EE 890/490 Mini Project 2 Report

Sayan Banerjee

## EE 890/490 Mini-Project 2 Report

**Paper Name:** Biomorphic Networks: Approach to Invariant Feature Extraction and Segmentation for ATR

All the images used here are Gray-Scale binary images. The feature vector of an image emerges as invariant structure in the aggregate histogram of inter-spike intervals (1/10 m-seconds) from all neurons of the network.

A basic image of 12 X 12 pixel is fed into a single layer network that consist of spiking neurons arranged in the 12 X 12 array format. The dynamics of a neuron can be summarized as follows, when potential of a neuron deltaV(t,i,j) reaches time varying threshold of Vth(t,I,j), it instantly drops to its resting state. From there potential of that neuron stats to grow by a time dependent exponential function until it reaches to its time dependent threshold. Threshold of a neuron at any given time is dependent on its neighbors’(max 8) inputs to the Neuron under consideration convoluted with an exponential time variant decay function. So, the Threshold of a Neuron, modulates the spike timing and frequency of spike of the Neuron. Thus, the Neurons, which are highly connected in the network spikes frequently than the others. The strength of the modulation is controlled by a constant beta. When neuron associated with a pixel whose value is “0”, it can’t fire because modulation a neuron is receiving from its neighbors gets nullify forcing the neuron to be silent.

The architecture of the network considered same as mentioned in the paper. One neuron is connected all its nearest neurons (max 8) and all the connections are bi-directional.

The time constant “meu” in exponential potential buildup of neuron is 12 times shorter than time constant, tau, of the exponential decay function. This causes slow depression in threshold of a neuron at the same time the rate of buildup of potential of a Neuron is higher. These together with the modulation received from its neighbors make the “more” connected neurons associated with positive pixel value, spike faster than the rest.

The simulation results are shown below with associated input images. It is possible to see from the results that when we are rotating, scaling an image and compare the result of the simulation with the original picture, it is showing the same/invariant pattern. When we change the picture the result of the simulation produces different pattern. This is happening because even though we are rotating or scaling a picture the basic structure remains same. In the original and rotated/scaled images the neurons associated with mostly connected position has higher firing rate and the result is the aggregate histogram of inter-spike intervals (1/10 m-seconds) from all neurons of the network. When we are scaling the image, there is difference in scale because the total number of neurons associated with the positive pixel value is decreasing, hence producing cumulatively less number of spike in network. But the pattern remains same. The shift comes from the fact that the higher or/and uniform image intensity lowers the thresholds of the neurons. The invariant structures of the output patterns are closely related to characteristic local features of the input pattern.

For an instance the basic structure of a “T” remains same throughout the course of rotation and scaling and the pixel(s) residing at the “joint” of two lines has(have) more connected and hence has(have) higher firing rate than others, hence producing the invariant structure in the output in the form of histogram.

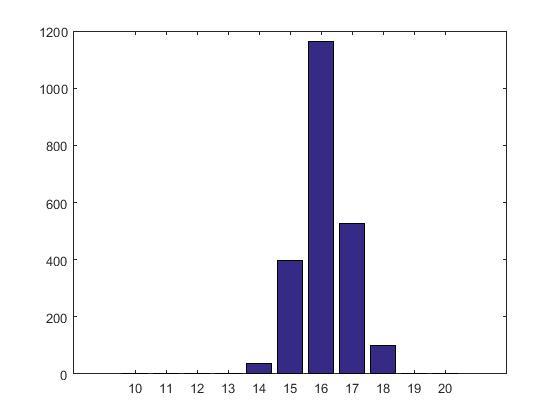
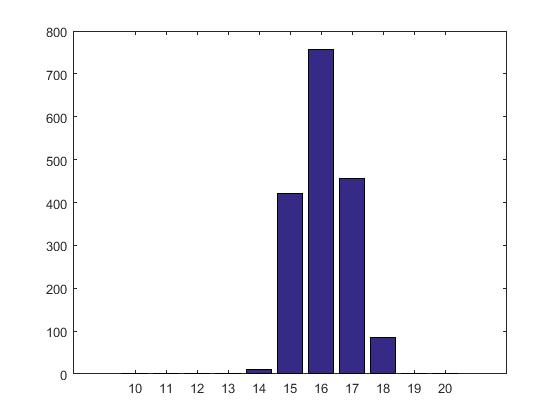
**Graphs:**

