Eurographics Workshop on Intelligent Cinematography and Editing (2016)
M. Christie, Q. Galvane, A. Jhala, and R. Ronfard (Editors)

Analysing Cinematography with Embedded Constrained Patterns

Hui-Yin Wu and Marc Christie

IRISA / INRIA Rennes Bretagne Atlantique

Abstract

Cinematography carries messages on the plot, emotion, or more general feeling of the film. Yet cinematographic devices are often overlooked in existing approaches to film analysis. In this paper, we present Embedded Constrained Patterns (ECPs), a dedicated query language to search annotated film clips for sequences that fulfill complex stylistic constraints. ECPs are groups of framing and sequencing constraints defined using vocabulary in film textbooks. Using a set algorithm, all occurrences of the ECPs can be found in annotated film sequences. We use a film clip from the Lord of the Rings to demonstrate a range of ECPs that can be detected, and analyse them in relation to story and emotions in the film.

Categories and Subject Descriptors (according to ACM CCS): I.2.10 [Artificial Intelligence]: Vision and Scene Understanding—Video Analysis

1. Introduction

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] from annotation tools such as Insight [MWS*15], as well as reanimate camera trajectories that are extracted from tools like voodoo tracker [SDM*14]. Visualization tools such as Cinemetrics [Bro11] show the color distribution of scenes in films, and allows the comparison of multiple films.

Apart from applications to video streaming platforms, tools for automatic analysis of film sequences can also provide educational benefits to prospective film makers. In cognitive film studies, statistics run on annotated shot data have provided insights on how directors make stylistic choices that convey story events and emotions simply by composing actor positions in images [Cut15] [Bor85]. The effect of framing composition on the audience has also drawn interest from multimedia analysts to understand what emotions are tied to visual features such as shot size [CBL11] or shot types [SBA*15]. Many theorists go so far as to propose that film, like written language, has some form of grammar and semantics [BS12] [Tse13].

Yet when searching for examples in actual film data, we are limited by our human capacity to observe, record, and generalise these grammar rules and semantics. We are distracted by the multitude of color, movement, sound on the flashing screen. There are currently no general purpose languages or tools for automated analysis of cinematography in film. This is due to three reasons. First, film cinematography has no fixed implementation from director to director. As shown in Figure 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. Thus a language to describe film cinematography 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.

In this paper we present Embedded Constraint Patterns (ECP), which sets out to tackle the challenge of analysing complex shot arrangements. Our purpose is to provide a simple–and familiar–set of vocabulary for cinematography that can be used to describe and formulate observations. Such a tool would allow users to formulate, collect data, and more easily reject, accept, or adjust different observational hypotheses. Setting out from a film analyst's observations, we can formalise grammatical rules and film idioms of cinematography found in film textbooks such as [TB09] [Zet07].

Here we propose an implementation of the Patterns cinematography language [WC15] as a film query language. ECPs can be defined using the Patterns language, and a solver searches movies for ECPs in the annotated data. Our goal for the implementation of the ECPs is to use the Patterns vocabulary to (1) easily and flexibly define groups of cinematographic constraints into a number of ECPs

DOI: 10.2312/wiced.20161098

Figure 1: These two sequences in *Lord of the Rings* and *Godfather* both show the intensify ECP (moving gradually closer to the actor).

(to address the difficulty of providing a unifying definition between various directorial styles), and (2) query on annotated film data and extract montages of matching seqeunces. An ECP represents the semantics of framing and frame sequencing over a number of shots, such as the intensify ECP shown in Figure 1. The vocabulary consists of on-screen properties that the user can easily observe–such as size, angle, position–and sequencing constraints on these properties. This approach may depart from traditional descriptions of cinematographic elements, but is a more direct way to observe and analyse the meaning of on-screen compositions and the underlying messages they carry.

ECPs can be generally used to query a variety of annotated data on cinematographic composition. Its main strength is in its ability to search for long sequences pertaining to a number of framing, movement, or target position constraints. Thus, ECPs has practical applications to improving film analysis techniques for event detection, viewer recommendations and movie search engines for online film services, and educational tools through learn by example.

