Constraint satisfaction programming for video summarization

Sid-Ahmed Berrani, Mohammed Haykel Boukadida, Patrick Gros

Constraint satisfaction programming for video summarization

Sid-Ahmed Berrani, Haykel Boukadida
Orange Labs – France Telecom
35510 Cesson-Sévigné, France
{*sidahmed.berrani, haykel.boukadida*}*@orange.com*

Patrick Gros
Inria
35042 Rennes, France
patrick.gros@inria.fr

***Abstract*—This paper addresses the problem of automatic video summarization. The proposed solution relies on constraint satisfaction programming (CSP). Summary generation rules are expressed as constraints and the summary is created using the CSP solver given the input video, its audio-visual features and possibly user parameters (like the desired duration). The solution clearly separates production rules from the generation algorithm, which in practice allows users to easily express their constraints and preferences and also to modify them w.r.t. the target application. The solution is extensively evaluated in the context of tennis match summarization.**

***Keywords*-Video summarization; Dynamic video skimming; Constraint satisfaction programming.**

I. Introduction

Given the huge amount of available video content, automatic video summarization tools are becoming essential for many applications. For instance, broadcasted TV programs are now available through TV-on-demand services or stored within personal video recorders. They are becoming more present in our everyday life, and their volume is constantly and very quickly increasing. Search or recommendation engines can be used in this context to find/select a content to watch. However, given that the video content is very rich and very difficult to finely describe using high-level and semantic features, these engines can only be used to reduce the search space to a short list. The final choice is generally achieved by the user after having a look to the content.

Basically, a video summary is composed of a subset of images of the original video. These images are either presented as still images, or as a set of video segments. The first case, commonly called static video abstract, is a small collection of representative images called keyframes that are carefully extracted from the video. Each of them represents the visual content of a part of the video. The second case, also known as dynamic video skimming, is a set of video segments of the original video. In this case, the summary is also a video. It conserves the dynamic properties of the original video and consequently it is more pleasant to watch than a static abstract. It is also more expressive since it includes both audio and visual information.

In practice, to be useful, a video summary should be created taking into account the video content itself but also users preferences. The first criterion related to the video content insures that the summary covers parts of the video that are interesting for the target use case. As for user preferences, they specify what should be included in the summary based on what the user is interested in and on his/her constraints like for instance, how much time he/she has to watch the summary. Visualization conditions, i.e. how the summary will be played and on which device (TV set, mobile, tablet...), may also be taken into account and may influence the summary creation.

If this has to be done manually, the video summary creation would be prohibitively expensive given the huge amount of available content in real-world applications. Automatic solutions are hence required. Existing solutions are reviewed in Section II. They use audio-visual content analysis or external sources of information (like tweets for live TV programs). They rely on three components: (1) Features describing/related to the content, (2) A set of high-level parameters to be set by the user (e.g. the summary duration) and a set of internal parameters (e.g. thresholds), and (3) An algorithm that produces the summary. This algorithm encodes the rules to be used to generate the summary *and* the mechanism that makes use of the rules, the features and the parameters to produce the summary. Thus, existing solutions are generally finely tuned for a specific type of videos. They are also "rigid" in the sense that if the user would like to modify even slightly the way the summary is produced, the method has to be completely reviewed and a potentially large number of internal parameters has to be tuned again. This is due to the full integration of rules within the summary generation algorithm.

In this paper, we propose a novel automatic video summarization method that relies on constraint satisfaction programming (CSP) [1]. CSP comes from artificial intelligence and has been used for problem solving in different fields, like software optimization for air traffic and boarding gates management, vehicles construction, schedules, staff turnover management... The idea is to clearly separate summary production rules from the algorithm that generates the summary. The rules are expressed as constraints. The summary generation algorithm is the CSP solver. This way, we get rid of internal parameters (the CSP solver has none) and we provide users with a solution that allows them to easily express their constraints and preferences and to modify them

with respect to the application and the context.

Even if the solution is appropriate for both static video summaries and dynamic video skimming, we focus in the paper on this later kind of summaries. The first step consists in a video shot segmentation followed by a video content analysis in order to extract all the necessary features for the summary generation. The summary is composed of a set of segments where each segment is a shot or a part of a shot.

