Processing Historical Ottoman Archives: Past, Present, and Future

Ediz ŞAYKOL, Ph.D.

Past

E. Saykol, A.K. Sinop, U. Gudukbay, O. Ulusoy, E. Cetin.

Content-Based Retrieval of Historical Ottoman Documents Stored as Textual Images,

IEEE Transactions on Image Processing,

Vol.13, No. 3, pp. 314-325, March 2004.

To store historical archives

- A content-based retrieval system can provide efficient access to the documents from the Ottoman Empire archives
- Historians want the documents in historical archives to be stored in image form because the documents include not only text but also drawings, portraits, miniatures, signs, ink smears, etc., which might have an associated historical value.
- Another reason for the digitization of these documents is that the hard copies of some of the documents are deteriorating as time passes.

Mirasül - Hat Sahife 193

Kalem-i Şecerî (Bitki şekilleri ile yazı)

Examples of handwritten Ottoman textual images:

(a) textual image containing a miniature painting and

(b) a document containing plant sketches.

Storing Ottoman scripts

- The Ottoman script is a connected script based on the Arabic alphabet. A typical word consists of compounded letters as in handwritten text. Therefore, an ordinary textual image compression scheme for documents containing isolated characters cannot encode Ottoman documents.
- Here, the document images are compressed by constructing a **library of shapes** that occur in a document, and the compressed textual images contain pointers into the **codebook** for each occurrence of these shapes together with the **coordinate information**.

Textual Codebook Model

984 16 83 7 8	884 2 −54 14 20	1539 5 55 8 7
556 2 −28 7 9	2000 920 13 9	228 11 −45 10 13
2322 0 62 7 8	1187 6 −34 12 22	1008 14 −35 13 21
1838 7 69 7 8	2384 9 −38 7 7	1253 2 35 7 7
1802 5 −18 7 13	1711 7 0 10 15	2062 5 −10 10 12
2258 5 −8 15 29	97 15 2 7 8	1624 0 13 8 8
819 6 −43 11 12	2337 1 58 9 10	1597 4 −35 10 13
682 21 −15 8 39	97 15 46 7 8	433 1 −47 11 11
1802 6 23 8 16	2456 0 3 11 27	1804 36 3 7 31
1378 7 −5 8 46	2331 8 −8 14 33	2332 0 31 14 23
2263 14 −29 7 44	1455 11 12 8 24	1838 8 17 7 8

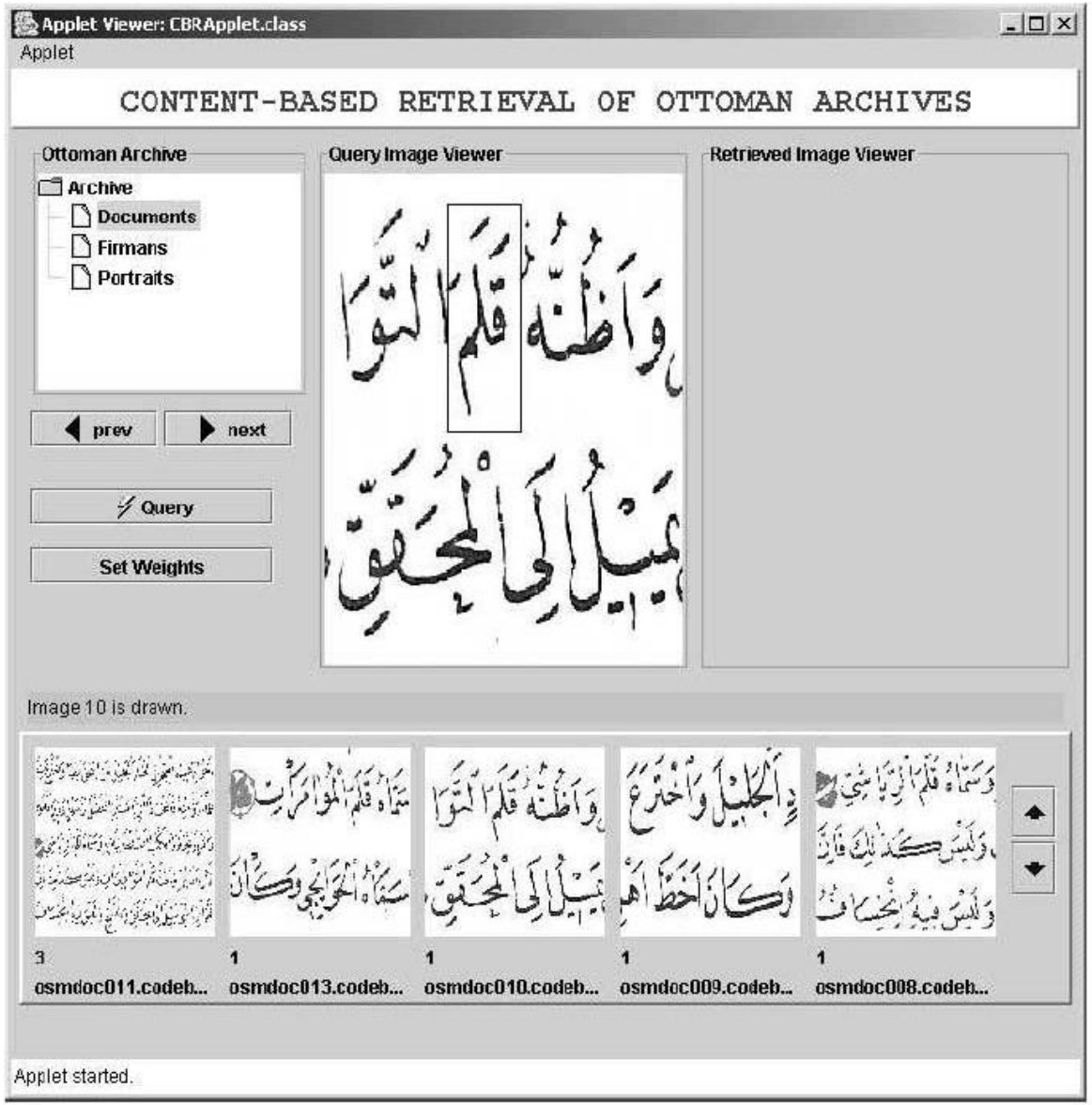
...

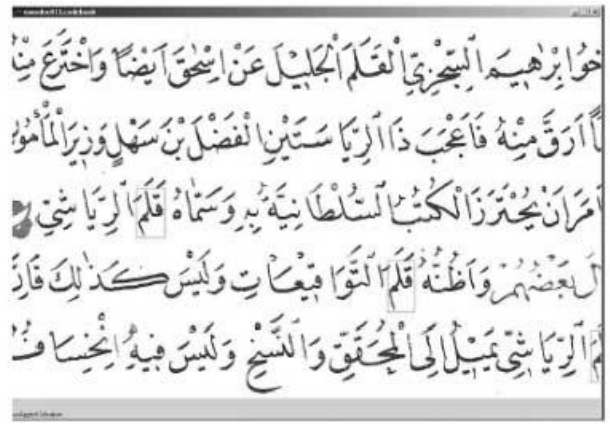
The first of five integers in a row is the symbol identifier, and the remaining four values are
x-*offset*,
y-*offset*,
symbol-width, and
symbol-height, respectively.

