Film Editing for Third Person Games and Machinima

Marc Christie, Christophe Lino, Rémi Ronfard

Film Editing for Third Person Games and Machinima

Marc Christie
INRIA/Université de Rennes 1, France
marc.christie@inria.fr

Christophe Lino
INRIA/IRISA, France
christophe.lino@inria.fr

Remi Ronfard
INRIA/LJK/Université de Grenoble, France
remi.ronfard@inria.fr

ABSTRACT

Generating content for third person games and machinima requires knowledge in cinematography (camera placement, framing, lighting) and editing (cutting between cameras). In existing systems, such knowledge either comes from the final user (machinima) or from a database of precompiled solutions (third person games). In this paper, we present a system that can make decisions about editing and automatically generate a grammatically correct movie during game interaction. While previous work described techniques for choosing camera setups and generating virtual footage, most of them do not propose general computational frameworks to express variations in directorial styles. We introduce a framework for generating a well-edited movie from such footage, based on film grammar. Our system computes a sequence of *shots* by simultaneously choosing which camera to use, when to cut in and out of the shot, and which camera to cut to. We cast film editing as a cost minimization problem in the space of possible shot sequences and provide an efficient search algorithm for computing the edit with minimal latency. In contrast to previous work, our method precisely examines all possible edit points and relies on a well-founded model to account for shot duration.

Keywords

Camera Planning, Automatic Film Editing, Film Grammar, Optimization Methods

1. INTRODUCTION

In many applications, it is useful to present 3D animation in a cinematic style, which means selecting appropriate cameras and appropriate inter-cutting between cameras to properly convey a narrative. In third-person games and machinima, cameraworks and editing are key elements to convey dimensions such as mood and tension, which in turn provide a greater feeling of immersion through the re-use of well-known cinematic techniques. This creative task requires both a strong knowledge of these techniques and the ability to handle complex animation and editing tools.

Game engines provide little support for cinematography and film editing techniques. Even with advanced machinima tools such as Moviestorm [1] and iClone [2], the control of cameras and cross-cutting remains a complex task (though Moviestorm provides a number of convenient control features for screen composition). One exception is the Magicam option in the online machinima system Text-to-Movie by Xtranormal [3] which automatically chooses cameras and cuts in two-person dialog scenes. Yet the system provides limited flexibility in choosing editing or camera styles.

This paper proposes a general computational framework which provides the user with an interactive process to generate a cinematic editing of a 3D animation. Our framework proposes scores for shots and cuts, independently of the film idioms used to generate them. The score for a shot is based on the Hitchcock principle of showing action from the best angle [11] and the score for a transition between shots is based on the working practices of film and television [13, 14]. In contrast to related work, we account for a precise enforcement of pacing (rhythm at which cuts are performed) and study in detail the best moment to perform cuts. This framework has been implemented as part of a system for controlling multiple virtual cameras in real time while the scene animation is computed, and evaluating the footage generated from each camera against a mathematical model of film grammar.

We cast the problem of film editing as selecting a path in time through a collection of takes (a take is a continuous sequence of images from a given camera) and precisely deciding when to cut in and out the takes. We propose an algorithm suitable for online editing which uses an efficient best-first search technique. The algorithm relies on short-term anticipation to improve quality in cuts and produce movies consistent with the rules of cinematography and editing, including shot composition, continuity editing and pacing.

Our approach work in three succesive steps. We first procedurally compute a collection of takes from the events occurring in the 3D animation (the collection of takes is large enough to make the search problem strongly combinatorial

[1] www.moviestorm.co.uk
[2] www.reallusion.com/iclone
[3] http://www.xtranormal.com

by generating many different camera configurations – 80 in average per action). All takes are cut into fragments of duration Δt and individually evaluated to establish their fragment quality. In a second step, a graph is constructed (referred to as the *editing graph*), in which nodes represent a fragment of a take, and arc represent either *cut-arcs* or *shot-arcs*. A cut-arc represents a cut from a take i at time t to a take $j \neq i$ at time $t + \delta t$. A shot-arc represents continuity (i.e. no cut) in a take i between time t and time $t + \Delta t$. In a final step, a search process is performed through this graph to compute a traversal of the animation sequence.

The paper is organized as follows. First, we review related work and compare it to our method. Then we introduce a cost function for shots, transitions and pacings, illustrated-with many examples. Then, we provide a fast method for computing the minimum-cost edit in near-real-time. Finally, we present some results and provide directions for future work.