The different criteria related to the summary are modeled and expressed as a constraint satisfaction problem. The advantages of the approach are numerous:

1) CSP is a suitable tool that allows us to express different kinds of criteria using a set of meaningful and high-level constraints.
2) Constraints can be formulated using a set of high-level parameters that can easily be modified in order to produce a new summary. Moreover, there is no need to set any internal parameter.
3) CSP allows specifying *hard* and *optimization* constraints. Hard constraints can be used to model a condition that must be satisfied. Optimization ones, however, can be used to minimize or maximize a specific property. In other terms, this allows the user to express "nice to have" criteria.

The rest of the paper is organized as follow: Section II reviews existing works in the field. In Section III, we describe our approach. Section IV is dedicated to the evaluation process, which is a crucial and difficult task when working on automatic video summaries. Experimental results are shown in Section V. Finally, Section VI concludes the paper.

II. Related Works

Although it is a relatively recent research domain, various works dedicated to automatic video summarization have already been proposed. Existing techniques are presented in the following four subsections.

A. Attention model-based approaches

These approaches proceed by computing a score for each base unit of the input video. A base unit can be a second, few seconds, a frame, a shot... The aim of the score is to measure the "importance" of the base unit by modeling and simulating the attention of viewers. Using this score, a skimming curve is created and used for excerpt extraction by setting a threshold on the curve: Excerpts with values above this threshold are selected. In practice, scores are calculated using low-level features that are correlated to human attention. In [2], summary generation is based on a visual attention model that is computed using the color, the luminosity, human faces, object motion and the number of regions of interest in a frame that can catch users' attention. In [3], the combination of partial attention models is provided by implementing linear and non-linear fusion schemes. In [4], feature scores for each frame are calculated using features like face and text occurrence and audio volume. Conditions on these features are expressed as inequalities and the final set of frames to be included into the summary is composed of frames that satisfy all of the conditions. To avoid over-segmentation, a greedy method is used to maximize the total score and minimize the number of excerpts.

Although these approaches are simple and provide a good formalism to state and to solve the problem, obtained results remain limited. Underlying low-level features are basically weighted and summed-up in order to construct the attention curve and this does not guarantee a correlation with what the user is interested in. These approaches also involve too many parameters that are difficult to set up and particularly the weights of low-level features and the threshold that is finally applied on the attention curve. On the other hand, it is very difficult to take into account instructions that might be expressed in natural language by an expert or a final user. Finally, once the model is trained, it is also very difficult to modify it in order to take into account a new type of videos or a new criterion.

B. Social short messages-based approaches

Recently, many research studies started exploring the association of a textual content to the video in order to determine important segments to be included in the summary. Twitter is a valuable source of such textual content. In [5], the number of tweets related to a live TV program that are posted per base unit is computed. This allows drawing a curve whose maxima likely correspond to events of interest. In [6], a selection technique of representative tweets is proposed. It relies on a clustering method that takes into account the temporal information. The summary is composed of the set of selected tweets with their timestamps.

These approaches are very powerful but are limited to highly popular live TV programs that generate a high number of live tweets. Tweets have to be associated with others features. Alone, they can detect socially important events, but do not guarantee that a summary based on these events covers the whole interesting parts of the program.

C. Summary as an overview

Other works aim to provide an overview which is used to give users an idea on the whole video content by measuring similarity between different parts of the video and by eliminating redundancy. In order to choose representative images that are different from each other and that well represent the video content, a comparison of all the video frames between them is performed. In [7], shots are classified into clusters based on their visual similarity. The longest shot is retained to represent the cluster. In [8], shots are classified using a minimum spanning tree and introduces a graph-based audiovisual alignment algorithm to align the video summary and the audio summary. The video summary is created by

eliminating visual redundancy and the audio summary is created using the latent semantic analysis technique applied on speech transcripts. Another straightforward way to produce overviews consist in compressing the original video by speeding up the playback [9]. Although the time is reduced, the video property is distorted and audio comprehension is affected.

D. Highlight-based approaches

Another way to create a summary is to rely on the detection of highlights. One of the earliest works of this approach is the Informedia project [10]. It relies on audio- and video-based extracted features. An audio skimming is created corresponding to keywords that are extracted from the audio transcript using TF-IDF. A video skimming is also created using detected faces, text, and camera motion.

Other techniques have also been proposed for detecting highlights in specific videos like sport or news videos. For instance, [12] addresses baseball videos and [13] soccer videos. Both use Markov chain models driven by an EM algorithm in order to detect play and break phases. As for news videos, hot events are detected [11]. The solution relies on measuring the similarity on news events and on a graph-based clustering. Hot events are detected using clusters properties: The size of the cluster and the globality of the event (i.e. its presence in different channels and in different periods of time).

