Automatic home video editing on music

F.R.T. Weers
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
research@weers.dev

ABSTRACT

A lot of videos are recorded during events (such as symposia), trips or other special occasions (like a wedding). Watching all the raw content is boring and takes a lot of time. Edited videos can summarize the experience in a fun and interesting way, but movie-editing still requires a lot of manual work and expertise. To solve this issue, we propose a method to automatically select important parts of photos and videos, analyze music for beats and use this to generate a short movie with transitions in rhythm. Automatic selection of important parts is based on motion detection, infidelity and simplicity. Combining the chosen song with the photos and videos is a constrained optimization problem and two solutions are given.

Keywords

Video editing, Video summarization, Automatic after-movie, Beat synchronization, Constrained optimization problem

1. INTRODUCTION

Nowadays, everyone has a mobile phone with a camera that is able to record videos. When family or friends are talking about how good their holidays were, they still use pictures to convey the feeling though. The minority is sharing videos to summarize their vacation, because the videos are often found to be long and boring to watch. It is also very hard to find the interesting parts of the many videos to reduce the viewing time. A compilation, or after-movie, would be more entertaining and would summarize the experience in a short amount of time, but creating such a movie is a very time-consuming task and requires experience with extensive software. This research will investigate whether we can automatically generate a professional after-movie of events, weekends or complete week trips. We can also use the same approach to summarize a more professional event, like a symposium. Since no manual editing would be required anymore, no money has to be spent on hiring people to do so and more money can be used to increase the value of the actual event.

To solve this, automatic selection of interesting images and parts of videos has to be done. Beats are detected on the chosen song and the selected media is combined with the beats to get an after-movie. Combining the analyzed media with the beats is a constrained optimization problem. The proposed method can generate after-movies in less than 5 minutes for long weekends. Media from a week trip will take longer to analyze and thus exceeds the 5 minutes.

31st Twente Student Conference on IT July. 5th, 2019, Enschede, The Netherlands.

Research questions

The following research questions will be answered in the paper.

RQ1 How do we improve automatic movie-generation of home-videos?

- **RQ1.1** What algorithms should be used to determine what parts of videos and photos are interesting?
- **RQ1.2** What algorithms should be used to recognize beats in the provided music?
- **RQ1.3** What algorithms should be used to get a high quality combination?
- **RQ1.4** How do we get reasonable efficiency when combining music sections with video sections?

RQ2 How do the generated movies using the proposed method compare to (generated) movies using existing software/methods?

Background

In order to understand further decisions made in both this paper and other researches, some background information is required. This section will provide the necessary information.
To select what photos are used in the after-movie, or what part of a video, the quality of the content is determined. Whether a video or photo is of high quality is affected by a few factors. Quality is defined by looking at e.g. contrast, colors, brightness, jerkiness, infidelity and/or orientation [10, 18, 26, 27, 20, 16, 11]. Motion, like a tilt, is also very useful for categorizing video quality [16, 11, 24]. Important parts of videos or photos are also defined by looking at the number of visible faces [11].
Music is used in the resulting movie as background audio and all transitions should be in rhythm. It is shown that listeners judge the identical video to be of higher quality when it has higher-fidelity audio [17]. It is assumed that synchronizing video and audio segments enhance the perception of both. This is common practice in the film industry.

2. RELATED WORK

A lot of research has been done around analyzing videos, photos and music. Automatic video editing, or generation of after-movies, has also been researched before, but to a smaller extent. We will discuss the related work in this section.

As discussed in the previous section, various properties of a photo are important to decide whether the photo is of high quality. Various researches [4] have been done where photo quality is automatically assessed and even improved by editing. Face recognition is a well-established research area. A lot of algorithms [6, 19] exist to detect how many faces are seen and who is in the picture.

Various algorithms that are used to determine whether a photo is of high quality, are also used to detect if (part of) a video is of high quality. Since running the algorithms every frame is too slow, other researches ran it every 15 frames and assume no important changes are missed [22]. Cutting in a video should also respect its corresponding audio, a cut while someone is in the middle of a sentence is not preferred [18] for example.

Various studies about analyzing music have been done. Beat [11] and tempo [8] detection have been used in the past.

Previously, two main methods have been used to combine the analyzed audio and video.

Align to video In this case video summarization is the main goal. Cuts should be visually appealing and the viewer should understand the story of the movie. Most related work [23, 28] has used this approach to summarize events or tv series.

Align to audio With this method, audio or time is leading. This is used to summarize e.g. concerts with multiple viewing points [20, 21, 12] or to generate music-videos.

Evaluating

Whether video editing is done right, is mostly subjective in the end. No standard evaluation method exists for video editing, but earlier works have both subjective and objective evaluations. The content of the finished movie can be compared to the content that was decided to be valuable beforehand. After editing the movie, high quality images/videos should be used and included in the result [11]. The movie should also use properties of the music, like beats, for transitions [11]. Computation time is also considered during evaluation [25]. To subjectively evaluate the result, users have been asked to rate movies [25, 11].