2. RELATED WORK

In one of the first contributions to automated camera planning, He et al. [6] proposed hierarchical state machines (HFSM) for camera placement and cross-cutting between cameras. Their approach assumes a finite number of *film idioms* suitable for prototypical situations in the animated scene. The higher levels in the HFSM are responsible for choosing the appropriate film idioms and camera setups, while the lower levels are responsible for cutting between the cameras in each setup. Film idioms can be effective in cases where the configuration space for actors and their actions can be enumerated. But even in such case, they tend to be repetitive and predictable, and attract attention away from the immersive experience of the game or the story. If the configuration space becomes large, as in scenarios with multiple stages and actors, the cost of encoding every possible idiom becomes too large.

For simple scenes with only one or two actors, film idioms provide a sensible solution because (1) the full vocabulary of possible shots is

As noted by many researchers and film analysts, a story can be broken down into dramatic beats, i.e. primitive actions. Dramatic beats typically involve only one or two actors at any given time. Each beat defines a line of action around which a film idiom can be defined. That approach was taken by Lino et al [7] with good results, but film idioms cannot guarantee that the resulting movie will be adequate, because secondary actors (not part in the primary action) may now interfere with the primary actors and furthermore create misleading shot transitions.

In this paper, we propose to combine a film idiom approach similar to [7] with an introspection mechanism that allows to alleviate this problem by generating a large number of candidate idioms, and computing a score for the shots and cuts proposed by each idiom. That strategy allows to detect poor shot composition and transition choices and combine candidate idioms into a single, consistent editing solution. Furthermore, we introduce novel constraints for controlling the pacing of the editing.

Our work can be closely related to the Virtual Director of Assa et al [1] but in a very different context. Their method focusses on showing the physical motion present in the scene, rather than its narrative content. As a result, their criteria for choosing views is based on the correlation between motion in the 3D scene motion and motion in each view. To control shot duration, the authors introduce an *accumulated view erosion* function resulting in a decay in the score of each view after a given amount of time but is not consistent with previous work in film aesthetics.

Our approach is also related to the Cambot machinima system of Elson and Riedl [5]. Similar to our system, Cambot receives a sequence of *narrative goals* from an external narrative engine. The authors also cast the problem of film editing as a shortest-path problem in a graph of candidate film takes. The score of a shot sequence is computed from user-provided aesthetic constraints. As an offline system, Cambot computes a global solution given the entire script, using a combination of breadth-first and dynamic programming techniques. We improve on this system in several respects. First, we introduce a set of generic aesthetic constraints consistent with established working cinematic practices [14] and film editing [13] and provide a detailed computational model implementing such constraints. Second, we place constraints on shot durations, allowing to more precisely set the *in* and *out* cutting point for each shot. Such constraints make the problem harder to solve with dynamic programming techniques. Third, we propose a fast algorithm allowing to find a suboptimal solution for both online and offline situations.

3. FILM GRAMMAR RULES

Our approach to film editing consists in enumerating all possible combinations of camera choices and evaluating them with a cost function. In this section, we review the components of our cost function. We make the assumption that the cost function is additive, i.e. the cost of a sequence is a simple sum over succesive frames in the sequence. The cost per fragment (a fragment is a part of a take of duration Δt) is evaluated as a weighted sum of all violations of the rules of frame composition. And similarly, the cost of a cut is evaluated as a weighted sum of all violations of the rules of editing. In this simplified scheme, the rules of fim grammar are directly translated into a global cost function.

Thus the cost associated with shot i at time t is given by $C^{SHOT}(i)$, and the cost associated with a cut from shot i to shot j at time t is given by $C^{CUT}(i,j)$:

$$C^{SHOT}(i) = \sum_k w_k cost_k(i,t)$$
$$C^{CUT}(i,j) = \sum_l w_l cost_l(i,j,t)$$

We further assume that the cost of a complete sequence $s(t) \in [1,M],\ t \in [1,N]$ of N fragments using M takes is the sum of the costs for all of its fragments and cuts :

$$C(s) = \sum_t \left(\sum_k w_k C_k^{SHOT}(s(t),t) + \sum_l w_l C_l^{CUT}(s(t),s(t+1),t) \right)$$

Note that we have made a number of strong assumptions. To fairly represent the editing process, the scoring of next shots

and cuts should obviously depend on the past history of all previous shots. At this step we only consider the previous shot. Though restrictive, this hypothesis enables an efficient search process which can run online (directly selecting the best view and cut as actions occur) or offline.

