Applied Artificial Intelligence

ISSN: 0883-9514 (Print) 1087-6545 (Online) Journal homepage: http://www.tandfonline.com/loi/uaai20

Film sequence generation strategies for automatic intelligent video editing

Sean Butler & Alan Parkes

To cite this article: Sean Butler & Alan Parkes (1997) Film sequence generation strategies for automatic intelligent video editing, Applied Artificial Intelligence, 11:4, 367-388, DOI: 10.1080/088395197118190

To link to this article: https://doi.org/10.1080/088395197118190

Published online: 26 Nov 2010.

FILM SEQUENCE GENERATION STRATEGIES FOR AUTOMATIC INTELLIGENT VIDEO EDITING

SEAN BUTLER and ALAN PARKES
Computing Department, SECAMS, Lancaster University, Lancaster, UK

In this article, we describe an approach to generic automatic intelligent video editing, which was used in the implementation of the LIVE system. We discuss cinema theory and related systems that recombine film fragments into sequences. We introduce LIVE and describe the architecture and implementation of the system. We discuss the generic film fragment generation rules implemented within the system, with examples of input and results. Finally, we discuss the future direction of our research.

Continued reduction in storage costs, accompanied by an increase in computing power and the development of sophisticated video compression techniques, is rapidly leading to a situation where computer video retrieval and presentation is becoming an unwieldy process. One approach to reducing the size of video databases is to facilitate reuse of individual video clips. Film is not unique in being a medium where a given fragment, if taken out of context and combined with other fragments, changes its meaning. Music, Natural Language, etc., have similar properties. That shots of film can take on differing meanings according to context has been established since the early days of the cinema. In particular, experiments carried out in the early part of the twentieth century showed that even neutral material, when juxtaposed with other material, could result in different meanings being attributed to the material.

In our work, we are interested in assembling meaningful sequences of film from fragments stored in an annotated video database. The sequences are assembled to satisfy an externally generated query. The query is essentially a conceptual specification of a very simple story (i.e., event or sequence of events). From this specification, which contains no filmic directions, our system creates one or more alternative queries, each representing the realization of the simple story as a sequence of film fragments. Thus our system could be regarded as a (simplified) representation of a film editor who is given a collection of film fragments and told by someone not aware of film editing techniques to produce a given story.

Revised November 1996.
This work is supported by the EPSRC.
Address correspondence to Sean Butler, Computing Department, SECAMS, Lancaster University, Lancaster LA1 4YR, UK. E-mail: [seanlapp]@comp.lancs.ac.uk

Applied Artificial Intelligence, 11:367–388, 1997

0883-9514/97 $12.00 + .00

CINEMA THEORY: Context, Order, and Meaning

Bloch (1988) asserts that the output of film editing systems should be based on domain comprehension, that is to say that output from these systems should be correct and based upon knowledge of film editing. The definition of *correct,* as applied to films, is a major area of research in its own right. Like most "languages," film does not have a definitive specification of its syntax and semantics, but rather a loose collection of descriptive and prescriptive statements of particular effects from a wide variety of sources (Andrew, 1976; Arnheim, 1958; Balasz, 1972; Branigan, 1984; Carroll, 1980; Eisenstein, 1948, 1949; Metz, 1974; Parkes, 1989a; Pudovkin, 1968; Spottiswoode, 1955; Tudor, 1974).

Shortly after the 1917 revolution, Russian film directors Kuleshov, Pudovkin, and Eisenstein had insufficient footage to proceed with their own original works and so started reediting preshot footage. Kuleshov (ca. 1920), as reported by Pudovkin (1968), conducted the earliest experiment to highlight important aspects of film juxtaposition. Visualize five clips of film described as follows:

1. A young man walks from left to right.
2. A woman walks from right to left.
3. They meet and shake hands. The young man points.
4. A large white building is shown, with a broad flight of steps.
5. The two ascend the steps.

> The pieces, separately shot, were assembled in the order given and projected upon the screen. The spectator was presented with the pieces thus joined as one clear, uninterrupted action; a meeting of two young people, an invitation to a nearby house, and an entry into it. Every single piece, however, had been shot in a different place; for example, the young man near the GUM building, the woman near Gogol's monument, the handshake near the Bolshoi Teatr, the white house came out of an American picture (it was, in fact, The White House), and the ascent of the steps was made at St. Saviour's Cathedral. What happened as a result? Though the shooting had been done in varied locations the spectator perceived the scene as a whole (Pudovkin, 1968, pp. 88–89).

The effect most successfully illustrated by the above example Pudovkin called "creative geography." However, the consequences of the geographic interpretation of this experiment can also be applied to purely event based fictions. In other words, we can create fictional *space* with editing, and we can also create fictional *narratives.*

Obviously, the juxtaposition of several shots of disjointed locations will not in itself create the desired filmic space in the spectators mind. Further constraints must be placed on the content of the shots for a filmic space to be created. These constraints largely concern *continuity* and are determined by the real world knowledge of the director and editor.

Imagine a person leaving one room, then a cut, followed by the same person closing the door of a corridor behind them. Clearly, any sequence of shots designed to portray two rooms joined in such a way must show the character in the same state when they entered the second room as when they left the first room (or at least as closely as possible). Other constraints are also imposed, by convention, such as the 180° rule, e.g., in the above sequence, if a person leaves the room toward the left of the image then they should enter the next shot from or on the right of the image. These, and other, constraints are used to achieve *continuity editing,* which is a common approach to film editing.

