ViVid: A Video Feature Visualization Engine

Jianyu Fan[1](✉), Philippe Pasquier[1], Luciane Maria Fadel[2], and Jim Bizzocchi[1]

[1] Simon Fraser University, Vancouver, Canada
{jianyuf, phillipe.pasquier, jimbiz}@sfu.ca
[2] Federal University of Santa Catarina, Florianópolis, Brazil
liefadel@gmail.com

Abstract. Video editors are facing the challenge of montage editing when dealing with massive amount of video shots. The major problem is selecting the feature they want to use for building repetition patterns in montage editing. It is time-consuming when testing various features for repetitions and watching videos one by one. A visualization tool for video features could be useful for assisting montage editing. Such a visualization tool is not currently available. We present the design of ViVid, an interactive system for visualizing video features for particular target videos. ViVid is a generic tool for computer-assisted montage and for the design of generative video arts, which could take advantage of the information of video features for rendering the piece. The system computes sand visualizes the color information, motion and texture information data. Instead of visualizing original feature data frame by frame, we re-arranged the data and used both statistics of video feature data and frame level data to represent the video. The system uses dashboards to visualize multiple dimensional data in multiple views. We used the project of Seasons as a case study for testing the tool. Our feasibility study shows that users are satisfied with the visualization tool.

Keywords: Video features · Data visualization

1 Research Topic

Generative video refers to creating videos using generating systems [1]. Potentially, multiple results can be produced, because a generative video is built using montage editing to connect shots and transitions that are drawn from a database. We present the design of the ViVid, which is a generic system for video feature visualization. We present a case study about "Seasons[1]", a generative media project, which uses a variety of computational processes to build the audio-visual output. "The system for Seasons builds video sequencing and transitions based on procedural rules and video metatags. Simultaneously, the system composes and mixes music and soundscape tracks that incorporate both semantic and affective elements of the video into their own aesthetic rules." [2].

[1] Seasons is a generative video from Simon Fraser University Generative Media Project coordinated by Jim Bizzocchi, Arne Eigenfeldt, Philippe Pasquier, and Miles Thorogood.

A. Marcus and W. Wang (Eds.): DUXU 2017, Part III, LNCS 10290, pp. 42–53, 2017.
DOI: 10.1007/978-3-319-58640-3_4

2 Film Editing

In film editing, there are two techniques widely used. The first is continuity editing, which uses sequences of shots to create a sense of real-time and mask the time and space. For example, a filmmaker wants a scene in which a person is cutting trees. The first shot shows the entire tree. When the person is about to cut the tree, the shot turns to the middle view. It becomes the close view when the axe is on the tree. This three-shot sequence is an example of continuity editing.

Another widely used technique is Montage editing. A "montage sequence" is a short segment in a film in which narrative information is presented in a condensed fashion" [3]. It has been used in various genres, such as science fiction, silent film, propaganda, sports, commercials, and city films. In the movie 2001: A Space Odyssey, the director depicted the development of a man evolved from apes to humans. For each shot, the creature evolves to a higher stage. This montage sequencing is based on the concept of evolution and time.

In the silent film, Battleship Potemkin, which is directed by Sergei Eisenstein, the montage theory was tested. Different types of montages were used, such as metric montage, rhythmic montage, tonal montage, over-tonal montage and intellectual montage, [4].

- Metric: The length of video segments follows the number of frames
- Rhythmic: The length of the shot suggests a pattern and the cutting points create visual continuity
- Tonal: Each shot reflects certain emotions to enhance the emotional experience in the viewers
- Overtonal: It is a combination of metric, rhythmic, and tonal montage
- Intellectual: It uses a combination of shots that are not from the original film to create new meanings

As for Olympic propaganda films, directors not only used sequences of shots of different athletes to show the diversity of the Olympic games, but also used the different stage of competitions to tell audiences the opening of the Olympic game. They used the concept of sport, development, and emotion, to build their sequences. This is an example of Overtonal montage.

Motion is widely used in the montage sequencing. In the city documentary Berlin Symphony of a Great City, the director used a sequence of shots containing objects moving in the same direction to indicate the train is coming to Berlin. This is an example of Rhythmic montage.

Other features have been used in montage editing include color, shape, texture, scale, photographic, focus, visual complexity, motion intensity, content, and emotion. Our system can be an assistant tool for selecting features and videos for montage editing.

