IRENA Experiments

<u>Scene learning</u>

1-Simple background (White Background) + Simple movement (Uniform Motion)
2-Simple background (White Background) + Complex movement (Moving with different speed)
3-Complex background (No White Background) + Simple movement(Uniform Motion)

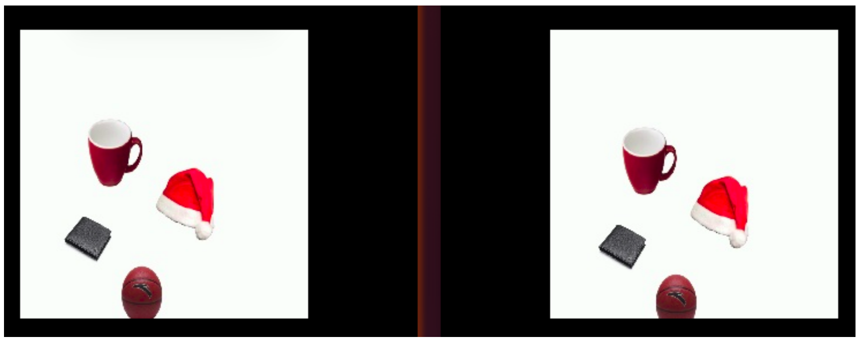
1-Simple background(White Background) + Simple movement(Uniform Motion)

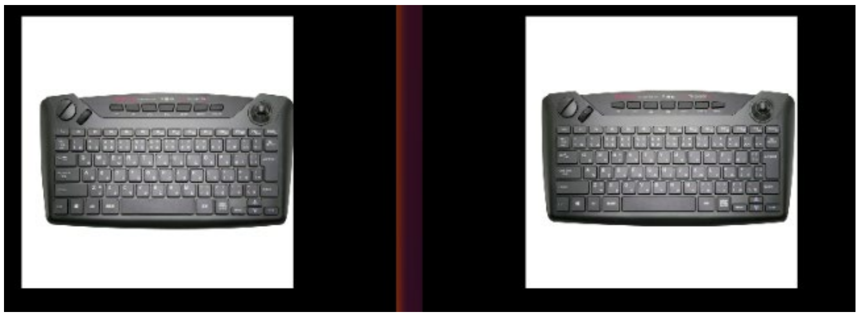
Experiments:
x direction + uniform motion
y direction + uniform motion
y = ¾ x direction + uniform motion
Here, I only show the experimental results of y = ¾ direction + uniform motion.

1.1 Training pictures（video）：
pic_ori_simple_uniformmotion_xy.avi

1.2 Test Pictures：

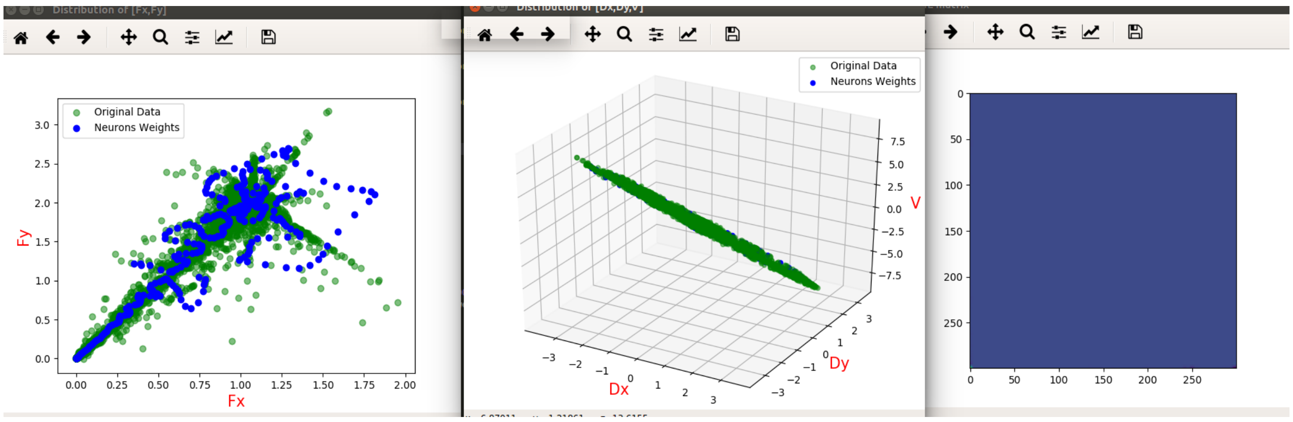
Frame1 Frame2





input：[Frame1_Gx, Frame1_Gy, Frame12_V]
output:[Frame12_Fx, Frame12_Fy]