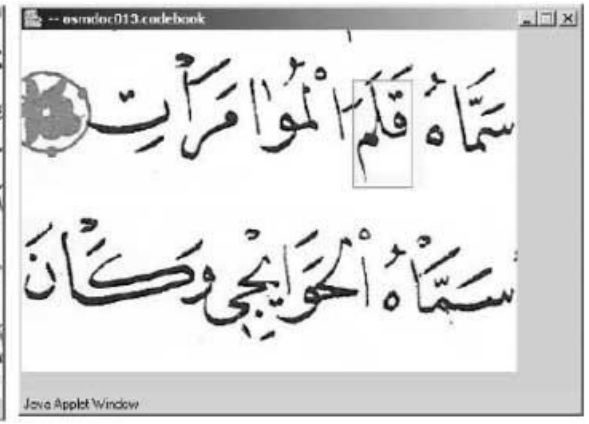
The offset coordinates denote the Euclidean distance between the lower left corners of the two adjacent symbols.

content-based retrieval system

- Query specification is performed by identifying a region containing a word, a phrase, or a sentence in a document image.
- The resulting documents are ranked in the decreasing order of their similarity to the query image, which is determined by the total number of symbols matched with the query region.
- Ordering of document images based on symbol-wise similarities yields more realistic partial matches than whole picture comparisons.







A query example.

(a) Document image where the query image is specified by a rectangle.
(b) First retrieved document with 3 matches.
(c) Second retrieved document with 1 match.

The matched keywords in (b) and (c) have different scales compared to the keyword in the query image.

Innovative Features

- The document images queried do not have to be binary images, they can be gray level or color images.
  - This makes the system capable of handling **all possible documents** in original forms without any possible distortions due to binarization.
- Symbols in the codebook are extracted from the document images by a **scale-invariant** process, which enables effective and consistent compression for all historical and cultural documents in textual image form.
- The techniques can easily be tailored to other domains of archives containing printed and handwritten documents.

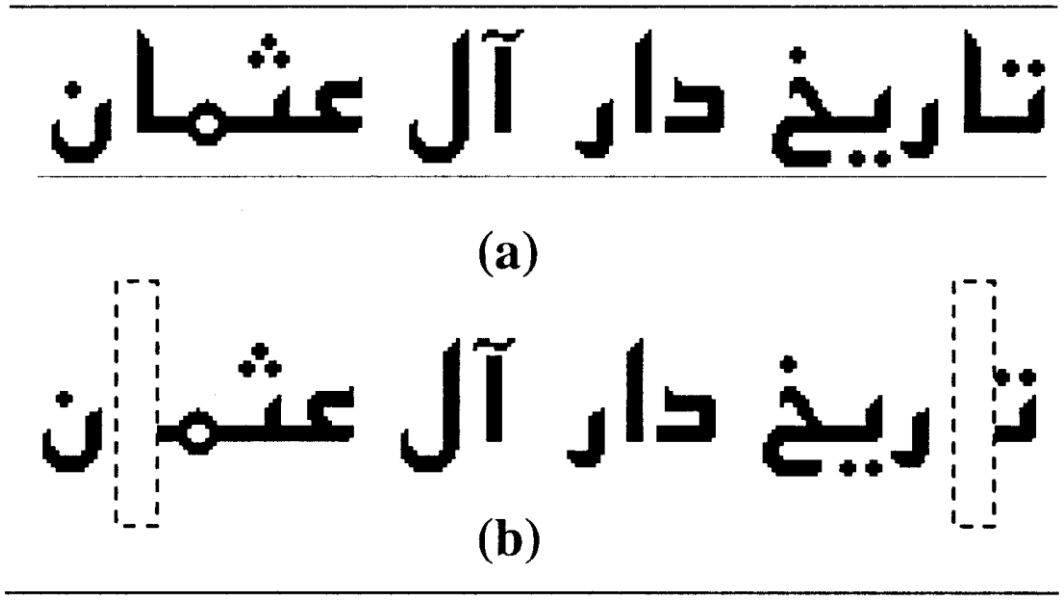
Character (or symbol) extraction

- The character-wise partitioning of the documents written in the Ottoman script is very difficult.
  - there are isolated symbols corresponding to single letters as well as
  - long connected symbols which are combination of letters
- Usually, longer symbols include some letters that can also be found separately inside the document.
- When the smaller isolated symbols are encoded and removed from the document, longer symbols split into smaller symbols.
- Each of these smaller symbols may also be contained in other connected symbols.

Symbol Extraction Process

- The accuracy of this method depends on the fact that there can be enough isolated symbols to split longer connected symbols,
  - a valid assumption in historical Ottoman documents.
- The symbols extracted in the **first pass** of the method are encoded and removed from the document, then the longer symbols split into smaller symbols for the succeeding passes of the method.
- The deletion of the extracted symbols from the document image is performed by sliding the symbol image throughout the document image.
- The algorithm stops when all of the symbols are extracted from the document image during these multiple passes.

Sample Step in Deletion



Deletion of a symbol from a script image corresponding to a date:

(a) image before deletion and
(b) image after deletion of an **L-like** symbol,
which appeared as an isolated symbol extracted earlier
(shown with dashed rectangular regions).

Pseudo-code of Document Compression

---

1. Extract all isolated symbols from D, and put them in CCD.

2. If SL is not empty, split further the items in CCD by checking each element in SL, and augment CCD with the new symbols formed after split.

3. Pick an item S from CCD.

4. By keeping the aspect ratio of S, check the similarities within the elements of CCD.

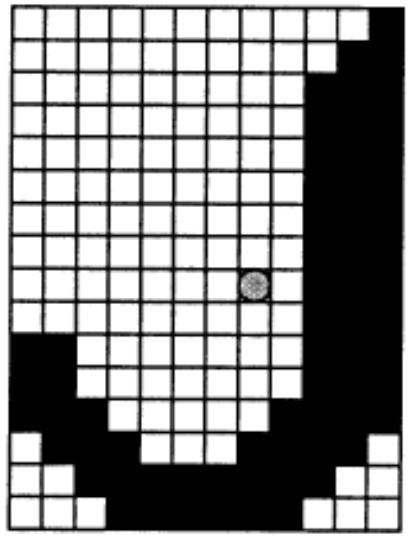
5. For every found highly correlated region within another element S' in CCD, clip that region from S' and encode the location of S in S' into CD.

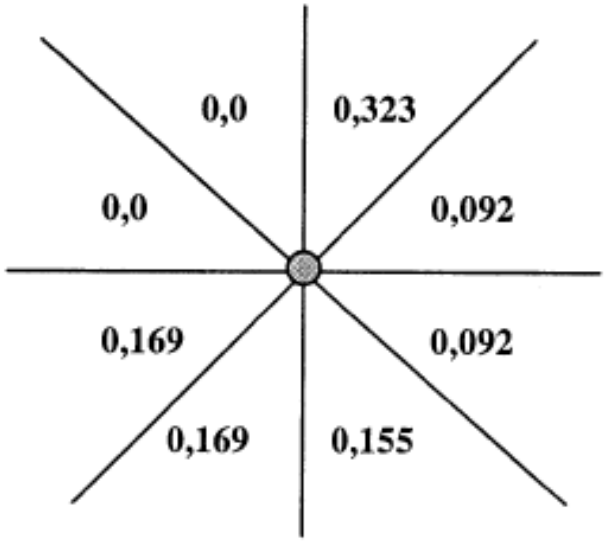
6. Add S to SL, and remove S from CCD.