2.1 Video Data

We extracted video feature data of the Seasons' video database. Original videos have the resolution of 1920×1080, which cause a massive amount of computation

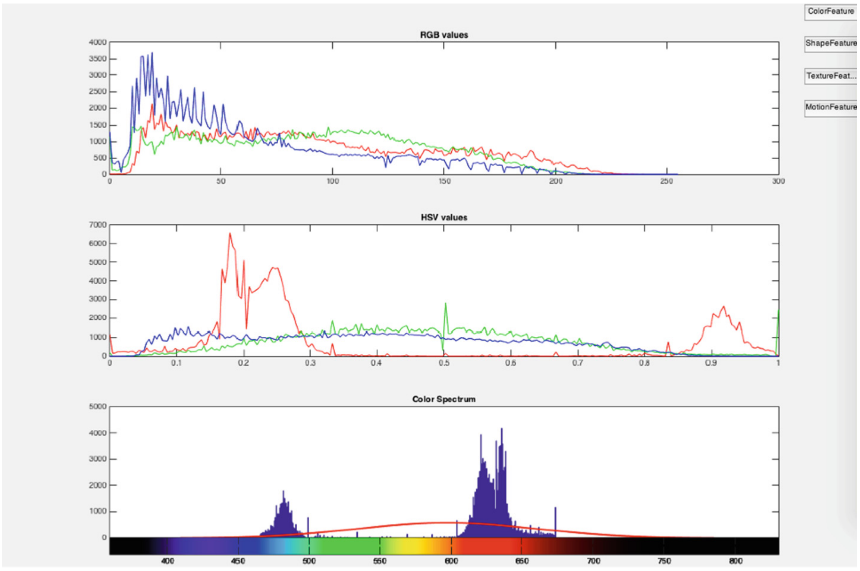

Fig. 1. A simple visualization of original color feature data of a target video (Color figure online)

regarding feature extraction and data visualization. We lower the resolution of video and extract features. Since there are many dimensions in video features, we separate video features into three categories including color, motion, and texture features. We used Matlab[2] to extract RGB and HSV values as color features, which are represented as a 2D chart, where the horizontal axis shows the frames and the vertical axis shows the value, as shown in Fig. 1. Matlab also extracts optical flows as motion features and represents this feature in each frame using directional lines as shown in Fig. 2.

We argue that these forms of visualization are not intuitive for users. To address this problem we selected only hue, saturation, and brightness, which we believe are more related to human perception of colors. Also, we rearrange the original video feature data for better visualization. It is hard to visualize each feature dimension frame by frame in real time. Therefore, we use probabilistic distribution to represent hue, saturation, and brightness. For motion features, we calculated the motion intensity value, motion direction value per frame and overall motion direction of the entire video. The motion intensity data is frame level data. That is also the case of texture feature visualization, which was represented using entropy and contrast. For texture features, we used entropy and contrast.

With both frame level feature data and statistics feature data of the entire video, users can have a macro view of the whole video that saves more time and has the micro observation of details of features. All the video features are extracted by using Matlab.

[2] Matlab is a developed by Mathworks.

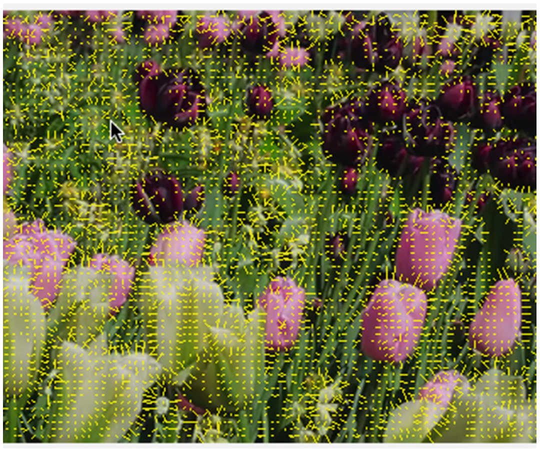

Fig. 2. Simple visualization of original motion feature data of a video (optical flow) (Color figure online)

We rearrange the original video feature data by using multi-view and representing each feature based on its probability distribution.