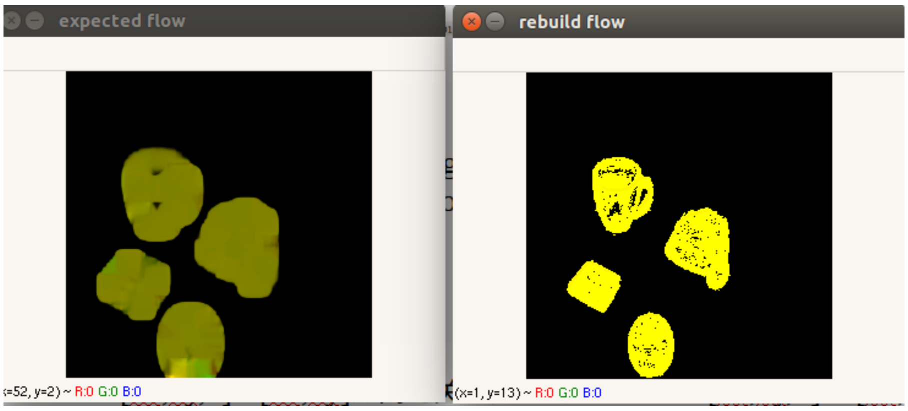
1.3 The weights' distribution and the HL matrix(**Epoch = 199, 1 hour**)



1.4 Expected optical flow image(Generated by calcOpticalFlowFarneback() function in Opencv) and reconstructed optical flow image :

Note: The color means the direction of moving.
The color saturation means the speed.

We can see they are similar with each other in this situation.

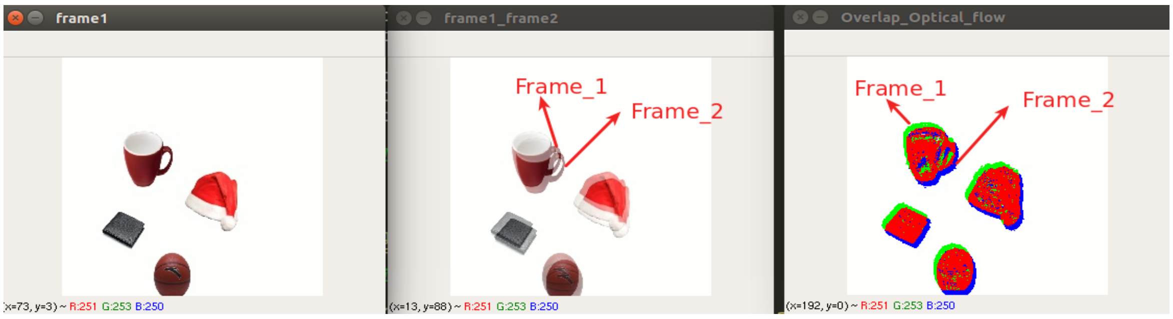


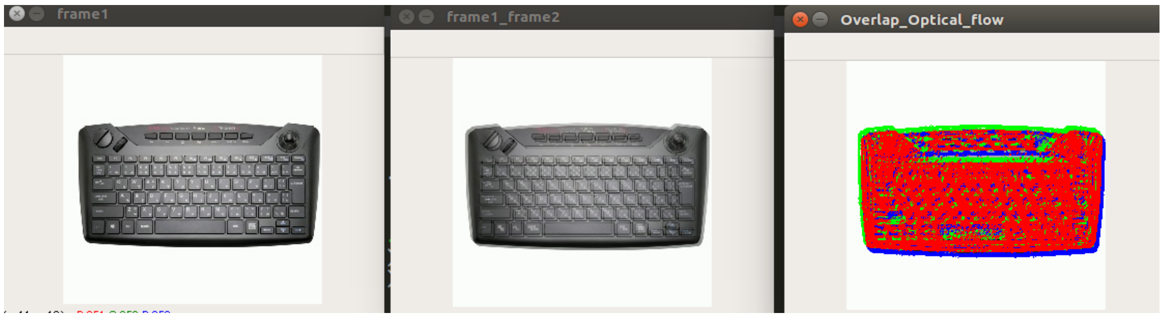
1.5 Prediction Process and the Final display.

Here are 2 consecutive frames(frame_1 and frame_2). We feed our model with [Gx,Gy,V], and we expect to get a dense optical flow image[Fx,Fy] which allow us to predict the position of the moving object in frame2.

Process:
1.5.1- Input the data [Gx,Gy,V]
1.5.2- Get the data[Ux,Uy]
1.5.3- The moving distance of frame_1 with respect to each pixel in frame_2 is recorded in the matrix [Ux,Uy]. We add the corresponding distance in [Ux,Uy] to position of each pixel in frame_1, then we will get its new position in frame_2.

Final display:


frame1
frame1_frame2
Overlap_Optical_flow
Frame_1
Frame_2
Frame_1
Frame_2
(x=73, y=3) ~ R:251 G:253 B:250
(x=13, y=88) ~ R:251 G:253 B:250
(x=192, y=0) ~ R:251 G:253 B:250



In figure3:

Green + Red means the position of moving object in frame_1, and Red+Blue means the expected position of moving object in frame_2. We can see our model can learn optical flow equation well in this situation.

