Project Report

ECE-S 523:
Detection and Estimation Theory

Prof. Fernand Cohen

Zexi Liu

June 4th, 2010

1. Objective

In this project we need to apply the knowledge of detection and classification learned in the course ECES-532 to detect the object in two images. We need to classify these images as the background (class 1) and the object (class 2). Also we need to find the pdf's and the parameters of the two classes.

2. Description

In this project, two images are given. The images are of the size (128×128). It is known that there are at most 2 classes (object and background) in each image. The object and background are of different distribution. There can be multiple objects in each image, but the objects are blocks with certain size instead of scattered points.

3. Algorithm (K-S test)

First, the whole image was divided into small blocks. One of them was chosen as the reference block. Then the rest of them are tested homogeneous or non-homogeneous against the reference block. This can be done by using the K-S test, which can differentiate whether two collections of data are from one class or not. In the end, the whole image can be classified into two classes.

Second, merge the homogenous blocks into one class. After merging, the histogram will be used to estimate the distribution and MLE will be used to estimate the corresponding parameters. Another K-S test is recommended in order to verify the whether the data is of assumed distribution.

As the distributions of two classes are known, this information can be taken into consideration to update our decision. The distribution of class 1 is used to test the whole image and classify the different blocks into two classes. Then this new classification is compared with that in the K-S test stage. If there is any change in the decision, the new homogenous blocks are merged and the parameter estimation is updated. These procedures are repeated until there is no change in the classification.

For example if the i-th block is of class 1 in the K-S test stage, however in the verification stage it belongs to class 2, then the decision is made that it should belong to class 2. Because in the verification stage, the knowledge of the pdf is already obtained, we can make better classification based on this knowledge.

This classification is sensitive to the block size. In this project, 32x32 blocks were chosen since they gave reasonable results. If the block size is too small, there is not enough information in each block to make good decision; if the block size is too large, it is highly possible that one block will contain both the background and the object.

It is also important to choose the reference block. The upper-left corner block was chosen as the reference block under the assumption that the probability of the object appearing in the corner is small.

4. Results

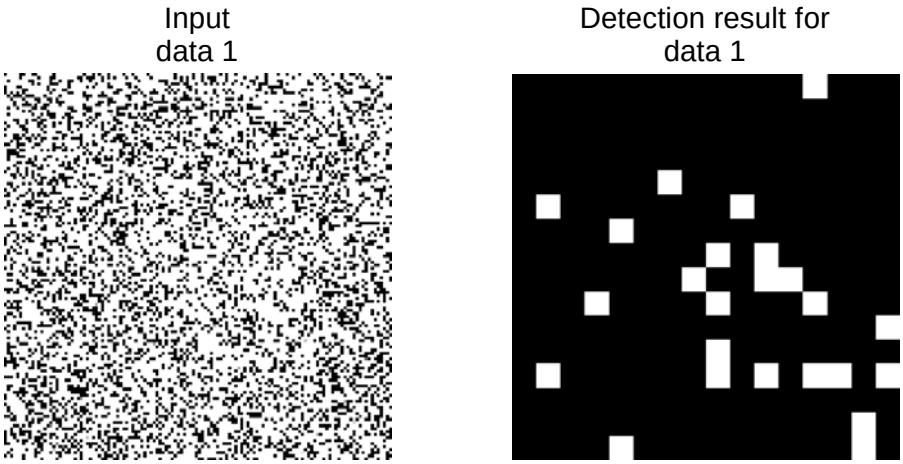

Fig. 1. Detection result for data 1 (8x8 block)