2. Related Work

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 summarization 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]. However, these approaches focus on a single aspect of film analysis, and cannot take into accound the multimodality of film cinematography, i.e. the combination of multiple elements of size, angle, composition, light may result in different meanings.

To bridge the gap between the viewer's emotion in relation to cinematography style, machine learning has been applied to understanding how multiple film features contribute to how films communicate with the audience. Both [CBL11] and [MBC15] use the markov chain to learn film parameters. [CBL11] observes the correlation between shot sizes and audience emotions; [MBC15] focuses on learning transition parameters to replicate director style. However, how weights for machine learning features are calculated makes it difficult to draw clear observations on how style correlates to emotions.

In virtual camera control, film idioms and continuity rules have been popularly adapted to constraint-based systems. An early example is the DCCL language for planning sequences of camera movements [CAH*96]. More recently, vocabularies for cinematography have been developed apart from Patterns. Most notably, the Prose Storyboard Language (PSL) [RVB13] was designed based on actual film practice on shot composition, including vocabulary for elements like size, region, or movement. In the implementation in [GCR*13], PSL targets autonomous camera steering and composition, conducting a search for suitable camera positions based on PSL constraints. However, films aren't just composed of a number of images strung together. In film, there is movement, there is cause-effect, there is a progression of story, and these are expressed and supported by the camera. Patterns provides specific vocabulary for constraints on relations between shots and sub-sequences. Since our main purpose in this paper is for analysis of cinematographic storytelling, we choose to use the base vocabulary set from Patterns for constructing ECPs.

3. Patterns Language Overview and ECP Structure

Patterns is based on the elements from widely-used film textbooks including [Zet07] [TB09]. In this section, we outline the structure and vocabulary of the language, as well as its grammar to construct ECPs.

A ECP sets a number of constraints on a sequence of shots. There are 3 types of constraints: (1) framing, (2) shot relations, and (3) sub-sequence constraints. An ECP can contain multiple framing and relation constraints. Further constraints such as length and sub-sequences can also be added to ECPs. Below, we describe each category in detail.

3.1. Framing Constraints

Framing constraints restricts how actors are arranged on-screen, mainly including four aspects: *size*, *angle*, *region*, and *movement*. The selection of each of these constraints has its basis in film practice.

The constraints are discussed in terms of actors. We adopt a very broad definition of actors that incorporates humans, animated creatures, and objects.

3.1.1. 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. We categorise 9 sizes 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 are shown in Figure 2.

Figure 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).

3.1.2. Angle

Corresponding to the three axes in 3D, the camera can be rotated on the horizontal, vertical, and roll axis. Examples of each type of angle can be found in Figure 3.

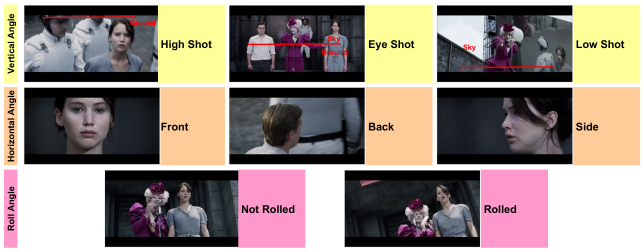

Figure 3: Angles of characters on the screen include the vertical, horizontal, and roll angle.Vertical angles can be tricky to observe. One trick is to estimate the skline in each shot (roughly indicated in this figure). Shots with higher angles have higher skylines, while lower shots have lower skylines. Screenshots come 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).

Tilted shots (with a roll angle) can be adopted for the purpose of showing disorientation of the actor, or distortion of the environment.

3.1.3. 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 $4-split$ or $9-split$ regions, or simply $top/bottom$ and $left/right$.

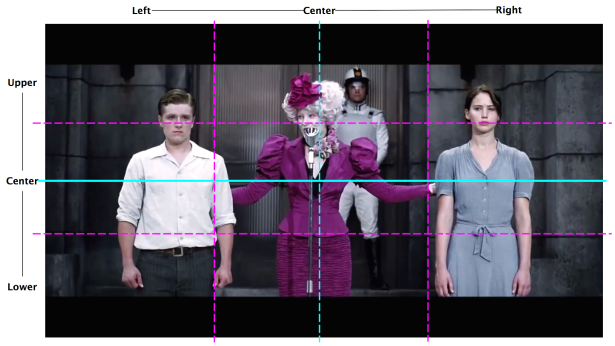

Figure 4: There are many ways to describe on-screen positions of actors. In film literature, the most well-known one is the 9-split standard (purple dotted lines; also referred to as the rule of thirds), where the screen is split into three regions horizontally and vertically. Patterns provides vocabulary for $9-split$, $4-split$ (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.