These approaches are very specific and generally require a training phase for each kind of videos.

III. Proposed method

A. Overview and novelty

The proposed method is based on CSP [1]. It is a powerful tool that allows us to express in a high-level language a set of constraints over the video. It is sufficiently flexible to define different kind of constraints related to the content of the video but also to the user preferences. Using CSP, a user can add at any time a new constraint or modify an existing one and get a summary without having to review the whole model or to modify any internal parameter.

The general working scheme of the proposed solution is depicted in Figure 1. A set of low-level features is extracted from the visual and the audio signals of the input video. In our case, the chosen base-unit of the video is the shot[1]. On the other side, a set of constraints are defined. These are applied to the extracted features and used by the solver in order to build a summary. In this context, the summary results from the resolution of a constraint satisfaction program, and it is not necessarily unique. A set of summaries can thus be found and they all satisfy the considered constraints.

[1]A shot is a contiguous sequence of frames taken by the same camera without interruption.

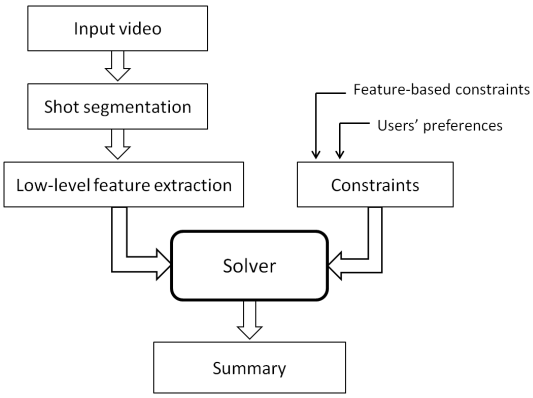

Figure 1. General scheme of the proposed solution.

Another important and interesting feature of CSP is the ability to formulate *hard* constraints but also *optimization* ones. The first ones have to be satisfied. The second ones can be used to minimize or maximize a certain property in the summary. As we will see later on, this is very helpful in practice as it avoids to hardly and precisely define each criteria. The solver is just noticed to optimize it.

In the rest of the paper, we use the following notations:

- n: the number of shots of the input video,
- S_i: the i-th shot of the input video,
- $Sfbegin_i$: the index of the first frame of shot S_i,
- $Sfend_i$: the index of the last frame of shot S_i,
- $Snbr_i$: the number of frames of shot S_i.

B. Introduction to CSP

A constraint satisfaction problem is defined by the triplet (X, D, C):

- A set of variables $X = \{X_1, X_2 \dots X_n\}$.
- A set of domains of variables $D = \{D(X_1), D(X_2) \dots D(X_n)\}$, where $D(X)$ is a finite set of possible values that a variable X can take.
- A set of constraints $C = \{C_1, C_2 \dots C_m\}$, where C_i is a constraint defined by a subset of variables on which it relates and expresses a property that must be satisfied by the corresponding variables. C_i is thus a relation over a set of variables.

CSP relies on a constraint solver that aims to find a feasible solution (values of X) that respects constraints (C). This is based on three principle mechanisms: domain filtering, constraint propagation and search for solutions. The problem is considered as a combination of sub-problems for which effective methods of resolution are provided. Each sub-problem is derived from a constraint. The sub-problem resolution removes values from the domain of variables that cannot belong to any sub-problem solution taking into account the possible values of other variables domains. It is a filtering mechanism that aims to reduce the domain of each variable. After each change on the domain of a variable,

constraints involving this variable are applied again to filter domains of other variables that may be impacted by that change. This mechanism is called constraint propagation. Finally, the remaining search space is browsed in order to assign successively a value to each variable and, hence, to create a solution.

There are three main approaches for solving constraint satisfaction problems: Backtracking search, local search, and dynamic programming. The first one is currently the most widely used. It is based on building a search tree where each level of the tree corresponds to a variable, each node in that level is a possible value of the variable and branches out of a node represent choices that should respect defined constraints. Backtracking search consists thus in a depth-first traversal of a search tree.

In addition, CSP enables to optimize a cost function g. The idea is to maximize or minimize a function of the variables. The solver uses initially a first backtracking search to find any solution (Sol_0) satisfying defined constraints. Then it adds a new constraint in the form $g(A) < g(Sol_0)$ (resp. $g(A) > g(Sol_0)$) in the case g has to be minimized (resp. maximized). Here, A is the current solution. This process is repeated for each new encountered solution.

