on developing an algorithm that upholds logicality of story events in stories involving complex temporal structures such as flashbacks. The main contribution of this work lies in increasing the replay value of stories by taking into account that temporal rearrangements of story events on the discourse level can also create a new variation of the story. This was the basis to our extended work on **evaluating how the audience perceived stories with temporally rearranged events**, and more specifically flashbacks. The importance of this work lies in the cognitive modelling and prediction of how the audience would interpret inserted flashbacks events based on a model of the viewer's memory. We stand apart from previous work on flashback generation by expanding the possible discourse functions of flashbacks not only just to fill in a logical gap to the story, but also to remind the audience, change their beliefs at different points throughout the story, and to reinforce a belief.

Our second contribution on **Patterns: analysis of cinematography storytelling** in Chapter 4 moved away from interactive systems towards visual–mostly film–storytelling by developing the concept of embedded constraint patterns (ECPs) for the purpose of analysing film storytelling. This chapter also introduced the revised *Insight* annotation tool, which allowed us to annotate a database of shots. The novelty of this work lay in bringing together annotated film data and pattern-matching techniques on high-level stylistic choices based on story made by the director over long sequences, setting the work apart from image processing techniques that operate on visual features in individual frames or shots. We have shown how analysis of ECPs in film clips can efficiently analyse film style over large amounts of film clips, indicate trends, and point out abnormalities among a dataset.

The first part Chapter 5, **data-driven cameras for interactive storytelling** applies the concept of ECPs from Chapter 4 to the generative storytelling system we developed

in Chapter 3 to automatically select camera positions that pertain to ECP defined constraints, in real-time, with user interactions. This was challenging in the aspect that the camera is not pre-calculated, and must adapt to the player's choices in the story, but still pertain to a number of film editing styles. There are two important contributions of this work that have not been seen in previous work. The first is bringing camera discourse techniques into game-like environments in the form of framing compositions annotated from real films, and ECPs that are commonly used by filmmakers, which creates a film-like narrative experience within the gameplay. The second is the virtual cinematographer component that does not pre-compute camera positions, but instead, in real time, reacts to context information, selects shots that avoid occlusion, and fulfils constraints for long sequences of style over multiple shots.

Finally, the second part of Chapter 5 then applies ECPs to a smart editing interface for virtual cinematography that shrinks the three stages of animating, rendering, and editing phases of the virtual camera in the animation pipeline in one interface, that shows the end result in real-time. This work has contributions in the aspects of creativity, practical filmmaking, and also in education. On the aspect of creativity, since the framing specifications and also ECPs come from real movies or real film practice, the interface allows users to mix and blend various styles from films into a single sequence, and experiment with different cuts or event sequencing possibilities. From the practical aspect, the interface directly takes a leaf from existing idioms and terminology that is used in real filmmaking. It shrinks the amount of time and cost needed to prototype the filming of a scene, since all outputs are presented in real time. From an educational aspect, the interface can automatically provide live feedback on whether a certain framing composition selected by the user may violate a specific style or continuity rule that the user also wants to match, and thus . We believe this tool opens up the possibility of more refined tools for virtual camera editing for film students and artists alike.

2 Future Work

There are a number of narrower topics that could be extended from each contribution, which we have described at the end of each chapter. What is the next step for the future development of cinematic discourse in interactive, immersive, and engaging stories that take place in virtual environments? Here, we provide a wider perspective on the overall contributions of this thesis on the areas that have not been sufficiently addressed in this thesis, but are nevertheless important.

2.1 User Evaluations

Evaluating a storytelling system is a challenging topic in the field of interactive storytelling due to the complex roles the user of these systems can take in the process: viewers, players, designers, artists, analysts, and storytellers, all of which appear in this thesis. However, unlike traditional media, in an interactive storytelling system, a person often has more than one role. In our current work, we have encountered two new types of users of our systems that are a blend of various roles, which we believe would be :

The Player-Viewer (Film Cognition, User Studies): a person playing a game would often be referred to as the player. However, many games now offer entertainment not just in skill and visuals, but also in the story. Playing a game can also be a film-like narrative experience in which the player also takes on the role of the viewer of the film. Central to our work is the understanding of how the cinematography in the game impacts the dual roles of the player (e.g. Does the cinematography interfere with the gameplay? Does the cinematography respond to the player's actions in the game?) and the viewer (e.g. Does the cinematography respond to the viewer's inner emotions? Does the cinematography guide these emotions and perception of the story?). The evaluation of the player-viewer would require understanding not just in virtual cinematography, but also human-computer interaction evaluations on the player part, and cognitive film studies on the viewer part.

The Storyteller-Artist (Computational Creativity, User Studies): The story and the discourse are intertwined when it comes to cinematography in interactive stories: the story affects how the camera should be placed to convey a certain scenario, but in turn, the way the camera guides the player's emotions would also affect their interactions, and indirectly, the outcome of the story. The designer of smart tools for creating game cinematography must then understand the needs of both the storyteller and the visual artist (a.k.a. cinematographer).