This work will contribute to the research topic by using new and more up-to-date algorithms for media analysis. Properties of the song will be used to model transitions and these will also be used to combine the media with the music in an entertaining way. Furthermore, no other research has been done considering video editing on music for a relatively long time (last, most related one was in 2004 [11]), while the entire tech industry is moving at a rapid pace. As a final note, we add a computational time constraint of 5 minutes to the software.

3. METHODOLOGY & APPROACH

This section will describe what steps will be taken to complete the research. The overall program flow can be found in Figure 1.

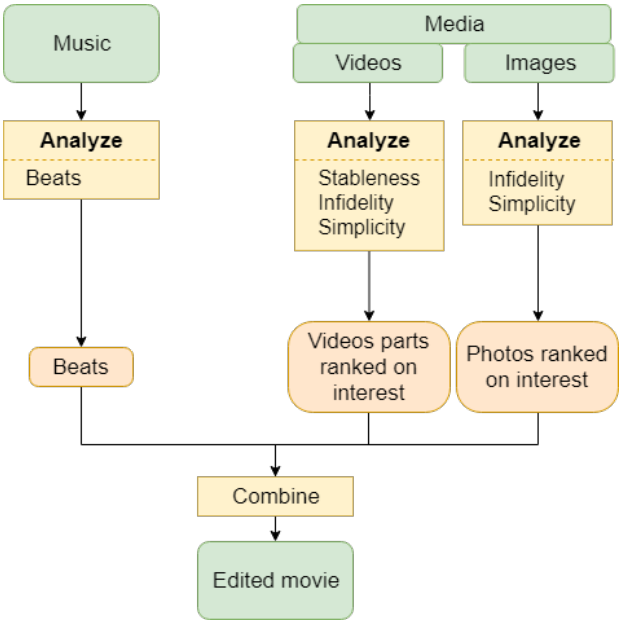

Figure 1. The overall program flow.

3.1 Approach

Analyzing media

All videos and photos have to be analyzed and ranked on interest. Interest is calculated by formula 1.

$$\begin{aligned} interest = {} & w_{stability} * stability + \\ & w_{infidelity} * infidelity + \\ & w_{simplicity} * simplicity \end{aligned} \tag{1}$$

Stability is calculated using motion estimation, which is implemented in the MPEG encoding domain using the FFmpeg library [7]. Motion vectors are extracted from the MPEG videos and an affine motion model is estimated. A motion vector $\begin{bmatrix} x \\ y \end{bmatrix}$ is expressed as

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{2}$$

at the position of a macroblock $\begin{bmatrix} X \\ Y \end{bmatrix}$, where

$\phi = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \end{bmatrix}$ is the parameter estimated from the Motion Vector Fields (MVF) of the undergoing frame by using the RANSAC method. We choose the RANSAC method instead of the least square method because of better results [2]. Vector elements of the parameter are used to get pan (horizontal movement), tilt (vertical movement) and zoom (forward/backward movement) [16] and rotation [14] values:

$$pan = b_1 \qquad tilt = b_2$$

$$zoom = \frac{1}{2}(a_{11} + a_{22}) \qquad rotation = \frac{1}{2}(a_{21} - a_{12})$$

The four motions are calculated for each frame. For zoom detection, sign validation is performed to improve accuracy. To estimate motions from the MVF parameters more accurate, magnitude- and temporal thresholding are applied. For the simple movements (pan and tilt) we only consider the movement if its magnitude is at least 1. For the other motions (zoom and rotation), we use 0.015 as the minimum. To remove very short motions that are not

significant, a temporal threshold is applied. A motion has to be consistently detected for at least 15 frames. These values are supported by earlier work [14]. Based on the detected motions, the video is split up into subshots. To reduce the number of subshots that are too short, motions are split into two categories: primary motions (zoom and rotation) and secondary motions (pan and tilt). If a motion starts, a new subshot is made if no primary motion is currently detected. If a primary motion ends, and no other primary motions are currently detected, a new subshot is created.

Using the motion values, the unstableness of a frame is computed as $\frac{max\{D_i\}}{T_\theta}$ (from [16]), where D_i is the number of direction changes of a component $i(i = pan, tilt, zoom)$ and T_θ is the duration of subshot θ.

The unstableness of a subshot is calculated by the percentage of frames that are considered unstable. A subshot has two regions, an important region and a soft region. The important region is the hard minimum of the video and if the subshot is chosen to be in the final movie, the whole important region should be in the movie. To decide what part is the important region and what part is the soft region, we start in the middle and detect the very first frame on either side that is defined unstable. From that frame to the start (or end) of the subshot is defined as soft. The part in the middle is defined as important. The soft region is used to make the subshot fit the beat-space and make transitions in rhythm.