Let us examine why the Kuleshov sequence of shots works as a geographic whole. Several of the elements in the above sequence (people, object, actions) occur in more than one shot. Additionally, some of the attributes (e.g., location and color) are shared by different elements across a number of shots:

1. the people
2. the color white
3. the city

Therefore only these three have continuity considerations applied to them, in the following ways:

1. The man and woman are the same actors throughout (continuity of characters). Although it is not stated in the report, one assumes that they also have the same appearance in all the shots in which they appear.
2. The steps that the couple ascend are white, as is the building, and are taken in close-up so we cannot see their actual location (continuity of shared attributes, color, concealment of unique attributes, location).
3. In many ways, cities can be similar when photographed (a certain anonymity exists, which creates a continuity of location through ambiguity of potentially distinguishing features).

So, the illusion of a contiguous space and time is created by common attributes shared by certain elements of the film content. In the Kuleshov example, these attributes are shared by the characters (man and woman), the locations (various locations in Moscow), and the objects with which they interact (the white steps and the white house).

A related result was obtained in another Kuleshov experiment. Identical shots of a famous Russian actor (Moszhukin) were juxtaposed with shots of some soup, a woman in a coffin, and a little girl. Pudovkin reported that the audiences noted the actor's ability to portray the subtle and varied emotions of hunger, sadness, and affection, despite the fact that the facial expression of the actor was exactly the same in all three sequences.

The above discussion serves to highlight the following key points:

- When a fragment of film is viewed in isolation, its meaning and content are likely to be largely unchanged on each viewing. However, when that same piece of film is combined with others in a sequence, the meaning of the fragment often changes. The meaning of the sequence cannot be determined simply as the sum of the meanings of the fragments, rather it is something greater than this.
- Despite the fact that cinema theory is regarded as an art, it defines rules and patterns that lend themselves to computational implementation, and can therefore be used as the basis of automatic intelligent video editing systems.

IMPORTANT SYSTEMS

The key factor common to systems that are relevant to our research is that they contain editing rules to support film production. In such systems, once an output goal has been identified, the system should, if necessary, apply rules to reorder or assemble the output from film fragments that were not explicitly requested (rather than supplying the best match from a database, as in "traditional" content-based media retrieval systems). Few systems currently exist that do this. However, a number of systems allow the user to request film based on content and then specify high-level editing principles manually (Chua & Ruan, 1995; Sack & Don, 1993) and, as such, are editing support tools. A third type of system supports retrieval of video in such an expressive and flexible manner that they are also editing support tools. However, the editing principle embedded in such systems is usually limited to simple juxtaposition, thus omitting familiar techniques such as inserts (Butler, 1996; Davis, 1995).

In summary, the research reviewed can be grouped into three types, concerned with

- Automatic intelligent film or video editing (Bloch, 1988; Sack, 1994; Nack, 1995; Butler, 1995)
- Film or video retrieval (Chua & Ruan, 1995; Sack & Don, 1993)
- Structure visualisation and support environments for editors (Davis, 1995; Butler, 1996)

In the following, the systems discussed have not been explicitly identified with one of the above categories, as most of the systems have aspects that belong in at least one of the categories.

Bloch's System

Bloch (1988) describes a system for generating film sequences. He makes the profound observation that film sequence generation, like natural language genera-

tion, should be based on domain comprehension and that film editing systems should aim to produce *correct* films. This aim is a major feature of our own research.

Bloch describes a nonexhaustive set of semantic slots for shots showing

- the *action* performed by the actors
- the *setting*
- the *actors*
- the direction of the *looks* of the main actors
- the *position* of the actors in the frame
- the direction and the speed of the apparent *motion* of the main actors or objects in motion

Bloch's system draws inferences concerning the "meaning" of a shot, based on the content of the semantic slots, by applying rules of filmic knowledge. These rules are adapted from techniques specified in film editing manuals. Bloch points out, however, that while manuals are useful sources of film editing techniques, the way in which they express these techniques lacks the precision required for computational implementation. Bloch's rules maintain continuity in the direction of a character's look, the position of actors, and motion, in successive shots. Bloch's system is limited by the small range of semantic slots implemented, which in turn, limits the amount of filmic knowledge that can be applied to the shots in the form of rules. Moreover, the user has to direct the output by requesting a specific number of shots. However, the scheme provides a useful foundation for automatic sequence generation. More recent systems, especially AUTEUR (Nack & Parkes, 1995), have surpassed Bloch's achievements.

IDIC

IDIC is a system used to generate video sequences from story plans by using segments from an archive of annotated video (Sack & Davis, 1994). IDIC creates trailers for *Star Trek, The Next Generation.* This constrained domain requires only simplistic sequence generation for two important reasons:

1. The narrative complexity of trailers is, as Sack and Davis correctly point out, limited. However, they do not state the full implications of these limitations for film editing. Trailers do not have to feature narrative continuity, as represented in conventional cinema and TV, as viewers appreciate a trailer as a disjoint summary of a much longer film.
2. In limiting themselves to a film database consisting solely of footage where characters wear the same outfits most of the time and where the action occurs mainly in the same set of locations, the researchers avoid a basic problem of film editing: editing for continuity.

Despite these limitations, IDIC is based on a thorough investigation of how planning techniques can be used to generate simple narratives from a constrained set of abstract narrative units, and the ways in which such narratives relate to film (or in this case, TV) fragments.

Stratification System

Aguierre-Smith and Davenport (1992) describe a system, Strata, which is a design environment for random access video. Though concentrating on the ethnographic annotation of video, they highlight the importance of context for the interpretation of a given annotation, a basic premise also adopted in our work. The annotations used in their system are keywords whose relevance occurs over an interval. They introduce the idea of multiple *partially overlapping* annotations, which we believe to be the key to effective video annotation, either for machine or human interpretation. Combinations of multiple keywords produce the layered data structure.

The system's visual representation occurs as a histogram, with the frame numbers of the video along the x axis, and the y axis consisting of a list of keywords. The result is a series of horizontal bars representing the span of relevance for each keyword, and the context for that keyword. Figure 1 shows a typical stratified representation.