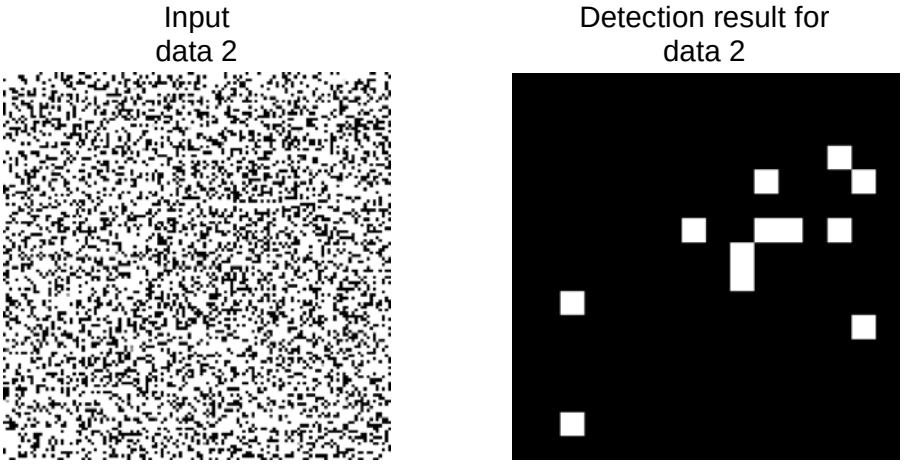

Fig. 2. Detection result for data 2 (8x8 block)

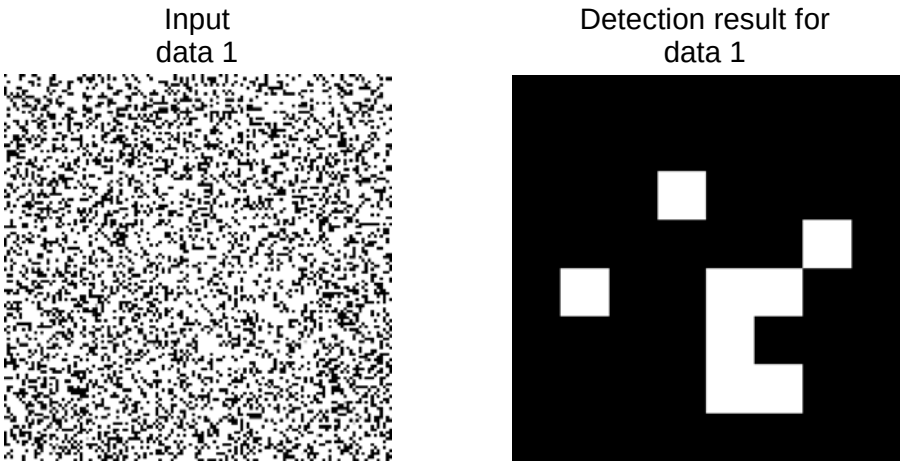

Fig. 3. Detection result for data 1 (16x16 block)

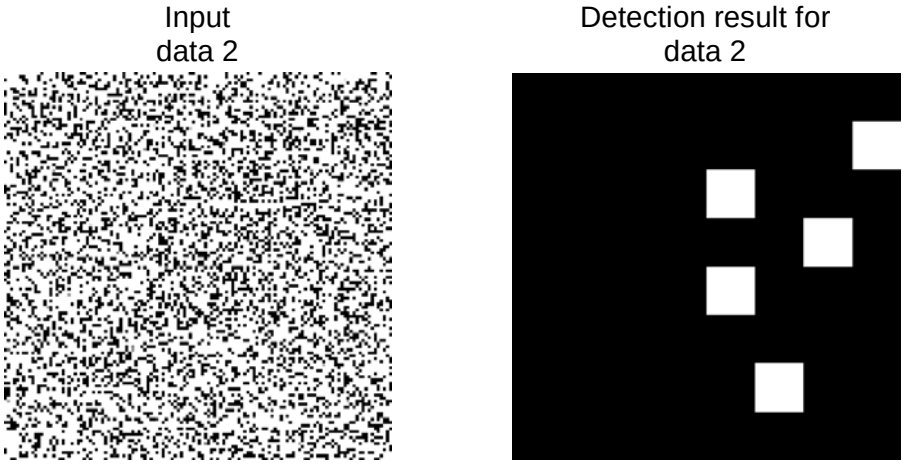

Fig. 4. Detection result for data 2 (16x16 block)