***X Axis*** *🡺 Aggregated Inter-Spike Intervals, Time, Unit = 1/10 msec.*

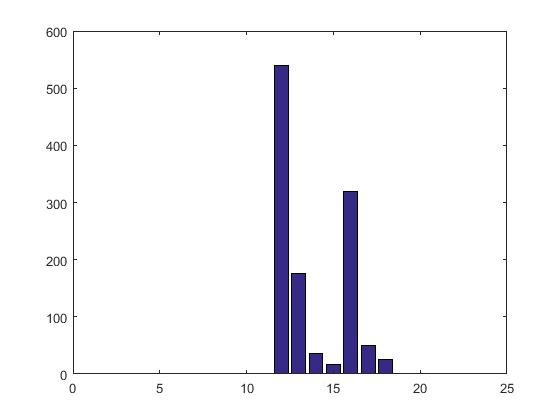
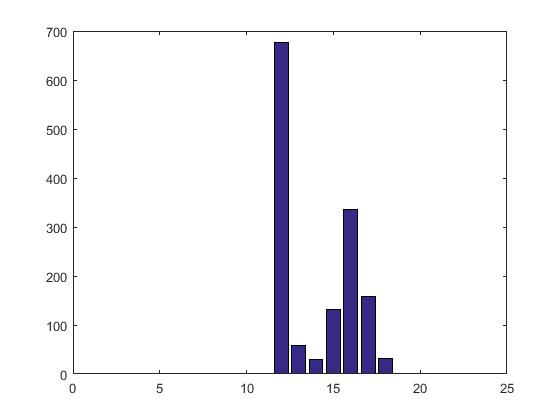
***Y Axis*** *🡺 Number of Spikes*

***Parameters:*** *h0* 🡺 0.08 , beta 🡺 0.25 , Vrest 🡺0 V

C:\Users\Sayan\AppData\Local\Microsoft\Windows\INetCacheContent.Word\tee.png C:\Users\Sayan\AppData\Local\Microsoft\Windows\INetCacheContent.Word\tee90.png

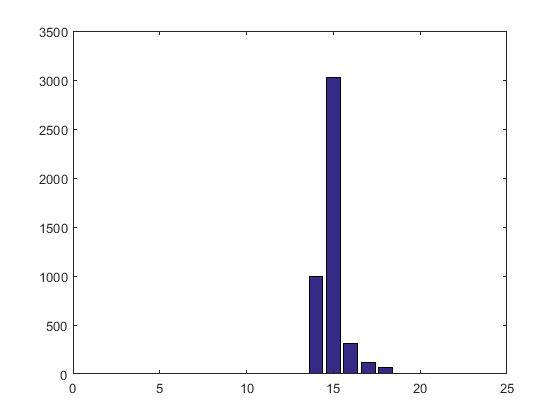
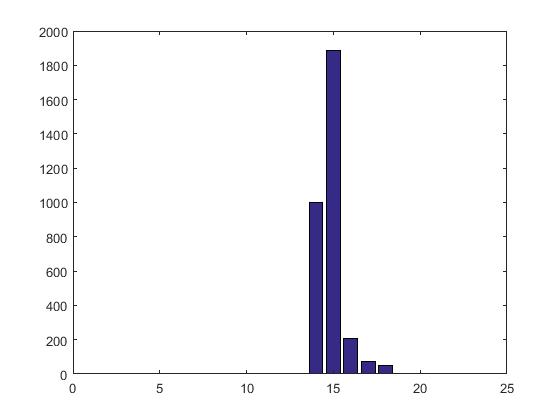
 

C:\Users\Sayan\AppData\Local\Microsoft\Windows\INetCacheContent.Word\cross.png C:\Users\Sayan\AppData\Local\Microsoft\Windows\INetCacheContent.Word\crossscale.png