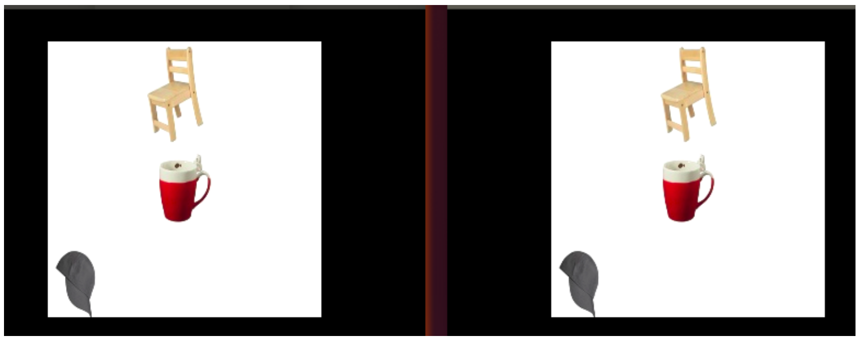
2-Simple background(White Background) + Complex movement(Moving with different speed)

Different speed:

Vcap = 0
Vcup = 0
Vchair > 0

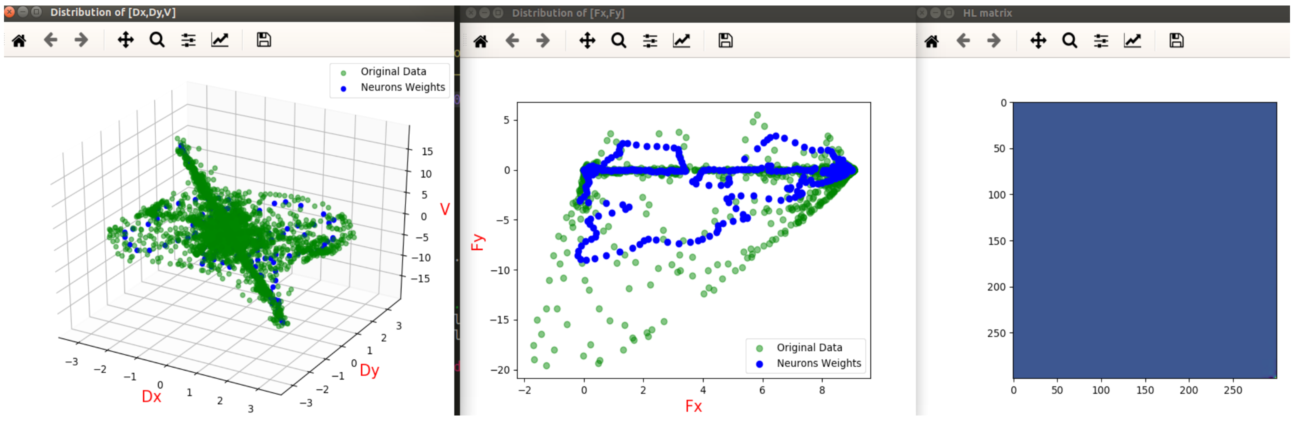
1.1 Training pictures（video） :
one_move_white_background.mp4

1.2 Test Pictures：Frame1,Frame2



input：[Frame1_Gx, Frame1_Gy, Frame12_V]
output:[Frame12_Fx, Frame12_Fy]

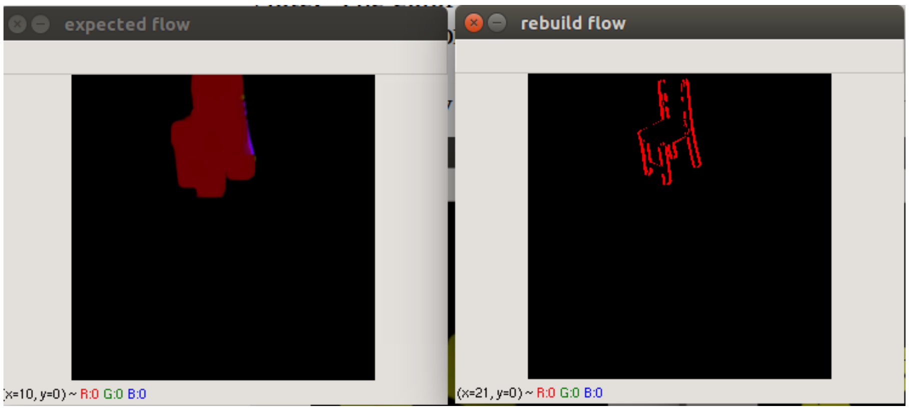
1.3 The weights' distribution and the HL matrix(**Epoch = 499, 2.5 hour**)



1.4 Expected optical flow image(Generated by calcOpticalFlowFarneback() function in Opencv) and reconstructed optical flow image：

Note: The color means the direction of moving.
The color saturation means the speed.

We can see they are similar with each other in this situation.