In-depth evaluations on the individual roles as well as the above combined roles requires more sophisticated experimental design adopting concepts from film cognition or HCI user studies, and the collection of qualitative, instead of quantitive, feedback from the participants.

2.2 Virtual Cinematography

Virtual camera techniques are at the core of delivering captivating visuals and well orchestrated animations. In this work we have primarily focused on the cognitive properties of camera placement and framing problems. However, good cinematography should also incorporate elements of movement, light, colour, and actor choreography. We believe in the potential of collecting real-film data, and trying to replicate this data using computer graphics and animation techniques, to use these elements to further express the story and emotion in the visual narrative.

Actor Choreography (Animation, Graphics, Film Theory): Currently the way we develop virtual camera techniques is to find ways of positioning a camera in order to capture the most interesting elements in a 3D scene, and to best tell the story of the scene. In reality, filmmakers actually tailor the shooting set and choreograph the actor's motions relative to the camera. Previous works have attempted to address virtual cinematography through a database of choreographed camera and actor positions. However, finding a more generalised solution to transforming a story scenario–such as from a film or theatre script–into a choreographed performance would greatly reduce the time needed to model and tailor a 3D scene, and could be useful for pre-visualisation of films before they are shot, and planning of the set with actors and technicians.

Movement (Animation, Graphics, Film Theory): Movement is a common way to express the emotions of the silent observer in a film. Long tracking sequences in films provide a continuous flow of events that keeps the viewer busy, but also brings awe to the beauty of the camera in revealing the story: with the camera following one actor then switching to another, characters entering and exiting the screen, events unfolding around the trajectory of the camera, as if the viewer were walking through the story themselves. Cameras shaking or with non-smooth paths can create a feeling of inner instability, turmoil, or disorientation of an actor or scene. Replicating the movement of cameras is another important way of expressing emotion, and also irreplaceable way to identifying a cinematographer's style. In our future work, we would hope to extend *Insight* annotation tool to create trajectories of target locations on the screen that could further be reproduced with a camera placement and movement algorithm, or set as constraints in *Patterns* to be tailored for other stories.

Aesthetic Quality of a Shot (Image Processing, Graphics, Computer Vision, Film Theory): In terms of the aesthetics of a shot taken in a 3D environment, our systems have only taken into account occlusion avoidance. However, whether a shot is visually pleasing can be a result of the lighting, colour, and visual arrangement on the screen, which in many instances, occlusion is a good thing. We can envision to learn from analysing shot compositions, colour, and lighting from collected film data to understand what the qualities of a good shot are. Using what we learn from films, we hope to expand the ECP-based cameras for interactive stories with more criterion to select shots that not just fulfil constraints and avoid occlusion, but are also pleasing to the eye.

2.3 Assisted Creativity

The storyteller is one of the end users we are targeting with our contributions. For tools to be accessible for the end user, many technical details have to be hidden under an interface. The challenge in creating assistive creativity technologies involves identifying the target user, as well as achieving a balance between he "smart" feedback from the tool while not limiting the user's creative potential. This can be further studied through user evaluations, and throughly understanding theories and principles of creativity. Here are a number of topics where our work can be potentially expanded into tools, interfaces, or research topics in assisted creativity:

Editing and authoring stories (HCI, AI Storytelling): A lot of work on AI story generation and interactive storytelling platforms have yet to be transformed into accessible interfaces for users from non-technical backgrounds. Yet it is our ultimate goal is to provide a set of integrated tools and applications for the whole authoring process of 3D interactive stories, starting from a story graph editing interface, then to tools for editing and generating 3D animation content, for editing cinematography and animations, and finally to pre-visualise and evaluate the story in real-time. With its virtual director and virtual cinematographer components, as well as a 3D content and animation manager, we believe the *Theater* could be developed into this stage for creating and simulating

interactive stories. This could also make way for the development of smart writing assistants that could check the logic of the story to alert the user to inconsistencies in the plots, or provide brainstorming tools to search and integrate new ideas that would become a branch in an existing interactive story.

Education and tutoring (HCI, Computational Creativity): The second target application of our work is for education of the next generation of new media content creators. Our interface for virtual cinematography was designed with this intent in mind. However, we could anticipate educational tools for editing interactive story graphs, or for creating interactive 3D animations. These tools not only enable the learner to create and experiment with their ideas, but also open areas of research in machine learning to process existing data on films, and create models of camera styles that could be applied to guiding educational activities, such as make corrections, provide feedback, and guide the user in the right direction.

Collecting data for film analysis (Video and Image Processing, Film Theory): In our work on ECPs for film analysis, we have shown the potential for analysing style over long sequences of shots. However, the film clips we analysed were hand-annotated, which would still require a lot of manual effort on the user's side. We are actively adding plugins for automatic face detection, motion flow, and colour and light analysis in our *Insight* annotation tool not only to speed up the data collection, but also to reduce human error. In the future extension of this work, we also hope to combine the workflow of defining and detecting ECPs into the annotation tool via user-friendly interfaces. This would also call for research in content-analysis of films that combine elements of visual style for film genre classification, scene segementation, or detecting the types of action taking place.