3 Method

We adopted the design science research, which is a method that addresses the development of an artifact while enables the researcher to learn a certain phenomenon [5]. Vaishnavi and Kuechler [6] suggested the design cycle used in this paper, which is composed of the following process steps: awareness of the problem; suggestion; development; evaluation; and conclusion.

The first step of the method involves the understanding the problem and the definition of the performance for the system. To do that, we adopted a systematic literature review. During the suggestion phase, we suggested a tool built based on multidimensional views of the multimodal data. The ViVid engine was developed based on bar charts visualization of saturation and brightness, radar visualizations for hue and overall motion direction, line graph visualization for motion intensity, pointer visualization for motion direction every frame, and bar charts visualization of contrast and entropy. The usability evaluation phase was conducted with montage editors, and the analysis of the results is written in the conclusion phase.

3.1 Related Works

The systematic literature review was performed in electronic databases: ACM library and Science Direct. As inclusion criteria, we opted only full papers published in the last five years. The keywords were chosen based on previous literature review and were defined as: Data visualization “AND” video feature, multiple windows “AND”

Table 1. Distribution of papers after applying the filters

Database	Data visualization "AND" video feature	Selected for reading	Multiple window visualization	Selected for reading
Science direct	41	2	75	0
ACM library	96	5	41	1
Total	137	7	116	1

visualization, which should appear in papers' title, keywords or abstract. In addition, only Arts and Humanities, Computer Science and Design sources were considered. Table 1 shows the quantity of papers found in each database and the quantity of papers selected for full reading after analyzing their abstract and relevance to this paper.

The problem of visualizing data extracted from videos is understood as visualizing multimodal data. For example, Rashid et al. [7] proposed a graph-based approach for visualizing and exploring a multimedia search result space. They explored the idea that each media can be accessed through different modalities (i.e., visual, acoustic or textual) depending on the considered low-level features or on the available metadata extracted from or associated with it. Their interface is composed of five panels: one for input and is a text window (query); one for the result of the query, which is also a text window; the next panel shows thumbnails of all media objects contained in the retrieved documents; for the media selected is possible to listen, watch or view; and finally a graph panel represents the selected media objects and its semantically related media objects. Multiple views are also used by Chandrasegaran et al. [8] to create a visual analytics framework named VizScribe that employs multiple coordinated multiple views that enable the viewing of multimodal data, such as audio, video, and text. Their interface layout is divided into two main sections: temporal view (time-series data) and transcript view (text). These data are displayed as interactive timeline views, such as video progress, transcript visualization, and sketch timeline. Video data is designed to answer the question: "what was happening when…?". The temporal view pane thus includes a video playback interface, with time encoded as a progress bar spanning the width of the pane.

Different signals and rates can also be synchronously visualized, played and re-arranged using the web interface by Mayor et al. [9]. They created the repoVizz, an integrated online system capable of structural formatting and remote storage, browsing, exchange, annotation, and visualization of synchronous multimodal, time-aligned data (audio, video, motion capture, physiological signals, extracted descriptors, annotations, etc.).

Other approaches to visualize video data take advantage of the data available from image domain. Liu and Shi [10] represented each frame by high-level attributes rather than visual features. They proposed a novel sequence for sequence architecture by using a pre-learnt image-captioning model to generate for video-description. Video sequencing can also be seen as synthesized visual storyline. Chen et al. [11] used clustering and automatic storyline extraction to generate these storylines. Video sequencing is a problem of detecting screen shot boundaries, which divides a video into groups with spatio-temporal similarities. Bhattacharya et al. [12] proposed a method to detect these boundaries and to generate video storyboards using the local features as well as the global features presented in the scene. Their method reserves the continuity by ensuring at least one frame from each shot avoiding scene kipping.

The content of a video can also be detected in real time. Tanase et al. [13] created the IMOTION, which is an interactive video retrieval system that offers a sketch-based user interface. The system recognizes in real-time the user's sketch and makes suggestions based both on the visual appearance of the sketch and semantic content.

4 Design Strategy

Based on the literature review, and given the multi-dimensional video feature data, we suggested multidimensional views and balanced spatial and temporal benefits using multiple views. ViVid is a tool created based on bar charts visualization of saturation and brightness, radar visualizations for hue and overall motion direction, line graph visualization for motion intensity, pointer visualization for motion direction every frame, and bar charts visualization of contrast and entropy. We listed the features that we used below.