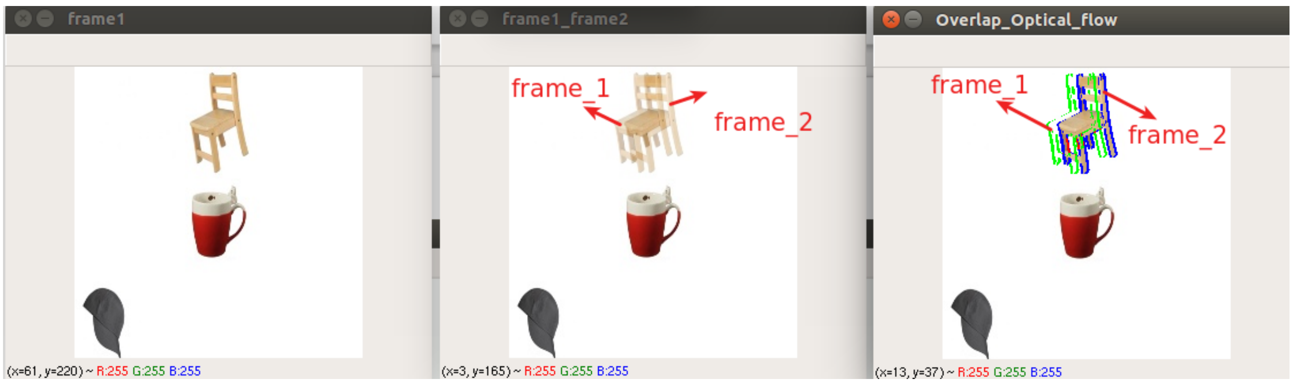


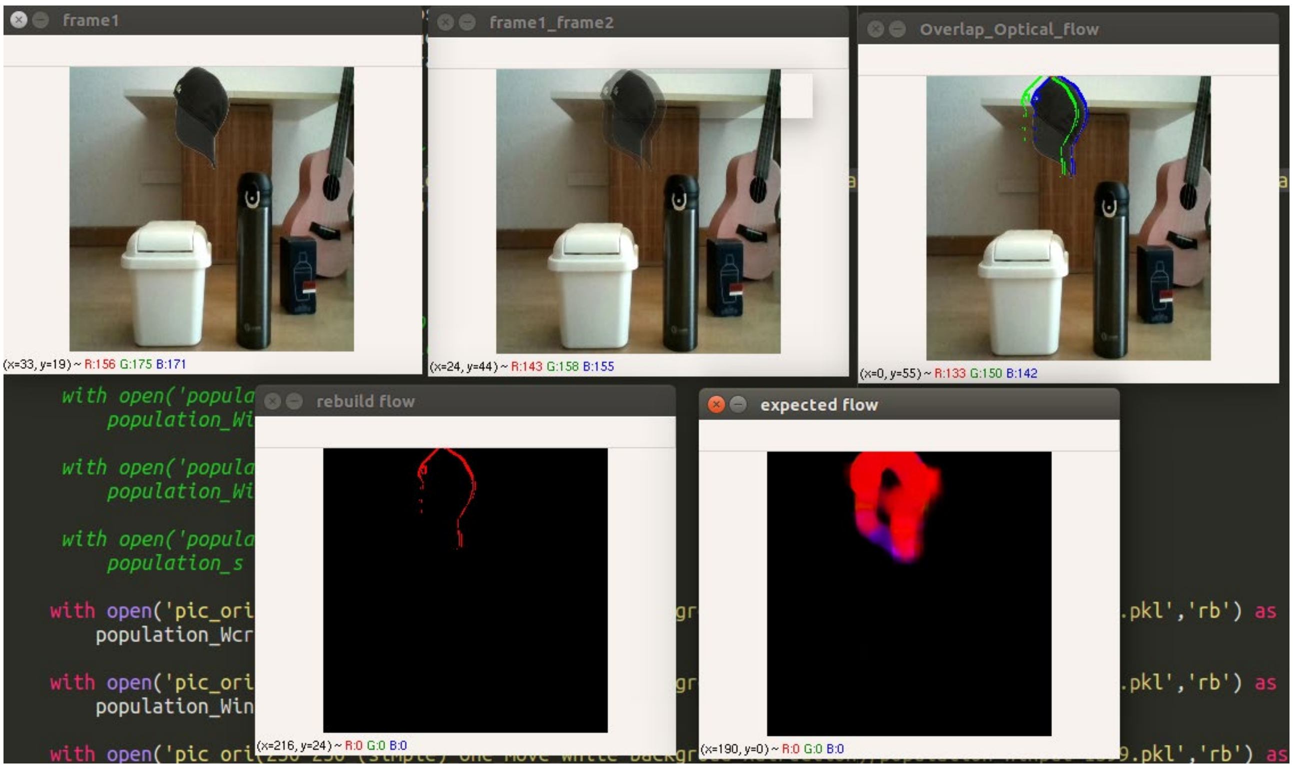
1.5 Prediction Process and the Final display.

Here are 2 consecutive frames(frame_1 and frame_2). We feed our model with [Gx,Gy,V], and we expect to get a dense optical flow image[Fx,Fy] which allow us to predict the position of the moving object in frame2.

Process:
1.5.1- Input the data [Gx,Gy,V]
1.5.2- Get the data[Ux,Uy]
1.5.3- The moving distance of frame_1 with respect to each pixel in frame_2 is recorded in the matrix [Ux,Uy]. We add the corresponding distance in [Ux,Uy] to position of each pixel in frame_1, then we will get its new position in frame_2.