3 Summary

One prominent feature of all of these future directions of cinematic discourse is the need for collaboration and knowledge sharing between different fields of study. Most of the creativity in interactive storytelling comes from the collaboration between various fields: with computer graphics and animations for engaging visuals, with interaction designers for a diverse and immersive experience, with storytellers to create exciting and depth-invoking content, and etc. But the ultimate purpose is to provide tools for telling and presenting stories that can increase the overall storytelling experience, and this remains our passion and motivation.

Bibliography

[AH11] Lee Averell and Andrew Heathcote. The form of the forgetting curve and the fate of memories. *Journal of Mathematical Psychology*, 55(1):25–35, 2011. 55, 58

[AK05] Dan Amerson and Shaun Kime. Real-time cinematic camera control for interactive narratives. In *ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 369–369. ACM Press, 2005. 33, 38

[All84] James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, 1984. 56

[AWCO10] Jackie Assa, Lior Wolf, and Daniel Cohen-Or. The Virtual Director: a Correlation-Based Online Viewing of Human Motion. *Computer Graphics Forum*, 29(2):595–604, 2010. 35

[Bar14] Camille Barot. *Scénarisation d'environnements virtuels. Vers un équilibre entre contrôle, cohérence et adaptabilité.* PhD thesis, Université de Technologie de Compiègne, 2014. 19

[BGL98] William H. Bares, Joël P. Grégoire, and James C. Lester. Realtime constraint-based cinematography for complex interactive 3D worlds. In *The National Conference On Artificial Intelligence*, pages 1101–1106. Citeseer, 1998. 33

[BL99] William H. Bares and James C. Lester. Intelligent multi-shot visualization interfaces for dynamic 3D worlds. In *Proceedings of the 4th international conference on Intelligent user interfaces (IUI 1999)*, pages 119–126, 1999. 33

[Bli88] Jim Blinn. Jim Blinn ' s Corner Where Am I ? What Am I Looking At ? *IEEE Computer Graphics & Applications*, pages 76–81, 1988. 30

[BM06] C.V. Buhusi and W.H. Meck. Interval timing with gaps and distracters: evaluation of the ambiguity, switch, and time-sharing hypothesis. *Journal of experimental psychology: Animal behavior processes*, 32(3):329–338, 2006. 55, 58

[BMBT00] William H. Bares, Scott McDermott, Christina Boudreaux, and Somying Thainimit. Virtual 3D camera composition from frame constraints. In *Proceedings of the Eighth ACM International Conference on Multimedia (ACMMM'00)*, pages 177–186, 2000. 33

[Bor85] David Bordwell. *Narration in the Fiction Film*. University of Wisconsin Press, 1985. 14, 15, 24, 55, 70

[Bra92] Edward Branigan. *Narrative Comprehension and Film*. Routledge, 1992. 14, 49

[Bro11] Frederic Brodbeck. Cinemetrics. http://cinemetrics.fredericbrodbeck.de, 2011. 26

[Bro12] Blain Brown. *Cinematography Theory and Practice*. Elsevier Inc., 2 edition, 2012. 16, 18

[BS12] John A. Bateman and Karl-Heinrich Schmidt. *Multimodal Film Analysis: How Films Mean*. Routledge, New York, 2012. 16

[BTM00] William H Bares, Somying Thainimit, and Scott Mcdermott. A Model for Constraint-Based Camera Planning. In *AAAI Spring Symposium*, Stanford, 2000. 33, 92

[BW14] John A. Bateman and Janina Wildfeuer. A multimodal discourse theory of visual narrative. *Journal of Pragmatics*, 74:180–208, 2014. 14

[BY08] Byung-chull Bae and R. Michael Young. A Use of Flashback and Foreshadowing for Surprise Arousal in Narrative Using a Plan-Based Approach. In *First Joint International Conference on Interactive Digital Storytelling, ICIDS 2008*, pages 156–167, Erfurt, Germany, 2008. Springer Berlin Heidelberg. 24, 25, 55

[CAH+96] David B. Christianson, Sean E. Anderson, Li-wei He, David H. Salesin, Daniel S. Weld, and Michael F. Cohen. Declarative camera control for automatic cinematography. *AAAI Conference on Artificial Intelligence*, 1996. 32, 34, 71, 92

[CB98] J.M. Corridoni and A.Del Bimbo. Structured Representation and Automatic Indexing of Movie Information Content. *Pattern Recognition*, 31(12):2027–2045, 1998. 27

[CBL11] Luca Canini, Sergio Benini, and Riccardo Leonardi. Affective analysis on patterns of shot types in movies. In *7th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 253–258, sep 2011. 27, 28

[CBL13] Luca Canini, Sergio Benini, and Riccardo Leonardi. Affective recommendation of movies based on selected connotative features. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(4):636–647, 2013. 26

