Efficient Video-Stream Filtering

Fabrizio Falchi, Claudio Gennaro, and Pasquale Savino
Institute of Information Science and Technologies, Italian National Research Council

Peter Stanchev
Kettering University

A new approach for video-stream filtering that makes use of the features representing video content and exploits the properties of metric spaces can help reduce the filtering receiver's computational load.

When information is delivered by news agencies, broadcast TV programs,[1,2] and even surveillance systems, users receive a huge amount of data, but they might be interested in only a limited part of it. The process of selecting only significant information is called *information filtering*. However, the complexity of the filtering process is linear with the number of streams, filters, and features used to represent the video data. The entire process must occur in real time, so it's likely that most of a systems processing power will be dedicated to filtering.

We propose a new approach to video filtering that makes use of simple additional information (that is, indexes) sent with the video, eliminating the need for users to digest video that wouldn't pass the filter anyway. Our approach requires a metric measure of similarity between the filter and the video representative (the feature); this measure is based on the well-known pivots technique.[3,4] Indeed, filtering quality depends on the metric measure adopted; however, our method is suited for any metric distance measure in that it won't affect filter capabilities but will significantly improve efficiency by more than one order of magnitude.

The scenario

Our approach is general and doesn't depend on a specific format to represent the video content, but we assume the use of MPEG-7 to provide a description of the videos. In particular, we concentrate on the MPEG-7 visual descriptors, which cover basic visual features, such as color, texture, and shapes. Each source sends a video stream associated with an MPEG-7 stream that contains this video stream's description. These streams move through a generic transmission channel (see Figure 1) to the receiver stations (for example, set–top–boxes, digital multimedia recorders, media center PCs, and so on). Each receiver station has several filters that select (from all the video streams that arrive to the station) and deliver to the user only the video frames considered relevant. Filtering is based on a comparison between each video frame with the filters, so that only frames similar to one of the filters are delivered to the user.

We can consider different description types of the video content. For example, the description can be based on the entire video—for example, on metadata, such as title, author, and so on—or on the video shots or scenes. It also can be based on several video representative frames. We assume that the MPEG-7 stream contains several visual descriptors for a subset of the video frames. We call these *selected frames* (S-frames). For simplicity, our system selects and analyzes a frame every five frames, which corresponds in the National TV Standards Committee television system to a frame rate of about six frames per second. The technique is not limited to just this simple method for the selection of the S-frames. Real application settings can adopt more semantically meaningful selection techniques, such as automatic selection of keyframes representing video shots or scenes. MPEG-7 visual descriptors—generated by the source station—describe each frame.

In this scenario, a query filter is an image whose representation is obtained using the MPEG-7 descriptors. The filter is compared with each S-frame using the metric similarity measure associated with the descriptor. This is the receiver station's task, that is, matching the queries (user submitted images) with the S-frames of all the streams received. When an S-frame passes the filter, the user sees all successive frames until the next S-frame.

To filter the stream, we could use a brute-force algorithm, which requires comparing all S-frames with the query filter, but this solution would overload the receiver station and reduce the number of channels we are able to filter simultaneously. We can calculate the total time spent by the receiver station for the

1070-986X/08/$25.00 © 2008 IEEE

Published by the IEEE Computer Society

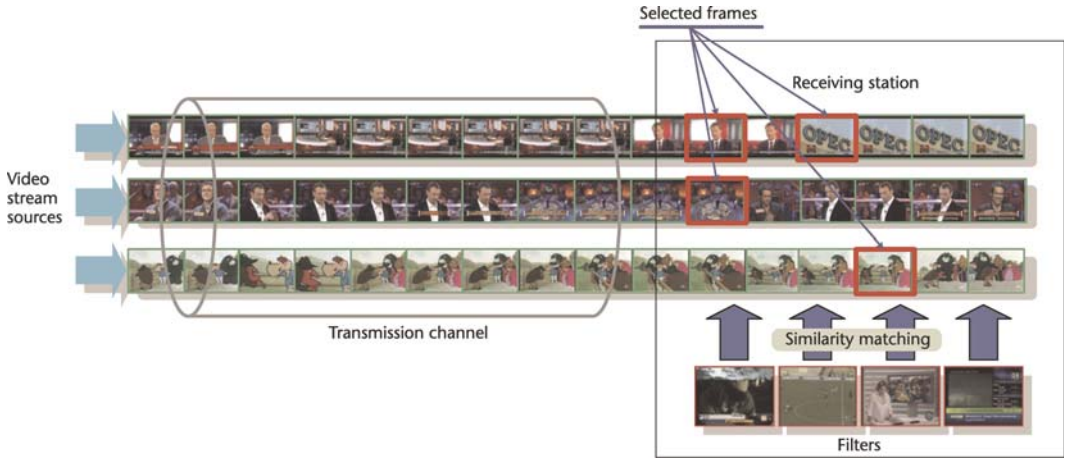

Figure 1. The scenario.

similarity evaluation as the number of sources multiplied by the number of query filters, multiplied by the number of MPEG-7 descriptors, multiplied by the average similarity computation time for the descriptors.

For example, using the MPEG-7 XM reference software[5], the average CPU time needed to evaluate the similarity between two visual descriptors represented with edge histograms is about 30 microseconds (μs). We measured this time on a 2.4-GHz Pentium 4 PC. This means that if we have 50 sources and 100 query filters, it takes $50 \times 100 \times 30\ \mu s = 0.15$ seconds only to evaluate the similarity using the edge histograms' descriptor of all the queries against one video frame. This implies that we cannot evaluate more than $1/0.15 \approx 6.67$ frames per second. This example makes clear the importance of adopting techniques to reduce the number of features extracted during the filter analysis.