Media Streams

Media Streams (Davis, 1995) is a system for retrieval and repurposing of video. However, on examining the retrieval-by-composition examples given on pages 185–193 of Davis' thesis, we see that although "Users can specify where shot boundaries should occur . . . [and] Media Streams in no way requires that the user formulate the segmentation of the query in order to retrieve/construct a video

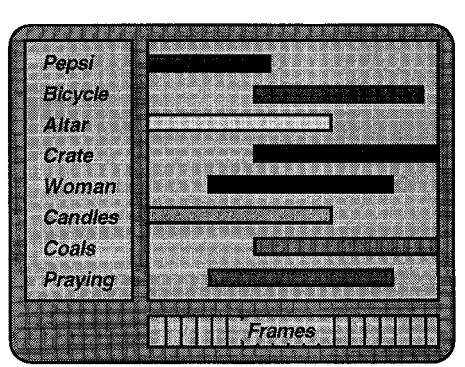

Figure 1. Impression of the stratification system.

sequence," in the examples given, the ordering of the segmentation matches the ordering specified in the query.

In the first example (A) the query asks for "a man walking away from the camera and then to the right." In the next example (B) the query asks for "a crowd making a noise with their mouths, a table is present and then an elderly man is eating." The result, in each case, is two fragments juxtaposed that indeed portray the required action, while preserving continuity.

Davis states that the two example results in case B are reaction shots, by virtue of the location of the actors. If this is the implemented method of creating reaction shots with the system, then the user must be aware that a reaction shot is (put simply) a two-part sequence and that the relationship between the shots is created by continuity of location and (say) direction of looks or other actions that suggest the same location. Strictly, the same location is not required as long as it is evident that the reacting character can see/hear the initial action. Examples C, D, E, and F described by Davis are all similar to the above, with continuity of location or action cited as the reason for the audience perceiving the resulting sequences as *reaction shots*.

The thematic sequence (G), is a further example where the ordering of the required fragments is specified explicitly in the query. Media Streams uses its semantic hierarchy to draw an inference regarding the causal relationship between fire and smoke, but otherwise the ordering of the query directly relates to the ordering of the result. Thus it appears that Media Streams does not take advantage of the true potential of editing to reorder material to realize a query, but in a way that does not explicitly reflect the ordering of that query.

We were unable to find, in Davis' thesis, any examples of reordering of material. Let us consider two cases where such reordering would be useful. The user might ask for X looking at Y, and the system may be unable to retrieve it in one shot. A system ought to be able to satisfy such a query by constructing a sequence of X looking out of a frame, followed by Y from an appropriate camera angle. In such an example, the ordering of the result can be said to differ from that specified by the user, since looking involves *concurrent* situations of actor and object, and the result shows actor and object in sequence. An alternative example of reordering arises with so-called parallel action, where the user requests events that occur simultaneously, but the system responds by interleaving the two actions (parallelism is discussed in more detail below).

The iconic language of Media Streams is an effective and expressive formalism for the annotation and retrieval of video sequences. However, it appears that any repurposing that occurs (as highlighted by Davis' own examples) does so owing to the explicit query requests of the user.

Struct

Butler and Parkes (1996) describe the Struct system, which visually displays the relationships between fragments of video during editing by a human operator.

This is done via a directly manipulatable *space-time* diagram. Types of cut such as *flashback* and *parallelism* have obvious visual shapes in the system. Figure 2 shows the interface to Struct. To our knowledge, this is the first system to visually represent film structure in this way. Struct supports the editor by enabling him or her to draw the structure of the film as he or she is constructing it. However, Struct contains no film editing rules, nor does it provide any automatic editing facilities.

Video Retrieval and Sequencing System

Chua and Ruan (1995) describe a video retrieval and sequencing system (VRSS) that supports indexing, retrieving, and sequencing of video information. As opposed to a stratified approach to content representation, the authors adopt a shot-based partitioning scheme. They justify this in terms of the excessive work involved in providing stratified annotations, and the familiarity with the shot as a unit on the part of traditional video operators.

The user of VRSS must select an editing rule, along with the concepts required in the sequence to be produced. Enabling the user to select a rule and apply it to the film content and required concepts has produced a unique tool for assisting humans to edit video. The design of VRSS implies that Chua and Ruan are obviously aware

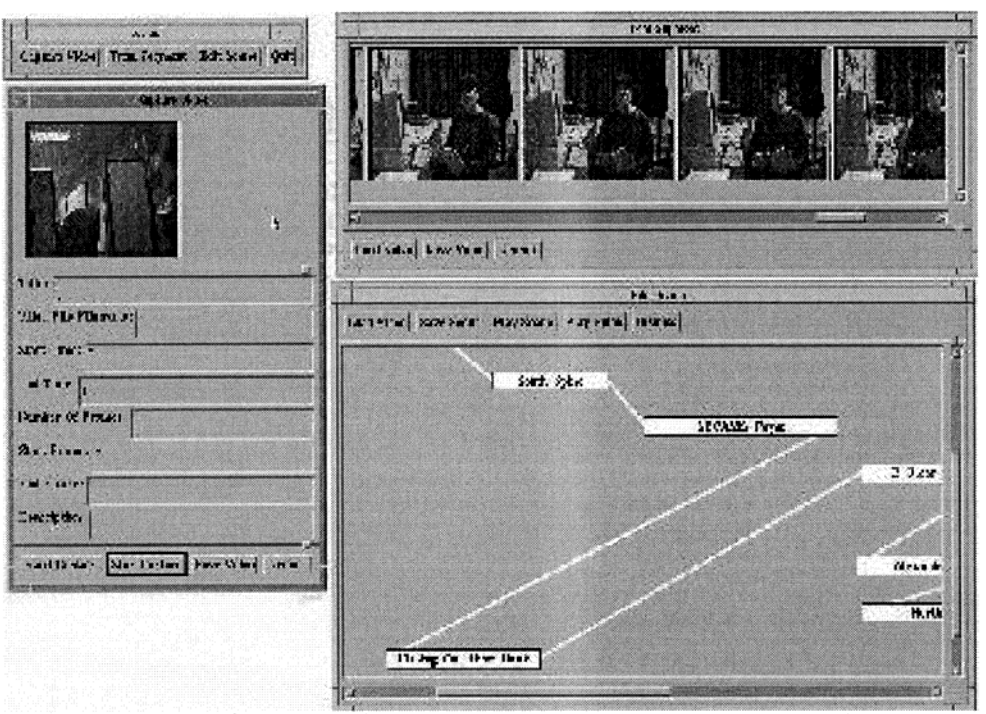

Figure 2. Struct interface presenting a flashback.

that retrieval and editing must occur together, due to the change in meaning of fragments when juxtaposed with others.