[Cha80] Seymour Chatman. *Story and Discourse: Narrative Structure in Fiction and Film*. Cornell University Press, 1980. 14, 23, 46

[CL12] Chia-Hao Chen and Tsai-Yen Li. Context-aware Camera Planning for Interactive Storytelling. In *9th International Conference Computer Graphics, Imaging and Visualization*, 2012. 38

[CLDM03] Nicolas Courty, Fabrice Lamarche, Stéphane Donikian, and Éric Marchand. A cinematography system for virtual storytelling. *Lecture Notes in Computer Science*, 2897:30–34, 2003. 37

[CM14] Gail Carmichael and David Mould. A Framework for Coherent Emergent Stories. In *Proceedings of Foundations of Digital Games 2014*, Florida, USA, 2014. 23, 25

[Coh13] Neil Cohn. Visual narrative structure. *Cognitive science*, 34:413–52, apr 2013. 14

[CON08] Marc Christie, Patrick Olivier, and JM Normand. Camera control in computer graphics. *Computer Graphics Forum*, 27:2197–2218, 2008. 29

[CPC+09] Fred Charles, David Pizzi, Marc Cavazza, Thurid Vogt, and Elisabeth André. Emotional Input for Character-based Interactive Storytelling. *Proceedings of the 8th Conference on Autonomous Agents and Multiagent Systems (AAMAS2009)*, 1:1381–1382, 2009. 20, 22

[CPFF05] Angelo E. M. Ciarlini, Cesar T. Pozzer, Bruno Feijó, and Antonio L. Furtado. Logic-based Tools for Interactive Storytelling. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology (ACE'05)*, pages 133–140, 2005. 23

[CrCWY12] Rogelio E. Cardona-rivera, Bradley A. Cassell, Stephen G. Ware, and R. Michael Young. Indexter : A Computational Model of the Event-Indexing Situation Model for Characterizing Narratives. In *Proceedings of 3rd Workshop on Computational Models of Narrative*, Istanbul, Turkey, 2012. 55, 58

[Cut15] James E. Cutting. The Framing of Characters in Popular Movies. *Art & Perception*, 3(2):191–212, 2015. 15, 70

[DB07] Leonard Daly and Don Brutzman. X3D: Extensible 3D Graphics Standard. *IEEE Signal Processing Magazine*, 24(6):130–135, 2007. 51

[dLPD+09] Edirlei E. S. de Lima, Cesar T. Pozzer, Marcos C. D'Ornellas, Angelo E. M. Ciarlini, and Antonio L. Furtado. Virtual Cinematography Director for Interactive Storytelling. In *The International Conference on Advances in Computer Enterntainment Technology*, pages 263–270, 2009. 36

[DZ94] Steven M. Drucker and David Zeltzer. Intelligent camera control in a virtual environment. In *Graphics Interface '94*, pages 190–199, 1994. 31, 92

[DZO+13] Nicholas Davis, Alexander Zook, Brian O'Neill, Brandon Headrick, Mark O. Riedl, Ashton Grosz, and Michael Nitsche. Creativity support for novice digital filmmaking. In *ACM SIGCHI Conference on Human Factors in Computing Systems*, page 651, 2013. 99

[Ebb85] Hermann Ebbinghaus. Memory: A Contribution to Experimental Psychology. *Annals of Neurosciences*, 2013;20(4), 1885. 55, 58

[EBY15a] Markus Eger, Camille Barot, and R. Michael Young. Impulse : a formal characterization of story. In *Proceedings of the 6th Workshop on Computational Models of Narrative (CMN'15)*, pages 45–53, 2015. 24, 56, 63

[EBY15b] Markus Eger, Camille Barot, and R. Michael Young. Merits of a Temporal Modal Logic for Narrative Discourse Generation. In *Proceedings of Intelligent Narrative Technologies 2015 (INT8)*, pages 23–29, 2015. 24, 56

[En04] Magy Seif El-nasr. An Interactive Narrative Architecture based on Filmmaking Theory. *International Journal on Intelligent Games and Simulation*, 2004. 36

[ER07] David K. Elson and Mark O. Riedl. A Lightweight Intelligent Virtual Cinematography System for Machinima Production. In *Proceedings of the Third AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE2007)*, Palo Alto, California, USA, 2007. 31

[FF06] Doron Friedman and Yishai A. Feldman. Automated cinematic reasoning about camera behavior. *Expert Systems with Applications*, 30(4):694–704, may 2006. 33

[FO12] Dezheng Feng and Kay L. O'Halloran. Representing emotive meaning in visual images: A social semiotic approach. *Journal of Pragmatics*, 44(14):2067–2084, 2012. 14

[FO13] Dezheng Feng and Kay L. O'Halloran. The multimodal representation of emotion in film: Integrating cognitive and semiotic approaches. *Semiotica*, 2013(197):79–100, jan 2013. 17

[Fro13] Jim Frost. Regression Analysis: How Do I Interpret R-squared and Assess the Goodness-of-Fit. http://blog.minitab.com/blog/adventures-in-statistics/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit, 2013. 64