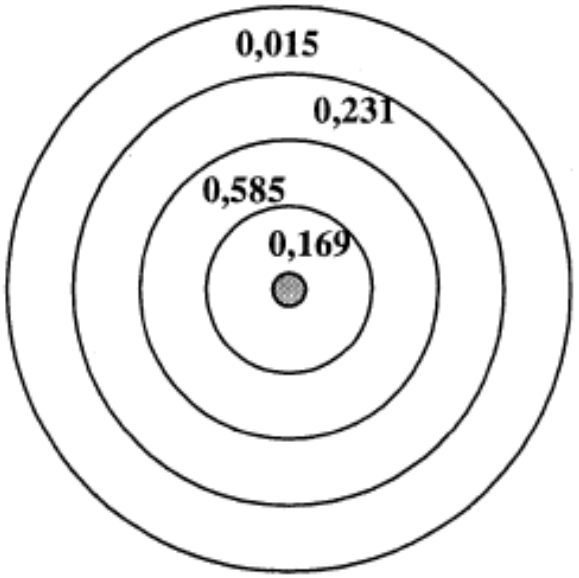
7. If CCD is not empty, go to Step 3.

Let **D** denote the document to be processed,
CCD denote the set of isolated symbols of D,
CD be the compressed textual image representation of D to be used in querying and retrieval, and
SL be the existing codebook at the time of processing D.

At step 3, while picking a isolated symbol from CCD, the **smallest symbol heuristic** can be used, which relies on the tendency that smaller sized symbols will divide other symbols more likely.







Angular Span of a symbol is a vector whose entries are the number of black pixels in Θ-degree slices centered at with respect to the horizontal axis. The entries are normalized by the area of the symbol.

Distance Span of a symbol is also a vector whose entries are the number of black pixels in between the concentric circles centered at cm with radius.
The entries are normalized according to the distance of the farthest two pixels on the symbol.

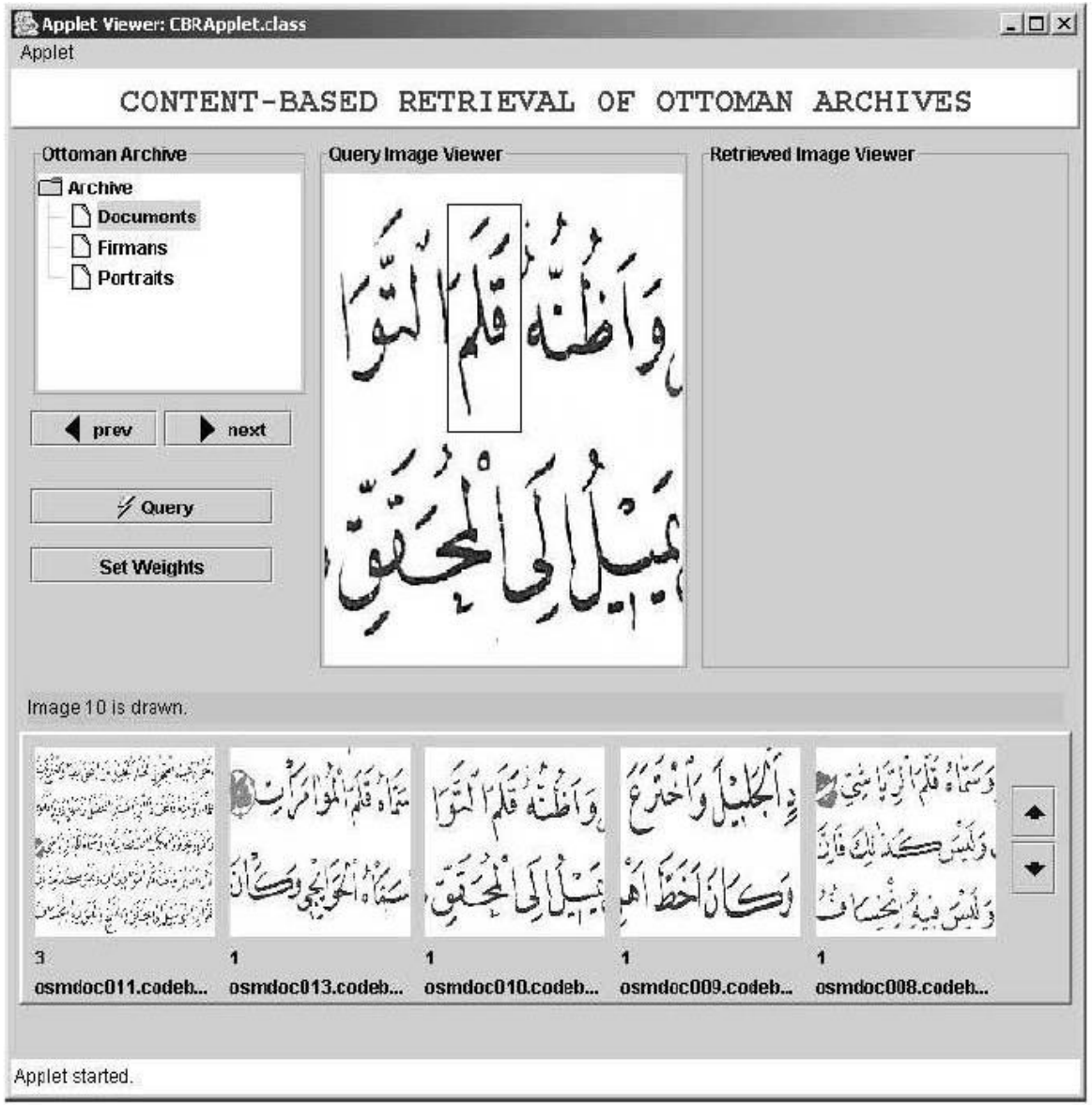
The symbol comparison method has to be **scale-invariant** to detect the different occurrences of the same symbol with different sizes.

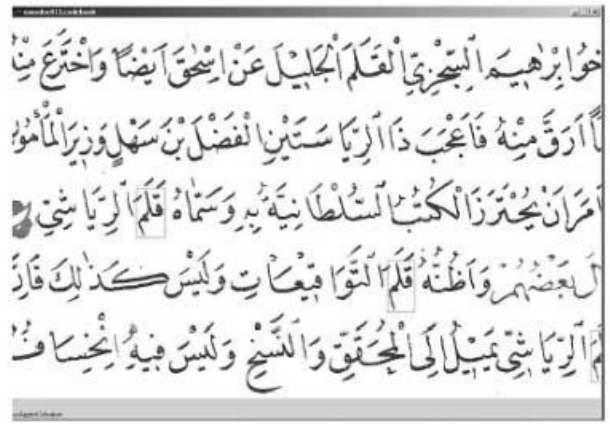
Rotation invariance up to a desired degree can be applied to detect minor rotations, e.g., slanted letters.

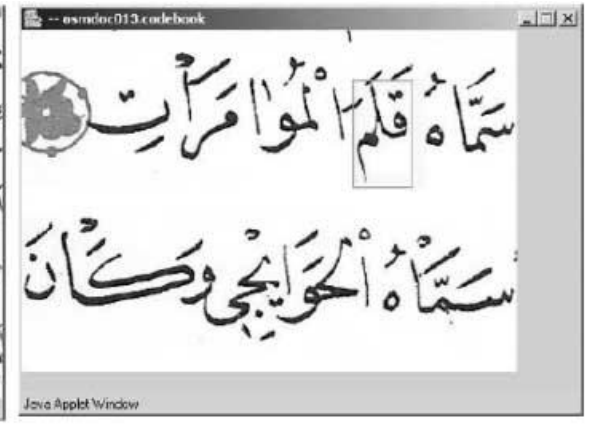
With angular span vector, rotating the symbols Θ-degrees is equivalent to circular shifting the angular span vector one slice.