Infidelity, simplicity and face detection are done on both images and videos. For subshots, we run it every 15 frames and take the average value. Low contrast and lack of colors are represented by infidelity [16] and calculated using formula 3.

$$infidelity = 1 - \sqrt{En(\theta)} \tag{3}$$

where $En(\theta)$ is the color entropy of a subshot θ. Let i(i=H,S,V) be the value in the HSV color space. The color entropy of this is defined as

$$En(i) = -\frac{1}{\log N} \sum_{i=0}^{N-1} p(i) * \log p(i) \tag{4}$$

where $p(i)$ is the probability of i appearing in an image/frame

$$p(i) = \frac{hist(i)}{\sum_{i=0}^{N-1} hist(i)}. \tag{5}$$

Since we already have access to the colors of frames in HSV space, because we use it for entropy, we also calculate the simplicity [13] of frames. To calculate the simplicity, we calculate the hue count. A histogram H, with 20 bins, is computed on pixels with *Value* in the range $[0.15, 0.95]$ and $Saturation > 0.2$.

$$N = \{i | H(i) > \alpha m\} \tag{6}$$

where m is the maximum value of the histogram and α is used to control the noise sensitivity. We use $\alpha = 0.05$, which is supported by [13]. We can then calculate the simplicity using

$$simplicity = 20 - ||N||. \tag{7}$$

Face detection is performed (using [5]) and is used to calculate the minimum required duration of an image. Beat detection is done on the music (implemented using [15]) and will result in a list of time-stamps of beats.

A survey is made to get data about grading of videos and images, and to get a formula for the minimum duration of an image in relation with the number of visible faces. In the survey, 30 videos will be shown with different stability, infidelity and simplicity values. Another 22 images will be shown with unique infidelity and simplicity values. As a final exercise, the users will be shown a small compilation of images and they will be asked whether the video goes too fast, too slow or was perfect considering the duration of individual images. Three of those compilations will be shown with 2, 4 and 6 faces (every compilation consisted of 5 images).

Combining image and music

When both images/videos and music have been analyzed, we have to select segments that fit properly with the music. Since selecting an image/video is not just affected by the score of the image/video, but also by relations between media, a graph is created that represents the analyzed data (see Figure 4 for an example). Nodes represent an image or a subshot of a video and edges represent the relation between two images/videos. An empty start node is added at the beginning and is used as the very first node of the after-movie. The 'Finish' node is added at the end and is used as the destination node. The graph is directed, acyclic and complete within the DAG domain. The *interest* of the target image/video is used as the weight of the edge, a distance bonus is added to increase selection of nodes at a certain distance from this node. This is used to improve overall coverage and makes sure no clusters of images/videos are selected for the final movie, but instead the whole raw data-set is used. The optimization problem of selecting images/videos can now be seen as a path-finding problem with the following constraints:

Weights The path should use the 'best' edges, by selecting edges with the highest score.

Duration The final movie should be as long as the supplied song is.

Coverage Instead of selecting a cluster of high quality images/videos, it is important that the whole event is covered. Thus media from the whole time-line should be used. The coverage of a path is calculated using Algorithm 1.

Algorithm 1: Calculating coverage of a path

Input: *path*: List of nodes, n: Number of nodes
Output: Coverage value
index = 0
time = $\frac{1}{n}$* song_duration
result = [ ]
foreach *node* **in** *path*:
    target = index * time
    if *target* > *node.time_stamp* **and** *target* < *node.time_stamp* + *node.duration*:
        cov = 0
    else:
        `# Take squared distance`
        cov = (target - node.time_stamp)^2
    result.append(*cov*)
    index += 1
distance_to_ideal = mean(*result*)
coverage = 1 / *distance_to_ideal*
return *coverage*

The requirement that the video should be chronologically ordered is constrained by the graph design. Both the dura-

tion and coverage constraints make this an NP-hard problem, since it is at least as hard as the Weight Constrained Shortest Path Problem. We have implemented two ways to solve this problem:

Parameter learning (PL) Using BFS (or A*) we find a path that uses edges with the highest weight. The path is evaluated after it has finished and both the number of remaining nodes as the number of unused beats of the song are checked. Both these values are used to determine whether we selected too many or too few items from the graph. The *distance_factor* is increased or decreased. After a few runs a selection is made that fits the music duration.

Heuristic After the graph is created, a tree is constructed (an example of such a tree is shown in Figure 2). The tree keeps track of possible paths, their total duration and their average edge weight. Paths stop exploring once they reach a duration that is equal to the length of the song.