To improve the receiver's filtering capabilities, we can use two approaches:

- at the receiver station, organizing a large number of filters using a conventional access structure (such as an M-tree[6]); or
- at the sender station, precomputing the similarity between some S-frames to reduce the number of similarity computations made at the receiver station between the query and the S-frame.

The first solution doesn't require any further computation at the sender station. But it involves the implementation of a query index structure to match S-frames against several queries. The text retrieval field has exploited this approach.[7] We can extend the method to our application scenario by using an access structure for measuring the similarity between S-frames.[6,8,9] This solution's drawback is that it's only worthwhile when the number of queries is large. With the second solution, we exploit the computational power of the sender station. We achieve this by adding some precomputed similarity information to the MPEG-7 stream to increase the receiver station's performance. The overhead for generating this precomputed information is minimal because its computation is negligible compared with that for the MPEG-7 stream production.

The objective of our research is to analyze the performance behavior of this second approach, which, to the best of our knowledge, has never been explored.

Metric spaces and pivot filtering

A convenient way to assess the similarity between two objects is to apply the metric notion by determining the closeness (the distance) of the objects.[10] Treating data items as objects of a metric space brings a great advantage in universality, because many data types and information-seeking strategies conform to the metric point of view. Therefore, an indexing technique based on metric objects is extendible in the sense that it can be applied to many specific search problems.

Metric space

Metric space is a term belonging to mathematical set theory. The space is not empty but is rather a mathematical set of objects, such as

a feature set. The only knowledge we start with is the distance among the space's objects, which we view as a measure of the objects' dissimilarity. For instance, if the distance $d(x, y)$ between two objects x, y is 0, then the objects are identical. More formally, a metric space $M = (D, d)$ defined by a domain of objects (features, points) D and a total (distance) function d. For any distinct objects $x, y, z \in D$, the distance must satisfy the following properties

$$\begin{aligned} &d(x, x) = 0 && \text{(reflexivity)} \\ &d(x, y) > 0 && \text{(strict positiveness)} \\ &d(x, y) = d(y, x) && \text{(symmetry)} \\ &d(x, z) \leq d(x, y) + d(y, z) && \text{(triangle inequality)} \end{aligned}$$

The technique we propose in this article to reduce the cost of the filtering process only assumes that the features used to represent the video content, and the corresponding distance functions, can be represented as a metric space. Indeed, this is the case with the MPEG-7 descriptors we use.

Pivot-based filtering

The efficiency of the filtering is obtained by reducing the number of similarity measures between the query object and the target objects. The basic idea of the pivot-based algorithms is to exploit the knowledge of a set of precomputed distances between one object of the data set, called a *pivot*, and all the objects from the data set. Formally, given a pivot p, a generic object x, and a query object q, the triangle inequality corresponds to the three following statements:

$$\begin{aligned} &d(x, q) \leq d(x, p) + d(q, p) \\ &d(x, p) \leq d(x, q) + d(q, p) \\ &d(q, p) \leq d(x, q) + d(x, p) \end{aligned} \quad (1)$$

Combining the last two elements of Equation 1 and applying the symmetry property, we can define an upper bound $d^H(x, q)$ and a lower bound $d^L(x, q)$ for the distance $d(x, q)$,

$$\begin{aligned} &d^L(x, q) = |d(q, p) - d(x, p)| \leq d(x, q) \\ &d^H(x, q) = d(q, p) + d(x, p) \geq d(x, q) \end{aligned} \quad (2)$$

We can improve efficiency by not evaluating the distance between the query and the object $d(x, q)$, which is on the right side of the previously mentioned inequalities. Because a range query requires selecting all objects x such that $d(x, q) \leq r$ (where r is the query range), we can use the lower bound value for the distance $d(x, q)$ to avoid its evaluation. In fact, when

$$d^L(x, q) > r$$

x does not belong to the result set of the query q. We refer to this check as an *exclusion test*. Here, we assume that the distances $d(q, p)$ and $d(x, p)$ are precomputed.

When this test fails, we can use the upper bound $d^H(x, q)$ to select objects without the direct evaluation of their distances to the query. In fact, when

$$d^H(x, q) \leq r$$

then x belongs to the result set. We refer to such a check as an *inclusion test*.

Figure 2 illustrates the principle of this pivoting technique in a 2D Euclidian space. In Figure 2a, we show an example of an exclusion test. We know the distance between any object and p. The area shown in blue represents the region of objects x that don't belong to the query result. In Figure 2b, we show an example of an inclusion test. The orange circle represents the region of objects that don't need to be evaluated and belong to the result set. More details about the pivoted-based algorithms are available elsewhere.[3,4] Bustos, Navarro, and Chavez systematically studied selecting pivots,[11] and proposed and tested several strategies. However, these strategies aren't suitable for data streams (as needed in our case) but are suited for static data sets. For this reason, in the next section we develop an approach for selecting the pivots specifically designed for metric data streams.