A last term we add in the evaluation of a sequence is a score for the pacing of cuts and the duration of shots, which are important factors in the rhythmic perception of films [10]. When cutting from camera i to camera j at time t, this additional term takes into account the previous cutting time $u(t)$ to estimate if the duration $t - u(t)$ of the closing shot is right. The additional term can be written as a sum over all cuts:

$$C^{PACE} = \sum_{s(t) \neq s(t+1)} \left(C^{PACE}(t - u(t)) \right)$$

3.1 Shot composition

The scoring of a fragment is computed as a weighted sum over scores on action, visibility, and composition of characters on the screen.

3.1.1 Action

The Action term measures the amount of the scene action which is missed in the given fragment. We define a cost table $A[a, s, p]$ which specifies the effectiveness of a shot size s and a profile angle p conveying an action a. For example, one can prefer framing a character speaking using a medium close-up from a front viewpoint. The computation of the Action term over fragment k is expressed with a weighted sum over all actions a occuring during this fragment. The weighting is expressed by the importance $imp(a)$ of an action in the scene ($type(a)$ represents the type of the action). $s_{a,k}$ represents the size of the shot for main character involved in action k and $p_{a,k}$ profil angle of main character involved in action k.

$$C^{ACTION}(k) = \sum_{a} imp(a) A[type(a), s_{a,k}, p_{a,k}]$$

With a suitable choice of the $A[a, s, p]$ coefficients, the action cost term can serve to enforce the Hitchcock principle, which states that the screen size of actors should be proportional to their importance in the scene [11]. In our implementation, we tabulate the values in $A[a, s, p]$ with four action types: facial actions (speak, listen, watch, turn, nod), hand actions (lift, pour, drink, eat), feet actions (walk, sit down, stand up) and no action (idle). Figure 1 illustrates the preferences for shot sizes and profiles for the special case of two hand actions : pour and drink.

3.1.2 Visibility

A good shot is a shot which maximises the visibility of the actions performed by the protagonists. To account for this, we express a visibility cost C^{VIZ} which measures the overlapping area between projected bounding boxes of protagonists:

$$C^{VIZ} = \sum_{i,j} area(box_i \cap box_j)$$

where box_i is the projection of the bounding box of actor i on the screen. Figure 2 displays three shots with increasing visibility scores in a scene with two actors.

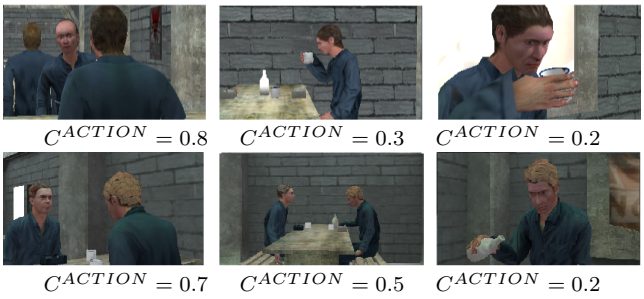


Figure 1: Action scores. Top: Action Drink. Bottom: Action Pour. Left: Poor. Middle: Better. Left: Best.

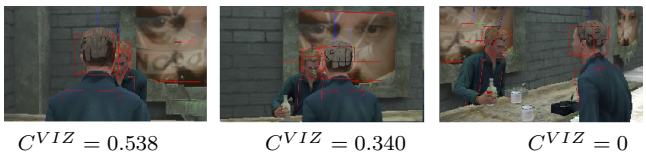


Figure 2: Visibility scores. Left: Poor. Middle: Better. Right: Best.

3.1.3 Look room

Following common practice, we favor shot composition where each actor is given enough screen space relative to the image frame, especially in the direction where he is looking. This is easily computed with the following cost function o

$$C^{LOOK} = \sum_{i} \delta(e_i + \lambda(s_i) g_i)$$

where e_i is the screen position for the middle of the actor's eyes, g_i is the screen vector of the actor's gaze direction, λ is a scalar function of the actor's screen size s_i and δ is the indicator function for being on-screen (within the image frame).