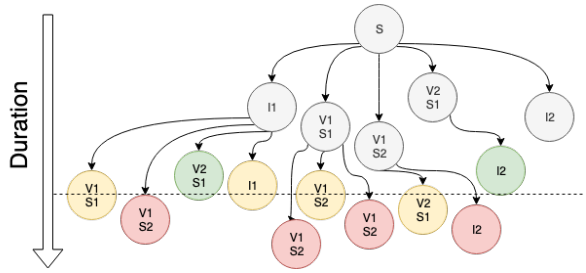

Figure 2. A tree representing possible paths.

In both cases a video should fit the beat-space. Figure 3 shows how this is done. A subshot has an important region and a soft region. The important region is the hard minimum of the video. If that does not fit properly within the beats, the soft region is used to fill the rest and make it align properly with the beats. If the soft region of a subshot is not long enough, the left or right node will be used (as long as they have the same source) as extra soft region. If it still does not fit, this node cannot be used and another one will be tried.
Both solutions result in a path with information about the duration of the chosen images/videos. Using FFmpeg [7] the content will be properly scaled and concatenated to create the movie.

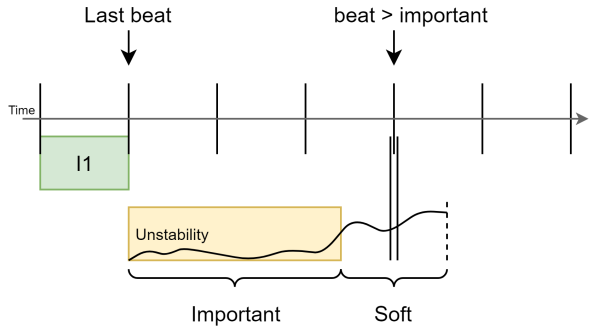

Figure 3. The fitting process at work.

Optimization

To make sure the video is edited within a reasonable amount of time, full decoding of the videos should be avoided during pre-processing. Decoding is a computationally expensive operation and thus, optimized algorithms should be used to analyze the video without decoding it. Camera motion estimation can be done by partially decoding the video using motion vectors from the MPEG encoding [16, 14, 1]. Parallelization is a good way to increase performance. Music analysis can be done simultaneously with media analysis. Analysis of different photos and videos can also be done in parallel. Part of the heuristic approach to solving the path finding problem can run in parallel as well. Instead of creating the tree graph using only one thread, we can split the task in multiple smaller tasks. Each child of the very first node will get their own processing space and will create the tree for a small part. While it is still a very computationally expensive task, it will improve the performance.

3.2 Evaluation

While many parts of video-edited movies are subjective, some objective evaluations are possible and will be defined.

Subjective evaluation

A group of people will be asked to rate edited videos. With the same content, three different ways will be used to edit the video.

- Videos automatically edited by the proposed method
- Videos automatically edited by other state of the art methods (Google Photos [9] and iMovie [3])

Using these results we can determine whether our video editing solution is better than the state of art, but we can also determine what can be improved in general on automatic video editing.

Objective evaluation

To determine whether the resulting video is objectively of high quality we use several statistics, based on the defined rules.

Quality of chosen photos Based on infidelity and simplicity we can detect whether an image is of high quality. A picture with lower quality should not be used if a better picture is available.

Quality of chosen video clips Based on motion, audio and features that define a high quality video, the content in the resulting video should be of high quality.

Photo duration Every photo should have a minimum duration so that viewers can understand what they are seeing. This minimum duration should always be respected.

Unique Every photo or video in the movie should be unique.

In rhythm Close to 100% of the transitions should be on beat.

Computation time The complete video should be done within a certain amount of time. We set this at 5 minutes.

4. RESULTS

The results will be discussed in this section.

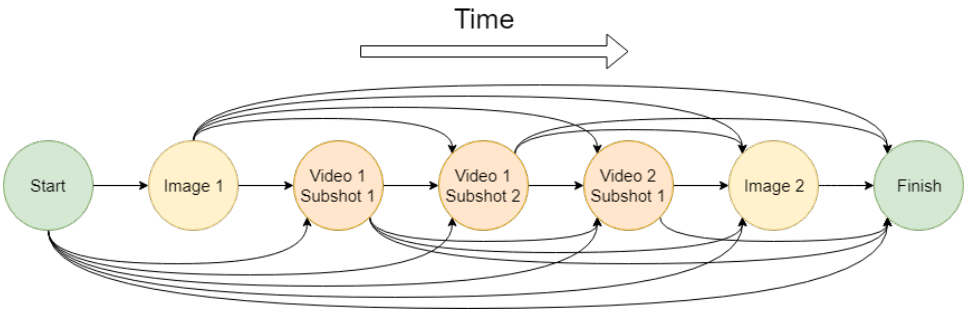

Figure 4. An example of a graph representing the raw footage.

Media Analysis

Figure 5 shows the different motions after applying magnitude thresholding. In Figure 6 temporal thresholding has been applied as well and the different subshots (the vertical lines) have been assigned.