Querying and Ranking

- Documents are ranked according to the *partial symbol-wise matching* scheme.
- For a query image of ***N*** extracted symbols, a match between the query image and a document ***D*** may contain ***m*** symbols, where $N/2 \leq m \leq N$ by default.
- The similarity of a retrieved document ***D*** is denoted ***m/N*** as for the match. If there are more than one such matches for a document, then the individual similarity measures are summed up to obtain the final similarity measure.







A query example.

(a) Document image where the query image is specified by a rectangle.
(b) First retrieved document with 3 matches.
(c) Second retrieved document with 1 match.

The matched keywords in (b) and (c) have different scales compared to the keyword in the query image.

Present

E. Saykol,

Keyframe-based video mosaicing for historical Ottoman documents,
Turkish Journal of Electrical Engineering & Computer Sciences,

Vol. 24, No. 5, pp. 4254-4266, June 2016.

Video Mosaicking

- Mosaicking can be simply defined as the automatic alignment (or registration) of multiple images into larger aggregates.
- In many cases, it may not be easy to capture a larger image as a whole with enough resolution; hence a partial capturing scheme to cover the entire image is required.
- The existence of **overlapping regions** among consecutive (sub)images is an important clue to merge them in a mosaic image.
- Video frames can also be input such that the frames having sufficient overlapping regions in between need to be selected to achieve better performance.

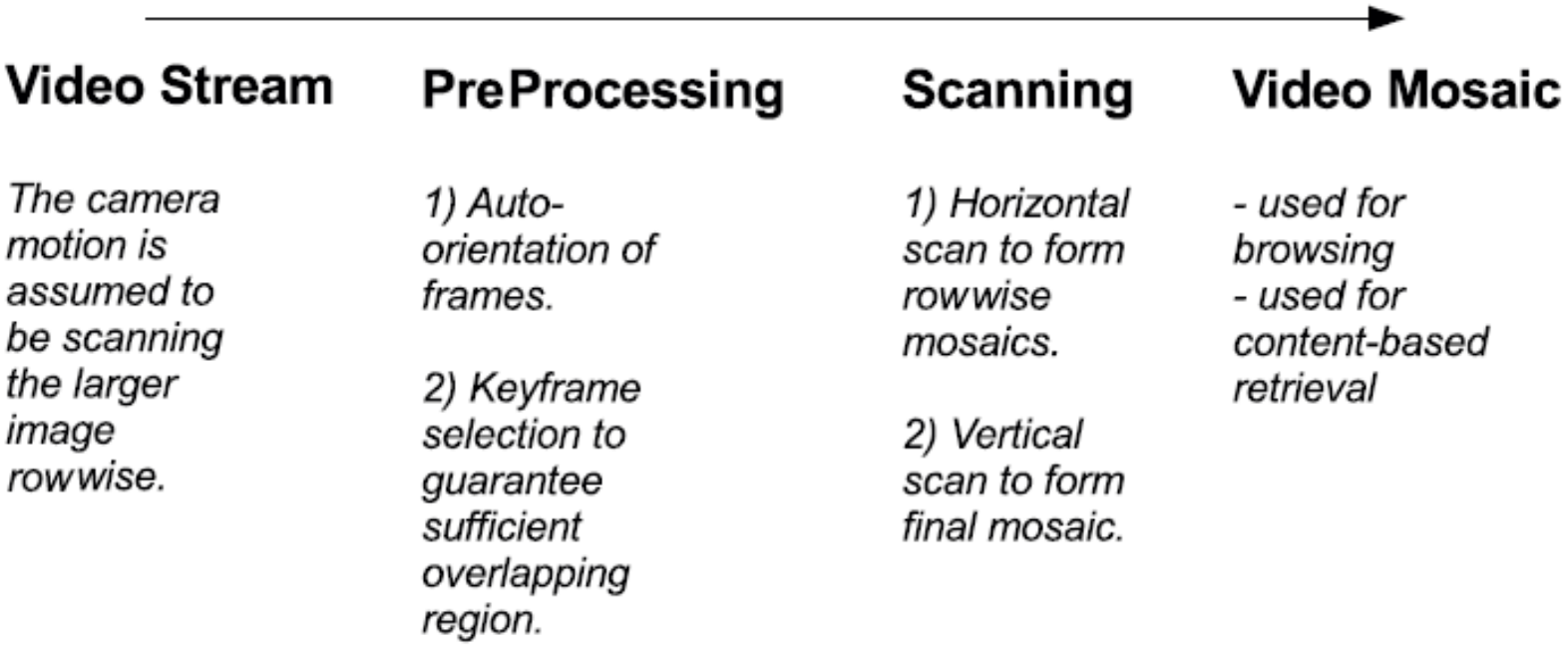
Where to use the Mosaick?

- The presence of mobile devices that are equipped with a camera has clearly facilitated image acquisition,
  - especially capturing document images of various types including **thick books** or **historical monuments**.
- Since these devices are also capable of capturing video, video mosaicking techniques can be utilized
  - to create high resolution historical document images for later use in **browsing** and **retrieval** systems.

To create high resolution images

- The video captures the frames while tracing the historical Ottoman documents in row major ordering.
- In the pre-processing steps,
  - the frames are oriented automatically by the help of tracking the optical flow, and
  - the keyframes are selected such that sufficient overlapping region in between is guaranteed.
- Then, the scanning phase traces these keyframes in horizontal and vertical manner to create the final video mosaic image in higher resolution.
- Both horizontal and vertical scan operations utilize distance vector to compute similarity between two adjacent images.

Video Mosaicking Process



Assumption:
a hand-held camera with constant zoom during camera motion to guarantee that the sizes/resolution of the captured frames
are equal.

The processing has two phases: the first phase includes auto-orienting frames and generating keyframes. The second phase includes horizontal and vertical scanning to produce video mosaic.

Contributions - 1

- The video mosaicing approach produces a **higher resolution image for the historical Ottoman document** when compared to a single image capturing the document as a whole.
- Instead of a single video, capturing an adequate number of single images that have sufficient overlapping regions can be used as an alternative;
  - however, it seems quite hard and erroneous for a user to capture a set of these images.
- Using a single video facilitates the tasks of the user during input acquisition.

Contributions - 2

- The preprocessing phase auto-orients the captured frames and produces keyframes with sufficient overlapping regions in between.
- This utilizes KLT-based optical flow analysis and provides keyframe-based image mosaicing.
- Hence, the processing deals only with a smaller set of frames (i.e. keyframes), which leads to an efficient video mosaicing approach in terms of time and space.

Contributions - 3

- During the scanning phase, the row and column similarities are computed based on the **distance vector**.

  «E. Saykol, U. Gudukbay, O. Ulusoy, A **Histogram-Based Approach** for Object-Based Query-by-Shape-and-Color in Image and Video Databases, **Image and Vision Computing**, Vol. 23, No. 13, pp. 1170-1180, November 2005.»

  - expressive in encoding the shape information based on the distribution of the pixels, and was invariant under scale, rotation, and translation.

  - Here, the distance vector is utilized to detect overlapping rows and **columns** efficiently between two keyframes while scanning.

```
Input:   V , the video stream;
Output:  M , the video mosaic;
Locals:  M_H(t) , the horizontal mosaic image at frame t
         M_V(r) , the vertical mosaic image at row r
         F , the set of oriented frames
         I , the set of keyframes
         r , the row count of the mosaic image

1.   F = autoOrientFrames(V)
2.   I = computeKeyframes(F)
3.   r = 0
4.   M_H(0) = I(0)
5.   for each keyframe t in I from 1 to end
6.       I(t) = pickNextKeyframe()
7.       M_H(t) = computeHorizontalMosaic(M_H(t-1), I(t))
8.       if M_H(t) = M_H(t-1)
         /* if empty horizontal merge */
9.           M_V(r) = M_H(t)
10.          r = r + 1
11.          M_H(t) = I(t)
12.      endif
13.  endfor
14.  for each row k from 1 to r
     /* skip first row image and up to last one */
15.      M_V(k) = computeVerticalMosaic(M_V(k), M_V(k-1))
16.  endfor
     /* copy last vertical merge to final mosaic image */
17.  M = M_V(r)
```

The first two steps correspond to the *preprocessing* phase.