[GCLR15] Quentin Galvane, Marc Christie, Christophe Lino, and Rémi Ronfard. Camera-on-rails : Automated Computation of Constrained Camera Paths. *ACM SIGGRAPH Conference on Motion in Games 2015 (MIG2015)*, pages 151–157, 2015. 31

[GCR+13] Quentin Galvane, Marc Christie, Rémi Ronfard, Chen-Kim Lim, and Marie-Paule Cani. Steering Behaviors for Autonomous Cameras. *ACM SIGGRAPH Conference of Motion in Games 2013 (MIG2013)*, pages 93–102, 2013. 35

[GDAPH05] Pablo Gervás, Belén Díaz-Agudo, Federico Peinado, and Raquel Hervás. Story Plot Generation Based on CBR. *Knowledge-Based Systems*, 18(4-5):235–242, aug 2005. 20

[Gen80] Gérard Genette. *Narrative Discourse: An Essay in Method*, volume 9. Cornell University Press, 1980. 24, 55

[GRCS14] Quentin Galvane, Remi Ronfard, Marc Christie, and Nicolas Szilas. Narrative-Driven Camera Control for Cinematic Replay of Computer Games. In *ACM SIGGRAPH Conference on Motion in Games 2014 (MIG2014)*, 2014. 38, 99

[GRLC15] Quentin Galvane, Rémi Ronfard, Christophe Lino, and Marc Christie. Continuity Editing for 3D Animation. In *AAAI Conference on Artificial Intelligence*, AAAI Press, Austin, Texas, United States, jan 2015. 36, 99

[GS11] Vittorio Gallese and Corrado Sinigaglia. What is so special about embodied simulation? *Trends in Cognitive Sciences*, 15(11):512–519, 2011. 15

[HCS96] Li-wei He, Michael F. Cohen, and David H. Salesin. The virtual cinematographer: a paradigm for automatic real-time camera control and directing. In *SIGGRAPH '96: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 217–224. ACM Press, 1996. 32, 33

[HHS01] Nicolas Halper, Ralf Helbing, and Thomas Strothotte. A Camera Engine for Computer Games: Managing the Trade-Off Between Constraint Satisfaction and Frame Coherence. *Computer Graphics Forum*, 20(3):174–183, 2001. 33

[HLT03] Alexander Hornung, Gerhard Lakemeyer, and Georg Trogemann. An Autonomous Real-Time Camera Agent for Interactive Narratives and Games. In *Proceedings of the IVA 2003: 4th International Working Conference on Virtual Agents*, pages 236—-243. Springer-Verlag, 2003. 37

[HTL14] Marissa Hoek, Mariët Theune, and Jeroen Linssen. Generating Game Narratives with Focalization and Flashbacks. In *3rd workshop on Games and Natural Language Processing*, 2014. 24, 25, 55

[Ish02] Toru Ishida. Q: A scenario description language for interactive agents. *Computer*, 35(11):42–47, 2002. 51

[JY10] Arnav Jhala and R. Michael Young. Cinematic Visual Discourse : Representation, Generation, and Evaluation. *IEEE Transations on Computational Intelligence and AI in Games*, 2(2):69–81, 2010. 38, 39

[KFF+15] Mubbasir Kapadia, Jessica Falk, Z Fabio, Marcel Marti, and Robert W Sumner. Computer-Assisted Authoring of Interactive Narratives. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (i3D)*, volume 2, 2015. 23, 25

[Kip10] Michael Kipp. Multimedia annotation, querying and analysis in anvil. *Multimedia information extraction*, 19, 2010. 26

[KKM+06] Stefan Kopp, Brigitte Krenn, Stacy Marsella, a Marshall, C Pelachaud, H Pirker, K Thórisson, and H Vilhjálmsson. Towards a common framework for multimodal generation: The behavior markup language. *Intelligent Virtual Agents*, 4133:205–217, 2006. 51

[KM02] Kevin Kennedy and Robert E. Mercer. Planning animation cinematography and shot structure to communicate theme and mood. In *Proceedings of the 2nd International Symposium on Smart Graphics 2002*, pages 1–8, New York, New York, USA, 2002. ACM Press. 37

[KMP77] Donald E. Knuth, James H. Morris, and Vaughan R. Pratt. Fast pattern matching in strings. In *SIAM Journal on Computing*, pages 323–350, 1977. 81

[KWB15] Christopher Kives, Stephen G. Ware, and Lewis J. Baker. Evaluating the pairwise event salience hypothesis in Indexter. In *Proceedings of the Eleventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE2015)*, volume 6, pages 30–36, Santa Cruz, California, USA, 2015. 67

[LC12] Christophe Lino and Marc Christie. Efficient Composition for Virtual Camera Control. In *Eurographics ACM SIGGRAPH Symposium on Computer Animation*, 2012. 30, 38, 96

[LC15] Christophe Lino and Marc Christie. Intuitive and Efficient Camera Control with the Toric Space. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2015*, 34(4), 2015. 26, 30, 31, 99