For a detailed description of CSP, the interested reader can refer to [1].

Many CSP solvers have been developed. The most popular ones are ILOG by IBM, OR-TOOLS by Google and Choco [14].

C. Problem modeling using CSP

As already stated, we focus on dynamic video skimming. The summary is thus composed of a set of segments. A segment is either a whole shot or a set of consecutive frames of a shot. We have chosen to allow the selection of a segment from a shot in order to enable a fine selection during summary building and to target only interesting parts. Since a shot can be very long and contain redundant information, it is not relevant to always select the whole shot. This allows our model to be more general and to address different kind of videos. The problem can thus be formulated as a selection process of segments from the video.

In our approach, we define a segment by a couple of variables: Its first frame and the number of frames it contains. For each shot, either it is completely selected, a single segment is selected or nothing is selected. We thus propose to assign two variables $(fbegin_i, fnbr_i)$ for each shot (S_i): $fbegin_i$ refers to the first frame of the segment within S_i and $fnbr_i$ to the number of frames of the segment. If nothing is selected from the shot, $fnbr_i$ equals 0.

Hence, variables of our program are:

$$\begin{cases} FBegin = \{fbegin_i,\ i = 1..n\} \\ FNbr = \{fnbr_i,\ i = 1..n\} \end{cases} \quad (1)$$

Where the domains of $fbegin_i$ and $fnbr_i$ are:

$$fbegin_i \in [Sfbegin_i\ ..\ Sfend_i]$$
$$fnbr_i \in [0\ ..\ Snbr_i]$$

In order to reduce the computational cost of the summary creation process, we propose to quantize variables' domains (e.g. by a quantization step of 12 for $fbegin$). The number of possible values for variables is reduced whereas the summary is almost not modified from the user point of view.

D. Constraints formulation

We distinguish five types of constraints:

1) Modeling constraints: These are used to insure the coherence of the model. In our case, we define two constraints:

$$fbegin_i + fnbr_i - 1 \ \leq\ Sfend_i \quad (2)$$

$$fnbr_i = 0 \implies fbegin_i = Sfbegin_i \quad (3)$$

Constraint (2) is used to make sure that the selected segment does not exceed shot boundaries. Constraint (3) allows the solver to avoid useless calculation: It does not have to act on $fbegin_i$ if the shot is not interesting and does not include a selected segment, that is, if $fnbr_i = 0$. As will be shown in the experiments in Section V-A, this constraint has a significant impact on the efficiency of the solver, i.e. its response time.

2) Global constraints: These apply to the whole summary. It concerns for instance the temporal distribution coverage, i.e. the ability to represent different parts of the original TV program in the summary. Another global constraint is the summary duration. It is one of the most important requirements that a user usually expresses. It is defined in (4).

$$dmin \ \leq\ \sum_{i=1}^{n} fnbr_i \ \leq\ dmax \quad (4)$$

Specifying an interval for the desired duration is important. This gives the solver the necessary *freedom* to explore neighboring solutions of the desired duration and increases the chance to find a solution. Section V-A provides an experimental study of this point.

3) Pruning constraints: This kind of constraints is used to eliminate parts of the video that do not fulfill a set of criteria. It can be related to a feature (5) or to the length of the selected segment (6).

$$(fnbr_i \times f_i)\ =\ 0 \quad (5)$$

$$(fnbr_i \geq Sdmin)\ \vee\ (fnbr_i = 0) \quad (6)$$

where f_i corresponds to the presence of the feature f in the i-th shot, $\vee$ denotes the logical operator *or* and $Sdmin$ is a threshold presenting the minimum duration of a segment to be included into the summary.

Constraint (5) means that if $fnbr_i > 0$, i.e. a segment is selected from shot S_i, the feature f must not be present in

that shot ($f_i > 0$). As for Constraint (6), it insures that each selected segment has a duration that is longer than $Sdmin$.

4) Neighborhood constraints: Constraints can also be about the properties of the neighborhood. For example, in sport videos, important events generally appear before applause. Formula 7 provides an example of such constraints.

$$(fnbr_i \neq 0) \implies (f_{i+1} \neq 0) \quad (7)$$

Here, for a segment to be selected from shot S_i, the feature f has to be present in the successor shot, S_{i+1}.