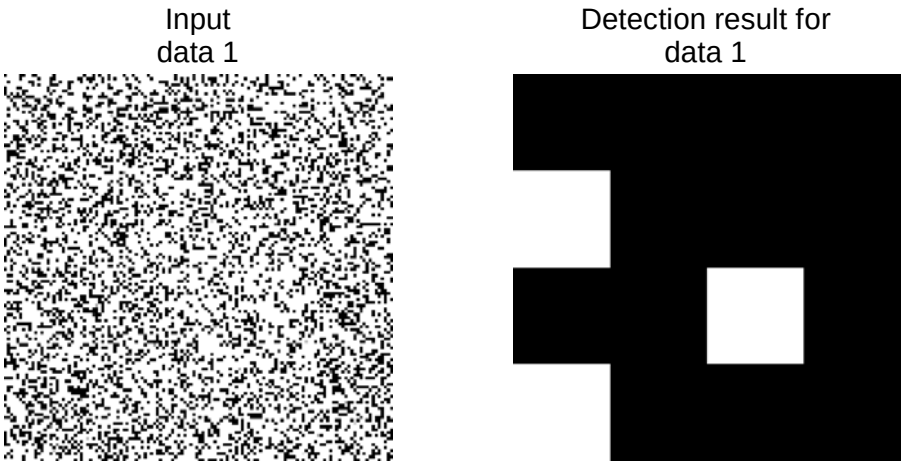

Fig. 5. Detection result for data 2 (32x32 block)

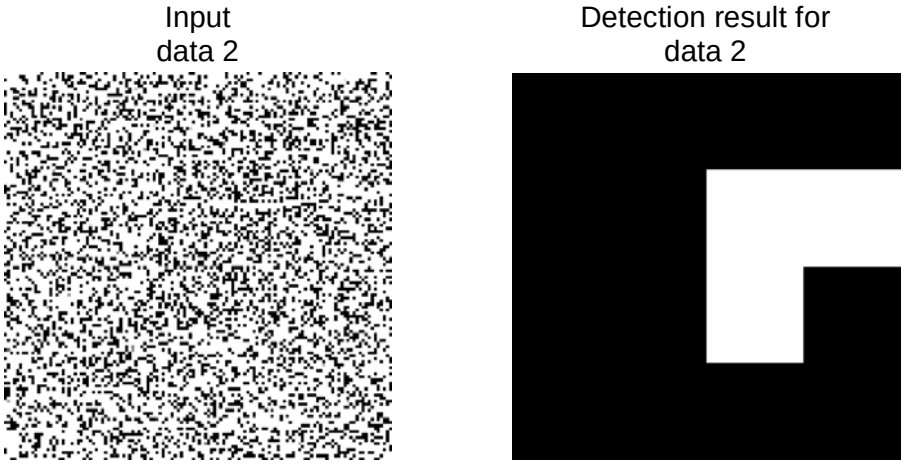

Fig. 6. Detection result for data 2 (32x32 block)

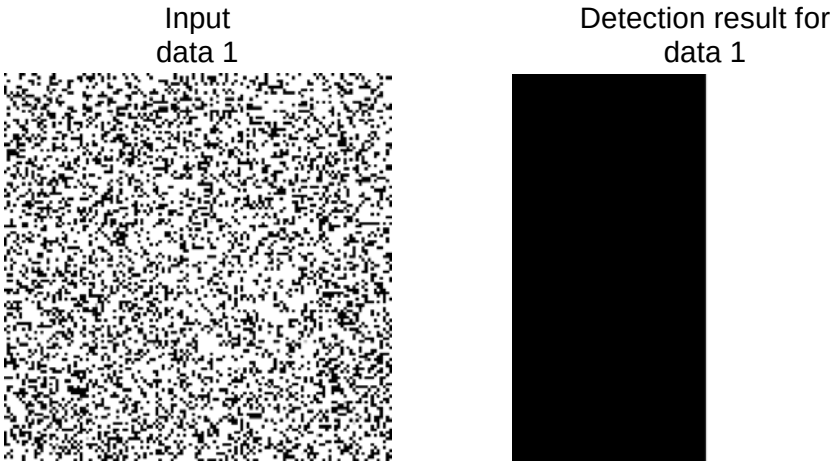

Fig. 7. Detection result for data 2 (64x64 block)