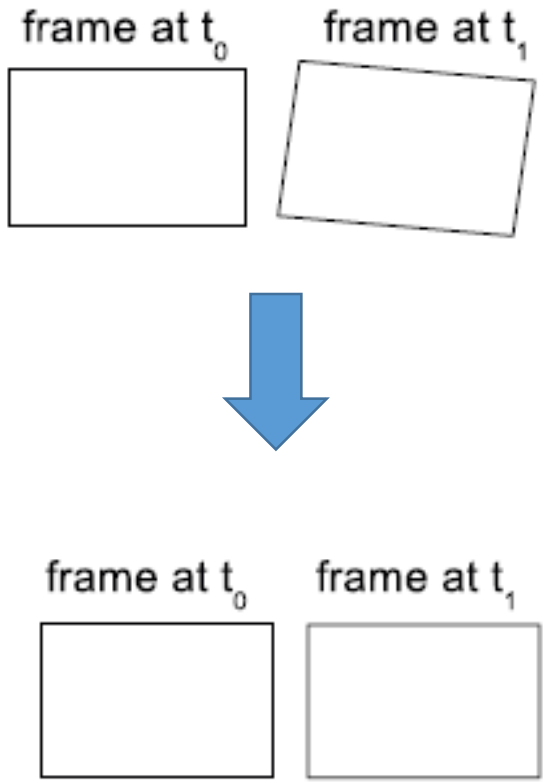
The *scanning* phase includes the operations listed between lines 3 and 17.
$M_H(t-1)$, $I(t)$, and $M_H(t)$ have the same height during the horizontal scan as a direct consequence of *computeHorizontalMosaic*(I_L; I_R).

During the horizontal scan, if an **empty merge** is encountered, this means that $I(t)$ is in fact the *first frame* of the next row; hence the row mosaic is formed in $M_H(t)$.

The mosaic of each row is then merged to form the final video mosaic.

Auto-orientation

- The optical flow can be seen as the velocity of the scene at each pixel.
- We employed the well-known KLT feature tracking algorithm.
- Major drawback of using optical flow is that the projected motion of an object depends on its distance to the camera.
- Since the zoom level and the distance of the camera can be safely assumed to be the same throughout the video, no problem.



Key Frame Generation

- Using the optical flow calculations in the frame at time *t*0 , the angle between the flow vector and the unit vector on the x-axis, is calculated based on the features/points in KLT.
  - A majority voting scheme is used to estimate the flow vector of the frame.
- If 5% threshold is exceeded for the frame at time *t*1, a new keyframe is created not to lose data at the boundaries.
- Then the frame at time *t*1 is rotated back to horizontal orientation (*as in previous slide*).
- The camera motion from the end of a row to the beginning of a new row can also be detected here, generating no keyframes at this motion.

Guaranteed Overlapping Region

- The keyframe selection scheme guarantees 20% overlap during the horizontal scan, by considering the width of the frames captured when the camera is moving.
- This necessary overlap condition is checked at each frame by tracking the velocity computed w.r.t. estimated flow.
- When the boundary value (i.e., 20%) is reached, a new keyframe is generated having at least 20% overlapping region.

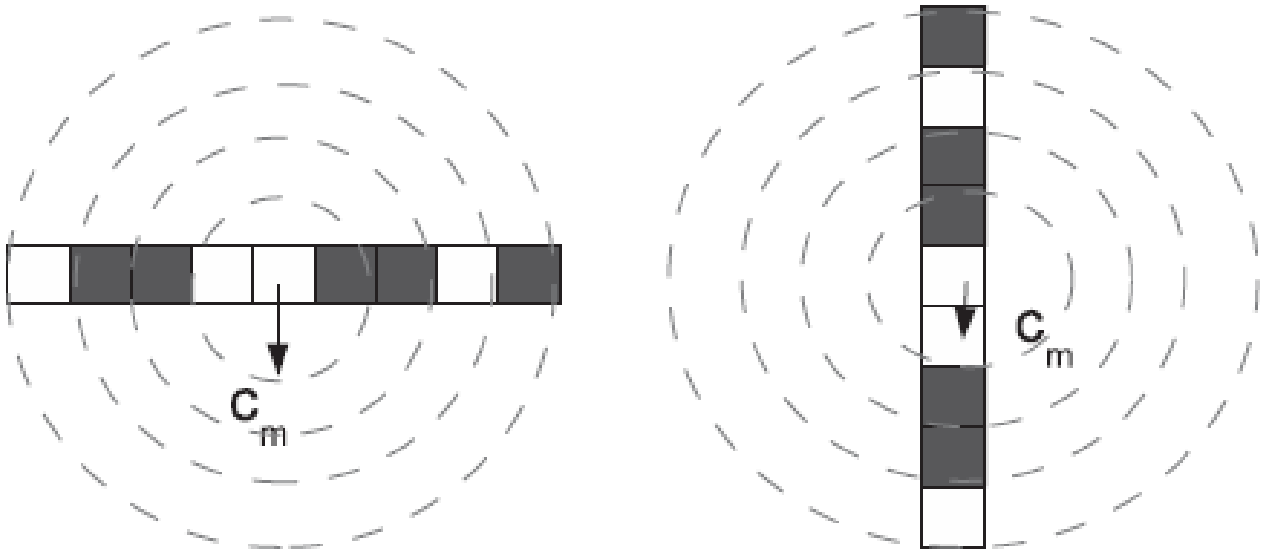
Scanning Phase: Horizontal

- *computeHorizontalMosaic*(*IL, IR*)
- The width of *IL* and *IR* might be different (since continuously merging) but their heights are equal since the same zoom level is assumed during camera motion and the frames are aligned automatically.
- The algorithm compares the similarity at each corresponding column on the left IL and on the right IR to conclude whether two columns are overlapping or not (using distance vector).

Scanning Phase: Vertical

- *computeVerticalMosaic*(*IU, ID*)
- merges two consecutive row images that are computed at the horizontal scan. Unlike horizontal merge, the dimensions of *IU* and *ID* can be different here.
- Since the distance vector that we use for similarity computation is scale-invariant, these different sizes would not be a problem as long as the vector is normalized with respect to the maximum distance value.