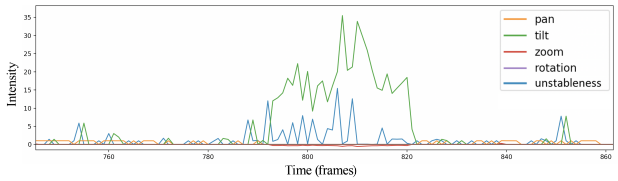

Figure 5. Motions after magnitude thresholding is applied.

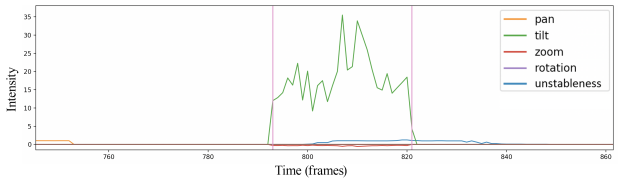

Figure 6. Final result of the motions, with two splits for subshots.

Graph creation

After all input is sorted chronologically (based on exif data), a directional acyclic graph (DAG) is created (see Figure 4). To speed up the combining of media with beats, the graph is simplified. Since one of the requirements is to have a high coverage, no edges are made between two nodes that have more than $0.3 * total\_media\_duration$ seconds between them. To improve coverage, a bonus is added to edges of a certain time distance. The bonus and offset values are calculated for every edge that is added to the graph. In equation 8 the *duration* is the duration of the source node and the *total_duration* is the sum of all durations.

$$offset = duration * \frac{total\_duration}{song\_duration} \tag{8}$$

$$bonus = 0.5 - (\frac{2}{offset} * (\Delta t - offset))^2 \tag{9}$$

In equation 9, Δt is the time in seconds between two media if one would chronologically sort all media and put it after each other.
The weight of an edge is then calculated by adding the *interest* and its *bonus* together.

User study

Appendix A shows the statistical summaries and results of the user study. The minimum required duration for images can be calculated using linear equation 10.

$$min\_dur = 0.099 * faces + 1.788 \tag{10}$$

The equation for the interest of a video is also derived from the results.

$$\begin{aligned} video\_interest = {} & w_{stability} * stability + \\ & w_{color\_quality} * color\_quality + \\ & w_{simplicity} * simplicity \end{aligned} \tag{11}$$

where $w_{stability} = 0.151$, $w_{color\_quality} = -4.056$ and $w_{simplicity} = -0.25$ gave good results (and are supported by Table 5).

$$\begin{aligned} image\_interest = {} & w_{color\_quality} * color\_quality + \\ & w_{simplicity} * simplicity \end{aligned} \tag{12}$$

where $w_{color\_quality} = -8.838$ and $w_{simplicity} = 0.014$ (again supported by Table 5).

Combining image and music

Because of the bad complexity of the heuristic solution, the heuristic approach is very slow. Even after adding pre-processing of the graph, which reduces the number of edges, we still had to limit the number of nodes that will be explored. Every node only checks its best 4 children, instead of all other nodes. While adding this rule removes some possible paths that are better, it speeds up the overall process. Without this rule, not even a 30 seconds path can be found within 5 minutes (excluding time spent on media analysis and rendering). Figure 7 shows experimental results. All computations were done on a 2017 Macbook Pro (4 cores, 2.8 Ghz, Intel Core i7).
We compare paths by calculating the average edge weight and its coverage (see Algorithm 1).
Both solutions, parameter learning and heuristic, give good paths (as can be seen in Table 1). While the heuristic approach will get (very close to) the absolute best path, it is not usable for large graphs or long videos. The growth rate of the algorithm is n^d, where n is the number of children that will be explored and d is the depth. Since PL simply uses BFS, runs it 10 times and adjusts its weight formula, its complexity is $O(V + E)$. From Figure 1 we can conclude that the average weight of the chosen edges is close to the average weight the heuristic approach gives. The coverage (calculated using Algorithm 1) is a lot better (40% better) with PL, since the heuristic approach tries to maximize the score and does not try to minimize its

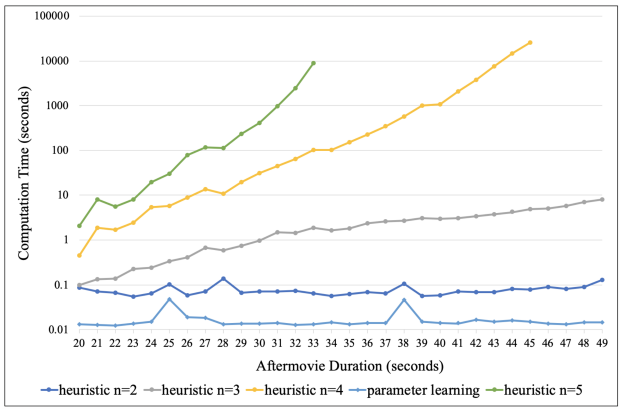

Figure 7. Results of the performance analysis.