The pivoted stream

We evaluate each MPEG-7 stream at the source station and extract n_d visual descriptors from each S-frame to represent the frame's content. The functions D_i and d_i normally support a generic descriptor i. The former extracts the feature from the image, and the latter evaluates the distance (or dissimilarity) between the two images' features. For instance, given two images x and y, we can evaluate the distance by applying d_i on the corresponding features $d_i(D_i(x), D_i(y))$. For

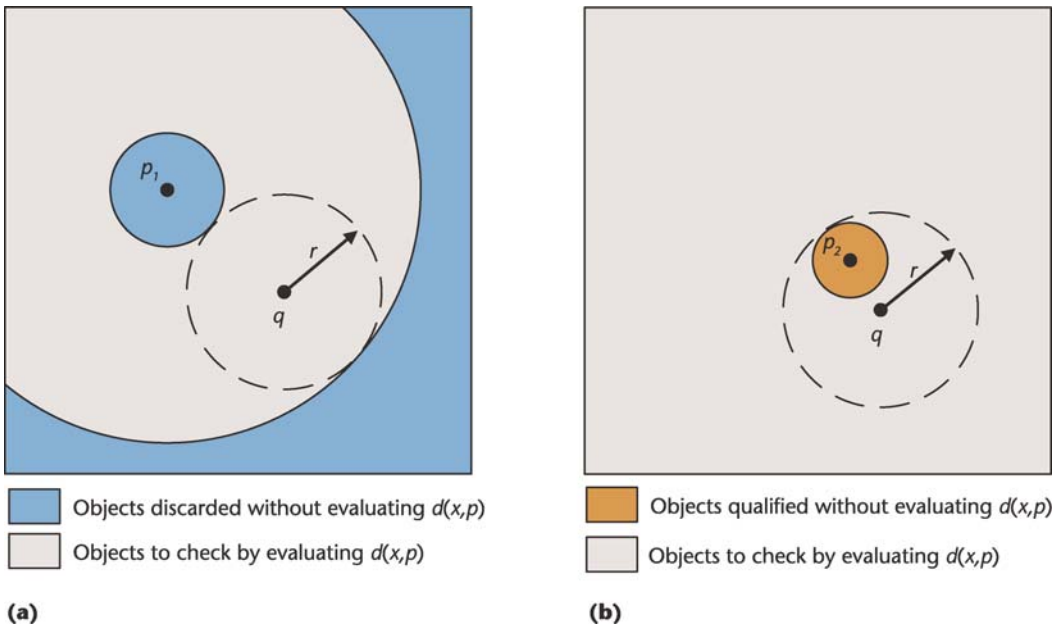

Figure 2. Example of an (a) exclusion test and (b) inclusion test.

simplicity, we use the notation $d_i(x, y)$ to indicate the distance between the features extracted by descriptor D_i. Similarly, we analyze each query filter and extract n_d visual descriptors.

We compare all S-frames with all query filters through the visual descriptors and the distance functions d_i, and use the symbols f and q to indicate a video stream S-frame and the query filter image, respectively. The user only sees S-frames whose distance to the query filter is lower than a certain bound r, that is $d_i(f, q) < r$. We call this a *range query* (the selection of all frames with a distance to the query lower than r), and use the pivoted stream technique to execute these queries.

Moreover, we assume that the receiver station maintains a set of n_q filters $q_1, \ldots, q_{nq}$. The pivoted stream principle elects one S-frame f as a pivot frame p. At the source station, the distances between the current pivot frame p and the successive S-frame f are precomputed. Figure 3 illustrates this principle.

These distance measures are attached to each S-frame f together with their MPEG-7 descriptors. In this way, during the filtering phase, the receiver can exploit these precomputed distances to skip some distance evaluations between the S-frames and the queries. New pivot frames are introduced in the stream periodically so that the distance between the

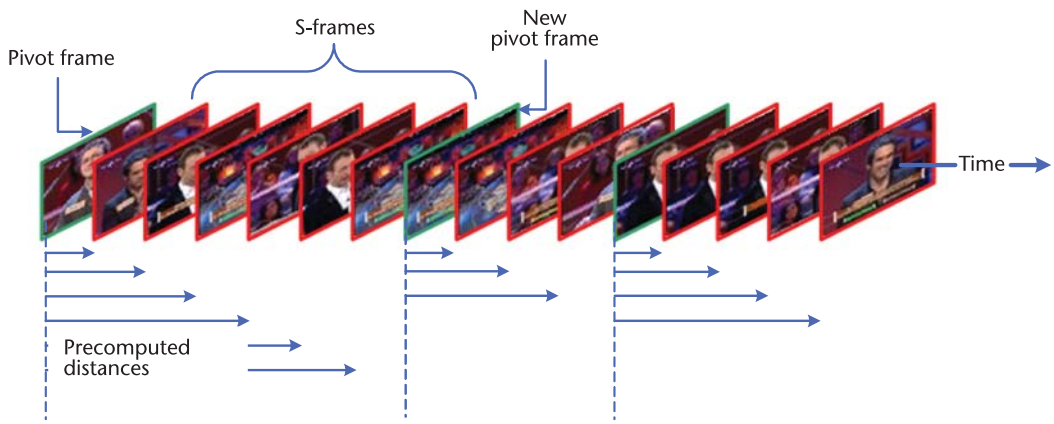

Figure 3. Illustration of the pivoted stream principle.

Figure 4. Pivot filtering algorithm for range queries.

```
for j = 1 to n_q
    for i = 1 to n_q
        # exclusion test:
        if d_i^L = (q_j,f) > r_i then continue;
        else
            # inclusion test:
            if not d_i^H = (q_j,f) ≤ r_i then
                # evaluate the distance:
                if d_i = (q_j,f) > r_i then continue;
            end if
        end if
        f → R_ij;
    end if
end for
```

S-frame and the pivot is sufficiently small. Note that we have n_d distinct pivots, one for each descriptor. To determine if an S-frame could become the pivot frame p_i of the i-th descriptor, we define n_d independent thresholds (s_1, ..., s_{n_d}) for each descriptor. If, for a given S-frame f, $d_i(p, f) > s_i$, the frame f is elected as a new pivot frame p and the distances of the following S-frames are evaluated with respect to the new pivot frame.