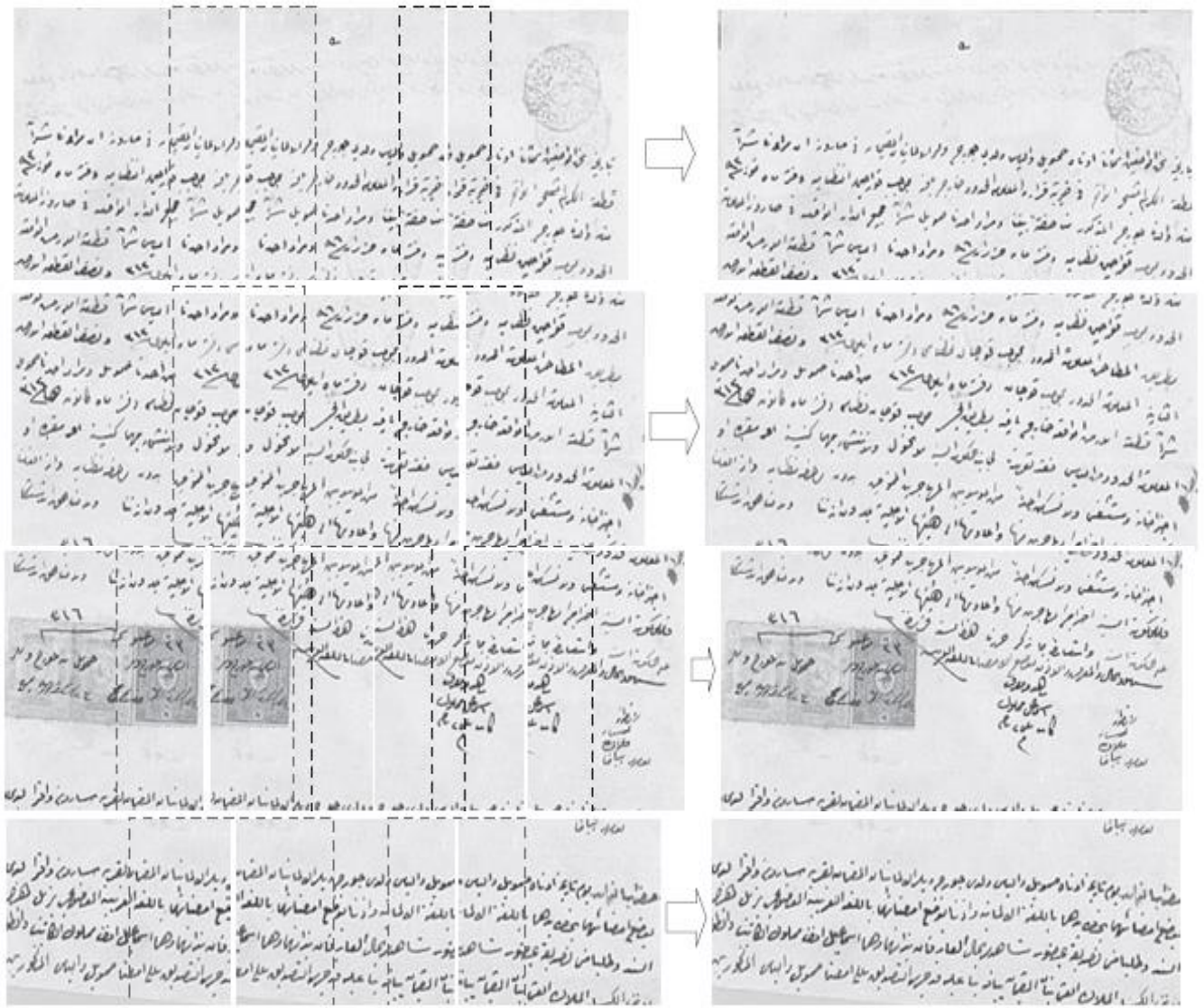
Distance Vector for Similarity



The distance vector is a 1-dimensional histogram whose entries are the number of pixels existing in between the concentric circles that are centered at the center of mass

Based on the existence of the black pixels in the concentric circles, the distance vector *D* is *D = [0; 1; 2; 1; 1]*.

After **normalizing with respect to the maximum distance**, which is 2 here, to guarantee scale-invariance, it can be written as *D = [0; 0.5; 1; 0.5; 0.5]*.



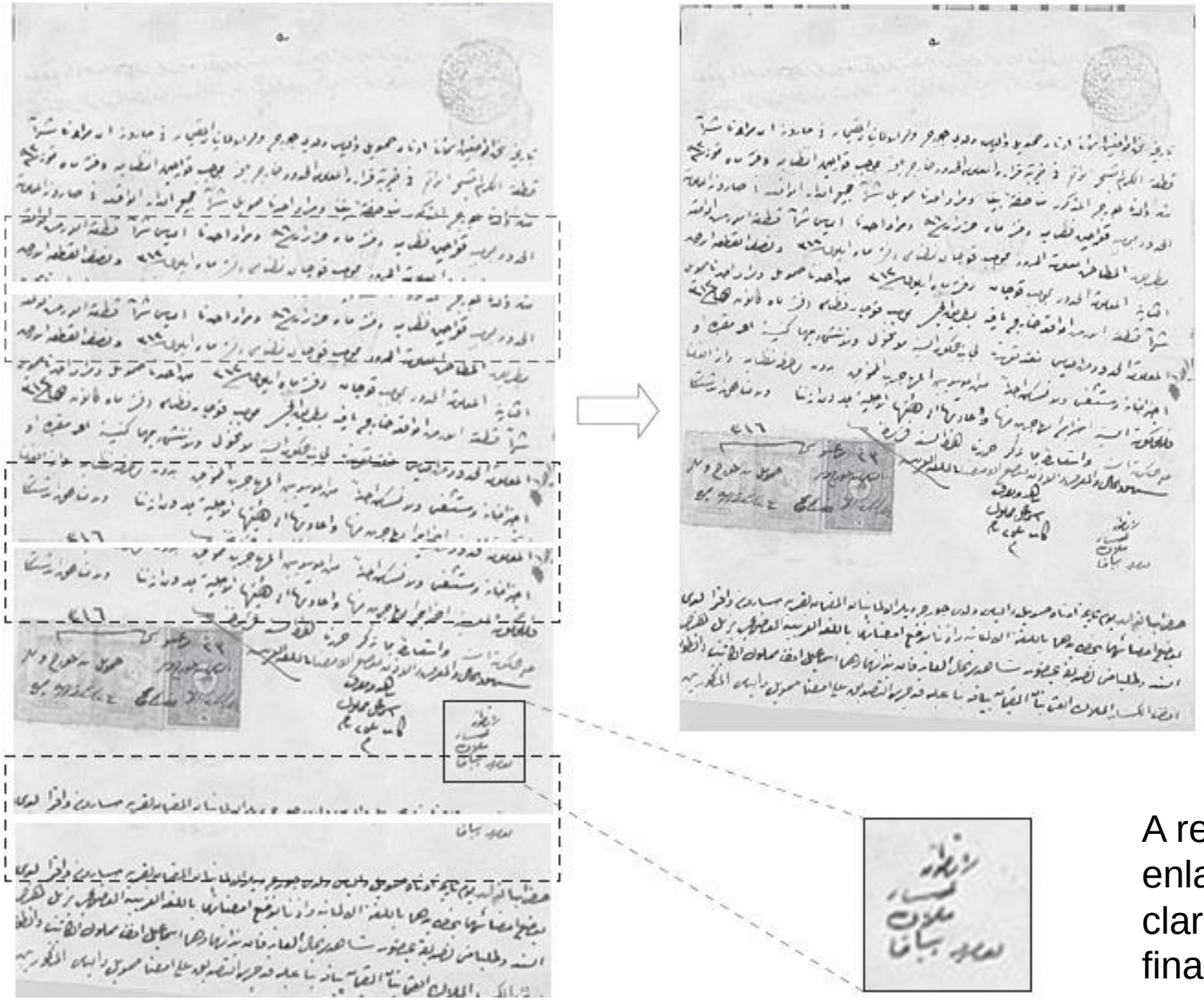
A sample historical document, namely a kushan, to illustrate the intermediate steps.

(a) Three frames of the 1st row and two overlapping regions with dashed lines,
(b) rowwise mosaic image of the 1st row.

(c) Three frames of the 2nd row and two overlapping regions with dashed lines,
(d) rowwise mosaic image of the 2nd row.

(e) Four frames of the 3rd row and three overlapping regions with dashed lines,
(f) rowwise mosaic image of the 3rd row.