[LCL+10] Christophe Lino, Marc Christie, Fabrice Lamarche, G Schofield, and Patrick Olivier. A Real-time Cinematography System for Interactive 3D Environments. In *2010 ACM SIGGRAPH Eurographics Symposium on Computer Animation*, number 1, pages 139–148, 2010. 34

[Lin13] Christophe Lino. *Virtual Camera Control using Dynamic Spatial Partitions*. PhD thesis, University of Rennes 1, 2013. 29

[Lön05] Birte Lönneker. Narratological Knowledge for Natural Language Generation. In *Proceedings of the 10th European Workshop on Natural Language Generation*, pages 91–100, 2005. 24, 25

[Lot00] Jakob Lothe. *Narrative in Fiction and Film: An Introduction*. Oxford University Press, 2000. 69

[MBC14] Billal Merabti, Kadi Bouatouch, and Marc Christie. A Virtual Director Inspired by Real Directors. In *AAAI Workshop on Intelligent Cinematography and Editing*, 2014. 27, 28, 36

[Met66] Christian Metz. La grande syntagmatique du film narratif. *Communications*, 8(1):120–124, 1966. 16

[MK06] Hugh Mccabe and James Kneafsey. A Virtual Cinematography System for First Person Shooter Games. In *Proceedings of International Digital Games Conference*, 2006. 35

[MKSB11] Daniel Markowitz, Joseph T. Kider Jr., Alexander Shoulson, and Norman I. Badler. Intelligent Camera Control Using Behavior Trees. In *The Fourth International Conference on Motion in Games (MIG2011)*, volume LNCS 5884, 2011. 38, 39

[MM09] Danielle S McNamara and Joe Magliano. Toward a Comprehensive Model of Comprehension. In *The Psychology of Learning and Motivation*, volume 51 of *Psychology of Learning and Motivation*, pages 297–384. Academic Press, 2009. 55, 58

[Mon07] Nick Montfort. Ordering events in interactive fiction narratives. In *The AAAI Fall Symposium on Intelligent Narrative Technologies*, pages 87–94, 2007. 24, 25

[Mon11] Nick Montfort. Curveship's Automatic Narrative Style. *Foundations of Digital Games*, pages 211–218, 2011. 24, 25, 55

[MS02] Michael Mateas and Andrew Stern. Towards integrating plot and character for interactive drama. *Socially Intelligent Agents*, 3:221–228, 2002. 20, 21

[Mur01] W Murch. *In the Blink of an Eye: A Perspective on Film Editing*. new world (for sure) Part 5. Silman-James Press, 2001. 92

[MWS+15] Billal Merabti, Hui-Yin Wu, Cunka Bassirou Sanokho, Quentin Galvane, Christophe Lino, and Marc Christie. Insight : An annotation tool and format for film analysis. In *Eurographics Workshop on Intelligent Cinematography and Editing, May 2015, Zurich, Switzerland*, page 1, 2015. 26, 28, 79

[NM05] Mark J. Nelson and Michael Mateas. Search-Based Drama Management in the Interactive Fiction Anchorhead. *Proceedings of the First Annual Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE2005)*, pages 99–104, 2005. 20, 21, 23, 25, 42

[NPAI06] Michael Nischt, Helmut Prendinger, Elisabeth André, and Mitsuru Ishizuka. MPML3D: A Reactive Framework for the Multimodal Presentation Markup Language. *Intelligent Virtual Agents*, 62(1):218–229, 2006. 51

[OR14] Brian O'Neill and Mark O. Riedl. Dramatis: A Computational Model of Suspense. *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, 2014. 55

[PCC10] Julie Porteous, Marc Cavazza, and Fred Charles. Narrative generation through characters' point of view. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS2010)*, pages 1297–1304. International Foundation for Autonomous Agents and Multiagent Systems, 2010. 20

[Pea09] Karen Pearlman. *Cutting Rhythms: Shaping the Film Edit*. Elsevier Inc., 2009. 17, 70

[Pro68] Vladimir Propp. *Morphology of the Folktale*. University of Texas Press, 2 edition, 1968. 45

[PTCC11] Julie Porteous, Jonathan Teutenberg, Fred Charles, and Marc Cavazza. Controlling Narrative Time in Interactive Storytelling. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS2011)*, volume 2, pages 1–8, 2011. 20, 24, 25

[RBLA13] Rémi Ronfard, Laurent Boiron, Inria Ljk, and Prior Art. The Prose Storyboard Language. In *AAAI Workshop on Intelligent Cinematography and Editing*, 2013. 26, 35, 38, 71, 92

[Rie09] Mark O. Riedl. Incorporating Authorial Intent into Generative Narrative Systems. In Sandy Louchart, David Roberts, and Manish Mehta, editors, *AAAI Spring Symposium: Intelligent Narrative*, pages 91–94. AAAI Press, 2009. 23, 25

[RL08] Mark O. Riedl and C Leon. Toward vignette-based story generation for drama management systems. *Workshop on Intelligent Technologies for Interactive Entertainment (INTETAIN)*, 2008. 20