Having n_d pivots doesn't imply a significant data-transmission overhead. Only the distance measured between the S-frame and the pivot gets sent in using this technique. If this distance is zero, the S-frame is considered a new pivot. In general, the S-frame can simultaneously act as the pivot of many descriptors.

Range queries

Given a data set of metric objects $X \subseteq D$, a range query retrieves all elements $x \in X$ within a distance r to q that is the set $\{\forall x \in X | d(q, x) \leq r\}$. We assume that search radius r_i is associated with each descriptor i, but we can evaluate the choice of r_i by experimenting with different possibilities in the real application and assessing user satisfaction. Figure 4 shows the algorithm for similarity range queries. During a generic time interval, the receiver station filters the source stream and produces the result sets $R_{i,j}$ (for each descriptor d_i and query q_j) formally defined as $R_{i,j} = \{\forall f_{\text{received}} | d_i(f, q_i) \leq r_i\}$. The algorithm's principle is straightforward. First it checks the two pivot tests explained in the previous section. If both of them fail, we must evaluate the distance $d_i(q_j, f)$ directly, to verify if it's smaller than r_i. In this case, we add f to the set $R_{i,j}$, which contains the range query result set for the descriptor i and query q_j. The choice of the threshold s_i affects the method's efficiency. If s_i is too small, many S-frames become pivot frames. If s_i it too large, we might suffer strong performance degradation.

Combining descriptors

The technique described so far is based on the use of a single visual descriptor at a time. However, to improve the filter's effectiveness, we can combine several visual descriptors. Using several visual descriptors improves the similarity measure's quality. For instance, a comparison of two images according to their color distribution and their objects' shape could provide more effective results than the use of a single descriptor.

Although we can use several types of feature combinations, we use a linear combination that is based on the Milos system and provides good retrieval effectiveness.[12]

Given a query q, and an S-frame f, we define the new combined distance d as

$$d(f,q) = w_1 d_1(f,q) + \ldots + w_{n_d} d_{n_d}(f,q) = \sum_{i=1}^{n_d} w_i d_i(f,q) \quad (3)$$

where w_i is the weight assigned to visual descriptor i. Note that the new distance d is still metric. In general, we can prove that a linear combination of metric functions d_i is metric.

We can compute the combined distance d (in Equation 3) by applying the algorithm described in the previous section to all visual descriptors. However, we can obtain an efficiency improvement if we use precomputed distances to obtain two bounds of the metric distance for combined descriptors. The technique requires the computation of both bounds $d^L(f, q)$ and $d^H(f, q)$ incrementally, separately adding each descriptor's contribution as follows:

$$d^L(f, q) = \sum_{i=1}^{n_d} w_i d_i^L(f, q),$$

$$d^H(f, q) = \sum_{i=1}^{n_d} w_i d_i^H(f, q)$$

During the evaluation we can exploit the d^L partial sum by testing if it's greater than the radius r. If the test is successful, we can avoid

Table 1. Video sequences in the TREC Video Retrieval Evaluation 2005 evaluation set used in our experimentation.

Sequence number	Sequence name
1	20041106_110000_MSNBC_MSNBCNEWS11_ENG
2	20041115_133000_MSNBC_MSNBCNEWS13_ENG
3	20041118_183000_NBC_NIGHTLYNEWS_ENG
4	20041118_230000_NBC_NBCPHILA23_ENG

the complete evaluation of d^L because, in this case, we are sure that the query doesn't belong to the result set.

Because the order in which the algorithm considers various descriptors when evaluating d^L is important, we dynamically adapt the algorithm by attempting to evaluate first the descriptors that produce greater values of the term , increasing the probability that $d^L > r$ before the end of the loop. Basically we swap the last two descriptors' evaluation order if the first one produces a small contribution.

As soon as we evaluate d^L and d^H, and if both inclusion and exclusion tests fail, the algorithm can take advantage of the computation already performed. It proceeds by substituting the terms $w_i d_i^L(f, q)$ and $w_i d_i^H(f, q)$ of the two summations d^L and d^H with $w_i d_i(f, q)$, which are the weighted distances between the current query and the frame f. The algorithm obtains this by iterating again over the descriptors. During the iteration we get increasingly precise values of the bounds. The aim is to terminate the algorithm before the complete evaluation of the real combined distance, that is, $d^L \equiv d^H \equiv d(f, q)$. Again, we try to keep the best order in the descriptor evaluation.

Performance evaluations

In the performance evaluation we aimed to determine efficiency improvements gained when using pivot filtering. We performed the evaluation using four video sequences of the TREC Video Retrieval Evaluation 2005 workshop's evaluation set (see Table 1 and http://www-nlpir.nist.gov/projects/trecvid/), with a total duration of about two hours. We use frame filters taken from the stream itself, which lets us test the worst-case performance because the probability that the pivot test will fail is higher for queries closer to the S-frames.

In the experiments, we used four queries that represent specific aspects of the video stream, namely

- a frame taken from a specific commercial (the commercial query),
- the beginning of the news (the news query),
- a frame taken from a basketball game (the basket query), and
- a frame taken from a weather forecast (the weather query).

We tested three MPEG-7 visual descriptors: scalable color, color structure, and edge histograms. We used the MPEG-7 XM reference software for their extraction. We based the similarity computation on the distance measures proposed in the XM reference software's MPEG group. The descriptors are vectors and their distances are metric. We used the following parameters for each descriptor: scalable color used 64 coefficients with 0 bitplanes discarded, color structure used 64 coefficients, and edge histograms with 80 coefficients and no parameters.