- Color: Red, Green, Blue, Hue, Saturation, and Brightness.
- Motion: Motion Intensity, Motion Direction.
- Texture: Entropy, Contrast.

The background color is set to white so that it contrasts sufficiently with the object. As for each graph, we utilize different colors because they correspond to different meanings in the data [14]. We adjust the size of each graph so that viewers can observe both details and macro information. We use perceptual techniques to focus user attention. For example, we select the color that has highest hue value within the hue wheel as the color for making saturation bar chart, brightness bar chart and motion direction radar graph. This also keeps views and state of multiple views consistent. For motion direction, we use the same dataset but different encoding including frame level motion direction visualization and overall motion direction visualization. Users have the freedom to control the tool, load database, load target video, play and pause the video and visualization. The video feature extraction is implement in Matlab. The ViVid system is implemented in Java.

4.1 Color Feature Visualization Interface

When a user clicks the "Color" button on the left corner, the interface will appear. Figure 3 shows the result. The video player will play the video frame by frame (2 frame per second in the target video). Color feature is represented using hue, saturation and brightness features.

Hue is defined as "the degree to which a stimulus can be described as similar to or different from stimuli that are described as: red, orange, yellow, green, blue, violet" [15]. Because the hue is usually described using a wheel, for hue value visualization, we adopted a radar graph to visualization overall hue value, which is the average of

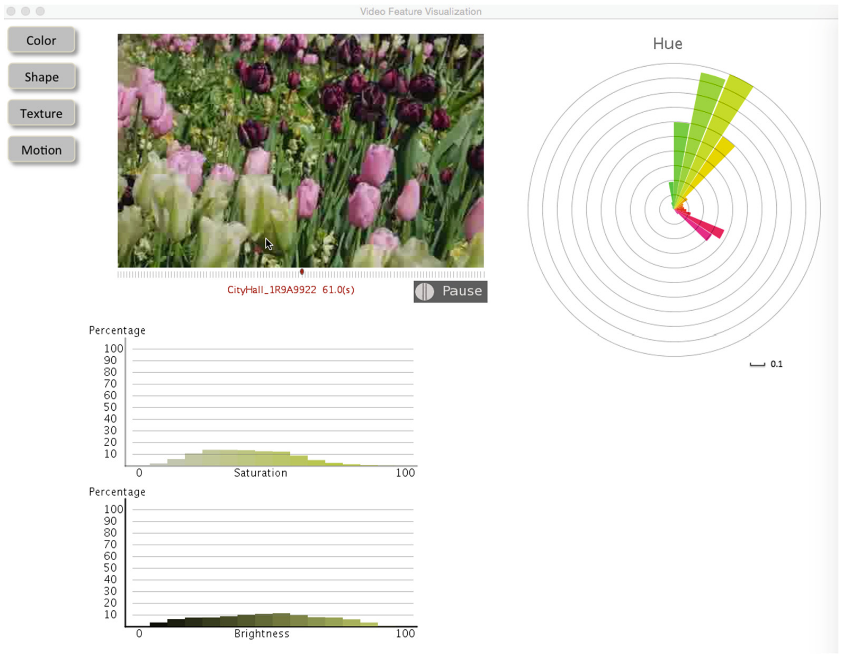

Fig. 3. Interface of color feature visualization (Color figure online)

every frame of a target video. We separated the hue wheel into 24 parts. In each part, we use inner circles to represent the number of pixels that have corresponding hue value. The highest value is normalized to 1. Between each circle in the concentric circle, the distance is 0.1. In Fig. 3, we can see that the highest hue value is light green, which is 1. The same normalization process is used to other values such as saturation, brightness, and motion direction. Once the system has found the highest hue value, it extracts the saturation and brightness values for this specific color. This brings users a consistent feeling of color style when watching the video and the visualization together. We normalized the biggest value of hue to 1 and made the visualization within a circle that is not big. Though it makes that the small hue value hard to be observed, it provides a better layout for users.

Saturation is the level of colorfulness, which also represents the level of purity of the color. The higher saturation value indicates less missing of colors. If we mix more colors together, the purity will decrease, and the picture will turn gray. We define the range of saturation value between 0 and 100 and separate the level of saturation into 16 parts, which is not too dense or sparse. The horizontal axis corresponds to each level of saturation, and the vertical axis represents the percentage of pixels that have the corresponding saturation value.