5) Optimization constraints: CSP gives the possibility to define optimization constraints which are preferably satisfied. Thus, we let the solver choose between different solutions, and favor some of them by maximizing or minimizing the given criterion. For each optimization constraint, a cost is associated. It can be for instance the number of frames containing a specific feature that has to be minimized. It can also be the number of frames between the end of the selected segment and the last frame of the shot. Asking the solver to minimize this number amounts to asking for selecting segments preferably at the end of shots.

The cost function f_{opt} can be written as:

$$f_{opt} = \sum_i \alpha_i.Opt_i \quad (8)$$

Where Opt_i denotes a computed cost to optimize and α_i is a weight of the corresponding cost. A weight can be positive or negative depending on the optimization type (minimize or maximize). It corresponds to a high-level parameter that encodes the importance of the corresponding criterion.

E. Feature extraction

In order to express constraints on the video content, a set of features has to be computed from the visual and/or the audio signals. The video is first segmented into shots using a method that is similar to the one proposed in [15]. Computed features are then projected on shots. The resulting data are then used by the solver as described before. The features used in our experiments are described in Section V.

IV. Evaluation

The quality evaluation of automatically generated summaries is a tricky task. Even when summarization is manually done by experts, resulting summaries are generally different. The process is highly subjective and very dependent on the target application and user interest. Moreover, there is no common and public video database of realistic size that can be used to compare different solutions[2]. In [18], a deep analysis of evaluation methods is provided. These are classified into three categories: (1) *Result description* where the evaluation comes down to a discussion of obtained results and internal parameters' influence, (2) *Objective metrics* are generally applied on static approaches where metrics are defined to compare keyframes and their redundancy, and (3) *User studies* where independent users analyze the obtained summaries and judge their quality. Each of these methods has its advantages and drawbacks. For instance, user studies seem to be at first sight the best method. In practice, it requires to define a rigorous protocol for selecting a significant number of users and for setting the evaluation measures, and this is rarely done in practice. It is also very subjective and should not be conducted without a well-defined target application.

In this paper, we propose to perform first a result description and analysis in order to show among others the *flexibility* of our solution, *our main claim*. We show in particular that we can easily add new constraints in order to modify the summary. We study also the impact of modeling constraints on the efficiency of the approach. In a second part, we properly evaluate the quality of summaries.

We have chosen to perform our experiments on tennis videos. These are well structured and non-interesting parts can easily be annotated. Furthermore, *editorial* summaries (ES) of most important matches are generally available and can be used for the evaluation. The quality evaluation consists then in computing the intersection between the ES and the automatically generated one. As a measure, we retained the ratio between this intersection and the duration of automatically generated summary. It is denoted in the rest of the paper by "$\bigcap\ with\ ES$". Obviously, the objective is not to have a full match between the two summaries. Even two editorial summaries that are generated by two TV channels for the same match are generally different. The idea here is to use this measure to perform relative evaluation of different automatically generated summaries with respect to the editorial one. We propose to use also another measure in which we consider non-interesting parts of the video. A volunteer annotates all the parts of the video that she/he considers as not interesting *at all* and that should not be included into the summary. The evaluation consists then in measuring the ability of our solution to discard these parts.

Another problem when evaluating video summarization approaches is the difficulty to compare the proposed approach with existing ones. The field is too wide and each solution is evaluated on a different video content type considering different criteria, and, as explained above, there is no public video database. This makes any comparative study almost impossible to do. In this paper, we have tried to overcome this difficulty by implementing three baseline solutions in order to show the advantages of our solution.

V. Experiments

Our test dataset is composed of 4 tennis matches, that correspond to a total duration of more than 8 hours. We denote these matches: M_1, M_2, M_3 and M_4. The two first

[2]In [16] and [17], the same videos are used for evaluation. These are however of very short duration (between 1 and 4 min per video) and can only be used for the evaluation of static video summarization techniques.

matches have editorial summaries (ES). For the two others, we have performed a manual annotation of non-interesting parts by independent volunteers, colleagues that have not participated to this work and that are tennis fans.

For each video, the following features have been computed:

- Dominant color descriptor (DCD): It provides the dominant colors in an image. We have used the MPEG-7 descriptor which detects 8 dominant colors at most.
- Speaker segmentation: This method partitions the audio signal into speech/non speech segments [19].
- Applause detection: We have used a method similar to [20]. It performs a segmentation into applause, speech, music and silence segments.
- Face detection and tracking: We have used the method developed in [21] for detecting and tracking faces in a video. It is based on Convolutional Neural Networks.