3.1.4. 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).

3.2. Shot Relations

Constraints can be placed not only on single framing compositions, but also between shots. There are two characteristics considering relationship between two shots. And the other is transitions, referring to the techniques that links two shots: a cut, fade, screen wipe, etc.; however, this is not our focus here.

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.

3.2.1. 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-distance*, *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 1–in which the camera becomes *closer* to the actors from one shot to another.

3.2.2. 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 $same-angle$ to describe angle relations between two consecutive shots.

In Figure 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.

3.2.3. 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 basic relation vocabulary for same *region* (the same 9-split or 4-split region), same/different $horizontal-region$, and same/different $vertical-region$.

Figures 8 and 9 show of how framing regions could be an interpretation of relations between actors. Figure 8 places actors in the same region to show agreement and compassion, while Figure 9 places two actors in opposite horizontal regions, reflecting a disagreement or enmity.

3.3. Sub-Sequences and ECP Structure

We will use the incremental construction of the intensify sequence to illustrate how sub-sequences (an embedded ECP) can be defined, and how an ECP is structured. We show how the combination of different vocabulary in Patterns can provide multiple definitions of the intensify ECP. Examples of intensify can be seen in Figure 1.

3.3.1. 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 6, 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 sub-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 6, 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.

3.3.2. 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, it can be achieved through setting ranges for constraints. A range parameter that can either be a continuous range expressed as $[x-y]$, a discrete list $(x,y,z...)$, or it can be one of the keywords of *initial* (which is equal to the list (1)), *all*, *none*, or *end*. In this case, we can define a pattern where shots 1, 3, and 5 are CU, and 2, 4, and 6 are LS, and shots 7 to 9 are MS as:

```
intensify{
    framing
        constraint: size=CU
        range: (1,3,5])
    framing
        constraint: size=LS
        range: (2,4,6)
    framing
        constraint: size=MS
        range: [7-9]
}
```

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
}
```

3.3.3. 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. We add a length constraint to intensify:

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

In Figure 6 even though $4 \rightarrow 5$ fulfils the relation constraint *closer*, and both $[4]$ and $[5-6]$ 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. Solver

The question now is that of finding all sequences that fulfill a user-defined cinematographic ECP. An ECP is composed of (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. `sub-sequence=shot`.

The solving process is expressed as a search over a sequence $\mathbf{S}$ of frames. 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 run in two stages. The first stage builds a cache of valid solutions as three sets of frame sequences **FC**, **RC** and **SC**. The first represents the sets of frames that satisfy each of the framing constraints mentioned in the ECP. The second represents the sets of frame/shot couples $[f_i, f_{i+1}]$ that satisfy the relation constraints, and the last represents the set of frame sub-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 2). 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).

4.1. 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 vocabulary 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 5 shows how the two constraints would be matched to a sequence of keyframes.

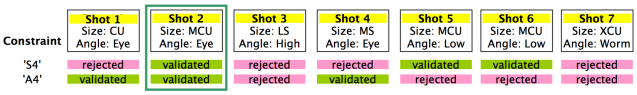

Figure 5: Constraints on single framings can be matched individually and validated or rejected directly. In this example, Frame 2 fulfils both constraints. We build a set that contains, for each constraint the set of valid frames.

Shot relations are more difficult to express due to the limitation that regular expressions cannot compare mathematical values. However, since the parameters for features in Patterns is 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.

Finally, sub-sequences are treated just like any other group of constraints. However the computation requires to run the search algorithm in the next section in order to return all sequences of keyframes that can be grouped into a sub-sequence.

Constraint	Shot 1 Size: CU Angle: Eye	Shot 2 Size: MCU Angle: Eye	Shot 3 Size: LS Angle: High	Shot 4 Size: MS Angle: Eye	Shot 5 Size: MCU Angle: Low	Shot 6 Size: MCU Angle: Low	Shot 7 Size: XCU Angle: Worm
'Closer'	validated	rejected	rejected	validated	validated	rejected	validated
'Same_Size'	validated	rejected	rejected	rejected	rejected	validated	rejected

Figure 6: In this example, both closer and same-size sets constraints on the framing size relation. Naturally, since their definition contradict each other, there is no overlap.

4.2. Search Algorithm

The search algorithm (ECPRecurse) relies on functions *isValidFrame()* to evaluate whether the frame f_i is within the set of valid frames **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 1 ECPSearch (ECP p, Sequence **S**)

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

Algorithm 2 ECPRecurse (ECP p, CurrentSequence **C**, Sequence **S**, Index s, ResultSet **R**)

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

5. Examples

We demonstrate how our solver analyses user-defined ECPs using the video clip of Lord of the Rings as an example †. The sequence is composed of 56 shots. The basic storyline follows the elvish lord Elrond who disagrees with the wizard Gandalf that men are untrustworthy. A flashback occurs in the middle of their dialogue, showing Elrond fruitlessly attempting 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-size. The two ECPs for framing regions are frameshare and opposition. Figure 7 presents all instances of found ECPs throughout the clip.

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

5.1. 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-size long. The definition of intensify is the same as the final step of our process in Section 3.3.3.

Our definition of same-size chain relies on the relation parameter $same-size$ 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{
    relation
        constraint: same-size
        range: all
    sub-sequence
        constraint: 'shot'
        range: all
    length
        constraint: length>=2
}
```