(g) Three frames of the 4th row and two overlapping regions with dashed lines,
(h) rowwise mosaic image of the 4th row.

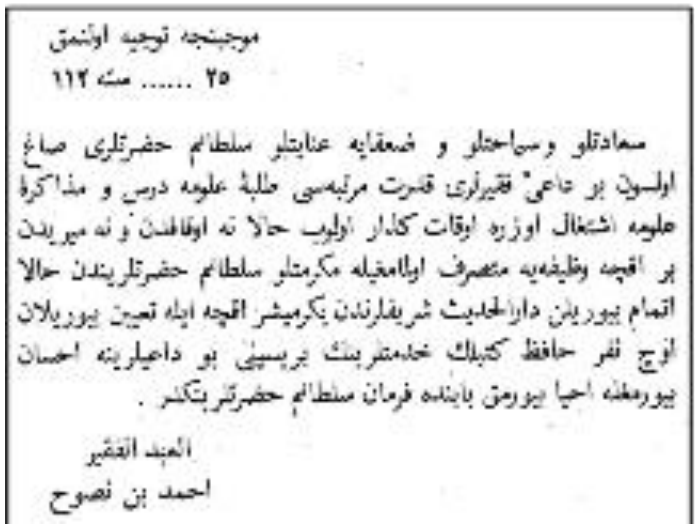


A rectangular region is enlarged 2 times and shown to clarify the visualization of the final mosaic image.

Performance Experiments

- The video capture operation traces the document in row major order,
  - the zoom level is constant,
  - the frames-per-second value is 24.
  - Each captured frame has 1280 x 960 resolution.
- The average number of frames is calculated with respect to these video data for each document.
- The average number of keyframes is based on the preprocessing phase of the video mosaicing approach, and similarly the video mosaic resolution is given in pixels on average.

Ottoman image dataset details	
Video duration (s)	3.21
# of Frames	77.6
# of Keyframes	9.2
Video mosaic resolution (pixels)	4444.9 x 5753.4

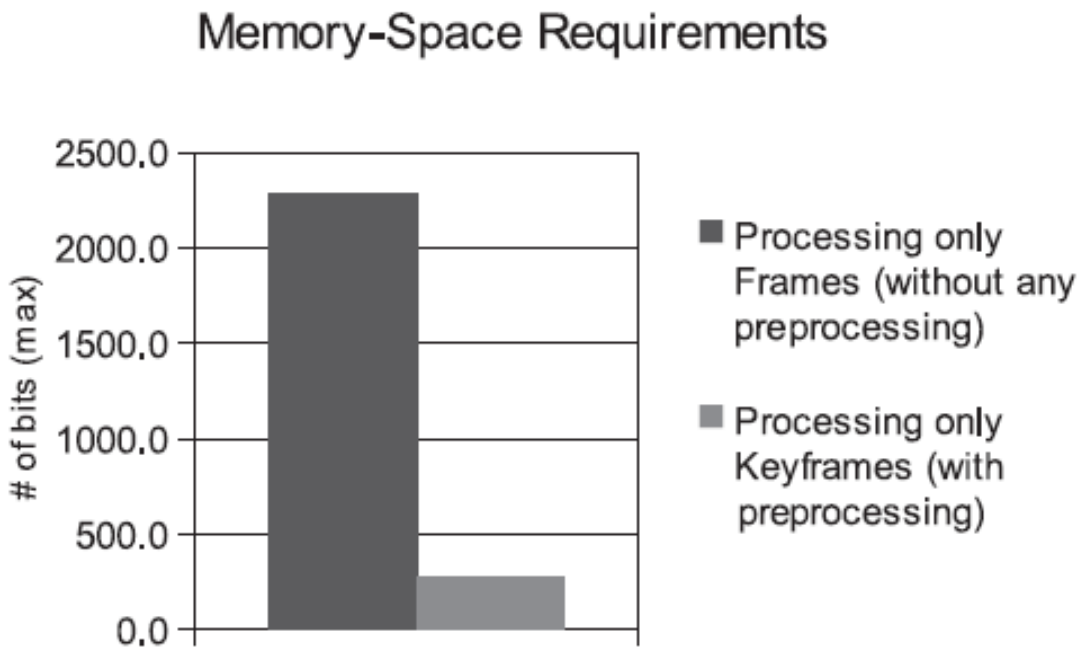


موجبنجه توجيه اولنمق
٢٥ ..... سنه ١١٢

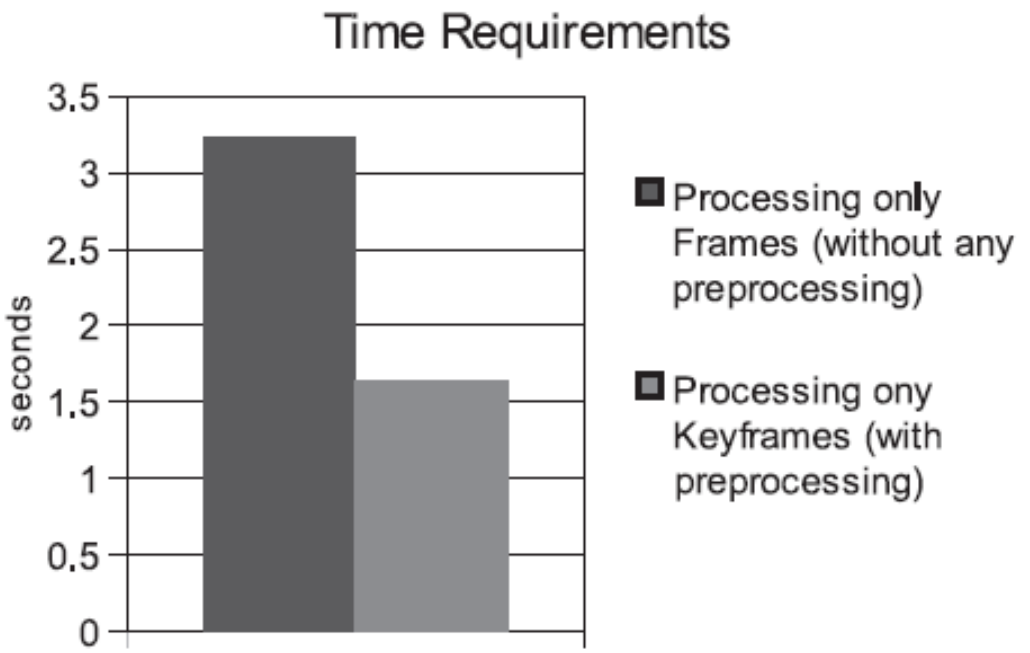
سعادتلو وسياحتلو و ضعفايه عنايتلو سلطانم حضرتلرى صاغ اولسون بر داعى فقيرلرى قدرت مرتبه‌سى طلبهٔ علومه درس و مذاكرهٔ علومه اشتغال اوزره اوقات كذار اولوب حالا ته اوقافدن و نه ميريدن بر اقچه وظيفه‌يه متصرف اولامغيله مكرمتلو سلطانم حضرتلريندن حالا اتمام بيورلين دارالحديث شريفلرندن يكرميشر اقچه ايله تعيين بيورىلان اوج نفر حافظ كتبك خدمتلرينك تربيسيلى بو داعيلرينه احسان بيورمغله احيا بيورمق بايننده فرمان سلطانم حضرتلرينكدر .