Final display:


frame1
frame1_frame2
Overlap_Optical_flow
frame_1
frame_2
frame_1
frame_2
(x=61, y=220) ~ R:255 G:255 B:255
(x=3, y=165) ~ R:255 G:255 B:255
(x=13, y=37) ~ R:255 G:255 B:255


frame1
frame1_frame2
Overlap_Optical_flow
rebuild flow
expected flow
(x=33, y=19) ~ R:156 G:175 B:171
(x=24, y=44) ~ R:143 G:158 B:155
(x=0, y=55) ~ R:133 G:150 B:142
(x=216, y=24) ~ R:0 G:0 B:0
(x=190, y=0) ~ R:0 G:0 B:0

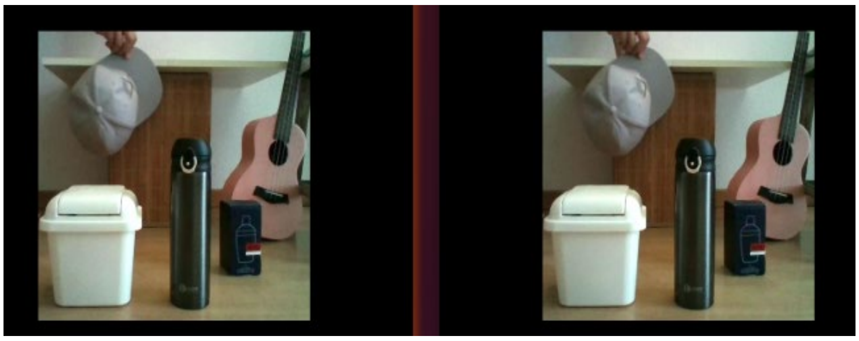
```
    with open('popula
        population_Wi

    with open('popula
        population_Wi

    with open('popula
        population_s

with open('pic_ori                    gr                    .pkl','rb') as
    population_Wcr

with open('pic_ori                    gr                    .pkl','rb') as
    population_Win

with open('pic_ori(250*250 (simple) one move white backgr                    9.pkl','rb') as
```

In figure3:

Green + Red means the position of moving object in frame_1, and Red+Blue means the expected position of moving object in frame_2. We can see our model can learn optical flow equation well in this situation.

3-Complex background(No White Background) + Simple movement(Uniform Motion)

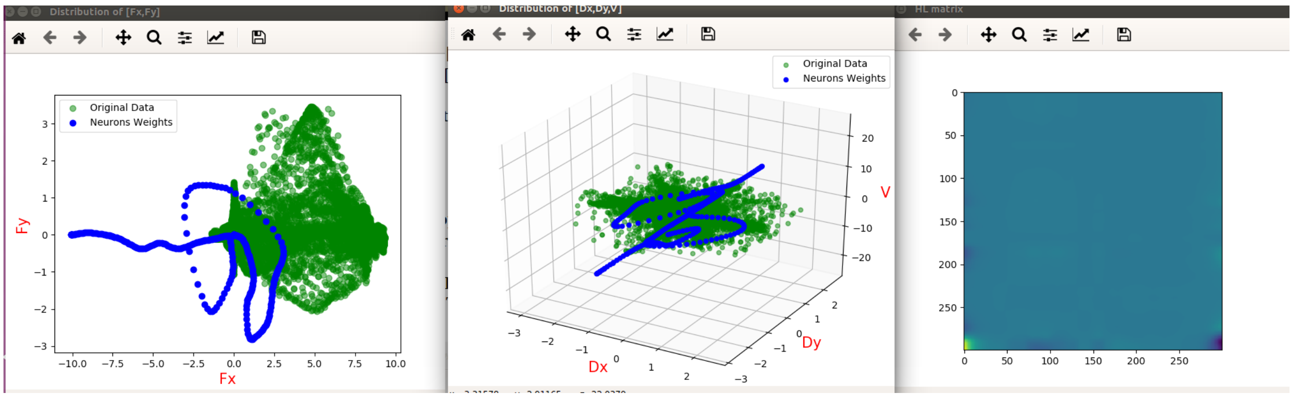
1.1 Training pictures（video）：
stable_scene_1.avi



1.2 Test Pictures：Frame1,Frame2

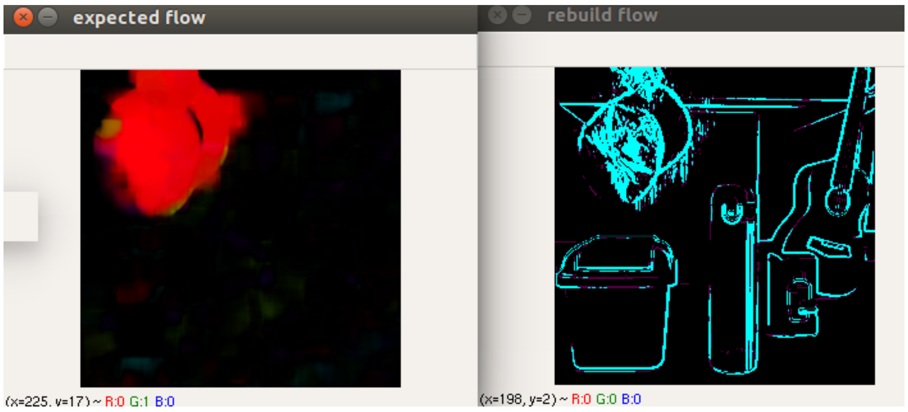
input：[Frame1_Gx, Frame1_Gy, Frame12_V]
output:[Frame12_Fx, Frame12_Fy]