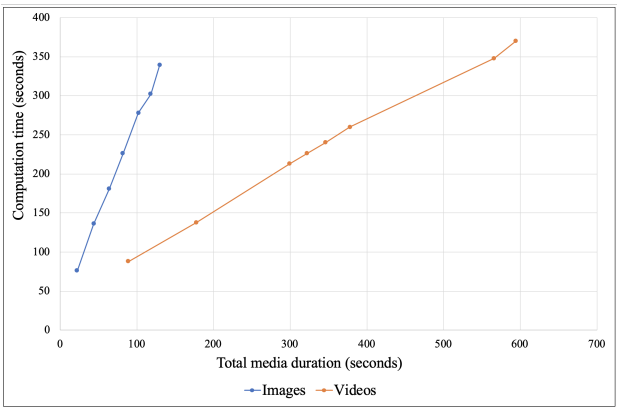

Figure 8. Input size vs total computation time (using PL).

coverage. Looking at the results of the two approaches of combining images and videos into a movie, parameter learning is a better way of solving the problem. The computation time is within the time limit, while the quality (average edge weight) is close to the heuristic approach.

Table 1. Path quality (after-movie duration from 20-50 seconds)

Type		Mean weight	Mean coverage^{-1}
Heuristic	n=2	7.3	140450
	n=3	7.3	141347
	n=4	7.3	141509
	n=5	7.3	139378
PL		6.2	80321

Performance

A big constraint in the research is the maximum computation time the software was allowed to use (5 minutes). Since the speed depends on a lot of factors, like the amount of media that is provided, the length of the chosen song, but also the specific MPEG encoding (more motion vectors or fewer I-frames result in longer computation times) and the resolution of the input, it is hard to get an accurate estimation of the performance. Figure 8 shows the effect of the number of input files to the total computation time (computed on a MacBook Pro 2017, 4 cores, Intel Core i7). Two lines are plotted, *Images* only contains images and *Videos* only contains videos. The computation time of a combination of videos and images can be found in-between the two extremes. All computations in the graph were done using PL. If we would use heuristic, the results would take on average more than twice as long. All computationally expensive tasks are written in C and C++ and run in parallel. One python program controls the output of each task and takes care of the program flow. A video is partially decoded once to calculate infidelity and simplicity.

Evaluation

Both a subjective and an objective evaluation have been performed. The following list contains all the objective requirements, with a small explanation on why the proposed method did or did not met it.

Quality of chosen photos Edge weights are based on the *interest* of an image. This is based on the infidelity and simplicity. In both heuristic and PL the edges with highest weight are chosen and thus this is respected.

Quality of chosen video clips Edge weights to a subshot are also based on the *interest* of the subshot. Similarly to photo quality, this is respected.

Photo duration Equation 10 is used to ensure that the minimum duration with relation to the number of visible faces is calculated. This minimum is always used and thus this is repsected.

Unique Every node in the graph is unique and since the graph is acyclic, no nodes can be used twice. Thus every image/video is unique.

In rhythm Images and videos are only used if they fit inside a beat (as explained in Figure 3). The beat detection is used to make sure everything is in rhythm.

Computation time The whole movie is created and rendered in less than 5 minutes, assuming the number of given images/videos and the duration of the song are reasonable.

We compared the proposed method with automatic movies generated using iMovie and Google Photos. Overall, aftermovies generated using Google Photos were considered to be of the highest quality. The proposed method came second and iMovie was third. The subjective results are further explained in Appendix B.

5. CONCLUSION

We have developed a system that selects high quality images/videos from a large list, detects beats in the chosen music and combines the two for an after-movie that has transitions in rhythm. While other research had already shown it was possible, our system takes less than 5 minutes and works for both images as videos. Our user study also gave more insights in how quality of videos/images can be assessed.

Discussion & Future work

The factors unstableness, simplicity and infidelity only affect the grade of a video by 28.3% (as can be seen in Table 4). This means that a lot more factors determine the actual quality of a video. Further research should be done to explore what these other factors are and how they affect the results. Similar comments can be made about the simplicity and infidelity that affect the grade an image gets (14.2%, from Table 6). To improve performance, more research can be done in solving the optimization problem (that we currently solve heuristically and using parameter