The system filters the S-frames using range queries and selects all S-frames whose distance from the query filter is lower than a certain value r. The brute force algorithm requires the comparison of all S-frames with the query filter. Pivot use can reduce the total number of distance computations. This number is equal to the number of similarity measures computed between the filters and the pivots plus the number of similarity measures between the filters and the S-frames, if the pivot test fails. All the experiments measure the percentage of distance similarity computations needed, when using the pivot method, with respect to the number of distance computations required when using the brute force algorithm. Within a given period of time, we denote the distance computation cost as

$$\rho_{i,j} = \frac{n_{i,j}}{n_f}$$

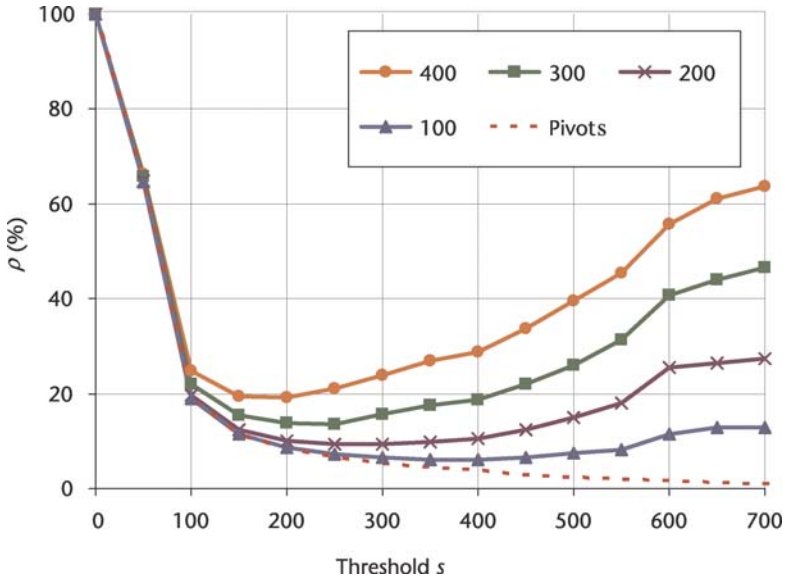

Figure 5. Distance computation cost ρ versus the threshold s for the commercial query. We use scalable color and show graphs at different query ranges and when we only use the pivots.

where $n_{i,j}$ is the number of distance computations for descriptor d_i and a given query q_j, and n_f is the number of selected frames ($0 < \rho_{i,j} \leq 1$). The objective is to obtain a value of ρ as small as possible ($\rightarrow$0), if we want to minimize the number of distance computations.

In general, the total number of distance similarities we evaluate depends on the query range r_i and the threshold s_i that the server uses to choose the pivot frames. The evaluation reported next analyzes performance ρ improvement obtained using the pivot method and varying these two parameters.

Performance sensitivity with respect to the threshold

In Figure 5, we report ρ as function of the threshold s_i for the scalable color descriptor and for different query ranges. The dotted line in the figure shows the percentage of distance computations caused only by pivot evaluations. This curve represents a lower bound of computation for the pivoted stream algorithm because the distance between the pivots and the query must always be calculated. Obviously, when the threshold value increases, this contribution decreases, because pivot frequency diminishes.

The four solid lines in Figure 5 represent the computation cost for distinct range queries as function of the threshold. In general, for a range query with $r > 0$, the number of distance evaluations decreases with the threshold until a certain point, after which the number grows again. This behavior occurs because, when the threshold is small, the number of distances is dominated by the distance evaluation $d(p, q)$ between the pivot and the query. As the threshold grows, the pivoted stream algorithm works progressively better. After a certain point, the number of pivot faults increases and the total number of distance computations grows. In other words, for a given search radius, there is a threshold for which the number of distance evaluations is lower. In Figure 6, we represent the following three curves:

- the best threshold value for each query range,
- a fixed threshold value equal to 250 (approximate average over the different optimal values), and
- the percentage of the qualifying S-frames.

The first curve provides an indication of performance variability with respect to threshold choice. The most significant and frequently used range values are the intermediate ones, between 100 and 400, because they return a reasonable number of results. For these values, the differences between the first two curves aren't significant, which means that the optimal choice of the threshold value is not critical. In general, the performance improvements are higher for lower query ranges.

In Figure 7, we show a synthetic view of performance improvement for the four test queries using the three feature descriptors for range queries. We use the following thresholds: 250 for scalable color, three for color structure, and 9.60 for edge histograms. Using these thresholds, 7 percent of the S-frames become pivots. We chose the thresholds as averages over the different optimal values, and we used a radius of 200 for scalable color, three for color structure, and five for edge histograms. This experimental result shows that the pivoted stream method gives significant performance improvements for all different queries and for all feature descriptors, with the edge histogram descriptor that gives the worst results for all four queries. We chose the radii of the range queries so that the number of results was on the order of tens. Only the commercial query for the edge histogram descriptor gives many more matches (several hundreds) due to the high color similarity

