Object recognition of Glagolitic characters using Sift and Ransac

Images in database: Glagolitic alphabet characters.

Algorithm

1) Prepare a database of images.

   Images characteristics :
   1. Scale changed images
   2. Rotation images
   3. Blurred images

2) Apply SIFT method on the images in the database.

3) Matching the images.

4) Apply RANSAC - RANdom SAmple Consensus,
   in order to estimate the parameters of the Sift model.

Input images - sample:

‘glag.jpg’:

'mislete.jpg'

SIFT method (David Lowe)

[image, descriptors, locs] = sift(imageFile)
This function reads an image and returns its SIFT keypoints.

Output:
image: the image array in double format

descriptors:
a K-by-128 matrix, where each row gives an invariant descriptor for one of the K keypoints. The descriptor is a vector of 128 values normalized to unit length.

locs:
K-by-4 matrix, in which each row has the 4 values for a keypoint location (row, column, scale, orientation). The orientation is in the range $[-\pi, \pi]$ radians.

Drawing Sift keypoints:

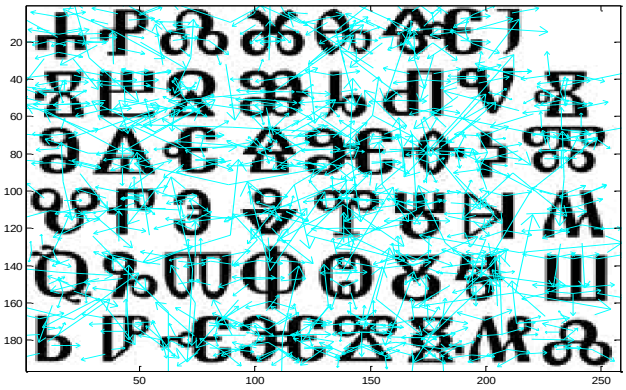
Showkeys (image, locs)

This function displays an image with SIFT keypoints overlayed.

Input parameters:

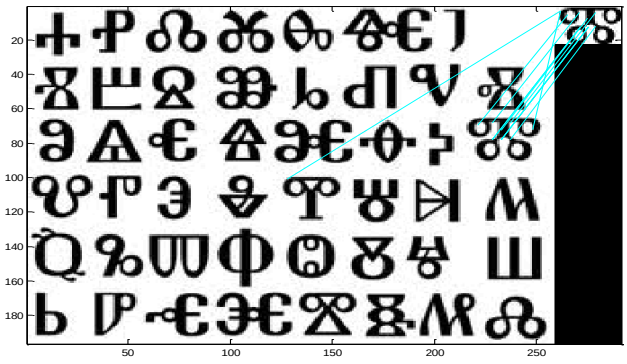
image: the file name for the image (grayscale)

locs: matrix in which each row gives a keypoint location (row,column, scale, orientation)



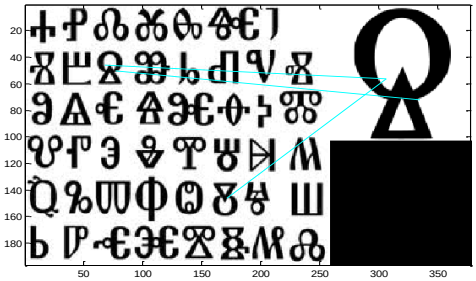
Match function

[matchLoc1 matchLoc2] = match(img1, img2);
This function reads two images, finds their SIFT features,
and displays lines connecting the matched keypoints.