learning). Currently, infidelity and simplicity is calculated and frames have to be decoded to get the HSV values of the frame. Instead of simply doing this every 15 frames, I-frames can be used to reduce overhead and get the HSV values without decoding unnecessary frames.
While analyzing the media, certain parameters are set (like the temporal thresholding minimum, or the fact infidelity and simplicity of videos are only analyzed every 15 frames). With high speed content, like sports, every 15 frames might be too little. With a very slow nature scene, analyzing every 15 frames might be unnecessary. A lot of methods have been shown that can detect what happens in a scene (using e.g. machine learning) and these can be used to first determine whether the setting is fast-paced or not. This would both improve accuracy of the results and it would increase performance in case we do not have to analyze as often as we currently do.
When trying to fit a node in a beat-space, its left and right nodes are used as extra soft region space. The quality of these two other nodes is not taken into account and the overall quality might be lower by using these to fill up space. When calculating a path using Parameter Learning, this should be taken into account when selecting a child node based on its score.
When using heuristic to get the best path, the mean edge weight is used as a measure. Coverage is calculated, but only used when comparing heuristic versus PL. It is currently unknown how much value people give to coverage versus quality (score). More research needs to be done to come up with an equation that objectively selects the best path, taking both coverage and score into account.
The system itself is built to support specific relations between footage (the edges in the graph). This means that similar coloring of two images can be rewarded or special transitions can be added if the same face is detected in two images/videos. There is currently not enough data that supports these features though and more user-studies should be performed.

6. REFERENCES

[1] J. Almeida, R. Minetto, T. A. Almeida, R. da S. Torres, and N. J. Leite. Robust estimation of camera motion using optical flow models. *Lecture Notes in Computer Science*, pages 435–446, 2009.

[2] J. Almeida, R. Minetto, T. A. Almeida, R. da S. Torres, and N. J. Leite. Estimation of camera parameters in videos sequences with a large amount of scene motion. 2010.

[3] Apple. imovie, April 2019.

[4] S. Bhattacharya, R. Sukthankar, and M. Shah. A framework for photo-quality assessment and enhancement based on visual aesthetics. *Proceedings of the international conference on Multimedia - MM '10*, 2010.

[5] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[6] K. Chengeta and S. Viriri. A survey on facial recognition based on local directional and local binary patterns. (17806160), March 2018.

[7] F. developers. ffmpeg tool (version 4.1.3), June 2019.

[8] J. Foote, M. Cooper, and A. Girgensohn. Creating music videos using automatic media analysis. 2002.

[9] Google. Google photos, April 2019.

[10] A. Hampapur, R. Jain, and T. E. Weymouth. Production model based digital video segmentation. *Multimedia Tools and Applications*, 1(1):9–46, Mar 1995.

[11] X.-S. HUA, L. LU, and H.-J. ZHANG. Ave - automated home video editing. 2004.

[12] T. Jehan, M. Lew, and C. Vaucelle. Cati dance. *Proceedings of the SIGGRAPH 2003 conference on Sketches applications in conjunction with the 30th annual conference on Computer graphics and interactive techniques - GRAPH '03*, 2003.

[13] Y. Ke, X. Tang, and F. Jing. The design of high-level features for photo quality assessment. June 2006.

[14] J.-G. Kim, H. S. Chang, J. Kim, and H.-M. Kim. Efficient camera motion characterization for mpeg video indexing. August 2000.

[15] McFee, Brian, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto. librosa: Audio and music signal analysis in python. *In Proceedings of the 14th python in science conference*, pages 18–25, 2015.

[16] T. Mei, X.-S. Hua, C.-Z. Zhu, H.-Q. Zhou, and S. Li. Home video visual quality assessment with spatiotemporal factors. (9482130):699 – 706, June 2007.

[17] W. R. Neuman. *Beyond HDTV : exploring subjective responces to very high definition television.* Media Laboratory Massachusetts Institute of Technology, Cambridge Mass., 1990.

[18] S. Pfeiffer, R. Lienhart, S. Fischer, and W. Effelsberg. Abstracting digital movies automatically. *Journal of Visual Communication and Image Representation*, 7(4):345 – 353, 1996.

[19] H. A. Rowley. Neural network-based face detection. page 149, May 1999.

[20] M. K. Saini, R. Gadde, S. Yan, and W. T. Ooi. Movimash. *Proceedings of the 20th ACM international conference on Multimedia - MM '12*, pages 139–148, Oct 2012.

[21] P. Shrestha, P. H. de With, H. Weda, M. Barbieri, and E. H. Aarts. Automatic mashup generation from multiple-camera concert recordings. *Proceedings of the international conference on Multimedia - MM '10*, 2010.

[22] M. Smith and T. Kanade. Video skimming and characterization through the combination of image and language understanding. (5823100), JAN 1998.

[23] T. Tsoneva, M. Barbieri, and H. Weda. Automated summarization of narrative video on a semantic level. (9880178), Sept 2007.

[24] K. Uehara, M. Amano, Y. Ariki, and M. Kumano. Video shooting navigation system by real-time useful shot discrimination based on video grammar. (8304301), June 2004.

[25] J. Wang, E. Chng, C. Xu, H. Lu, and Q. Tian. Generation of personalized music sports video using multimodal cues. (9370072):576 – 588, March 2007.

[26] M. M. Yeung, B.-L. Yeo, W. H. Wolf, and B. Liu. Video browsing using clustering and scene transitions on compressed sequences. *Multimedia Computing and Networking 1995*, Mar 1995.