In addition to modeling constraints (2) and (3), we have proposed the following constraints for the tennis application:

- Summary duration: It must be within an interval given as an input parameter.
- Speech segments must not be cut. The idea is to avoid in the summary to cut a person who is talking.
- Each selected segment must contain applause, or must have an adjacent selected segment that contains applause. This constraint insures that each sequence (set of adjacent selected segments) in the summary contains applause. Here, applause indicates that something interesting happens.
- Short isolated segments are discarded. The minimum duration threshold has been set to few seconds.
- In order to increase the number of interesting segments, we have proposed to maximize the presence of applause in the summary.
- Shots with a majority of frames containing a dominant color are maximized. These frames correspond of views on the whole tennis field and the corresponding shots are game shots, i.e. shots of ball exchange between players.
- The presence of faces in the selected segments is minimized. The idea is to minimize the number of frames where a face of a minimal height of 20 pixels is detected. These segments are less interesting than game segments that do not contain faces of that size.
- In the summary, we must have the match point. This is identified as the last game shot of the video.

Optimization constraints have been encoded as a cost function where an equal weight has been assigned to each of them.

In our experiments, we have used the freely available CSP solver Choco [14]. Experiments have been performed on a machine running an Intel Core-Duo 2.40 GHz CPU and with 4 GB of main memory.

A. Exp. 1: Study of the model

Formulation of all of the constraints and their implementation in Choco were straightforward and this is one of the main strength of the approach. The proposed model is very adequate and allowed us expressing all the constraints we have thought of.

In order to assess the efficiency of our model, we have focused on Constraint (3). We recall that this constraint aims at preventing the solver from exploring useless possibilities: If $nbr_i = 0$, this means that shot S_i is not selected and there is no need to act on the other associated variable $fbegin_i$.

For this, we have generated two summaries of M_2 with and without considering Constraint (3). The desired duration has been set between 4 and 5 min. Obtained results are described in Table I.

	Without Constraint (3)		With Constraint (3)	
	$\bigcap$ $with\ ES$	# of sol.	$\bigcap$ $with\ ES$	# of sol.
After 2 min	17%	12	20%	37
After 30 min	20%	54	23%	441

Table I
RESULTS ON M_2. IMPACT OF CONSTRAINT (3).

This table clearly shows that Constraint (3) has a direct impact on the efficiency of the solver. This constraint allows the solver to return more and better solutions in much less time. After 2 min and with Constraint (3), the solver returns 37 solutions among which a solution that contains 20% of the ES. To get the same result without Constraint (3), we have to wait 30 min.

We have also studied the influence of the duration constraint. Three cases have been considered: (1) specifying a fixed duration, (2) specifying an interval of 1 minute, (3) specifying a lower bound. Obtained results are described in Table II. These results were obtained after 1 hour. They show that the best trade-off is obtained when specifying an interval. It gives the solver freedom to explore more solutions while staying in the neighborhood of the desired duration.

Desired sum. dur. (D)	$\bigcap$ $with\ ES$	Dur. of sum.
$D = 5\ min$	21%	5 min
$5\ min \geq D \geq 4\ min$	27%	4 min 51 s
$D \geq 5\ min$	28%	6 min 51 s

Table II
RESULTS ON M_2. IMPACT OF THE CONSTRAINT ON DURATION.

We have performed a last experiment on the variation of the quality of the summary w.r.t. the time left to the solver. Again, using M_2, we have asked for a summary with a duration between 4 and 5 minutes, and we have analyzed obtained solutions after 20s, 2 min, 30 min... Obtained results are depicted in Figure 2. After 20s, 17% of the

obtained summary comes from the ES. This ratio increases with the time spent by the solver exploring the search space. It riches 33% after 3 hours. Here, we have provided the solver with optimization constraints without specifying any threshold.

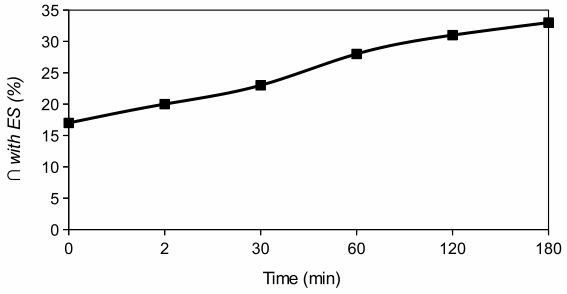

Figure 2. Quality of the summary vs. the time spent by the solver.