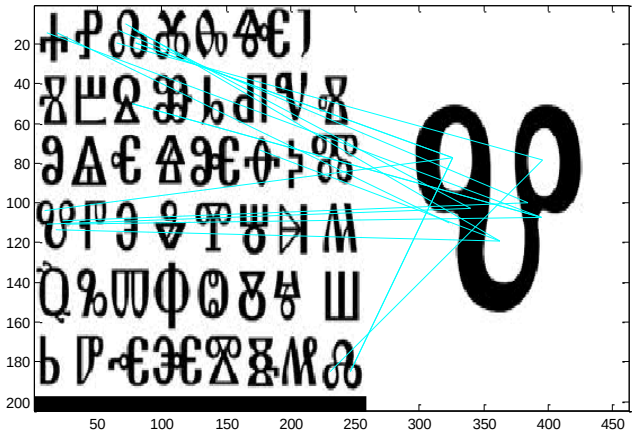


A match is accepted only if its distance is less than distRatio times the distance to the second closest match. (distRatio = 0.6)

Examples of rotation invariance:
Rotation of 180 degrees:



Another example:



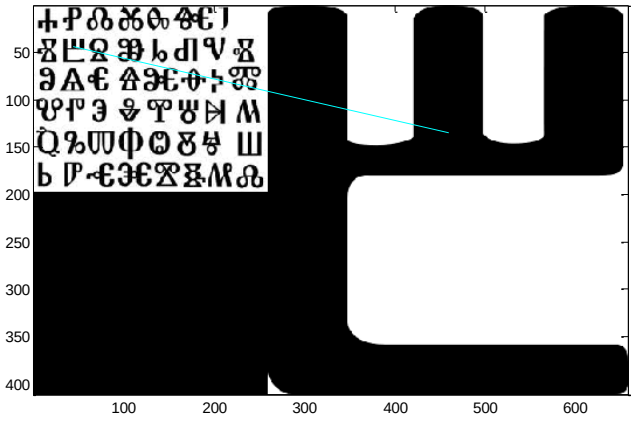
This example shows that the invariance to rotation leads to problems when recognizing characters.

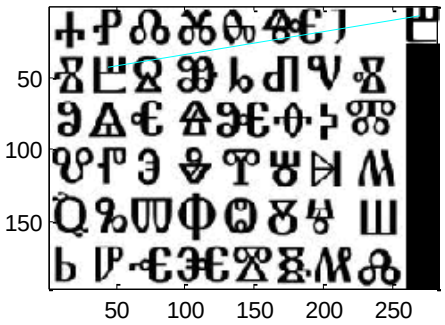
In the above example, glagolitic vede has the same topology as glagolitic dobro , rotated by 180 degrees.

Sift method cannot differentiate between them.

Scale invariance example:

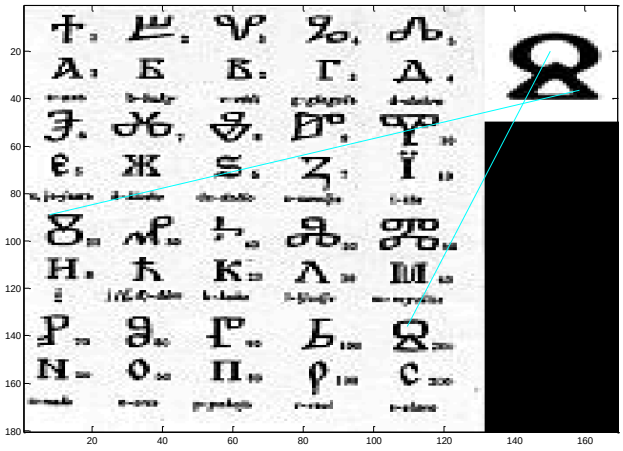
Tests on various sizes of glagolitic characters.





Blurred images example:

match('cyrilic.jpg','slovo.jpg')



RANSAC - RANdom SAmple Consensus

Is an iterative method to estimate parameters of a mathematical model from a set of observed data which contains outliers. Ransac rejects inconsistent matches.
Outliers - an observation that is numerically distant from the rest of the data.