3.2 Shot transitions

A transition between shots always causes discontinuity. The art of the editor is to choose transitions minimally intrusive transitions by selecting appropriate shots and moments for cutting. A complete theory of what makes a cut intrusive is not currently available, although there has been work in film aesthetics [8, 9, 2] and cognitive psychology [4, 12, 15] on the subject. For our purpose, we have found useful to compute the cost of a cut as a sum of terms measuring typical grammatical errors according to the classical style of continuity editing (continuities in the screen positions, gaze directions and motion directions of actors).

3.2.1 Screen continuity

The score of screen continuity in transitions prevents actors who appear in two successive shots to jump around the screen. Because the actor's eyes are the most important center of attention, we favor transitions which maintain the actor's eyes at the same screen location. We weight this term with the screen size of actors, so that continuity in the

$C^{SCREEN} = 0.596$ $C^{SCREEN} = 0$

Figure 3: Screen continuity scores. Left: cutting from left to right has the leftmost character jumping to the screen center resulting in a poor cut. Right: keeping the center character in the same screen location produces a smooth cut.

$C^{GAZE} = 1$ $C^{GAZE} = 0$

Figure 4: Gaze continuity. Left: the gaze direction of the main character changes between the left and right images, resulting in a poor cut. Right: keeping the gaze directions consistent results in a better cut.

foreground receives a larger reward than in the background. As a result, screen continuity is enforced by minimizing:

$$C^{SCREEN} = \sum_{i \in S_1 \cap S_2} a_{ij} \phi(|e_{i1} - e_{i2}|)$$

where $S_1 \cap S_2$ represents the set of actors shared between shots S_1 and S_2, e_{ik} is the screen location of the middle of the eyes of actor i in image k and ϕ is a non-linear function of the screen distance such as a sigmoid or threshold function. Each term in the sum is weighted with the average screen size $a_i = (s_{i1} + s_{i2})$ of the actor in the two images. Figure 3 shows examples of two cuts with increasing screen continuity scores.

3.2.2 Gaze continuity

Another important focus of attention when watching a movie is the gaze direction of actors. We propose a cost function that penalizes camera changes that cause apparent reversals in the actor's gaze directions. This cost is defined by:

$$C^{GAZE} = \sum_{i \in S_1 \cap S_2} a_i \delta(sign(xg_{i1}) - sign(xg_{i2}))$$

where xg_{ik} is the x coordinate of the gaze direction for actor i in image k and δ is the Kroneker symbol. Figure 4 shows the examples of two cuts with increasing gaze continuity scores.

3.2.3 Motion continuity

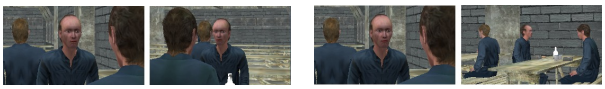
Motion direction of actors in two successive shots represents another focus of attention. The corresponding transition cost penalizes camera changes that cause apparent reversals in the actor's motion, defined as:

$$C^{MOTION} = \sum_{i \in S_1 \cap S_2} a_i \delta(sign(xm_{i1}) - sign(xm_{i2}))$$

where xm_{ik} is the screen motion of the actor's eyes in image k measured in the x direction, and δ is the Kroneker symbol.

$C^{MOTION} = 1$ $C^{MOTION} = 0$

Figure 5: Motion continuity score. Left: the main character's head is oriented differently in left and right shots, resulting in a poor cut. Right: keeping the head orientations consistent results in a better cut.



$C^{ORDER} = 1$ $C^{ORDER} = 0$

Figure 6: Left to right ordering scores. Left: the foremost character moves from the rightmost position to the leftmost position, resulting in a poor cut. Right: keeping the relative ordering of all characters results in a better cut.

Figure 5 shows examples of two cuts with increasing motion continuity scores.

3.2.4 Left to right ordering

The left to right ordering of actors is another important factor for ensuring screen continuity. Actors whose relative screen position are reversed appear to be jumping around, which attracts attention to the cut. We penalize such situations with the following penalty cost:

$$C^{ORDER} = \sum_{i,j \in S_1 \cap S_2} a_{ij} \delta(sign(xe_{i1} - xe_{j1}) - sign(xe_{i1} - xe_{j1}))$$

where xe_{ik} is the x coordinate of the actor's eyes in image k and δ is the Kroneker symbol. As in previous cases, each term is weighted with the average screen size $a_{ij} = \sum_{k=1}^{2}(s_{ik} + s_{jk})$ of the actors in the two images. Figure 6 shows examples of two cuts with increasing left to right ordering scores.