However, given the type of videos, if we consider that at least 1/2 of the summary should be game shots and 5% should be applause sequences, we can provide these thresholds in order to orient the solver and avoid it exploring useless possibilities. After applying these two thresholds, we get a summary whose ratio of intersection with ES equals 30% in 2 minutes.

This shows that the way CSP is used has a great impact of the efficiency. Using two simple and intuitive heuristics allows us to drastically reduce the solver response time.

We have thus used these two thresholds for all of the experiments that are presented in the rest of the paper.

B. Exp. 2: Flexibility of the solution

Another important criterion of summary generation is the flexibility of the solution, i.e. the ability to easily adapt the summary.

We have analyzed the distribution of segments of the two ES, and we have noticed that for M_2, 86.35% of the ES comes from the 2nd half of the match. If this has to be taken into account in the summary generation, we have only to add a constraint on the distribution of segments (CD) that maximizes the difference between the total duration of selected segments coming from the 2nd half and the total duration of selected segments coming from the 1st half of the match. We have generated two summaries for M_2 of durations between 4 and 5 min with and without using CD. Figure 3 presents the distribution of segments of the two summaries. Additionally, using this constraint, the quality of the generated summary w.r.t. the ES has been increased from 30% to 42%.

As this distribution property is not satisfied for all the matches, we have not included it into the set of constraints. The objective of this experiment was just to show that our method can easily take into account an additional rule, and this can be done without modifying any other constraint.

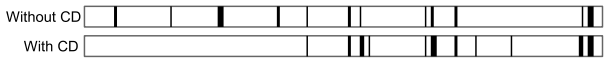

Figure 3. Impact of the constraint CD on the distribution of segments (in black) in the summaries of M_2.

C. Exp. 3: Quality evaluation

In order to assess the quality of generated summaries, we have considered three baseline methods:

- **A pure random shot selection (RS).**
- **A random selection of game shots (RG):** The summary is created as a random selection of game shots (these are identified using DCD).
- **A uniform selection of game shots (UG):** Starting from the beginning of the video, one game shot is selected each p encountered game shots. The value of p is fixed w.r.t. the desired summary duration.

Obtained results on M_1 and M_2 are presented in Table III. Four summaries with different durations (4-5 min, 9-10 min, 14-15 min and 19-20 min) have been computed for each video. Evaluations on M_3 and M_4 have been done w.r.t. the proportion of non-interesting segments in the generated summaries. Obtained results are presented in Table IV. For each summary generation, we have stopped the solver after 2 min and we have retained the last found solution.

Both Tables III and IV show that our solution outperforms baseline methods. Between 21 and 43% of segments composing our summaries are the same as those used for ES. We can also notice that our solution achieves the best results in discarding non-interesting segments. These represent in the worst case 22% of the duration of the summary. In Table IV, there are some missing values for RG and UG. The video M_4 corresponds to a short match and does not contain enough game shots.

We have also manually checked all the generated summaries and all of them contain the match point sequence.

VI. Conclusion and Future Work

In this paper, we have proposed a novel solution for video summarization that is based on CSP. The main advantage of our solution is its ability to clearly separate production rules from the summary generation algorithm. It also allows users to easily express high-level constraints and preferences thanks to the proposed model.

Future work will focus on how we can go a step further by proposing a high-level language that allows users to specify their constraints without dealing directly with the CSP tool. Currently, even if the rules and the solver are separated, we still have to write constraints in the CSP language. The idea is to create a new high-level language that makes the process even more transparent to the user.

Additional experiments on other types of contents are also part of our future work.

	M_1				M_2			
	4-5 min	9-10 min	14-15 min	19-20 min	4-5 min	9-10 min	14-15 min	19-20 min
RS	1%	6%	9%	9%	9%	14%	16%	14%
RG	8%	15%	10%	13%	15%	19%	25%	21%
UG	8%	16%	9%	12%	18%	18%	25%	22%
Our solution	43%	32%	28%	21%	30%	28%	32%	33%

Table III
RESULTS ON M_1 AND M_2. EVALUATION W.R.T. EDITORIAL SUMMARIES (ES).

	M_3				M_4			
	4-5 min	9-10 min	14-15 min	19-20 min	4-5 min	9-10 min	14-15 min	19-20 min
RS	41%	50%	44%	40%	32%	46%	32%	33%
RG	16%	28%	23%	22%	21%	23%	-	-
UG	31%	27%	23%	22%	23%	27%	-	-
Our solution	15%	11%	13%	9%	10%	8%	15%	22%

Table IV
RESULTS ON M_3 AND M_4. PROPORTION OF NON-INTERESTING SEGMENTS IN THE SUMMARIES.