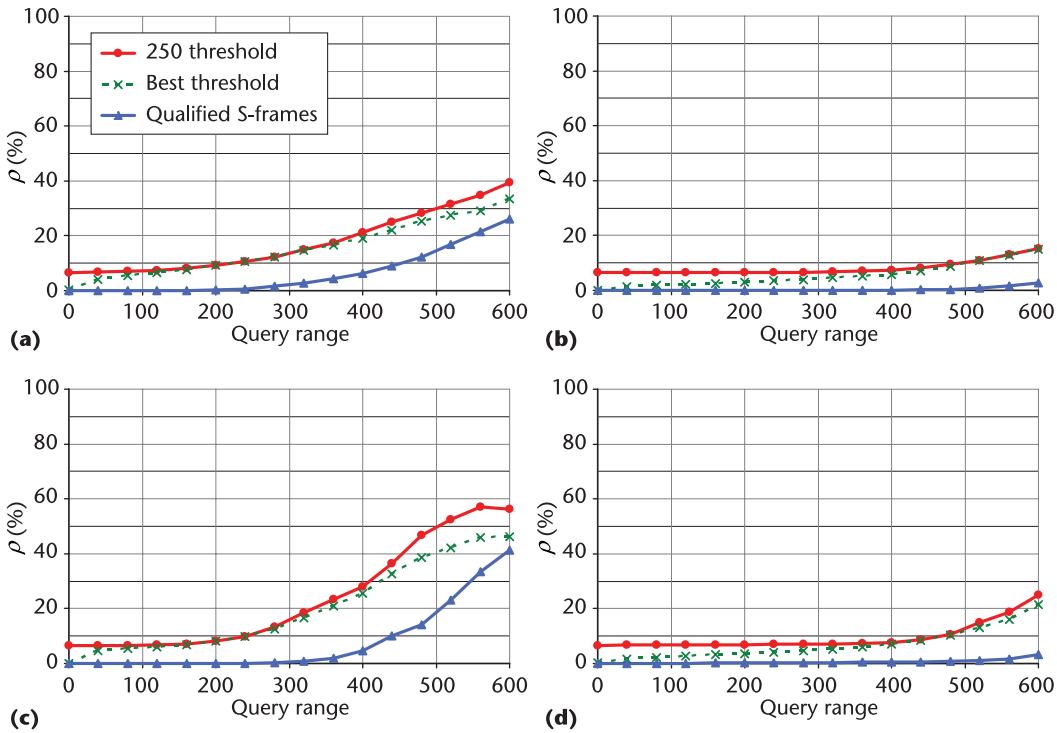

Figure 6. Distance computation cost ρ for different queries versus the query range using the scalable color descriptor: (a) commercial, (b) news, (c) basket, and (d) weather.

between different S-frames of this particular commercial.

Linear combination of descriptors

In the last set of experiments, we study system performance when we linearly combine the three visual descriptors (scalable color, color structure, and edge histograms). For convenience, we chose the weights w_i as the inverse of the radii of the range queries experiments. Consequently, assuming $r = 3$ for a combined range query, the number of results is the same order of magnitude as for the single descriptors. In particular, in our experiments we have chosen r so that if a frame f passes all three single descriptor filters, it also passes the combined descriptor filter. On the contrary, if the frame f passed the combined descriptor filter, it passes at least one of the single descriptor filters.

Figure 7 shows the global result of the query with combined descriptors for range queries. In this case, the cost represents the distance computation cost averaged over all the descriptors. In all the experiments, the advantage of the pivots is evident: ρ ranges from 2.62 percent of news to 4.46 percent of commercial. The number of results for the range queries is 14 for commercial, eight for news, 25 for basket, and 114 for weather.

Performance estimation

This section provides more general evidence of our approach's efficiency by developing a statistical estimation of the computational cost ρ. We must note that the pivoted filtering performance depends on data set characteristics and query choice. Therefore, in principle, it's not possible to give a full theoretical estimation of performance gains. However, with a certain data set as a given (the

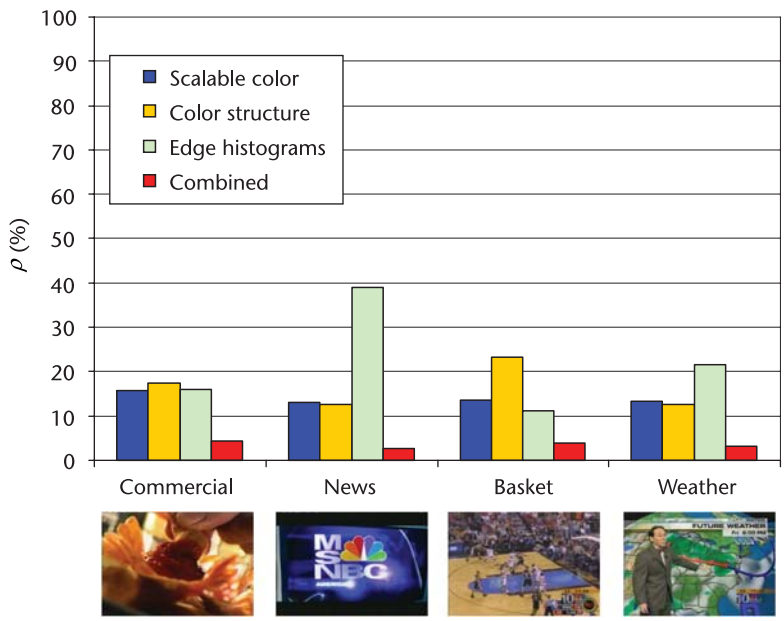

Figure 7. Distance computation cost ρ for different range queries and three different image descriptors combined.