[27] H. Zhang, S. Y. Tan, S. W. Smoliar, and G. Yihong. Automatic parsing and indexing of news video. *Multimedia Systems*, 2(6):256–266, Jan 1995.

[28] L. Zhang, B. Denney, and J. Lu. Sub-event recognition and summarization for structured scenario photos. *Multimedia Tools and Applications*, 75(15):9295–9314, Mar 2016.

APPENDIX

A. USER STUDY

A survey is made to construct an equation for the *interest* of an image or video and to estimate the minimum required duration an image should be shown. 39 people have participated.

For videos, three variables were included:

- Stableness
- Infidelity
- Simplicity

Images/videos were automatically chosen from a personal data-set of over 9000 images and videos.

10 videos were chosen with a wide range of stableness, but constant infidelity and simplicity. 10 videos were chosen with a wide range of infidelity, but constant stableness and simplicity. Another 10 videos were chosen with a wide range of simplicity values, but constant stableness and infidelity values. This resulted in 1170 data points and provided us with enough significance to draw conclusions.

For images, two variables were included:

- Infidelity
- Simplicity

11 images were chosen with a wide range of infidelity values, but constant simplicity values. Another 11 images were chosen with wide range of simplicity values, but constant infidelity values. This resulted in 858 data points, again enough to draw significant conclusions.

Linear regression is used to get two *interest* equations (for both images and videos) as can be seen in Table 5 for videos and Table 7 for images.

In the second half of the survey, people were asked whether a video that consisted of 5 images went too fast, too slow or was perfect. Everyone was shown 3 videos. The first video only contained images with 2 people. Whenever someone pressed 'too slow', the same video was shown, but now every image was shown 0.2 seconds less. If someone pressed 'too fast', every image was shown for 0.2 seconds more. If the user pressed 'perfect', the next video was shown. The following two videos contained respectively images with only 4 and only 6 faces. This resulted in 117 data points.

Again, linear regression is used to get to equation 10 which is concluded from 9.

B. SUBJECTIVE EVALUATION

The second user study was used to evaluate the quality of our system. 10 people were asked to send a folder of videos and photos. They also selected a song, knowing that it was going to be used for an after-movie. Every participant got three after-movies:

iMovie The whole album is imported and 'auto content' is turned on. The selected song is added as background music. If the duration of the song is not long enough, part of the video will not have any background music.

Google Photos All videos and photos are uploaded and using the 'create video' feature an after-movie is created. Since Google Photos only allows you to use a maximum of 50 photos/videos as input, the first 50 photos/videos of the album were used.

Proposed method Using PL an after-movie is created.

Every participant was asked to give a grade from 1-10, where 1 is equal to 'I have never seen worse' and 10 is equal to 'brilliant selecting and editing of the video!'. Based on the grades, a ranking was made for each participant. The best movie received 3 points, the second best received 2 points and the worst movie was equal to 1 point. Summing the points, Table 2 was created and the resulting ranking was concluded as:

1. Google Photos
2. Proposed method
3. iMovie

Next to that, they were asked to support their grade with a small comment on each movie. We have summarized some of the positive and negative comments for each software in Table 3.

Table 2. Second user study

Type	Points
Google Photos	27
Proposed method	20
iMovie	13

Table 3. User feedback

Software	Feedback
Google Photos	Transitions are nice. Total video is a bit short.
	Music is annoying.
Proposed method	Nice music.
	Showed a very long part of a video.
iMovie	Very nice transition effects.
	All media is added, instead of a selection. The video is way too long.

Feedback for all three methods is that a lot of duplicate photos are not filtered. Photos that only differ a little bit, are still included and make the movie less interesting. Most people (60%) also commented that the after-movies did not feel chronologically sorted, which might be the case if the raw data does not contain proper time stamp data.

Table 4. Video: model

R	R^2	Std. Error of the Estimate
.532	.283	2.211

Table 5. Video: residuals statistics

Model	Unstandardized B	Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.
(Constant)	-8.075	1.026		-7.872	.000
Unstableness	.151	.010	.400	14.767	.000
Simplicity	-0.25	.007	-0.92	-3.408	.001
CQ	-4.056	.392	-.261	-10.358	.000

Table 6. Image: model

R	R^2	Std. Error of the Estimate
.376	.142	2.095

Table 7. Image: residuals statistics

Model	Unstandardized B	Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.
(Constant)	8.054	.240		33.565	.000
Infidelity	-8.838	.759	-.374	-11.647	.000
Simplicity	.014	.009	.049	1.519	.129

Table 8. Faces: model

R	R^2	Std. Error of the Estimate
.463	.214	.311

Table 9. Faces: residuals statistics

Model	Unstandardized B	Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.
(Constant)	1.788	.077		23.185	.000
# Faces	.099	.018	.463	5.530	.000