3.2.5 Profile and size change

A type of transition which should be avoided is when the apparent size and profile angle of an actor remains the same in a sequence of two shots. The cut is interpreted as a sudden change in the actor's pose, also known as a jump cut. We model this transition cost as follows:

$$C^{JUMP} = \sum_{i \in S_1 \cap S_2} a_i \delta(s_{i1} - s_{i2}) \delta(|\theta_{i1} - \theta_{i2}| \leq 30)$$

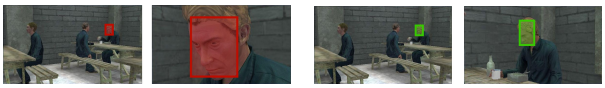
where θ_{ik} is the profile angle of actor i in image k. Thus, transitions on an actor with the same screen size and profile in both shots receive a penalty (see Figure 7).

We also penalize excessive changes in shot sizes, i.e. between a long shot and a close-up, because they make it more difficult for the viewer to recognize correspondences between actors in the two shots. This is enforced with the simple cost term

$$C^{SIZE} = \sum_{i \in S_1 \cap S_2} \phi(s_{i1} - s_{i2})$$

$C^{JUMP} = 1$ $C^{JUMP} = 0$

Figure 7: Jump cut score: transitions with insuffisant change in size or orientation wrt actors do not make good cuts (left) while transitions with significant changes make good cuts (right).



$C^{SIZE} = 1$ $C^{SIZE} = 0$

Figure 8: Size continuity. Left: cutting directly from medium close-up to long shot results in a poor cut (the change in shot range is too strong). Right: cutting to a medium shot results in a smooth cut.

where ϕ is a non-linear function of the size difference, such as a sigmoid or threshold function (see Figure 8).

3.2.6 *Action line continuity*

In addition to preserving the continuity of actors between successive shots, it is equally important to preserve the direction of the *line of action* between actors (usually the world line between their eyes). Thus, when cutting between actors A and B, the direction of the (AB) line needs to be preserved, even if the two actors are not simultaneously visible on-screen. This typical case occurs when actors are looking at each other: the cut is good when gaze directions cross each other in the successive shots, which we model as:

$$C^{LINE} = \sum_{i \in S_1, j \in S_2} a_{ij} \delta(sign(xe_{i1} - xej1) - sign(xe_{i2} - xej2))$$

where xe_{ik} is x coordinate of the actor's eyes in image k and δ is the Kroneker symbol. Figure 9 shows examples of two cuts with increasing action line continuity scores.

3.3 Shot durations

In his book *History of Film Style and Technology*, Barry Salt asserts that shot durations in movies generally follow log-normal distributions, similar to sentence lengths in natural language[10] and proposes to use the parameters of the log-normal distribution as a signature of film editing styles.

Because shot durations are such an important element of film editing style, we introduce a duration cost per shot, measuring the deviation from a log normal law, where d is the duration of the shot, μ and σ are resp. the mean and the standard deviation of the log normal law

$$C^{PACE}(d) = \frac{(\log(d) - \mu)^2}{2\sigma^2}$$

The values for μ andσ can be left under the control of the user, be inferred from the pacing of actions, or be inferred from example movie scenes.

3.4 Feature weight selection

$C^{LINE} = 1$ $C^{LINE} = 0$

Figure 9: Line of action score. Top: In the left image, the gaze direction of the main actor defines the line of action. Bottom: the gaze direction changes because the camera has crossed the line, resulting in a poor cut. Bottom: keeping the line of action result in a smooth cut

Putting everything together, we now build the cost function for an arbitrary sequence of key shots, as a weighted sum of all its features. We choose the weights with a linear discriminant analysis (LDA) of shots and cuts labeled as 'grammatical' and 'ungrammatical'. LDA finds linear combinations of the cost features that minimize the overlap of the transformed distributions [3]. We separately set the weights with a first LDA for the shot terms using shot examples, then for the cut terms using cut examples. Finally, we arbitrarily set the relative weights W^{SHOT}, W^{CUT} and W^{PACE} to 2, 1 and 1, respectively representing the weighting of C^{SHOT}, C^{CUT} and C^{PACE}

4. BEST-FIRST SEARCH

The computation of an optimal sequence of shots consists in searching the path of least cost in our editing graph. Algorithms to solve these classes of problems (dynamic programming, A*-based algorithms) remain computationally complex in practice. Additionally, the problem we address displays a very large collection of solutions which yield similar costs (e.g. consider the case of symmetric viewpoints around an actor), and for which an optimal computation seems unnecessary.