1.3 The weights' distribution and the HL matrix(**Epoch = 599 4 hour**)



1.4 Expected optical flow image(Generated by calcOpticalFlowFarneback() function in Opencv) and reconstructed optical flow image：

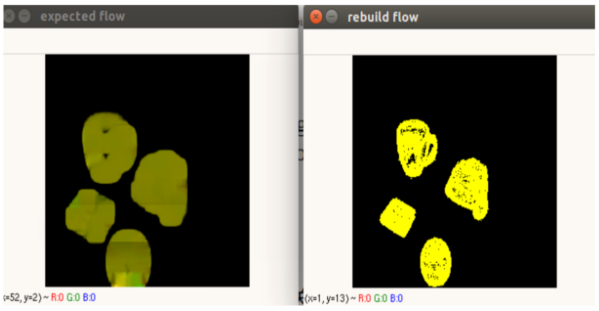
We can see they are quite different with each other in this situation.

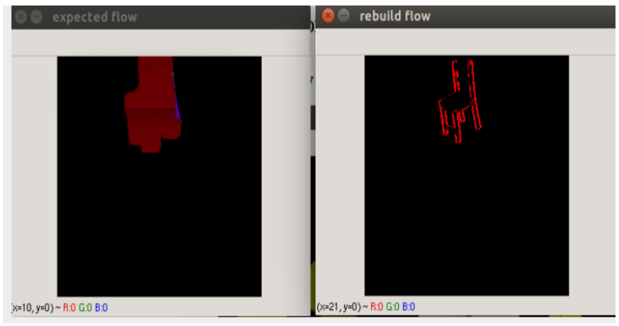


This is why I divided the experiment into 4 situation(**1-Simple background + Simple movement .2-Simple background + Complex movement 3.-Complex background+ Simple movement 4.Complex background + Complex movement**).As you can see, our model has the ability to learn optical flow equations, just need the right neurons and training time.

Note:

1- Compare with situation 1 Why there are only some contours in situation 2?

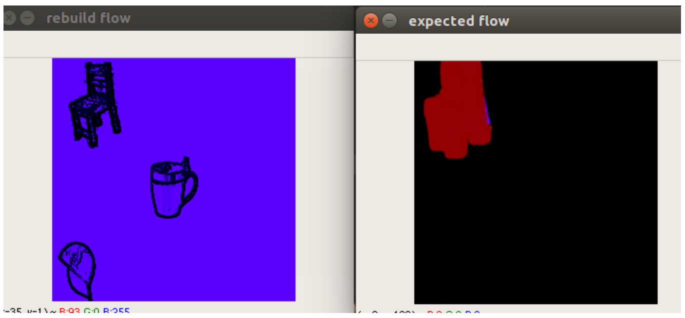




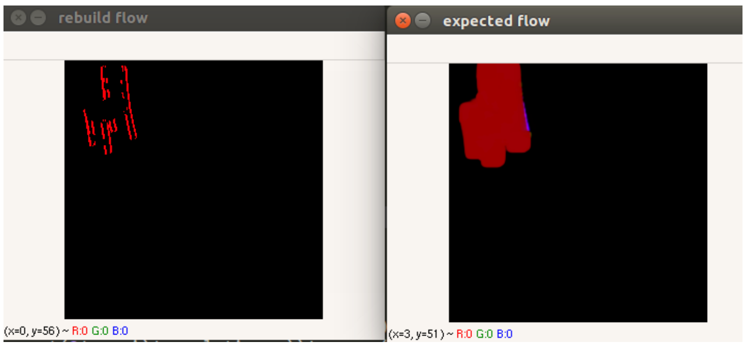
Answer:

The background(or the motion) in situation 2 is much more complicated than situation 1. We need more training time or more Neurons to get better results. The intermediate results are as follows:

Epoch = 99: (The number of red point : 1)



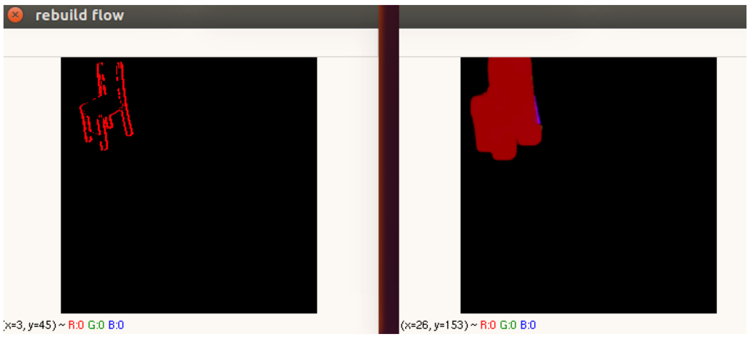
Epoch = 299:(The number of red point : 586)



(x=0, y=56) ~ R:0 G:0 B:0

(x=3, y=51) ~ R:0 G:0 B:0

Epoch = 599:(The number of red point : 754)