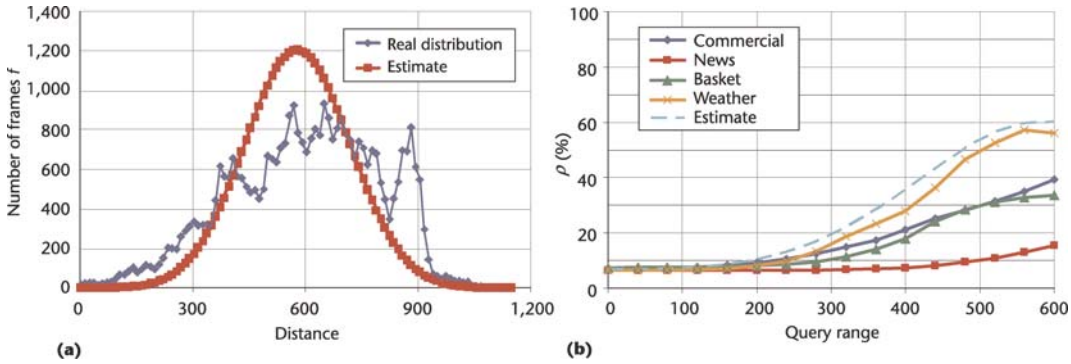

Figure 8. (a) Estimation of the $d^L(f, q)$ *distribution for the commercial query with a normal distribution. (b) Comparison of estimation of the distance computation cost* ρ *with the four queries for different range queries, using the scalable color descriptor.*

video stream), we try to predict performance with the upper bound computational cost ρ independent from the query. If we take the queries from the data set, the performance tends to decrease because the number of results is significant and the probability that the pivots will fail is high. For this reason, we estimate the distribution of the distances $d^L(f, q)$ and $d^H(f, q)$ by considering a certain number of S-frames, say $\tilde{q}_1 \ldots \tilde{q}_m$, of the stream as queries. The greater the value of m, the more precise the distribution's evaluation. We evaluate the mean $\tilde{d}_i$ and the standard deviation $\tilde{\sigma}_i$ of the distance of the i-th query $\tilde{q}_i$ from all the data set's pivot frames (for a given threshold)—that is,

$$\tilde{d}_i = E_f\{d(\tilde{q}_i, p)\}$$
$$\tilde{\sigma}_i^2 = E_f\{d(\tilde{q}_i, p)^2\} - \tilde{d}_i^2$$

Then we evaluate the corresponding mean values ($\tilde{\mu}_{qp}$ And $\tilde{\sigma}_{qp}$) over the entire set of m queries:

$$\tilde{\mu}_{qp} = E_i\{\tilde{d}_i\}$$
$$\tilde{\sigma}_{qp} = E_i\{\tilde{\sigma}_i\}$$

Assuming that the distance of queries and pivots from the other S-frames are statistically independent random variables (which is fairly true if the subset of queries and pivots are disjointed), and according to Equation 2, we obtain:

$$\mu_L = \tilde{\mu}_{qp} - \mu_{fp}$$
$$\mu_H = \tilde{\mu}_{qp} + \mu_{fp}$$
$$\sigma_L^2 = \sigma_H^2 = \tilde{\sigma}_{qp}^2 + \sigma_{fp}^2$$

where μ_{fp} and σ_{fp} denote the mean and the standard deviation for the distances between the frames and pivots. To determine the number of distance evaluations, we approximate the distribution of the distances $d^L(f, q)$ and $d^H(f, q)$ with normally distributed functions. This assumption is quite realistic (see Figure 8a).

Given a certain search radius r, the computation cost ρ can be finally evaluated by calculating the probability that $d^L(f, q) > r$ and that $d^H(f, q) \leq r$. Figure 8b shows this estimation's result compared with the four queries of the previous experimentations for several random queries $m = 100$. In the worst case, We can use this estimate to predict system performance for a given threshold.

Exploiting multiple pivots

This section describes how we extend the pivoted stream technique by using multiple pivots. The idea is straightforward: we send along with the video stream the precomputed distance between the last k pivot frames and the current S-frames. The case $k = 1$ corresponds to the single pivot discussed so far in this article. From the receiver station's perspective, this certainly means less distance computations. However, the sending station must evaluate several distances among the S-frames and the k pivot frames. Most of the computational effort for the sending station is due to the extraction of the visual descriptors, which is, in some cases, even three orders of magnitude higher than the distance computation cost.

Figure 9 shows the ρ costs when using multiple pivots. For short-range radii, the advantage is significant; as the search radius becomes larger, the advantage of using multiple pivots becomes negligible. Nevertheless, in the case $r = 120$, the percentage of results is 0.04 percent, which means that for a transmission of one hour and selecting six frames per second, we take on average $6 \times 60 \times 60 \times 0.0004 \approx 8.6$ results, which is quite reasonable for a real TV application. In this case, we have $\rho = 6.39$ percent with one pivot, $\rho = 1.88$ percent with eight pivots, $\rho =$

1.25 percent with 16 pivots, and $\rho = 1.04$ percent with 32 pivots. The case with 32 pivots exhibits a performance gain of about six times the single pivot case; these performance figures can justify the use of multiple pivots.

Conclusion

There are many possible applications in which the pivoted stream technique could be useful. For example, this method could help generate statistics to compute how many times commercials are transmitted or to select a television program's start time to begin recording it. Another possible use is in property rights protection for videos or images distributed by news agencies, such as Reuters. These agencies have special agreements for use of material they sell to TV broadcasters. A filtering application such as the one proposed in this article could help news agencies discover illegal use of this material. In addition to applications in television environment, other applications include earth satellite surveillance, police surveillance, and so on.