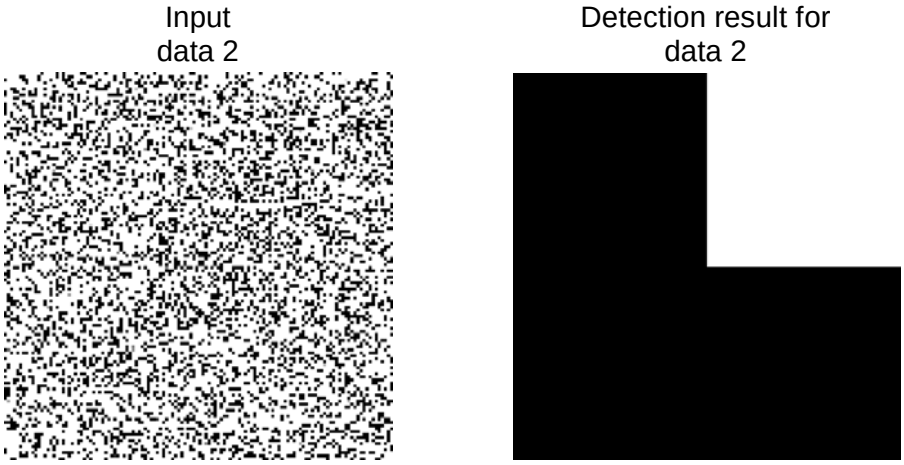

Fig. 8. Detection result for data 2 (64x64 block)

Table 1 block size vs. mean & variance

	μ for background	μ for object	σ for background	σ for object
8x8 block	48.6655	52.6089	84.7053	98.3218
16x16 block	48.8634	48.6982	84.5412	94.9018
32x32 block	48.9948	48.225	82.5856	96.6317
64x64 block	49.0456	48.265	83.2792	91.449