Splicer

Splicer results from the work of Sack and Don (1993) and, like Bloch's system, AUTEUR, and the LIVE system described below, features explicitly coded film editing rules. However, Splicer differs from the three systems mentioned immediately above in that the user of Splicer has to specify both the editing rule to be applied and the shots to which it is to be applied. The results are presented visually on a "video wall," which allows the user random access to the assembled sequence.

AUTEUR

Nack and Parkes (1995) describe AUTEUR, a system to construct thematic sequences from arbitrary annotated video material. At the present time, the exemplar theme is that of *humor*. Thus AUTEUR embodies strategies for film sequence generation and presentation of humorous concepts.

The annotation approach used in AUTEUR, although based upon multiple overlapping intervals, makes no effort to be compact. Such an approach allows the implementation of complex and subtle editing rules but limits the applicability of the strategies. The strategies that determine the joke structure are dependent upon filmic knowledge. This is indicated by the fact that to create *anticipation* in a joke (say), one part of the system specifically requests a *highlight* from the other.

AUTEUR successfully generates humorous sequences but only because, in its prototypical form, its video database and knowledge representations are kept to a manageable level.

LANCASTER INTELLIGENT VIDEO ENVIRONMENT

We now discuss our own research, which has resulted in the Lancaster Intelligent Video Editor (LIVE). LIVE is a system that edits film in response to an external request. It embodies a database of stored MPEG1 files, a database of video annotations, a rule base of cinematic primitives, and a set of scene creation strategies. For the video presentation there is an MPEG1 file editor and player.

Development Aims

Our aim was to produce a system to edit film taken from a database of annotated clips in order to produce a sequence matching an external request. We now present the goals we sought to achieve, and the constraints we applied, in implementing the LIVE system.

- Any person or system using LIVE to create a sequence must be able to do so with no knowledge of film editing. Thus the queries entered contain no specification of the editing techniques to be used in their film realization. This means that, for example, a story-generation system could be interfaced to LIVE, despite the fact that LIVE contains no knowledge of story generation. Conversely, the story generation system would need no knowledge of film editing techniques or scene structure. All that would be required is that both systems have common database content (e.g., if we wish the system to edit video to accompany stories about airplanes, the system must have available annotated video concerned with airplanes). To this end, a simple means-end planner is being developed into a narrative generator.
- Any annotation method used in LIVE should be cost effective in terms of time, resources and effort. To achieve this, annotations must be compact and reusable. The effort applied to annotating video for use in LIVE must be considerably cheaper than filming the sequences needed.
- The results of the system should be filmically correct. We have already touched on the issue of film correctness in the introduction and in the discussion of Bloch's work. The search for rules that generate filmically correct sequences is the single most demanding aspect of our work.
- The system should feature a flexible and extendable set of semantic slots with which people can create their annotations. Therefore any rules in the system should be capable of operating on data structures containing a minimal level of information, as well as those that are replete with slots whose significance is loosely defined.

Overview of the System

LIVE is implemented in Poplog Prolog and C on a Sparc 5, using the Sun Video Technologies XIL. There are four main components of the system.

Database of video—A set of compressed MPEG1 video files stored on hard disk.

Video annotations—A database of annotations applied to the video by a human, stored as Prolog clauses (discussed below).

Rule base of cinematic primitives—Film fragment construction rules, implemented in Prolog. They operate on the entered query and the annotations to edit parts of the Database and create fragments matching input concepts.

Scene creation strategies—A set of rules to analyze a sequence of the outputs from the cinematic primitives, and attempt to discern a possible scene structure. If such a structure exists, the rules reinforce the filmic presentation of that structure with an appropriate editing technique. This component is not discussed further in this article.

A query is entered and is compared with the contents of the database. Assuming there is no directly applicable clip of video, the query is fed to the rule base of

cinematic primitives. These rules attempt to reformulate the original query as a set of new queries, which will yield portions of film from the database. They do this by taking into account the query and the database contents. If this process is successful, and appropriate fragments of video are found to be available, then these are fed, in turn, into the scene creation strategies. These strategies attempt to mould the sequence given into a scene or set of scenes, by inserting additional footage.

Each of the sections of LIVE continually accesses the database of video annotations in order to maintain continuity as fragments are retrieved and assembled. This process is summarized in Figure 3.

Video Annotation

In this section we describe the video annotation methodology of LIVE. We focus particularly on the inference mechanism implemented, by providing examples of the flexibility that our approach offers in comparison with other schemes.