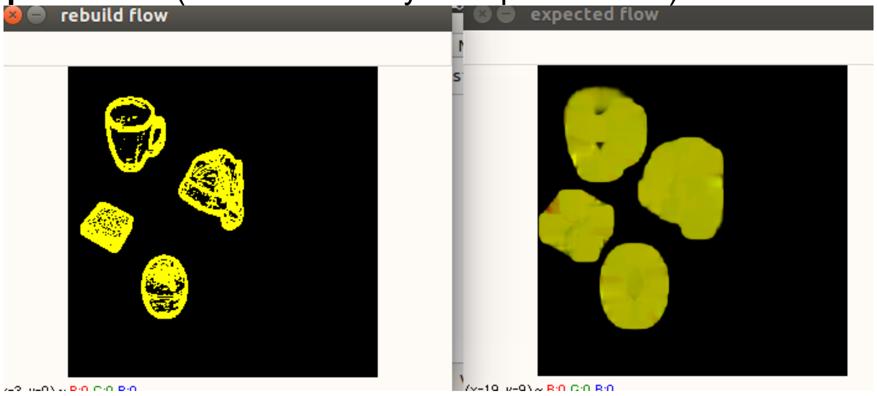


(x=3, y=45) ~ R:0 G:0 B:0

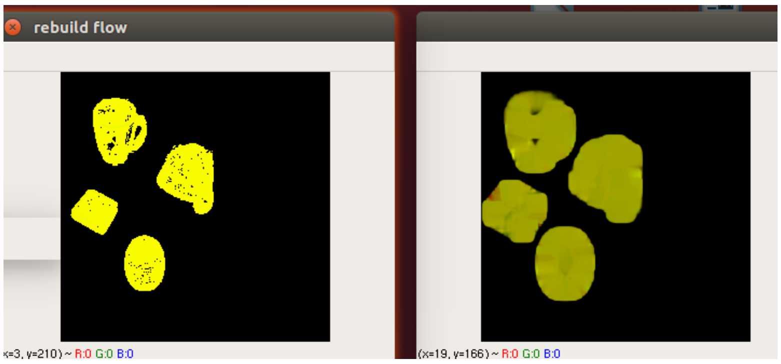
(x=26, y=153) ~ R:0 G:0 B:0

Here are intermediate results for situation 1.

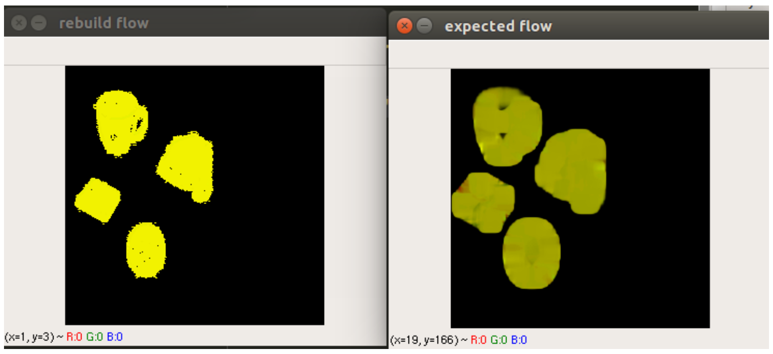
Epoch = 99: (The number of yellow point : 5169)



Epoch = 299: (The number of yellow point : 7292)



Epoch=599:(The number of yellow point : 7949)

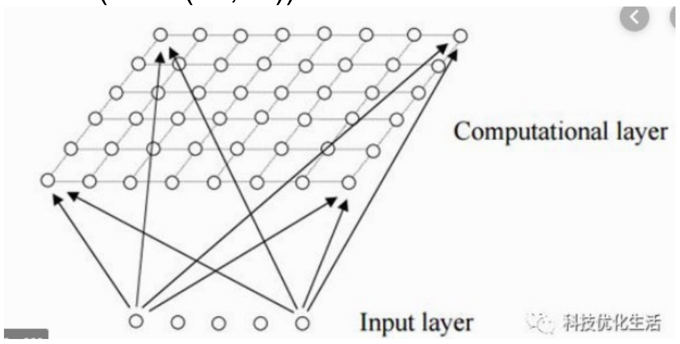


Scene understanding and classification capabilities

IRENA's model is based on object classification, and our ultimate goal is actually to achieve the multi-sensor fusion, not just to complete the classification. As you know, we use SOMs to learn the distribution of our input data, and then use SOMs as the input of HL to learn the correlation between SOMs. Actually, the essence of SOM is to classify our input data. Here, we can give them two example to show that our model actually include the object classification.

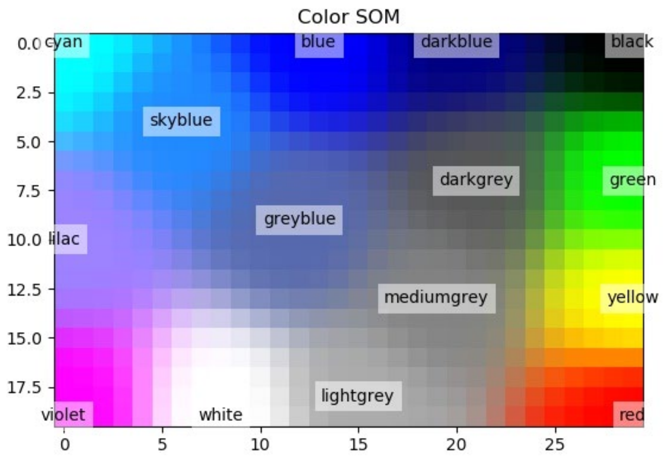
1-Color Classification:

1- Design a 2-D SOM(size=(20,30)) network.