[RLAR11] Mark O. Riedl, Boyang Li, Hua Ai, and Ashwin Ram. Robust and Authorable Multiplayer Storytelling Experiences. *Proceedings of the Seventh International Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 189–194, 2011. 23, 25, 42, 46

[RSS05] Zeeshan Rasheed, Yaser Sheikh, and Mubarak Shah. On the use of computable features for film classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(1):52–63, 2005. 27

[RU14] Roberto Ranon and Tommaso Urli. Improving the efficiency of viewpoint composition. *IEEE Transactions on Visualization and Computer Graphics*, 20(5):795–807, 2014. 30

[Rya06] Marie Laure Ryan. *Avatars of story*. University of Minnesota Press, 2006. 25

[SBA+15] M. Svanera, S. Benini, N. Adami, R. Leonardi, and A. B. Kovács. Over-the-shoulder shot detection in art films. In *International Workshop on Content-Based Multimedia Indexing*, volume 2015-July, 2015. 27

[SDM+14] Cunka Bassirou Sanokho, Clement Desoche, Billal Merabti, Tsai-Yen Li, and Marc Christie. Camera Motion Graphs. *Proceedings of ACM SIGGRAPH Eurographics Symposium on Computer Animation*, pages 177–188, 2014. 26

[SGKB13] Alexander Shoulson, Max L. Gilbert, Mubbasir Kapadia, and Norman I. Badler. An Event-Centric Planning Approach for Dynamic Real-Time Narrative. In *Proceedings of ACM SIGGRAPH Conference of Motion in Games 2013 (MIG2013)*, pages 99–108, Dubin, 2013. ACM Press. 20, 24, 25

[Smi97] Greg M. Smith. Local Emotions, Global Moods, and Film Structure. In *Passionate Views: Film, Cognition, and Emotion*, pages 103–126. 1997. 15

[Smi12] Tim J. Smith. The Attentional Theory of Cinematic Continuity. *Projections*, 6:1–27, 2012. 15

[SMP09] Mei Si, Stacy Marsella, and David V Pynadath. Directorial Control in a Decision-Theoretic Framework for Interactive Narrative. In *Interactive Storytelling.Second Joint International Conference on Interactive Digital Storytelling, ICIDS 2009, Guimarães, Portugal, December 9-11, 2009. Proceedings*, pages 221–233. Springer Berlin Heidelberg, 2009. 20, 22

[SW08] Han Sloetjes and Peter Wittenburg. Annotation by category: ELAN and ISO DCR. *LREC. European Language Resources Association*, 2008. 26

[SWM06] Ulrike Spierling, Sebastian A. Weiß, and Wolfgang Müller. Towards Accessible Authoring Tools for Interactive Storytelling. *Technologies for Interactive Digital Storytelling and Entertainment - Lecture Notes in Computer Science*, 4326:169–180, 2006. 21, 23, 25

[TB09] Roy Thompson and Christopher J Bowen. *Grammar of the Shot*. Focal Press, 2009. 15, 17, 73

[TBN00] Bill Tomlinson, Bruce Blumberg, and Delphine Nain. Expressive Autonomous Cinematography for Interactive Virtual Environments. In *Proceedings of the Fourth International Conference on Autonomous Agents (AGENTS'00)*, pages 317–324, New York, NY, USA, 2000. ACM. 38

[Tse13] Chiaoi Tseng. *Cohesion in Film: Tracking Film Elements*. Palgrave Macmillan, Basingstoke, 2013. 14

[Tur89] Maureen Turim. *Flashbacks in Film: Memory and History*. Routledge, 1989. 24, 55, 59

[TZ04] Wallapak Tavanapong and Junyu Zhou. Shot Clustering Techniques for Story Browsing. *IEEE Transactions on Multimedia*, 6(4):517–527, 2004. 27, 71

[WABBY15] David R. Winer, Adam A. Amos-Binks, Camille Barot, and R. Michael Young. Good Timing for Computational Models of Narrative Discourse. *6th Workshop on Computational Models of Narrative (CMN 2015)*, 45:152–156, 2015. 24

[WC03] Jihua Wang and Tat-Seng Chua. A cinematic-based framework for scene boundary detection in video. *The Visual Computer*, 19(5):329–341, 2003. 27, 71

[Wey97] Peter William Weyhrauch. *Guiding Interactive Drama*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1997. 21, 23, 25, 42, 43

[WY14] Stephen G. Ware and R. Michael Young. Glaive: A State-Space Narrative Planner Supporting Intentionality and Conflict. In *Proceedings of the Tenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE2014)*, pages 80–86, Raleigh, North Carolina, USA, 2014. 20, 22

[Zet07] Herbert Zettl. *Sight, sound, motion: Applied media aesthetics*. Wadsworth Publishing Company, 2007. 15, 16, 17, 72, 73

Film Bibliography

[Cin98] New Line Cinema. American History X, 1998.