Multiple Overlapping State Descriptions

To create a database that supports automatic intelligent film editing, we must store several types of information. This information is collectively known as *mise-en-scène*. Our database of video annotations uses an approach based upon Sowa's (1984) conceptual structures to describe the content of a interval of film. This formalism provides a complete scheme for concept description, combined with a basic inference mechanism. Conceptual graphs were established as a suitable method for the description of film concepts in the CLORIS system (Parkes, 1989a,

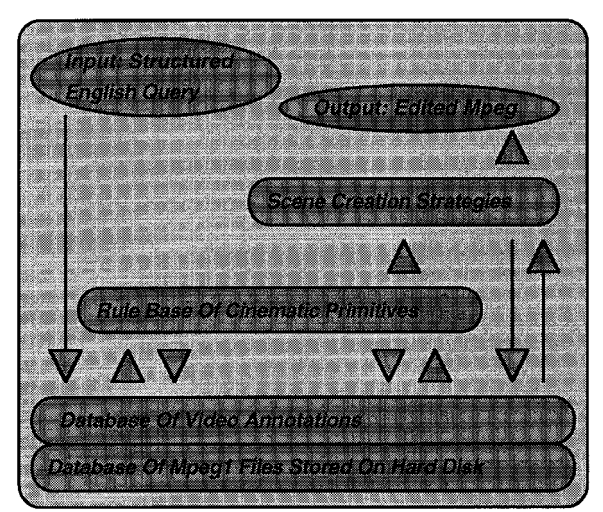

Figure 3. Overview of the structure of LIVE.

1989b). However, we extend the hierarchical approach to representation used in CLORIS by combining it with the stratification layers introduced by Aguierre-Smith and Davenport (Aguierre Smith & Davenport, 1992) also discussed above, and the objective aspects of the *setting* definition. The results of this are interval-based multilayered partially overlapping structures along with content annotations that are constrained to be as objective as feasible.

In actuality, we describe a state, then say that this state is true for a given length of film. However, the use of the term *state* is misleading, as the annotations are not necessarily mutually exclusive or unique. Having overlapping descriptions enables their combination into complex descriptions. It also enables a considerable reduction in the time taken to actually annotate the video. For example, in Figure 4, at frame 100 we know it is raining, that Fred is driving the bus, and that he is driving it fast.

Conceptual Structures

In the simplified version of Sowa's (1984) conceptual structures implemented within LIVE, a conceptual graph is a connected directed graph formed from concepts and conceptual relations. A *conceptual relation* is one of a finite set of relations such as agent, attribute, location, object, and a *concept* is an instantiation of a concept type.

A *concept type* is a member of a hierarchy (or lattice) of classes of entities, attributes, states, events, etc. (see Figure 5). The hierarchy facilitates simple reasoning about a conceptual graph (see Figure 6). This reasoning is primarily based upon inheritance within the hierarchy. So if a rule requests a piece of footage showing a car and one cannot be found, then by a process of generalization, and specialization, alternative vehicles will be used.

Example descriptions will reveal more about the approach than will further discussion. Let us create an example conceptual graph and say it is true for a specific interval of a film.

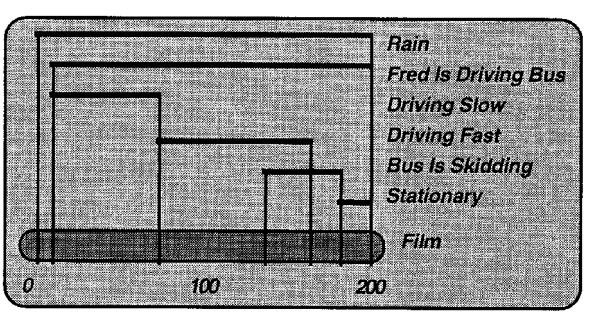

Figure 4. Overlapping interval descriptions.

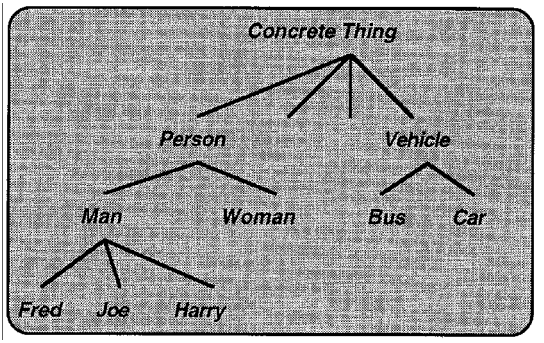

Figure 5. Conceptual hierarchy.

```
[alan, agent, drive], [car, object, drive]
                                        frames 100 to 200
```

Below we define additions to this concept. The first holds over a subinterval of that immediately above and thus is implicitly dependent on it:

```
[fast, manner, drive]                   frames 100 to 140
```

The second addition holds over an interval that overlaps with that associated with the first graph:

```
[veryfast, manner, drive]               frames 160 to 200
```

Thus we have now specified three intervals:

```
[alan, agent, drive], [car, object, drive],
 [fast, manner, drive]                  frames 100 to 140
[alan, agent, drive], [car, object, drive],
 [veryfast, manner, drive]              frames 160 to 200
```

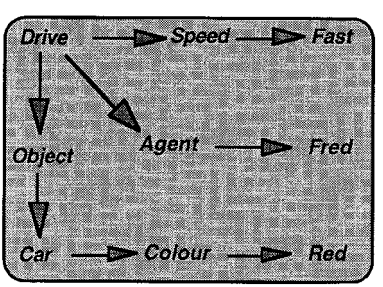

Figure 6. Conceptual graph.

```
[alan, agent, drive], [car, object, drive]
                                        frames 141 to 159
```

Note that, while similar to the stratification method discussed earlier, our scheme is more powerful, in that our interval annotations are nonatomic and can be used to support reasoning in ways discussed below.

The above approach is used to support our stated goal of a cost-effective annotation method by facilitating reuse of state descriptions. Mise-en-scène descriptions may also be represented in this way, and we now present a selection of such descriptions to highlight the advantages of our approach over more structured frame-based approaches.

