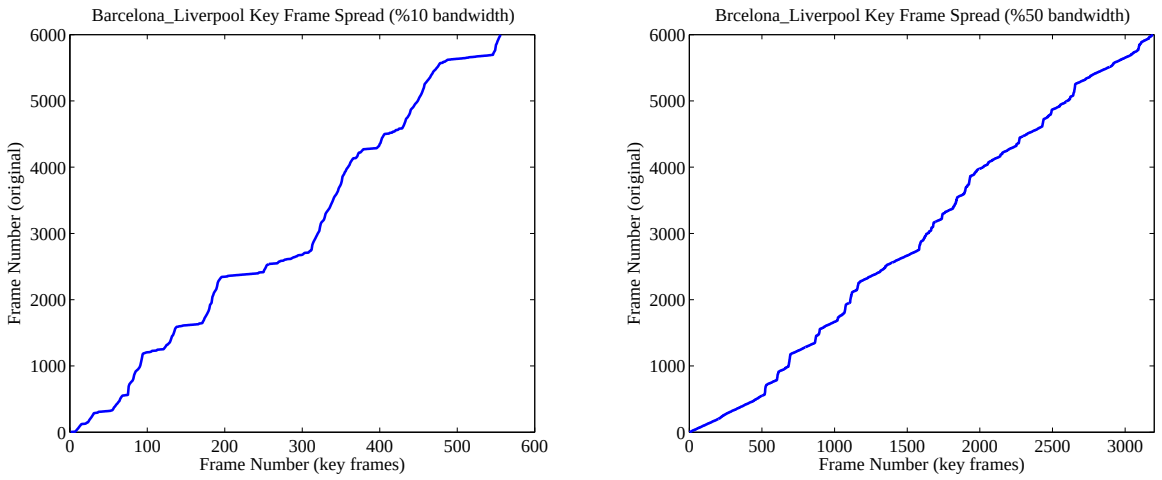


(a) (b)

Fig. 1. Key frames are well spread using the SVD based algorithm

I used the new code and ran it on several videos. Some of the videos are from our video library which is available online [1].

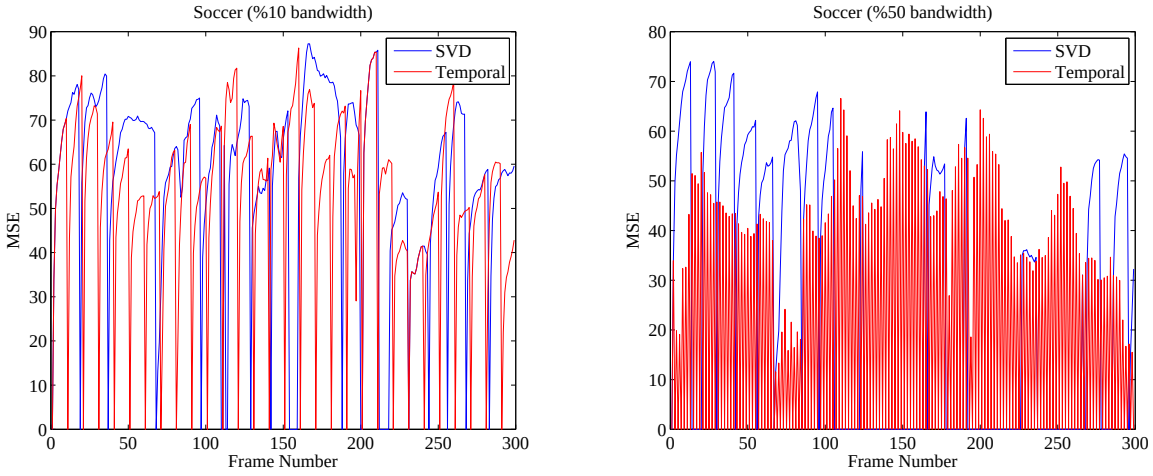
- It looks like the algorithm is spreading the key frames along time. This can be verified as in Figure 1. The graphs show the distribution of key frames with respect to original frames. They are produced by finding the appropriate thresholds which result in %10 and %50 bandwidth respectively. This property may be good for smooth motion when we have sufficient bandwidth. On the down side, video shots are not of equal length/importance/content, therefore we might need a guided key frame extraction scheme that extracts more key frames from some shots.
- Our algorithm was compared against temporal sampling where for example every 10th frames is selected as a key frame to achieve %10 bandwidth. MSE was computed for both cases. Frames from the original video that do not have any corresponding key frame were compared against the last key frame. For example suppose we have 10 frames, and we need to extract only 2 of them. SVD (what we call our method for now) and Temporal sampling are used for the task. For SVD all frames 1-7 would be compared against frame 1 and frames 8-10 would be compared against 8.