As we can observe from Figure 7, 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.

5.2. Study of Framing Region Evolutions

Figure 8: Example frameshare extracted by the query, with Isildur under the influence of the ring.

In the film clip, we 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 the actors appear in two consecutive shots on the same side of the frame such as in Figure 8–to show empathy. We define the frameshare ECP as:

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

Figure 7: Here we present the result of a search for 4 ECPs: intensify, same-size, frameshare, and opposition. These techniques reveal insight into the plot, such as when characters are in agreement (frameshare) or discord (opposition), and when the atmosphere is intensifying (intensify) or stable (same-size).

Its opposite, which we call opposition–hence, actors appearing on opposite left/right sides of the frame such as in Figure 9–is often used to show opposition or disagreement among actors. Using Patterns, opposition is defined as:

```
opposition{
    framing
     constraint: target_num==1
    relation
        constraint: horizontal-region!=same
        range: all
    sub-sequence
        constraint: 'shot'
        range: all
    length
        constraint: length>=2
}
```

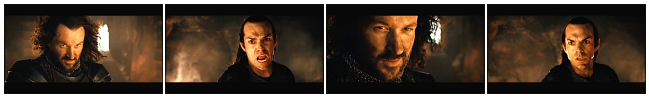

Figure 9: Example opposition extracted by the query, where Elrond and Isildur disagree with each other.

By defining the two ECPs, we easily extract uses of frame-share and opposition throughout the clip. Comparing two specific examples in Figures 8 and 9, 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 7, we also see 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 filmshare when portraying the arrival of surrounding allies to support their cause.

6. Limitations and Future Work

Our solver for ECP implements a full search on the framing database for sequences that fulfil constraints. This becomes computationally heavy when evaluating multiple constraints over a large database. More efficient algorithms could enhance the speed of of searching for complex ECPs. Moreover, a smarter interpreter with the ability to filter contradicting constraints (such as larger and smaller) can be useful for the user.

In the field of cognitive science for films, we have seen a rise in

film analysis that involves processing annotated data on the visual features of shots, and montaging. We envision that ECPs can become a plugin for film search engines, video streaming services, or film analysis tools. Combined with annotation tools, ECPs can provide insight into how films are structured, and how certain types of camera movements, shot compositions, and transition techniques contribute to the story, like in our *Lord of the Rings* example.

In our ongoing work, we plan to design *Patterns* as a plugin for a film shooting and editing tool in Unity 3D that will allow users to apply ECPs to their montages, which can be useful not only for film pre-visualisation, but also educational for film school students on experimenting with different film editing techniques.