Shot-type description. It is usually the case that video annotation schemes constrain the specification of shot-type parameters such as *long* and *medium* to apply to the whole of a shot, as for example, does AUTEUR (Nack & Parkes, 1995). It is actually more useful if such parameters can be defined for objects *within* the shot. LIVE permits this. For example, a shot may be a long shot of Alan but could feature a close-up of a lamppost in the foreground. This information can simply be attached to the interval descriptions from above; thus

```
[[alan, agent, drive], [car, object, drive],
[longshot, shottype, alan], [mediumshot, shottype,
car], [alan, agent, look], [lamppost, object, look],
[closeup, shottype, lamppost]]
```

The effect of this type of annotation is to allow further reuse of footage. The above interval is suitable if a close-up of a lamppost is required, or if a long shot of Alan was requested. Traditional annotation approaches concentrate on the main action of the interval and ignore the alternative framing.

Illumination. Suppose alan is walking into, and out from, the light cast by the lamp in our lamppost. The following two conceptual graphs specify this:

```
[alan, agent, walk], [street, location, alan],
 [dark, illumination, alan]          frames 100 to 150
[alan, agent, walk], [street, location, alan],
 [light, illumination, alan]         frames 151 to 200
```

Of course, the inheritance mechanism outlined in the previous section can apply to any aspect of a description. So the lighting section could be specified like

```
[alan, agent, walk], [street, location, alan]
                                     frames 100 to 200
[dark, illumination, alan]           frames 100 to 150
[light, illumination, alan]          frames 151 to 200
```

A genuine example of this can be found in many films featuring Marlene Dietrich. For example, in *Shanghai Express,* von Sternberg used additional top lighting, not only to highlight the star's cheekbones but also to suggest her forward projection out of the picture.

Effects. For each of the concepts in a conceptual graph, we may wish to state what type of special effects are involved, e.g., mononchromatic, color, iris:

```
[[sophia, agent, sunbathe], [beach, location, sophia],
[iris, effect, sophia]]
```

Thus the annotator can, and indeed is recommended to, apply characteristics previously applied only to the image as a whole, e.g., illumination and effects, to individual items portrayed. This is a further feature that distinguishes our annotation scheme from others.

Fragment Construction Rules

As discussed several times above, when film units are juxtaposed, their meanings combine to produce new meanings. The different ways of juxtaposing units of film can be regarded as rule based. However, a cautionary note is needed here, as stated by Monaco (1981, p. 140):

> Film has no grammar. There are, however, some vaguely defined rules of usage in cinematic language, and the syntax of film—its systematic arrangement—orders these rules and indicates relationships between them.

Thus, while we have implemented a set of film editing "rules," it must be appreciated that such rules do not work in all circumstances. As such, they might better be termed *strategies.* However, their formulation reflects our desire to create rules that are as nonspecific as possible.

General Operation of Rules

The rules are implemented as hierarchies of Prolog clauses and are divided into two sections. The first section determines whether the query entered satisfies the conditions needed for the rule to be applied. The second section is a hierarchy of transformations that can be applied to the query based upon the content of the database and the details expressed in the query. As we saw in the section above, Overview of the System, rules are applied only if no applicable portion of film can be found to exactly match the query. Because the queries take the same form as the annotations (graphs), the first section searches for a subgraph (generally a tree) of the query matching certain preconditions necessary for that rule to fire, for example, in the subjective shot [Agent, agent, look], [Object, object, look].

If the preconditions do not exist, then the rule fails, and another is tried. Assuming the above precondition exists, then the rule reformulates the user query to [Agent, agent, look], [unknown, object, look], ...

Further predicates extract information regarding the look, and regarding the agent, through Prolog pattern matching of the triples that make up the query/graph. If this further information cannot be found, then the rule continues with the minimum information. This first new query is presented to the database, which tries to retrieve an interval of video with the best match. The similarity measure is simply one where most of the triples that make up the arcs and nodes of the graphs are identical.

Assuming that an interval can be found, then the initially entered query, the new query, and the description of the retrieved video interval are used to construct a second new query, which is to be used to retrieve the second part of the subjective shot. To construct the second new query, the Object is retrieved from the initially entered query, then the direction of the look, and the location of the agent from the description of the first retrieved interval. This query is then fed into the database and a closest match found. The two resulting MPEGs are appended and saved as a single file.

Creative Geography

Creative geography is manifested in a sense of coherent location that we usually experience when we watch a film. To achieve it, we juxtapose events that feature common agents, or take place in common locations, or that feature other common attributes. If we show someone walking along a corridor, down some stairs, and then outside, the locations could be three places on different continents, but as long as the person and general architecture matched, the viewer would assume the events to be taking place with respect to the same building.

The element common to separate fragments of film may be different parts of the same action. As we see from the gunfight during *The Naked Gun 2½: The Smell of Fear,* the audience's expectations of the information missing from the portrayed action define the space in which the fictional action occurs. In the gunfight, we see two characters shooting at each other, both portrayed in separate shots. Then we see them together in the same shot, and they are only 1 meter apart.

The person, or some other common element, is the detail to which the viewers devote their attention and thus creates the link between the locations. Consider the following query, which requests a "leaving the building" episode, such as that discussed above, for the character "Dave":

```
[[[dave, agent, walk], [lab, location, dave]], [[dave,
agent, walk], [corridor, location, dave]], [[dave,
agent, walk], [carpark, location, dave]]]
```

The above query might be partitioned into separate queries based on the parentheses (which are used to indicate episode ordering). This yields a first query of

```
[dave, agent, walk], [lab, location, dave]
```

Upon being issued to the database, the query immediately above may yield the following description:

```
[dave, agent, walk], [lab, location, dave], [dave,
agent, wear], [jacket, object, wear], [paul, agent,
work], [lab, location, paul]
```

A second query is then generated from this result, after removing the original query, by extracting all the attributes from the elements common to both the first and the second of the short queries derived from the original user query. In the above case, *dave* and *walk* are the common element, so we extract those attributes of the description pertaining to them (note that in this case, walk has no associated attributes):

```
[dave, agent, wear], [jacket, object, wear]
[dave, agent, walk], [lab, location, dave], [dave,
agent, wear], [jacket, object, wear]
```

However, when the immediately preceding query is issued to the database, a closest match, rather than an exact match, is requested. This yields a number of potential results, each of which is juxtaposed with a copy of the original result to create new alternatives. The same process is applied to the third and any successive queries, yielding a small set of final results. A suitable output for our example is depicted in Figure 7.