2-Generate some Colors('black', 'blue', 'darkblue', 'skyblue', 'greyblue', 'lilac', 'green',red', 'cyan', 'violet', 'yellow', 'white','darkgrey', 'mediumgrey', 'lightgrey')

3-After 500 times training, we can get the final SOM picture below. We can see the neurons are classified very well. It proves that our model have the ability of classification.

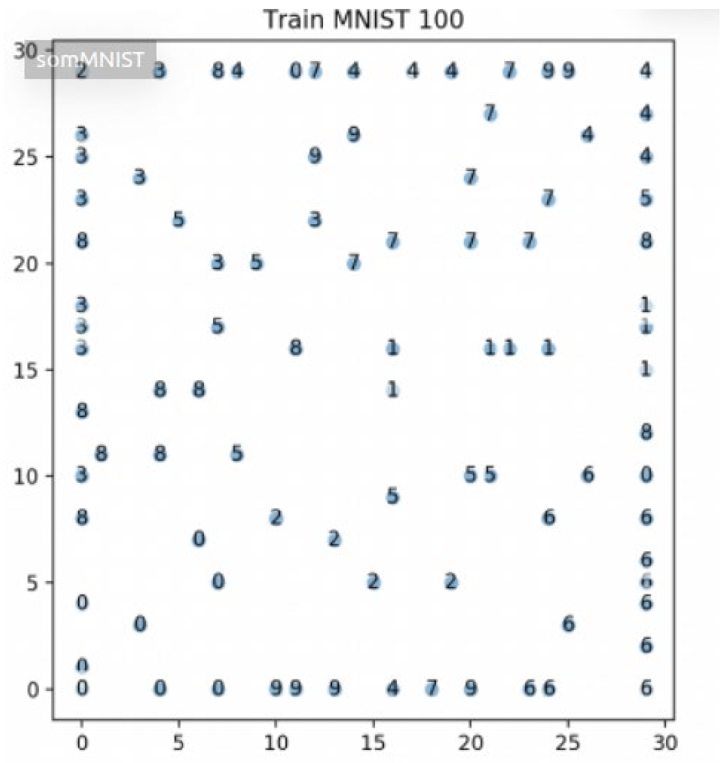


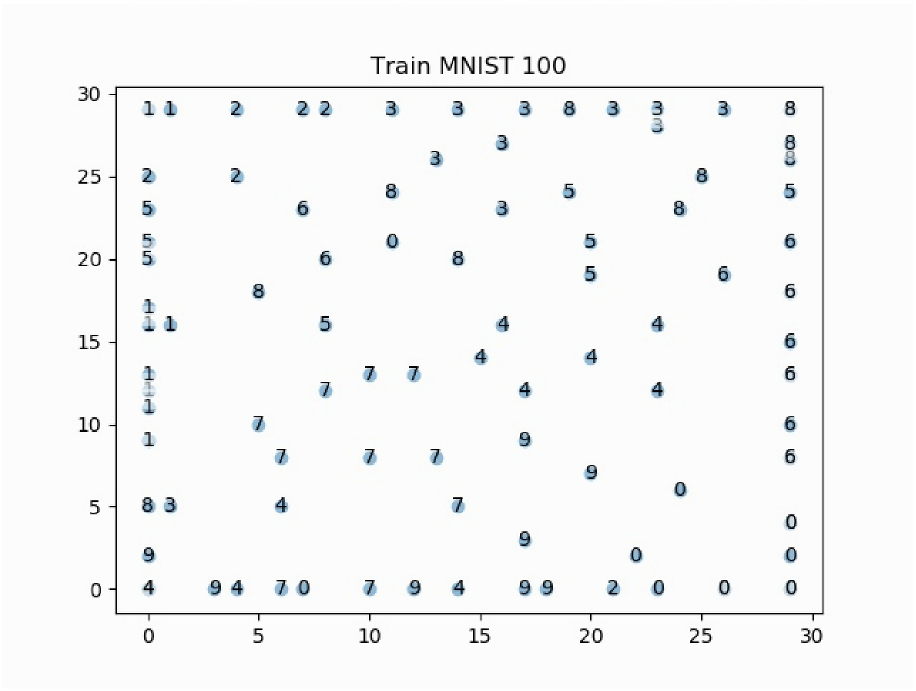
2-MNIST Classification:

1- Design a 2-D SOM(size=(30,30)) network.

2-Download MNIST data, and then choose 100 numbers(0-9) as the input of our SOM network.

3-After 200 times training, we can get the distribution of final SOM like picture 1





4-Input 10 test data. we can see the corresponding locations are activated. Each of the plotted MNIST test digits is well fitted into the target locations. Although digit 2 and 0 are not completely split, it seems to be resolved if the number of training iteration increases.