We propose a sub-optimal best-first search through the editing graph. Given that the quality of the overall sequence strongly depends on the output of this search process, we propose to enhance the search by proposing an *informed* best-first search. This informed search uses a sliding observation window over the next fragments to locally reason on the best moment for a transition.

At a given depth in the search process (i.e. advancement in time over the fragments), a critical decision needs to be made whether to stay within the current shot, or to perform a transition to another shot. To inform this decision, we rely on this observation window over the next w fragments. We study within this window the best transition to be performed, given the knowledge of the shots to come. If the best transition occurs immediately, the cut is performed. If the best transition occurs later in the observation window, we shift the window a fragment ahead and start the process over.

To compute the best moment for a transition inside the observation window we use an incremental scanning process. The process is illustrated in Figure 10. Given the current shot is c, for a given time t in the observation window and for each shot $i \neq c$, we compute the cost C^{CUT} of a possible transition from shot c to shot i, and we compare it to the cost C^{NOCUT} of staying in the current shot. If the

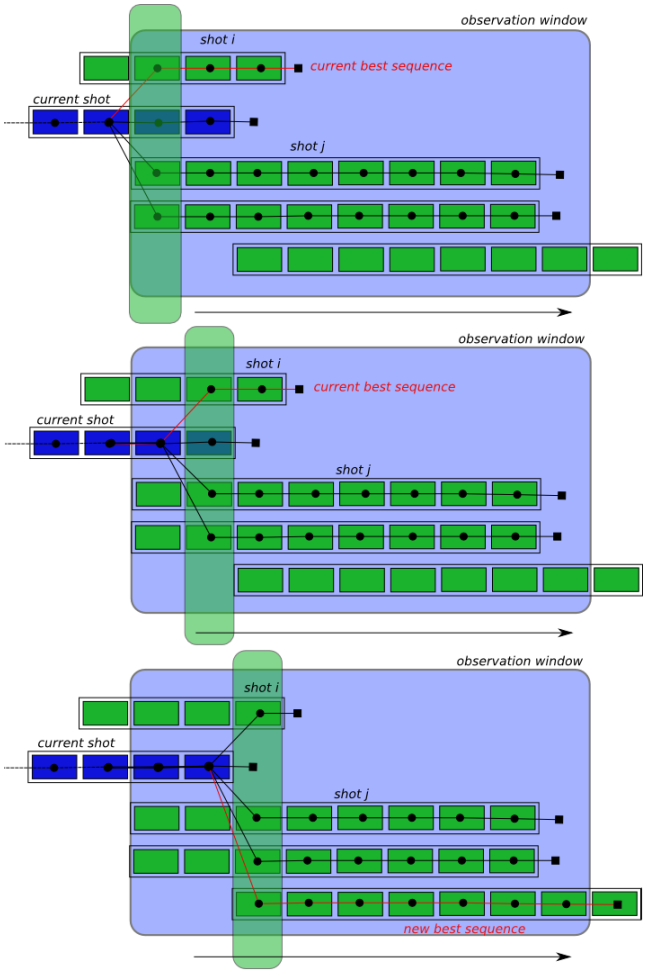


Figure 10: Searching for the optimal transition in the observation window to decide whether to cut or stay in the shot. Each green/blue rectange represents a fragment of a shot. On the top image, the current optimal shot sequence is drawn in red and recommends to cut at time t. A scanning process over the observation window (window in light blue) is performed to seek for a possible better moment. On the bottom image, a sequence with a better cost is found (displayed in red). The observation window is then shifted ahead one fragment in time and the process starts over.

cost of staying in the current shot c is the minimal cost (ie $C^{NOCUT}(c) \leq min_i C^{CUT}(c,i)$), we extend its duration by Δt and the observation window is shifted a fragment ahead. If there exists a shot i such that $C^{CUT}(c,i) < C^{NOCUT}(c)$ at time t, we need to know whether to cut at the current time t to shot i, or to wait for a better moment. To implement this, the process explores the successive fragments at $t+\Delta t, t+2\Delta t, ..t+w\Delta t$ in the observation window until a cost lower than $C_{CUT}(c,i)$ is found. In such case, the best cut occurs later and the observation window can be shifted a fragment ahead. Otherwise, t represents the best moment for a transition and a cut is performed towards shot i.