We think that our proposed approach opens many possible future lines of investigation and we intend to explore the use of other features, such as text (optical character recognition captions, subtitles, and so on) and audio descriptors. In addition, we can study further performance improvements if the number of query filters is large. In these cases, it would possible to organize the query filters using a specific access structure for metric spaces. **MM**

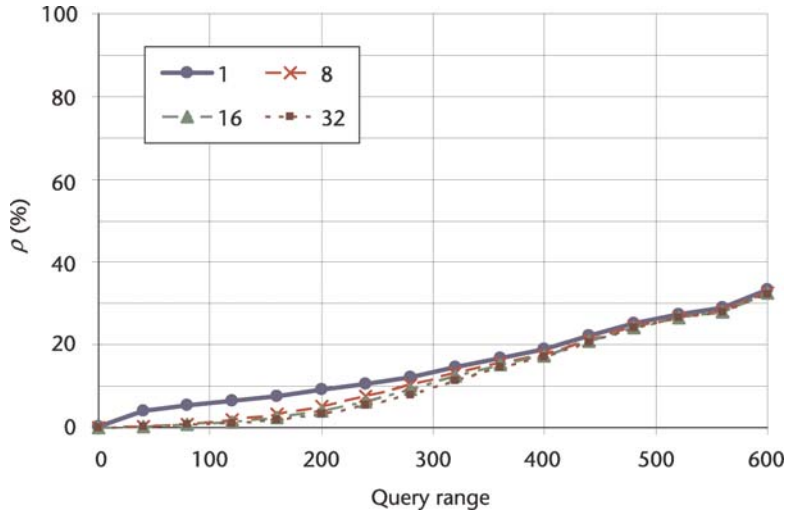

Figure 9. Distance computation cost ρ for different numbers of pivots.

References

1. S. Krishnamachari et al., ''Multimedia Content Filtering, Browsing, and Matching Using MPEG-7 Compacts Color Descriptors,'' *Proc. 4th Conf. Advances in Visual Information Systems* (Visual), LNCS 1929, Springer, 2000, pp. 200-211.
2. D. Zhong, R. Kumar, and S.-F. Chang, ''Real-Time Personalized Sports Video Filtering and Summarization,'' *Proc. 9th ACM Int'l Conf. Multimedia*, ACM Press, 2001, pp. 623-625.
3. W.A. Burkhard and R.M. Keller, ''Some Approaches to Best-Match File Searching,'' *Comm. ACM*, 1973, vol. 16, no. 4, pp. 230-236.
4. M. Shapiro, ''The Choice of Reference Points in Best-Match File Searching,'' *Comm. ACM*, 1977, vol. 20, no. 5, pp. 339-343.
5. *Information Technology—Multimedia Content Description Interface—Part 6: Reference Software*, ISO/IEC 15938-6:2003, ISO, 2003.
6. P. Ciaccia, M. Patella, and P. Zezula, ''M-Tree: An Efficient Access Method for Similarity Search in Metric Spaces,'' *Proc. Very Large Data Bases*, Morgan Kaufmann, 1997, pp. 426-435.
7. T.W. Yan and H. Garcia-Molina, ''Index Structures for Information Filtering Under the Vector Space Model,'' *Proc. 10th Int'l Conf. Data Engineering*, IEEE CS Press, 1994, pp. 337-347.
8. T. Bozkaya and Z.M. Özsoyoglu, ''Indexing Large Metric Spaces for Similarity Search Queries,'' *ACM Trans. Database Systems*, vol. 24, no. 3, 1999, 1999, pp. 361-404.
9. V. Dohnal et al., ''D-Index: Distance Searching Index for Metric Data Sets,'' *Multimedia Tools and Applications*, vol. 21, no. 1, 2003, pp. 9-13.
10. E. Chávez et al., ''Searching in Metric Spaces,'' *ACM Computing Surveys*, vol. 33, no. 3, 2001, pp. 273-321.
11. B. Bustos, G. Navarro, and E. Chavez, ''Pivot Selection Techniques for Proximity Searching in Metric Spaces,'' *Pattern Recognition Letters*, vol. 24, no. 14, 2003, pp. 2357-2366.
12. G. Amato et al., ''Milos: A Multimedia Content Management System for Digital Library Applications,'' *Proc. 8th European Conf. Research and Advanced Technology for Digital Libraries* (ECDL), LNCS 3232, Springer, 2004, pp. 14-25.

Fabrizio Falchi is a PhD student in information engineering at the University of Pisa and in informatics at the Faculty of Informatics of Masaryk University of Brno. He is also a research fellow with the Networked Mul-

timedia Information Systems Laboratory at the Institute of Information Science and Technologies, Italian National Research Council (ISTI-CNR). His research interests include similarity search, distributed indexes, multimedia content management systems, content-based image retrieval, and peer-to-peer systems. Contact him at fabrizio.falchi@isti.cnr.it.

Claudio Gennaro is a researcher at ISTI-CNR. His research interests include performance evaluation, similarity search, information retrieval, distributed and parallel systems, multimedia content management systems, and multimedia document modeling. Gennaro has a PhD in computer and automation engineering from Politecnico di Milano. Contact him at claudio.gennaro@isti.cnr.it.

Pasquale Savino is senior researcher at ISTI-CNR. His research interests include multimedia information indexing and retrieval, multimedia content management, and digital libraries. Savino graduated in physics from the University of Pisa, Italy. Contact him at pasquale.savino@isti.cnr.it.

Peter Stanchev is a professor at Kettering University, US, and professor and chair at the Institute of Mathematics and Informatics, Bulgarian Academy of Sciences. His research interests include multimedia systems, multimedia semantics, and medical systems. Stanchev has a PhD in mathematics and computer science from Sofia University. Contact him at pstanche@kettering.edu.