7. Conclusion

In this paper we have proposed a solver using the Patterns cinematography language for defining ECPs. We extract relevant data from annotated film clips to compose regular expression constraints, and conduct a full search on matches to find sequences that fulfil ECPs. A full example from Lord of the Rings is provided to demonstrate the approach's potential.

References

[Bor85] BORDWELL D.: *Narration in the Fiction Film*. University of Wisconsin Press, 1985. 1

[Bro11] BRODBECK F.: Cinemetrics. http://cinemetrics.fredericbrodbeck.de, 2011. 1

[BS12] BATEMAN J. A., SCHMIDT K.-H.: *Multimodal Film Analysis: How Films Mean*. Routledge, New York, 2012. 1

[CAH*96] CHRISTIANSON D. B., ANDERSON S. E., HE L.-W., SALESIN D. H., WELD D. S., COHEN M. F.: Declarative camera control for automatic cinematography. *AAAI Conference on Artificial Intelligence* (1996). 2

[CB98] CORRIDONI J., BIMBO A.: Structured Representation and Automatic Indexing of Movie Information Content. *Pattern Recognition 31*, 12 (1998), 2027–2045. 2

[CBL11] CANINI L., BENINI S., LEONARDI R.: Affective analysis on patterns of shot types in movies. In *7th International Symposium on Image and Signal Processing and Analysis (ISPA)* (sep 2011), pp. 253–258. 1, 2

[CBL13] CANINI L., BENINI S., LEONARDI R.: Affective recommendation of movies based on selected connotative features. *IEEE Transactions on Circuits and Systems for Video Technology 23*, 4 (2013), 636–647. 1

[Cut15] CUTTING J. E.: The Framing of Characters in Popular Movies. *Art & Perception 3*, 2 (2015), 191–212. 1

[GCR*13] GALVANE Q., CHRISTIE M., RONFARD R., LIM C.-K., CANI M.-P.: Steering Behaviors for Autonomous Cameras. *Proceedings of Motion on Games - MIG '13* (2013), 93–102. 2

[LC15] LINO C., CHRISTIE M.: Intuitive and Efficient Camera Control with the Toric Space. *Transactions on Graphics 34*, 4 (2015). 1

[MBC15] MERABTI B., BOUATOUCH K., CHRISTIE M.: A Virtual Director Using Hidden Markov Models. *Computer Graphics Forum* (2015). 2

[MWS*15] MERABTI B., WU H.-Y., SANOKHO C. B., GALVANE Q., LINO C., CHRISTIE M.: Insight : An annotation tool and format for film analysis. In *Eurographics Workshop on Intelligent Cinematography and Editing, May 2015, Zurich, Switzerland* (2015), p. 1. 1

[RSS05] RASHEED Z., SHEIKH Y., SHAH M.: On the use of computable features for film classification. *IEEE Transactions on Circuits and Systems for Video Technology 15*, 1 (2005), 52–63. 2

[RVB13] RONFARD R., VINEET G., BOIRON L.: The Prose Storyboard Language. In *AAAI Workshop on Intelligent Cinematography and Editing* (2013). 2

[SBA*15] SVANERA M., BENINI S., ADAMI N., LEONARDI R., KOVÁCS A. B.: Over-the-shoulder shot detection in art films. In *International Workshop on Content-Based Multimedia Indexing* (2015), vol. 2015-July. 1, 2

[SDM*14] SANOKHO C. B., DESOCHE C., MERABTI B., LI T.-Y., CHRISTIE M.: Camera Motion Graphs. *Proceedings of the Eurographics/ ACM SIGGRAPH Symposium on Computer Animation* (2014), 177–188. 1

[TB09] THOMPSON R., BOWEN C. J.: *Grammar of the Shot*. 2009. 1, 2

[Tse13] TSENG C.: *Cohesion in Film: Tracking Film Elements*. Palgrave Macmillan, Basingstoke, 2013. 1

[WC15] WU H.-Y., CHRISTIE M.: Stylistic Patterns for Generating Cinematographic Sequences. In *Eurographics Workshop on Intelligent Cinematography and Editing* (2015). 1

[Zet07] ZETTL H.: *Sight, sound, motion: Applied media aesthetics*. Wadsworth Publishing Company, 2007. 1, 2, 3