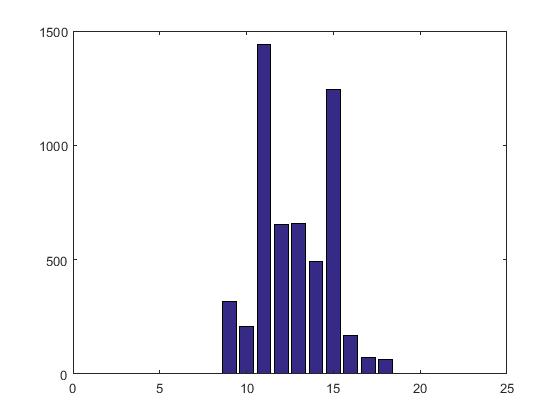
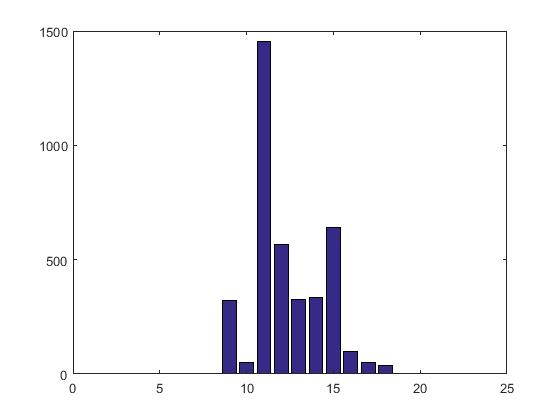


C:\Users\Sayan\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Rectangle_RtSc.png



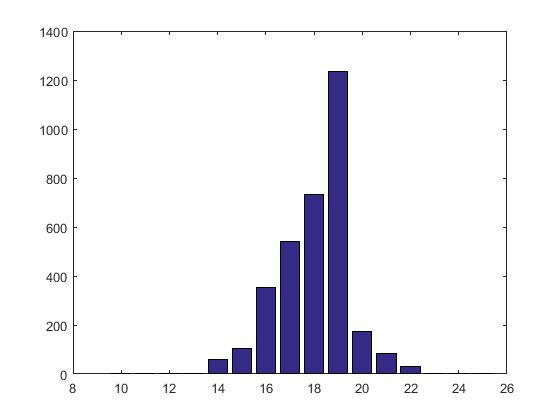
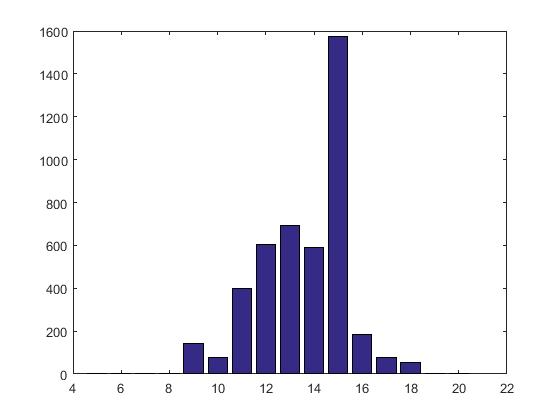
 

C:\Users\Sayan\AppData\Local\Microsoft\Windows\INetCacheContent.Word\TriangleTest.png C:\Users\Sayan\AppData\Local\Microsoft\Windows\INetCacheContent.Word\TriangleTest_Sc.png

Intensity 75% Intensity 100%

**MATLAB Code:**

%Project2.m

clear All;

clc;

I=double(imread('Cross.png'));

% for i=1:12

% for j=1:12

% if (I(i,j)==1.0)

% I(i,j)=0.75;

% end

%

% end

%

% end

V\_th=double(5.\*ones(12,12));

Record\_Keeping = double(zeros(144,2001));

tau=12.0;

meu=1.0;

h0=0.08;

beta=0.25;

V0=5.0;

E=4.8;

deltaVm=double(zeros(12,12));

S=double(zeros(12,12));

P=double(zeros(12,12));

Spike\_Count=zeros(144,1);

for t=1:2001

for i=1:12

for j=1:12

if (I(i,j)>0)

P(i,j)=neighbor\_contribution(Record\_Keeping, i,j, t, Spike\_Count,h0,tau);

S(i,j)=I(i,j)\*(1+beta\*(P(i,j)));

V\_th(i,j)=V0-S(i,j);

relative\_t=double(time\_passed\_since\_previous\_spike(Record\_Keeping, i, j, t));

deltaVm(i,j)=E\*(1-exp(-relative\_t/meu));

if (deltaVm(i,j)>=V\_th(i,j))

pos=Position(i,j);