Brightness feature represents the level of a source appears to be radiating or reflecting light, which is the perception elicited by the luminance of a visual target [16]. Similar to saturation, we separated the brightness value into 16 parts. The horizontal axis corresponds to each level of brightness, and the vertical axis represents the percentage of pixels that have the corresponding brightness value.

4.2 Motion Feature Visualization Interface

While the color feature is presented based on its statistics and distribution, motion feature is visualized based on each frame. We use real-time data to visualize the motion intensity and motion direction per frame. In addition, we used the statistics of motion direction to represent the overall motion direction of the video.

Motion Intensity is the level of motion between frames within the video. We used a timeline based line graph visualization for motion intensity [15]. This allows the viewers to observe the motion intensity and its variation within the video. The timeline of the video player is horizontally aligned with the timeline of the motion intensity graph. The vertical axis indicates how intense the motion is. We separate the vertical axis into four sections, including low motion, medium motion, high motion and super high motion; each section is highlighted by an individual color. Rule of the separating point is based on the multiple observations of visualization of many videos. The horizontal axis represents time (Fig. 4).

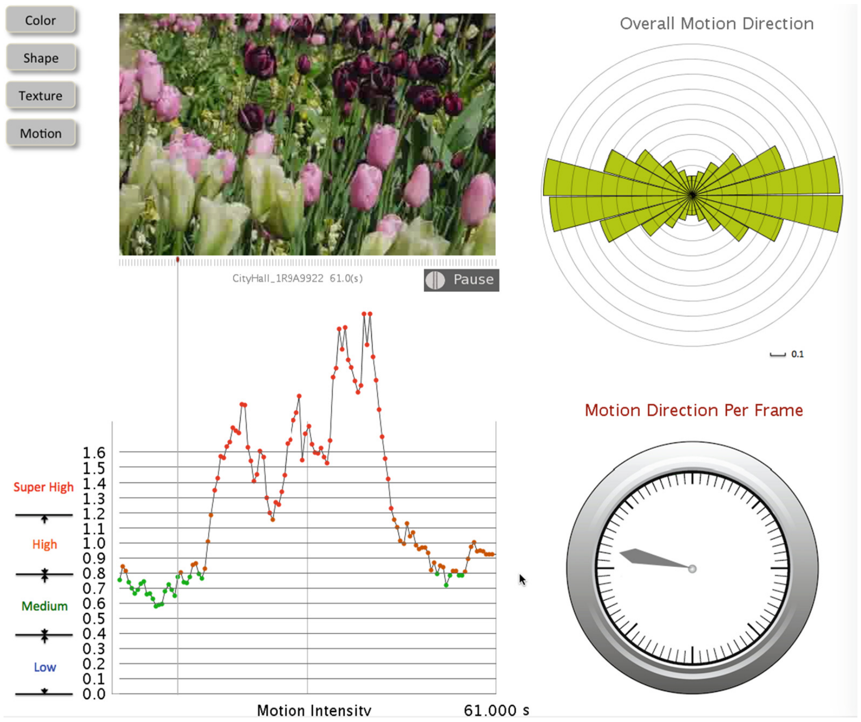

Fig. 4. Interface of motion feature visualization (Color figure online)

The bottom right corner shows the duration of the video. Since we want the user to compare the image of the frame and the feature of the frame at the same time, we used a pointer to connect the timeline with the horizontal axis of the motion intensity line graph. Each dot on line graph is corresponding to the time point of the video player. We enable the user to pause the player so that the user can view the feature data and image carefully. In the original data extracted by Matlab, we have optical flow vector of every moving point. To make a trade off and provide more useful information to users, we use statistics of the motion data of both frame-by-frame and overall video.

The motion direction can be defined using 360°. Therefore, we use the radar visualization for overall motion direction data, which is the average of every frame in a video. We separate 360° into 24 parts. We detect optical flow vectors and obtain its absolute value and its angle. Then we accumulate the optical flow vector value in each direction among 24 parts frame by frame and obtain the average motion direction value for an entire target video. In each part, we use the length of the radius of an arc to represent the number of pixels that moves toward the corresponding direction. When considering the absolute value, values of certain parts will be high and take space. Therefore, the biggest value of the motion direction is normalized to 1 to save space. Though it makes that the small hue value hard to be observed, it provides a better layout for users.