5. Codes

```
% ---------------------- Descriptions ----------------------------
% Code for ECES523 Final Project
% file name: ECES523_FinalProject.m
% % data created: 05/01/2010
% last revise: 06/07/2010
% ----------------------------------------------------------------


%% Initializations (load raw data)
clear;
clc;

BS = 32;  % block size
a1 = 0.1;
a2 = 0.05;

data1 = load('data1');
data2 = load('data2');

figure(1);
S = 32;
subplot(1,2,1);
imshow(data1);
title({'Input', 'data 1'}, 'FontSize', S);

figure(2);
subplot(1,2,1);
imshow(data2);
title({'Input', 'data 2'}, 'FontSize', S);


%% Process data 1
block_size = BS;       % 32x32

vector_1st_block = reshape( data1(1:block_size, 1:block_size),1, [] );

NoB = 128 / block_size;        % # of blocks
H = zeros( NoB, NoB );
background = [];

for i = 1:NoB
      for j = 1:NoB
            temp = data1( (i-1)*block_size+1 : i*block_size, ...
                                 (j-1)*block_size+1 : j*block_size );
            data_vec = reshape( temp, 1, [] );
            H(i,j) = kstest2( vector_1st_block, data_vec, a1, 'unequal' );
            if H(i,j) == 1
                  background = [ background data_vec ];
            end
      end
```

```
end

H %#ok<NOPTS>

m = mean( vector_1st_block );
sigma = std( vector_1st_block );

block_size = BS;
NoB = 128 / block_size;
H_old = ones( NoB, NoB );
H_new = zeros( NoB, NoB );
while sum( sum( H_old ~= H_new ) )
      H_old = H_new;
      background = [];
      object = [];
      for i = 1:NoB
            for j = 1:NoB
                  temp = data1( (i-1)*block_size+1 : i*block_size, ...
                                    (j-1)*block_size+1 : j*block_size );
                  data_vec = reshape(temp,1,[]);
                  H_new(i,j) = kstest( data_vec, ...
                                          [data_vec.' normcdf(data_vec,m,sigma).'], ...
                                          a2, 'unequal' );
                  if H_new(i,j) == 0
                        background = [ background data_vec ];          % if it is background, then merge
                  elseif H_new(i,j) == 1
                        object = [ object data_vec ];
                  end
            end
      end
      m = mean( background );               % for each time, update the parameter of background
      sigma = std( background );
end

H_new %#ok<NOPTS>
display('Parameters of the background:');
m %#ok<NOPTS>
sigma %#ok<NOPTS>

Detection_Output = zeros( 128, 128 );
for i = 1:NoB
      for j = 1:NoB
            Detection_Output( (i-1)*block_size+1:i*block_size, ...
                                    (j-1)*block_size+1:j*block_size ) ...
                                    =    H_new(i,j);
      end
end
figure(1);
subplot(1,2,2);
imshow(Detection_Output);
title({'Detection result for', 'data 1'}, 'FontSize', S);

display('Parameters of the object:');
mean_obj = mean(object) %#ok<NOPTS>
sigma_obj = std(object) %#ok<NOPTS>

kstest( object, [object.'    normcdf(object,mean_obj,sigma_obj).'], a2, 'unequal' )


%% Process data 2
block_size = BS;
vector_1st_block = reshape( data2(1:block_size,1:block_size), 1, [] );
NoB = 128/block_size;
```

```
H = [];
H_new = [];
background = [];
for i = 1:NoB
      for j = 1:NoB
            temp = data2( (i-1)*block_size+1:i*block_size, ...
                              (j-1)*block_size+1:j*block_size );
            data_vec = reshape(temp,1,[]);
            H(i,j) = kstest2( vector_1st_block, data_vec, a1, 'unequal' );
            if H(i,j) == 1
                  background = [ background data_vec ];
            end
      end
end

H %#ok<NOPTS>

m = mean(vector_1st_block);
sigma = std(vector_1st_block);
block_size = BS;
NoB = 128/block_size;
H_old = ones(NoB,NoB);
H_new = zeros(NoB,NoB);
while sum( sum( H_old ~= H_new ) )
      H_old = H_new;
      background = [];
      object = [];
      for i = 1:NoB
            for j = 1:NoB
                  temp = data2( (i-1)*block_size+1 : i*block_size, ...
                                    (j-1)*block_size+1 : j*block_size );
                  data_vec = reshape( temp, 1, [] );
                  H_new(i,j) = kstest( data_vec, ...
                                          [data_vec.'    normcdf(data_vec,m,sigma).'], ...
                                          a2, 'unequal');
                  if H_new(i,j) == 0
                        background = [background data_vec];          % if it is background, then merge
                  elseif H_new(i,j) == 1
                        object = [object data_vec];
                  end
            end
      end
      m = mean(background);              % for each time
      sigma = std(background);        % update the parameter of background
end

H_new %#ok<NOPTS>
display('the parameter of background')
m %#ok<NOPTS>
sigma %#ok<NOPTS>

Detection_Output = [];
for i = 1:NoB
      for j = 1:NoB
            Detection_Output( (i-1)*block_size+1 : i*block_size, ...
                                    (j-1)*block_size+1 : j*block_size ) ...
                                    = H_new(i,j);
      end
end
figure(2);
subplot(1,2,2);
imshow(Detection_Output);
title({'Detection result for', 'data 2'}, 'FontSize', S);
```

```
display('the parameter of object')
mean_obj = mean(object) %#ok<NOPTS>
sigma_obj = std(object) %#ok<NOPTS>

kstest( object, [object.'    normcdf(object,mean_obj,sigma_obj).'], a2, 'unequal' )
```