[Cin01] New Line Cinema. The Lord of the Rings: The Fellowship of the Ring, 2001.

[Dis02] Media Asia Distribution. Infernal Affairs, 2002.

[Fea05] Focus Features. The Constant Gardener, 2005.

[Fil94] Miramax Films. Pulp Fiction, 1994.

[Fil12] Lionsgate Films. The Hunger Games, 2012.

[PEA66] PEA. The Good, The Bad and The Ugly, 1966.

[Pic72] Paramount Pictures. The Godfather, 1972.

[Pic80] Warner Bros. Pictures. The Shining, 1980.

[Pic85] Universal Pictures. Back to the Future, 1985.

[Pic94a] Columbia Pictures. Shawshank Redemption, 1994.

[Pic94b] Paramount Pictures. Forrest Gump, 1994.

[Pic97] Columbia Pictures. Gattaca, 1997.

[Pic98] Buena Vista Pictures. Armageddon, 1998.

[Pic03] Columbia Pictures. Big Fish, 2003.

[Pic06] Warner Bros. Pictures. V is for Vendetta, 2006.

[Pic08] Paramount Pictures. The Curious Case of Benjamin Button, 2008.

[Pic09] Warner Bros. Pictures. Invictus, 2009.

[Pic11] Walt Disney Studios Motion Pictures. The Help, 2011.

[Pic15] Warner Bros. Pictures. Jupiter Ascending, 2015.

[Sho08] Shochiko. Departures, 2008.

Publications

[LWS+14a] Pei-Chun Lai, Hui-Yin Wu, Cunka Bassirou Sanokho, Marc Christie, and Tsai-Yen Li. A Pattern-based Tool for Creating Virtual Cinematography in Interactive Storytelling. In *Proceedings of 12th International Symposium on Smart Graphics, Springer LNCS*, Taipei, 2014.

[LWS+14b] Pei-Chun Lai, Hui-Yin Wu, Cunka Bassirou Sanokho, Marc Christie, and Tsai-Yen Li. An Experimental Platform for Customized Cinematography in Interactive Storytelling. In *Proceedings of Computer Graphics Workshop 2014 (Best Paper Award)*, Taipei, 2014.

[MWS+15] Billal Merabti, Hui-Yin Wu, Cunka Bassirou Sanokho, Quentin Galvane, Christophe Lino, and Marc Christie. Insight : An annotation tool and format for film analysis. In *Proceedings of Eurographics Workshop on Intelligent Cinematography and Editing, May 2015, Zurich, Switzerland*, page 1, 2015.

[WC15a] Hui-Yin Wu and Marc Christie. Placing Invisible Cameras: Conversation Between Cognitive Science and Automated Virtual Cinematography. In *Conference on Cognitive Science and Moving Images 2015 (SCSMI2015)*, London, 2015.

[WC15b] Hui-Yin Wu and Marc Christie. Stylistic Patterns for Generating Cinematographic Sequences. In *Proceedings of Eurographics Workshop on Intelligent Cinematography and Editing*, 2015.

[WC16a] Hui-Yin Wu and Marc Christie. Analysing Cinematography with Embedded Constrained Patterns. In *Proceeding of Eurographics Workshop on Intelligent Cinematography and Editing*, 2016.

[WC16b] Hui-Yin Wu and Marc Christie. Detecting Cinematographic Patterns. In *Conference on Cognitive Science and Moving Images 2016 (SCSMI2016)*, Ithaca, 2016.

[WC16c] Hui-Yin Wu and Marc Christie. Using Montage Patterns for Smart Cinematographic Editing. In *Proceedings of TAICHI 2016*, Taipei, Taiwan, 2016.

[WCL13a] Hui-Yin Wu, Marc Christie, and Tsai-Yen Li. Shaping Interactive Stories in 3D Environments. In *ACM SIGGRAPH Conference of Motion in Games 2013 (MIG2013)*, Dublin, Ireland, 2013.

[WCL13b] Hui-Yin Wu, Marc Christie, and Tsai-Yen Li. Stories Animated : A Framework for Personalized Interactive Narratives using Filtering of Story Characteristics. In *Proceedings of International Conference on Computer Animation and Social Agents*, Istanbul, Turkey, 2013.

[WLC15] Hui-Yin Wu, Tsai-Yen Li, and Marc Christie. Logic Control for Story Graphs in 3D Game Narratives. In *Proceedings of International Symposium on Smart Graphics*, 2015.

[WWLC15] Jou-Wen Wang, Hui-Yin Wu, Tsai-Yen Li, and Marc Christie. Design of Intelligent Editing System for Computer Animation Directors. In *Proceedings of Computer Graphics Workshop 2015*, Taipei, 2015.

[WYC16] Hui-Yin Wu, Michael Young, and Marc Christie. A Cognitive-Based Model of Flashbacks for Computational Narratives. In *Proceedings of Twelfth AAAI Conference of Artificial Intelligence and Interactive Digital Entertainment (AIIDE2016)*, Burlingame, USA, 2016.