Spike\_Count(pos,1)=Spike\_Count(pos,1)+1;

time=time\_passed\_since\_previous\_spike(Record\_Keeping, i, j, t);

Record\_Keeping(pos,t)=time;

end

end

end

end

end

Interval=zeros(1,201);

for b=1:144

for a=1:2001

if Record\_Keeping(b,a)~=0

spke=int32(Record\_Keeping(b,a)\*10);

Interval(1,spke)=Interval(1,spke)+1;

end

end

end

axis=[0.1:2.0:0.1];

figure

%histogram(Interval)

bar(axis, Interval(1:20));

%Position

function[m]=Position(i,j)

m=((i-1)\*12)+j;

end

% neighbor\_contribution

function [p]=neighbor\_contribution(Record\_Keeping, i,j, t, Spike\_Count,h0,tau)

p=0;

z=0;

%neighbor 1

pos=Position(i-1,j-1);

x=Spike\_Count(pos,1);

t\_1=t;

while x>0

if(Record\_Keeping(pos,t\_1)>0)

y=(t-t\_1)\*0.1;

z=z+h0\*exp(-y/tau);

x=x-1;

end

t\_1=t\_1-1;

end

%neighbor 2

pos=Position(i-1,j);

x=Spike\_Count(pos,1);

t\_1=t;

while x>0

if(Record\_Keeping(pos,t\_1)>0)

y=(t-t\_1)\*0.1;

z=z+h0\*exp(-y/tau);

x=x-1;

end

t\_1=t\_1-1;

end

%neighbor 3

pos=Position(i-1,j+1);

x=Spike\_Count(pos,1);

t\_1=t;

while x>0

if(Record\_Keeping(pos,t\_1)>0)

y=(t-t\_1)\*0.1;

z=z+h0\*exp(-y/tau);

x=x-1;

end

t\_1=t\_1-1;

end

%neighbor 4

pos=Position(i,j-1);

x=Spike\_Count(pos,1);

t\_1=t;

while x>0

if(Record\_Keeping(pos,t\_1)>0)

y=(t-t\_1)\*0.1;

z=z+h0\*exp(-y/tau);

x=x-1;

end

t\_1=t\_1-1;

end

%neighbor 5

pos=Position(i,j+1);

x=Spike\_Count(pos,1);

t\_1=t;

while x>0

if(Record\_Keeping(pos,t\_1)>0)

y=(t-t\_1)\*0.1;

z=z+h0\*exp(-y/tau);

x=x-1;

end

t\_1=t\_1-1;

end

%neighbor 6

pos=Position(i+1,j-1);

x=Spike\_Count(pos,1);

t\_1=t;

while x>0

if(Record\_Keeping(pos,t\_1)>0)

y=(t-t\_1)\*0.1;

z=z+h0\*exp(-y/tau);

x=x-1;

end

t\_1=t\_1-1;

end

%neighbor 7

pos=Position(i+1,j);

x=Spike\_Count(pos,1);

t\_1=t;

while x>0

if(Record\_Keeping(pos,t\_1)>0)

y=(t-t\_1)\*0.1;

z=z+h0\*exp(-y/tau);

x=x-1;

end

t\_1=t\_1-1;

end

%neighbor 8

pos=Position(i+1,j+1);

x=Spike\_Count(pos,1);

t\_1=t;

while x>0

if(Record\_Keeping(pos,t\_1)>0)

y=(t-t\_1)\*0.1;

z=z+h0\*exp(-y/tau);

x=x-1;

end

t\_1=t\_1-1;

end

p=z;

end

% time\_passed\_since\_previous\_spike

function [relative\_time] = time\_passed\_since\_previous\_spike(Record\_Keeping, i, j, t)

relative\_time=0.0;

flag=1.0;

i\_1=Position(i,j);

j\_1=t;

current=double(j\_1);

while j\_1>1

if (Record\_Keeping(i\_1,j\_1)>0.0)

flag=double(j\_1);

break;

end

j\_1=j\_1-1;

end

if t~=1

relative\_time=double((current - flag ))\*0.1;

end

end