Parallelism, Crosscutting, Parallel Action

Parallelism is used in many films, often to show two or more separate threads of plot occurring at the same time, though possibly in separate locations. A clichéd example is the cavalry and the Indians both converging on the same wagon train. To achieve parallelism, we interleave two or more events, as depicted in Figure 8.

Figure 7. Creative geography.

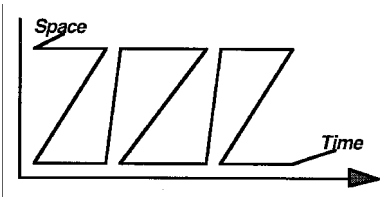

Figure 8. Parallelism.

We saw in the section above, Multiple Overlapping State Descriptions, that our annotation method has the side effect of creating segmentations of the stored footage at state boundaries. Such state boundaries indicate places in the footage where units of action are complete, and so provide a convenient location at which to crosscut.

The following example requests two actions that occur at the same time:

```
[[frank, agent, type], [sparc, object, type], [sean,
agent, read], [manual, object, read]]
```

Assuming that footage showing both actions occurring in the same shot cannot be found, then this query is partitioned according to agent and event triples, yielding two new queries:

```
[[frank, agent, type], [sparc, object, type]]
[[sean, agent, read], [manual, object, read]]
```

These two queries are then issued separately to the annotation database and the results interleaved at the segmentation points. If in the above example, while frank is typing at the computer, he also uses the mouse, or the environment changes, and these factors are captured in the annotations, then the system will use the boundaries of these actions as places to make the insertions of the footage (similarly divided) of sean reading. If no segmentation points exist, the insertion points can be arbitrarily generated (though this has not been implemented).

Subjective Shot, Point of View

Carroll (1980) discusses the notion of filmicity and filmic correctness and, in doing so, defines how to achieve a subjective shot:

> If an actor casts a glance out of a frame of the shot (casts a look of outward regard), the shot immediately following will be interpreted as a subjective-shot, that is, as a shot from the actors point of view—revealing what it is he looked at (Carroll, 1980, p. 71).

To show a subjective shot, we show a person looking at something out of the frame, then cut to a shot of the entity at which the person is assumed to be looking. For example, consider the following query:

```
[dave, agent, look], [sean, object, look], [up, direc-
tion, look]
```

If this query fails in the database, it is changed by the subjective shot rule into two new queries:

```
[[dave, agent, look], [unknown, object, look], [up,
direction, look]], [[sean, X, Y], [up, shotangle,
sean]]
```

Example output from the above query is shown in Figure 9. It is important to note that the angle of the actor's look in the first shot must be similar to the camera angle of the second shot and the actor must not feature in the action of the second shot. It is possible to present the viewer with more evidence that a subjective shot is occurring by making it appear as if the actions take place in the same location. Alternatively, less evidence is given if the direction of the look is violated. To this end, a hierarchy of subjective shot rules has been implemented, which ranges from specific and effective, to general and less effective.

Highlight

Pudovkin (1968) draws parallels between humans viewing their environment and the ways in which events are portrayed by film:

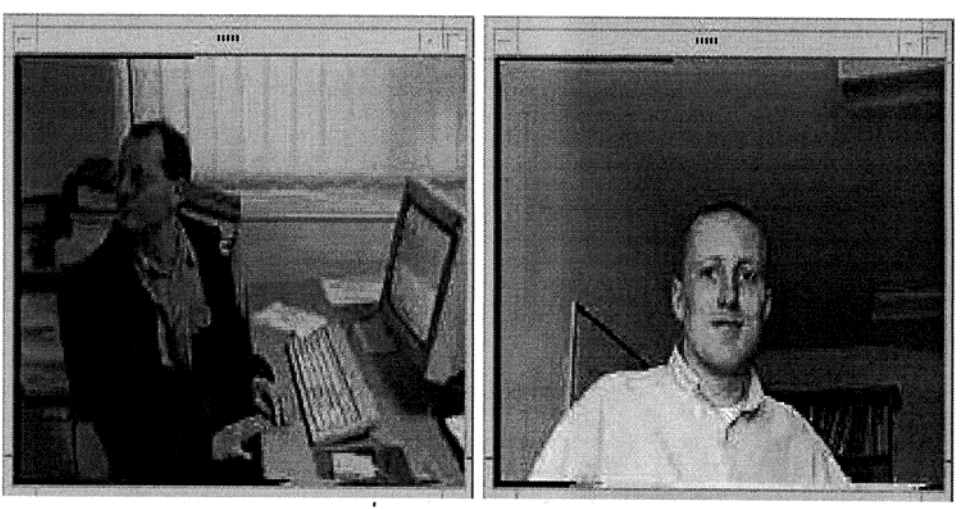

Figure 9. Subjective shot.

> the nearer we approach a regarded object, the less material appears simultaneously in our field-view; the more clearly our investigating glance examines an object, the more details we perceive and the more limited and sectional becomes our view (Pudovkin, 1968, p. 90).

He goes on to say that when we look at an object from a short distance,

> We no longer perceive the object as a whole, but pick out the details with our glance in order, thus receiving by association an impression of the whole that is far more vivid, deeper and sharper than if we had gazed at the object from a distance, and perceived the whole in a general view, inevitably missing detail in so doing (Pudovkin, 1968, p. 90).