REFERENCES

[1] F. Rossi, P. Van Beek, and T. Walsh, *Handbook of constraint programming*. Elsevier Science, 2006.

[2] Z. Longfei, C. Yuanda, D. Gangyi, and W. Yong, "A computable visual attention model for video skimming," in *Proc. of the Int. Symp. on Multimedia, Berkeley, USA*, Dec. 2008.

[3] Y.-F. Ma, X.-S. Hua, L. Lu, and H.-J. Zhang, "A generic framework of user attention model and its application in video summarization," *IEEE Trans. on Multimedia*, vol. 7, no. 5, pp. 907–919, 2005.

[4] R. T. M. Lyu and I. K. C. Kink, "Video summarization using greedy method in a constraint satisfaction framework," in *Proc. of the Int. Conf. on Distributed Multimedia Systems, Miami, USA*, 2003.

[5] A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller, "Twitinfo: aggregating and visualizing microblogs for event exploration," in *Proc. of the Conf. on Human factors in computing systems, Vancouver, BC, Canada*, May 2011.

[6] H. Takamura, H. Yokono, and M. Okumura, "Summarizing a document stream," in *Proc. of the EU Conf. on IR Research, Dublin, Ireland*, Apr. 2011.

[7] Y. Gong and X. Liu, "Video summarization with minimal visual content redundancies," in *Proc. of the Int. Conf. on Image Processing, Thessaloniki, Greece*, vol. 3, Oct. 2001.

[8] Y. Gong, "Summarizing audiovisual contents of a video program," *EURASIP Journal on Advances in Signal Processing*, no. 2, pp. 160–169, 2003.

[9] D. Ponceleon, A. Amir, S. Srinivasan, T. Syeda-Mahmood, and D. Petkovic, "Cuevideo: automated multimedia indexing and retrieval," in *Proc. of the ACM Int. Conf. on Multimedia, Orlando, USA*, Oct. 1999.

[10] M. A. Smith and T. Kanade, "Video skimming for quick browsing based on audio and image characterization," School of Computer Science, Carnegie Mellon University, Tech. Rep. CMU-CS-95-186, Jul. 1995.

[11] Y. Peng and C.-W. Ngo, "Hot event detection and summarization by graph modeling and matching," in *Proc. of the Int. Conf. on Image and Video Retrieval, Singapore*, Jul. 2005.

[12] Y. Rui, A. Gupta, and A. Acero, "Automatically extracting highlights for tv baseball programs," in *Proc. of the ACM Int. Conf. on Multimedia, Los Angeles, USA*, Oct. 2000.

[13] A. Ekin, A. M. Tekalp, and R. Mehrotra, "Automatic soccer video analysis and summarization," *IEEE Trans. on Image Processing*, vol. 12, no. 7, pp. 796–807, 2003.

[14] N. Jussien, G. Rochart, X. Lorca *et al.*, "Choco: an open source java constraint programming library," in *Proc. of the Workshop on Open-Source Software for Integer and Constraint Programming, Paris, France*, May 2008.

[15] A. Hanjalic, "Shot-boundary detection: unraveled and resolved?" *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 12, no. 2, pp. 90–105, 2002.

[16] S. E. Fontes de Avila, A. P. Brandao Lopes, A. da Luz Jr., and A. de Albuquerque Araújo, "Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method," *Pattern Recognition Letters*, vol. 32, no. 1, pp. 56–68, 2011.

[17] P. Mundur, Y. Rao, and Y. Yesha, "Keyframe-based video summarization using delaunay clustering," *Int. Journal on Digital Libraries*, vol. 6, no. 2, Apr. 2006.

[18] B. T. Truong and S. Venkatesh, "Video abstraction: A systematic review and classification," *ACM Trans. on Multimedia Computing, Communications, and Applications*, vol. 3, no. 1, Feb. 2007.

[19] M. Collet, D. Charlet, and F. Bimbot, "Speaker tracking by anchor models using speaker segment cluster information," in *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing, Toulouse, France*, vol. I, May 2006.

[20] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeature speech/music discriminator," in *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing, Munich, Germany*, vol. 2, Apr. 1997.

[21] F. Mamalet, S. Roux, and C. Garcia, "Real-time video convolutional face finder on embedded platforms," *EURASIP Journal on Embedded Systems*, no. 1, 2007.