  The results shown in Figure 2 indicate that neither of the methods outperforms the other in terms of MSE. The reason could be that in both cases key frames are spread (almost) uniformly. Moreover, had SVD resulted in non-uniform key frame distribution, the results would have been worse. I feel using MSE and similar metrics are not appropriate for comparison as they ignore the content. And

TABLE I

EXTRACTING KEY FRAMES

Original	1	2	3	4	5	6	7	8	9	10
SVD	1							8		
Temporal Sampling	1					6				

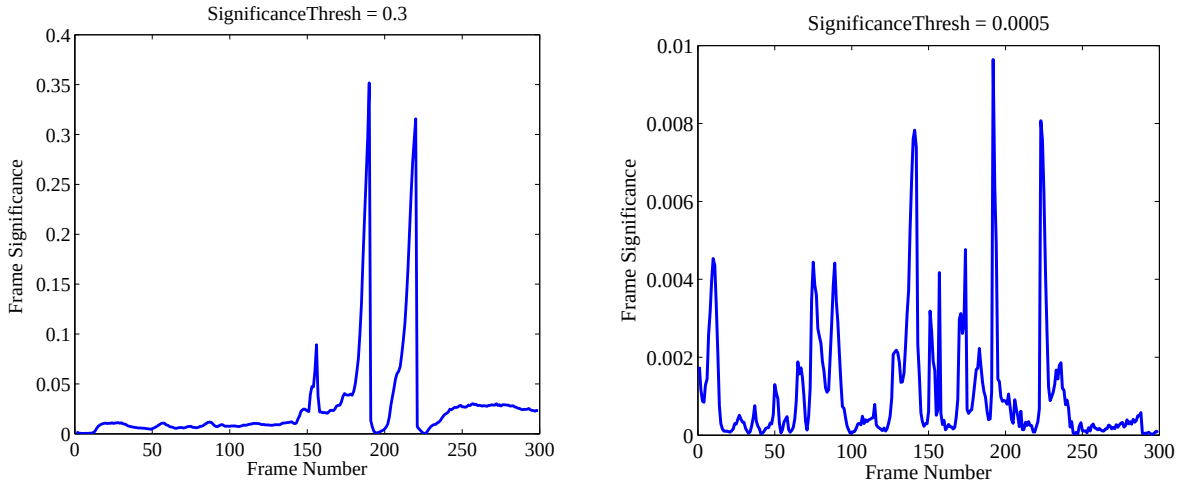


(a) (b)

Fig. 2. Compating MSE for SVD based algorithm and Temporal Sampling

this is the whole point of our method. For example to compute MSE two frames from different shots might be compared against each other which does not make sense. We also tried SSIM that claims to capture the structure of the image. However, both MSE and SSIM are very sensitive to camera motions. Therefore, even a small spatial shift/rotation/scaling would result in a bad MSE/SSIM value.

- I think the metric computation should be done shot-wise and then added up in some manner. Thus we are not comparing irrelevant frames from different shots against each other. We need to make sure that our algorithm selects the best key frames for each shot minimizing some metric (in best case). To find/devise such a metric I need to know more about the nature of the key frames and the mathematical theory behind it. One technical question is also how to set the significanceThresh to detect the shot boundaries as different values result in different graphs with few or many peaks depending on the value (see Figure 3)
- This is only an idea: Color histograms have proved to be good for shot boundary detection as there



(a) (b)

Fig. 3. Frames with peak significance for diffrent values of SignificanceThress

is much change from one shot to the next. Can we use it only for this purpose and then extract the key frames later to minimize some metric as an optimization problem? I feel color histograms do not reflect the change WITHIN the frames. For example if a player is running across the field the color histogram does not change while we would like to capture this event.

REFERENCES

[1] NSL Video Library and Tools, “http://nsl.cs.sfu.ca/wiki/index.php/video_library_and_tools.”