Because the user needs to observe the arc of motion direction within the video, we added motion direction per frame graph. This graph doesn't involve any intensity data but direction. When the video is playing, the pointer will move frame-by-frame to indicate the direction of the video.

4.3 Texture Feature Visualization Interface

We chose to visualize the entropy and the contrast feature to provide the information of texture of the video to users. Texture feature visualization is based on each frame. We used the distribution of texture features to represent the video (Fig. 5).

Entropy represents the level of randomness. Specifically, in image processing, entropy is a measure of the amount of information, which must be coded for by a

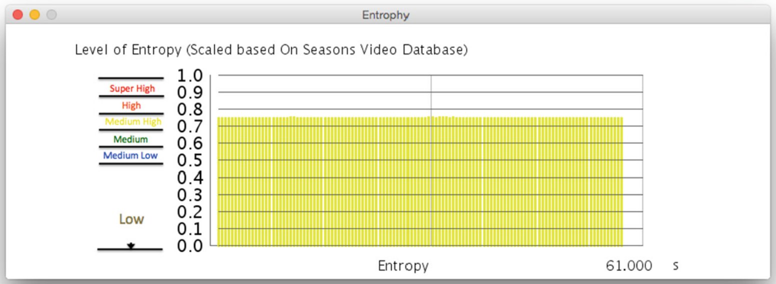

Fig. 5. Visualization of the entropy feature (Color figure online)

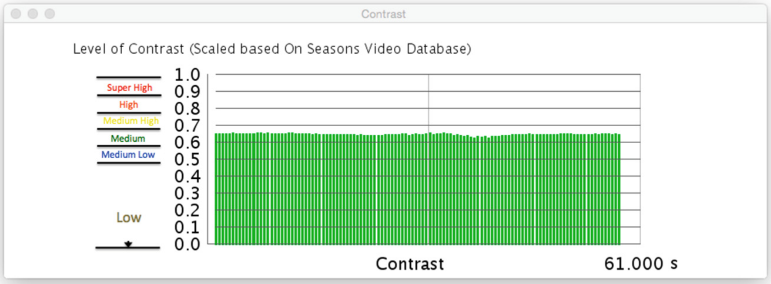

Fig. 6. Visualization of the contrast feature (Color figure online)

compression algorithm. If the image has higher entropy, such as an image of flowers, it contains more information. Low entropy images, such as those containing a white wall, have less information. Therefore, the high entropy images cannot be compressed as much as low entropy images. Similar to motion intensity, we used timeline based line graph visualization for entropy, which allows the visualization of this evolving entity over time. This allows the viewers to observe the change of entropy. The timeline of the video player corresponds to the timeline of the entropy graph. The vertical axis indicates how high the level of entropy is. We separate the vertical axis into six sections, including low, medium-low, medium, medium-high, high and super high; each section is highlighted by an individual color. The idea of the design is similar to the motion intensity (Fig. 6).

Contrast indicates how distinguishable regarding color that objects are within an image. This property is determined by the difference of color between the objects within an image. Similar to the design of visualization of Entropy, we used timeline based line graph visualization.

5 Usage, Feasibility Study and Considerations

The ViVid engine was built to help editors during the montage phase of a video. Thus, the system extracts and presents hue, saturation, brightness, motion direction and texture of a shot using multiple views. The interface was designed to focus attention on these data. The ViVid engine was used for the creation of the automatic music video generation system, DJ-MVP [17]. The systems used a color heuristic method for video selection. The target video segment is either closest or the furthest in the HSV color space to the previous video segment. This mechanism enabled the generative video to become more diverse regarding color. Examples can be found online.[3]

To evaluate the system, we conducted a feasibility study to understand how intuitive the interface is perceived by the users. The study was carried out at Simon Fraser University, and the system was presented using a 60” TV. Eight people were asked to

[3] https://vimeo.com/channels/djmvp.

discuss what they understood of each feature and how accurate these features were represented. Users gave positive feedbacks toward our video feature visualizations. They also made suggestions to improve the tool.

First, the system should accept frame or intervals (regions) or any length. So far, the tool presented in this paper finds the highest hue value for the whole shot and extracts the saturation and brightness values for this specific color.