Ransac input and output

INPUT:

X = input data. The data id provided as a matrix that has dimesnsions 2dxN where d is the data dimensionality and N is the number of elements

options = structure containing the following fields:

sigma = noise std

P_inlier = Chi squared probability threshold for inliers (i.e. the probability that an point whose squared error is less than T_noise_squared is an inlier) (default = 0.99)

T_noise_squared = Error threshold (overrides sigma)

epsilon = False Alarm Rate (i.e. the probability we never pick a good minimal sample set) (default = 1e-3)

Ps = sampling probability (1 x size(X, 2)) (default: uniform, i.e. Ps is empty)

ind_tabu = logical array indicating the elements that should not be considered to construct the MSS (default is empty)

validateMSS_fun = function that validates a MSS Should be in the form of:

flag = validateMSS_foo(X, s)

validateTheta_fun = function that validates a parameter vector Should be in the form of:

flag = validateTheta_foo(X, Theta, s)

est_fun = function that estimates Theta. Should be in the form of:

[Theta k] = estimate_foo(X, s)

man_fun = function that returns the residual error. Should be in the form of:

[E T_noise_squared] = man_fun(Theta, X)

mode = algorithm flavour

'RANSAC' -> Fischler & Bolles

'MSAC' -> Torr & Zisserman

max_iters = maximum number of iterations (default = inf)

min_iters = minimum number of iterations (default = 0)

max_no_updates = maximum number of iterations with no updates (default = inf)

fix_seed = true to fix the seed of the random number generator so that the results on the same data set are repeatable (default = false)

reestimate = true to resestimate the parameter vector using all the detected inliers (default = false)

verbose = true for verbose output (default = true)

notify_iters = if verbose output is on then print some information every notify_iters iterations.

If empty information is displayed only for
updates (default = [])

OUTPUT:

results = structure containing the following fields:

Theta = estimated parameter vector
E = fitting error obtained from man_fun
CS = consensus set (true -> inliers, false -> outliers)
r = rank of the solution
iter = number of iterations
time = time to perform the computation

options = options (including the defaults set by the algorithm)

Homography

To compute the Theta, we compute the homography between the point pairs X1, X2 by calling:
[H A] = HomographyDLT(X1, X2, mode)

Theta = H(:)

A homography is an invertible transformation from the real projective plane to the projective plane that maps straight lines to straight lines.

Marco Zuliani (Ransac toolbox) uses The Normalized Direct Linear Transform (nDLT) Algorithm.

Homography

An invertible transformation from the real projective plane to the projective plane that maps straight lines to straight lines.

Key stages in runSift driver program:

```
[matchLoc1 matchLoc2] = match(img1, img2);
Match function calls sift function for both images.
```

Set RANSAC options.
Form the input data pairs:

```
X1 = matchLoc2';  - The transpose of matchLoc1
X2 = matchLoc1';  - The transpose of matchLoc2

X = [X1; X2];
```

Run Ransac :

```
[results, options] = RANSAC(X, options);
```

Ransac estimates the vector of parameters Theta.

Ransac results:

Starting RANSAC
Minimal sample set dimension = 4
Squared noise threshold = 73.563766, (assuming Gaussian noise, for sigma = 1.000000)
Iteration = 1/ 1361. Inliers = 5/ 12 (rank is r = -82.21007017)
Iteration = 2/ 449. Inliers = 6/ 12 (rank is r = -78.86619437)
Iteration = 3/ 26. Inliers = 10/ 12 (rank is r = -32.87832753)
Iteration = 4/ 26. Inliers = 10/ 12 (rank is r = -32.42028464)
Iteration = 114/ 26. Inliers = 10/ 12 (rank is r = -32.28285257)
Estimating the parameter vector... Done
Final number of inliers = 10/12
Converged in 1001 iterations (1.727473 seconds)

Usage Example:

Use Ransac for choosing the best distance ratio in matching process.

distRatio parameter: Only keep matches in which the ratio of vector angles from the nearest to second nearest neighbor is less than distRatio.

Table with results for various distRatio:

Distance ratio	0.55	0.6	0.66	
Number of inliers (correct matches)	9/10	10/12	10/15	

Discussion of results

The results show that Sift method is invariant to scale changes, rotation changes and blur. In the case of glagolitic characters, the sift produces relatively few matches. The sift method still produces too many outliers (false matches).