We can model this close-up examining of objects and rely on the audience's ability to construct a whole from parts to create *false* impressions. To show that an entity possesses a particular attribute, when explicit footage is unavailable, we can show that object in a long or medium shot, then close up to a similar item portraying that attribute. However, if the attribute of the entity required is its location, we can show a close-up of the entity in any location with an embedded long shot of the entity at the desired location.

Consider this query:

```
[[frank, agent, walk], [path, location, frank],
[frank, agent, smile]]
```

From this, two new queries are generated:

```
[[frank, agent, walk], [path, location, frank]]
[[frank, agent, smile], [closeup, shottype, frank]]
```

When the first is given to the database, the description of the resulting footage is used to augment the second, by extracting as many of the original attributes as possible.

```
Result: [[frank, agent, walk], [path, location, frank],
[frank, agent, carry], [newspaper, object, carry]]
New Second Query: [[frank, agent, smile], [closeup,
shottype, frank], [path, location, frank], [frank,
agent, carry], [newspaper, object, carry]]
```

The resulting footage returned from the first of the above pair of queries is partitioned and used to bracket the result of the second query. The second query is presented to the database requiring only a best match. Figure 10 is an example output displaying such a *highlight,* and we see that in the middle image, although

Figure 10. Highlight.

the actions and locations are correctly presented, Frank is not shown carrying a newspaper.

CONCLUSIONS

This article has shown that the LIVE system operates with a set of query transformation rules that fire in response to queries that do not explicitly specify the juxtaposition of particular shots. In this way, we have developed a system that allows the user to formulate queries based on the meaning of the desired footage, and the system interprets this meaning and maps it onto the content of the footage available in the database. Thus the query is expressed nonfilmically, but the decisions as to how the query is to be expressed filmically are made by the system itself.

The possibility of ambiguity due to natural language being used in the annotations has not been encountered. This is because the rules as described in this article do not rely on any specific values for particular attributes. This was part of a conscious design rationale, in order to make the system efficient and flexible. The result is that a given community of users can agree on their own values for attributes such as light levels, or angle of view in the case of a subjective shot, which works by similarity rather than relying on specific values.

The similarity measure for comparing queries with the database contents can in situations where the best match is a slight match, produce results unsuitable for juxtaposition in sequences; however, this is not generally a problem.

There are a number of limitations to our prototype. Currently, no two rules can operate at the same time. For example, the system will not generate a sequence featuring crosscutting if one of the threads needs to have the subjective shot rule applied to be realized. This is a severe limitation and needs to be rectified in future versions. Future work also includes the implementation of *event-abbreviation,* described by Butler and Parker (1996), a form of linear deletion as described by Carroll (1980).

LIVE does not contain a *complete* cinema grammar. It is debatable whether a cinema grammar can ever be complete. However, we have designed LIVE so that its grammar can be allowed to evolve, as any "natural language" grammar should.

REFERENCES

Aguierre Smith, T. G., and G. Davenport. 1992. The stratification system: A design environment for random access video. MIT Media Lab Technical Report.

Andrew, J. D. 1976. The major film theories—An introduction. New York: Oxford University Press.

Arnheim, R. 1958. *Film as art.* London: Faber & Faber.

Balasz. 1972. *Theory of the film.* New York: Arno Press and *The New York Times.*

Bloch, G. R. 1988. From concepts to film sequences. In *RIAO 88,* 760–767. Cambridge, MA: MIT.

Branigan, E. R. 1984. *Point of view in the cinema—A theory of narration and subjectivity in classical film.* Amsterdam: Mouton Publishers.

Butler, S., and A. Parkes. 1996. Space time diagrams for film structure visualisation. *Signal Processing: Image Communication* 8(4):269–280.

Carroll, J. M. 1980. *Towards a structural psychology of cinema.* Amsterdam: Mouton Publishers.

Chua, T., and L. Ruan. 1995. A video retrieval and sequencing system. *ACM Transactions on Information Systems* 13(4):373–407.

Davis, M. 1995. Media Streams: Representing video for retrieval and repurposing. Ph.D. diss. MIT, Cambridge, MA.

Eisenstein, S. 1948. *Film sense.* London: Faber and Faber Ltd.

Eisenstein, S. 1949. *Film form.* New York: Harcourt, Brace, World.

Metz, C. 1974. *Film language: A semiotic of the cinema.* New York: Oxford University Press.

Monaco, J. 1981. *How to read a film.* New York: Oxford University Press.

Nack, F., and A. Parkes. 1995. AUTEUR: The creation of humorous scenes using automated video editing. In *IJCAI-95 workshop on entertainment and AI/ALife, Montreal, Quebec.*

Parkes, A. P. 1989a. An artificial intelligence approach to the conceptual description of videodisc images. Ph.D. thesis, Lancaster University.

Parkes, A. P. 1989b. The prototype CLORIS system: Describing, retrieving and discussing videodisc stills and sequences. *Information Processing and Management* 25(2):171–186.

Parkes, A. P. 1989c. Settings and the settings structure: The description and automated propagation of networks for perusing videodisk image states. In *Proceedings of the SIGIR '89, Cambridge, Massachusetts.*

Pudovkin, V. I. 1968. *Film technique and film acting.* London: Vision Press Limited.

Sack, W., and M. Davis. 1994. IDIC: Assembling video sequences from story plans and content annotations. In *Proceedings of the IEEE international conference on multimedia computing and systems, May 14–19, Boston, Massachusetts.*

Sack, W., and A. Don. 1993. Splicer: An intelligent video editor. Project4: Intelligent interface software design workshop. MIT Media Lab Technical Report.

Sowa, J. F. 1984. *Conceptual structures: Information processing in mind and machine.* Reading, Mass.: Addison-Wesley.

Spottiswoode, R. J. 1955. *A grammar of the film—An analysis of film techique.* London: Faber & Faber.

Tudor, A. 1974. *Image and influence.* London: George Allen & Unwin Ltd.