Second, it is useful to normalize the vertical axis of saturation and brightness bar chart to make three visualizations of HSV consistent. Third, we should change the color of the arrow in the motion direction figure so that it matches the color in the motion intensity figure. Both suggestions help users to read multiple views. In addition, these perceptual cues make the relationship among views more clearly to the user.

Fourth, the interface should represent hue, saturation, and brightness divided into more than 24 parts to make the visualization more precise. This suggestion might increase cognitive attention demanded to perceive the information. Ideally, the information extracted should provide users a stable context of analysis without adding complexity. Thus, providing more detailed information might be an option provided by the system, while the default could remain as 24.

In addition, to obtain a clear understanding of each feature and the visualization tool, users recommend us to give a detailed explanation of each feature in the interface. These explanations would be useful when learning about the interface. All these feedbacks will be considered in further development of the tool.

Acknowledgement. We would like to acknowledge the Social Sciences and Humanities Research Council of Canada and Ministry of Education CAPES Brazil for their ongoing financial support. And we would like to thank the reviewers, who through their thoughtful comments have been assisting with this publication.

References

1. Galanter, P.: What is generative art? Complexity theory as a context for art theory. Digit. Creat. (2009)
2. Bizzocchi, J.: Ambient Video (2012). https://ambientvideo.org/seasons/
3. Eisenstein, S.: Film Form: Essays in Film Theory. Harcourt Brace and Company, New York (1949)
4. Walid, M.: Eisenstein and Montage: Battleship Potemkin, essay. https://www.academia.edu/7875638/Eisenstein_and_Montage_Battleship_Potemkin
5. Bunge, M.: Philosophical inputs and outputs of technology. In: Doner, D. (ed.) The History and Philosophy of Technology. University of Illinois, Champaign (1979)
6. Vaishnavi, V., Kuechler, W.: Design Research in Information Systems (2011). http://desrist.org/design-research-in-informationsystems
7. Rashid, U., Viviani, M., Pasi, G.: A graph-based approach for visualizing and exploring a multimedia search result space. Inf. Sci. **370–371**(20), 303–322 (2016)
8. Chandrasegaran, S., Badam, S.K., Kisselburgh, L., Peppler, K., Elmqvist, N., Ramani, K.: VizScribe: a visual analytics approach to understand designer behavior. Int. J. Hum.-Comput. Stud. **100**, 66–80 (2016)

9. Mayor, O., Llimona, Q., Marchini, M., Papiotis, P., Maestre, E:. repoVizz: a framework for remote storage, browsing, annotation, and exchange of multi-modal data. In: Proceedings of the 21st ACM International Conference on Multimedia, MM 2013, October 2013
10. Liu, Y., Shi, Z.: Boosting video description generation by explicitly translating from frame-level captions. In: Proceedings of the 2016 ACM on Multimedia Conference, MM 2016, September 2016
11. Chen, T., Lu, A., Hu, S.: Visual storylines: semantic visualization of movie sequence. Comput. Graph. **36**(4), 241–249 (2012)
12. Bhattacharya, S., Gupta, S., Venkatesh, K.S.: Video shot detection & story board generation using video decompositio. In: Proceedings of the Sixth International Conference on Computer and Communication Technology, ICCCT 2015, September 2015
13. Tanase, C., Giangreco, I., Rossetto, L., Schuldt, H., Seddati, O., Dupont, S., Altiok, O.C., Sezgin, M.: Semantic sketch-based video retrieval with autocompletion. In: Companion Publication of the 21st International Conference on Intelligent User Interfaces, IUI 2016 Companion, March 2016
14. Few, S.: Practical Rules for Using Color in Charts, Perceptual Edge Visual Business Intelligence Newsletter February (2008)
15. Van Wijk, J.J.: Cluster and calendar based visualization of time series data. In: Proceedings of INFOVIS 1999, pp. 4–9 (1999)
16. Vadivel, A., Sural, S., Majumdar, A.K.: Robust histogram generation from the HSV space based on visual colour perception. Int. J. Sig. Imaging Syst. Eng. InderScience **1**(3/4), 245–254 (2008)
17. Fan, J., Li, W., Bizzocchi, J., Bizzocchi, J., Pasquier, P.: DJ-MVP: an automatic music video producer. In: Proceedings of the Advances in Computer Entertainment Technology Conference, ACE 2016, November 2016