العبد الفقير
احمد بن نصوح

Performance Gains

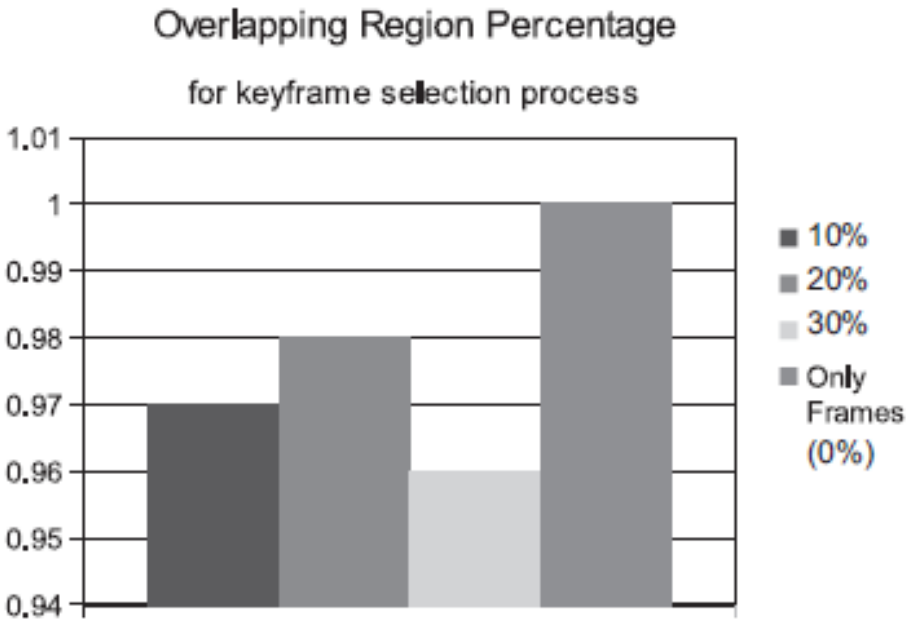


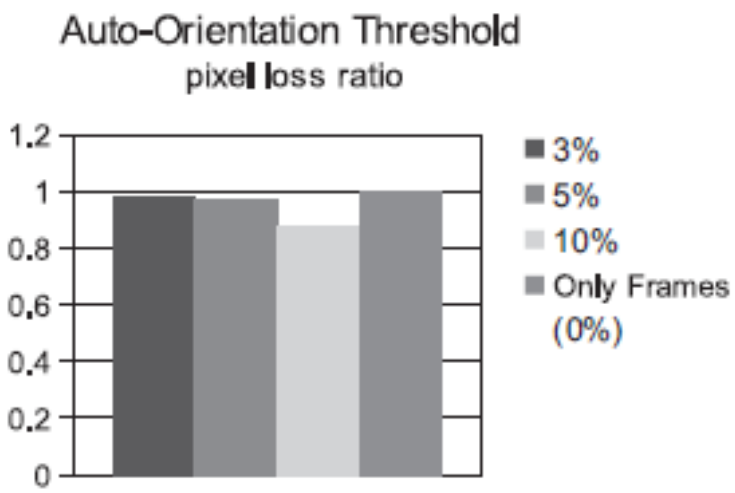
The use of keyframes for video mosaic generation yields 12% memory space requirements as opposed to processing frames,

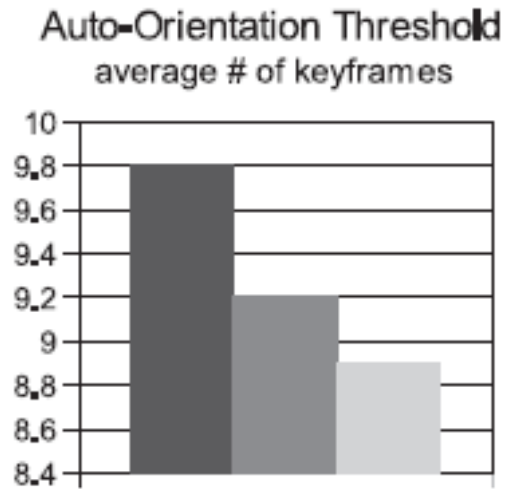


Total processing time is also reduced 49% in keyframe processing.

Evaluations to set appropriate thresholds







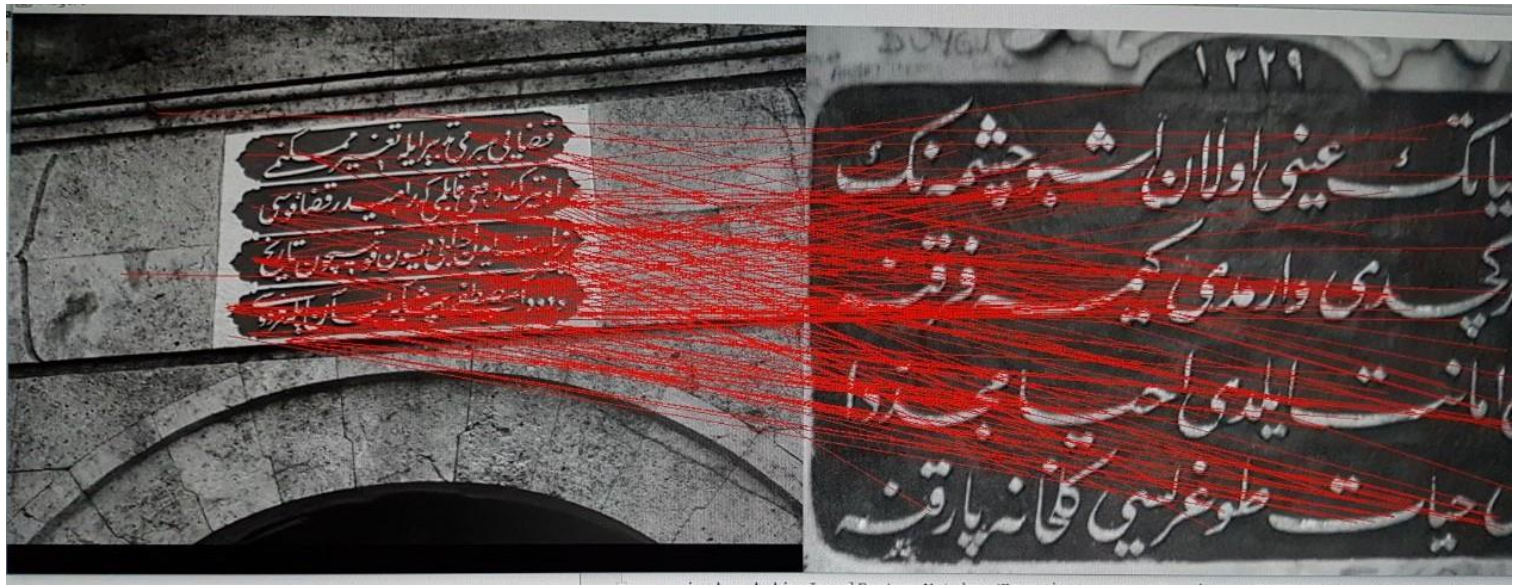
Based on these evaluations, the most appropriate value is 5%. Here, the percentage of the overlapping region is set to 20%, and the distance vector row/column similarity threshold T_D is set to 90%.

To sum up

- A video mosaicing technique for historical Ottoman documents is presented to produce a high resolution video mosaic image from a captured video for later use in browsing and/or content-based retrieval systems.
- The video mosaicing technique scans the captured frames first horizontally to create **row mosaics** and then vertically to create the **final mosaic**.
- This scanning phase relies on the auto-oriented keyframes that are generated in the preprocessing phase based on mainly tracking the optical ow.
- The assumptions include constant zoom during camera motion.

Future

Monument Identification



To be used as Landmark Detection ..

To be integrated with Content-based Retrieval Systems …

Thank you for your attention!