Figure 11: Snapshot of our system in action, showing the on-going actions (bottom part), the costs (middle part) and the selected shot (top part). The illustrated shot clearly displays the action (the character is eating), and respects head-rooom and lookroom., with a sufficient visibility even if another character is in the frame.

We fixed the size of the observation window to the mod m of the log normal law ($m = log(\mu)$): $w = m/\Delta t$. Indeed, the evaluation over the window only considers one possible transition between shots, which optimally occurs every m seconds. Any window size larger than the mod would provide an erroneous estimation (considering one cut when two may occur) and any size smaller would not fully consider the cost of the pacing over the next shot.

5. EXPERIMENTAL RESULTS

Experimental results have been generated using our system on the 1984 animation content and annotated scene [4]. The duration of the animation is 3 minutes, with over 70 actions occurring with overlaps.

Currently, the duration Δt of a fragment is set to $250ms$ (i.e cuts between shots may occur every $250ms$). For each action occurring in the scene, we generate between 50 shots (single-character actions) and 80 shots (two-character actions), corresponding to classical cinematic viewpoints.

Figure 11 presents a snpashot of our system in action, with details on the current action and values of all cost functions. When executed offline, our fully automatic method takes 15 seconds on the three-minute sequence in 1984. When executed online, our method works in real-time with a latency of approximately two seconds.

[4]kindly made available by the authors of [7] at `http://www.cameracontrol.org/1984-content`

6. CONCLUSION

We have introduced a novel framework for virtual cinematography and editing which adds an evaluation function to previous approaches. Experimental results demonstrate that our approach is efficient in separating correct from incorrect shot sequences. Furthermore, we have introduced an efficient search strategy for finding the best sequence of shots from a large number of candidates generated by traditional film idioms.

In future work, we would like to extend the approach to higher-level criteria, including story advancement, repetition and rhythm and possibly even emotion. More generally, an important research issue is the discovery of film idioms and styles from examples. We believe the proposed approach makes it possible to add style-specific terms in the evaluation function and we will investigate varying the scoring function to generate movies with varying styles.

7. REFERENCES

[1] J. Assa, L. Wolf, and D. Cohen-Or. The virtual director: a correlation-based online viewing of human motion. *Comput. Graph. Forum*, 29(2):595–604, 2010.

[2] T. Berliner and D. J. Cohen. The illusion of continuity: Active perception and the classical editing system. *Journal of Film and Video*, 63(1):44–63, 2011.

[3] R. Duda, P. Hart, and D. Stork. *Pattern Classification (2nd ed.)*. Wiley, 2000.

[4] G. D'Ydewalle and M. Vanderbeeken. *Perceptual and cognitive processing of editing rules in film*, pages 129–139. Elsevier Science, 1990.

[5] D. K. Elson and M. O. Riedl. A lightweight intelligent virtual cinematography system for machinima generation. In *AI and Interactive Digital Entertainment*, 2007.

[6] L. He, M. Cohen, and D. Salesin. The virtual cinematographer: a paradigm for automatic real-time camera control and directing. In *SIGGRAPH '96*, pages 217–224, 1996.

[7] C. Lino, M. Christie, F. Lamarche, G. Schofield, and P. Olivier. A real-time cinematography system for interactive 3d environments. In *Proceedings of SCA*, 2010.

[8] W. Murch. *In the Blink of an Eye: A Perspective on Film Editing*. Silman-James Press, 2001.

[9] M. Ondaatje. *The Conversations: Walter Murch and the Art of Editing Film*. Knopf, 2004.

[10] B. Salt. *Film Style and Technology: History and Analysis (2nd edition)*. Starword, 2003.

[11] H. G. Scott and F. Truffaut. *Hitchcock-Truffaut (Revised Edition)*. Simon and Schuster, 1985.

[12] T. J. Smith. *An Attentional Theory of Continuity Editing*. PhD thesis, University of Edinburgh, 2005.

[13] R. Thompson. *Grammar of the Edit*. Focal Press, 1993.

[14] R. Thompson. *Grammar of the Shot*. Focal Press, 1998.

[15] J. M. Zacks and J. P. Magliano. *Film, Narrative, and Cognitive Neuroscience*. OxfordUniversity Press, 2010.