Game AI

Project 3: Behavior Programming

Alina Arunova
Phil Matyash
Marina Mircheska
Aditya Kela
Sattar Rahimbeyli

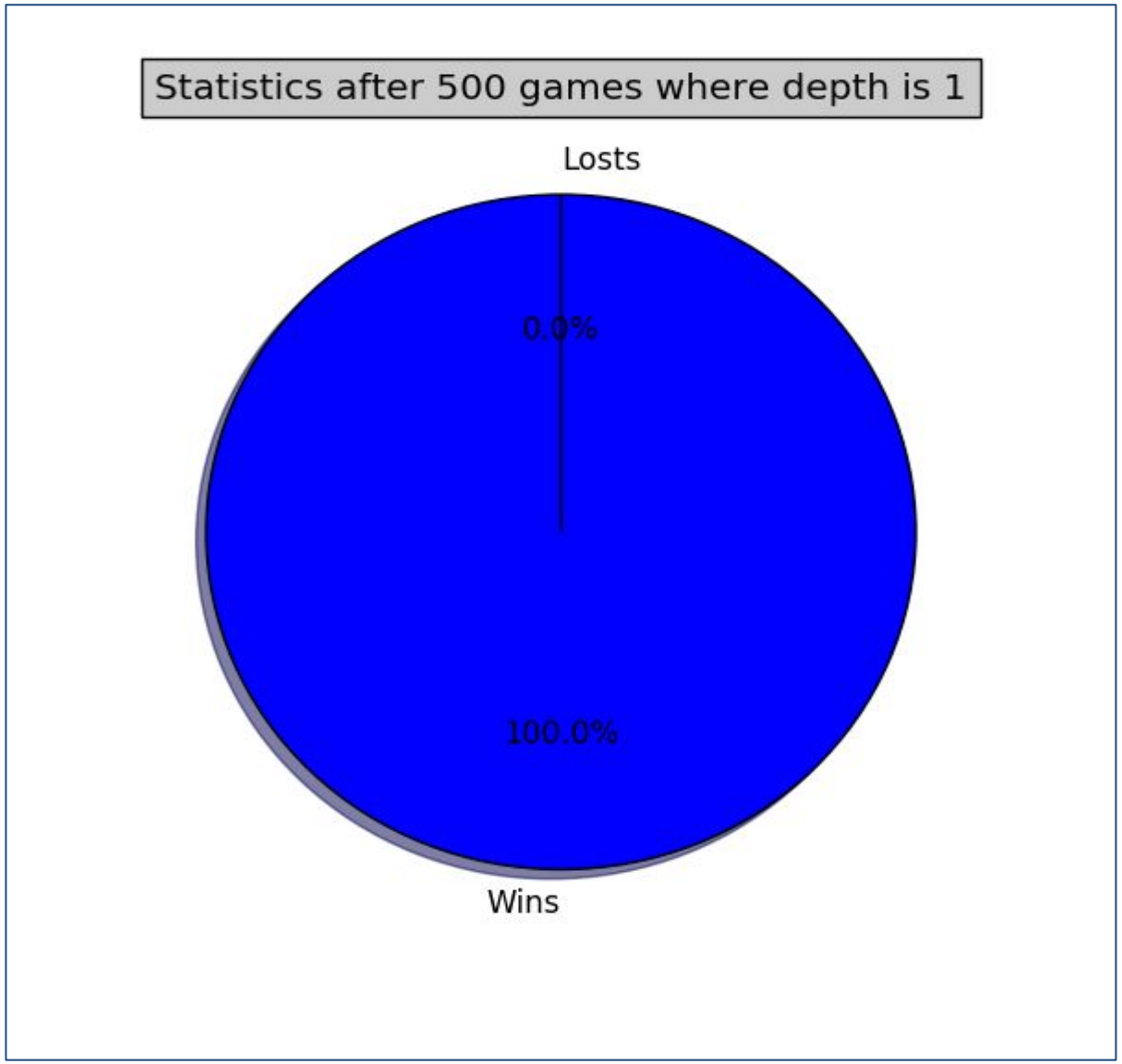
Outline

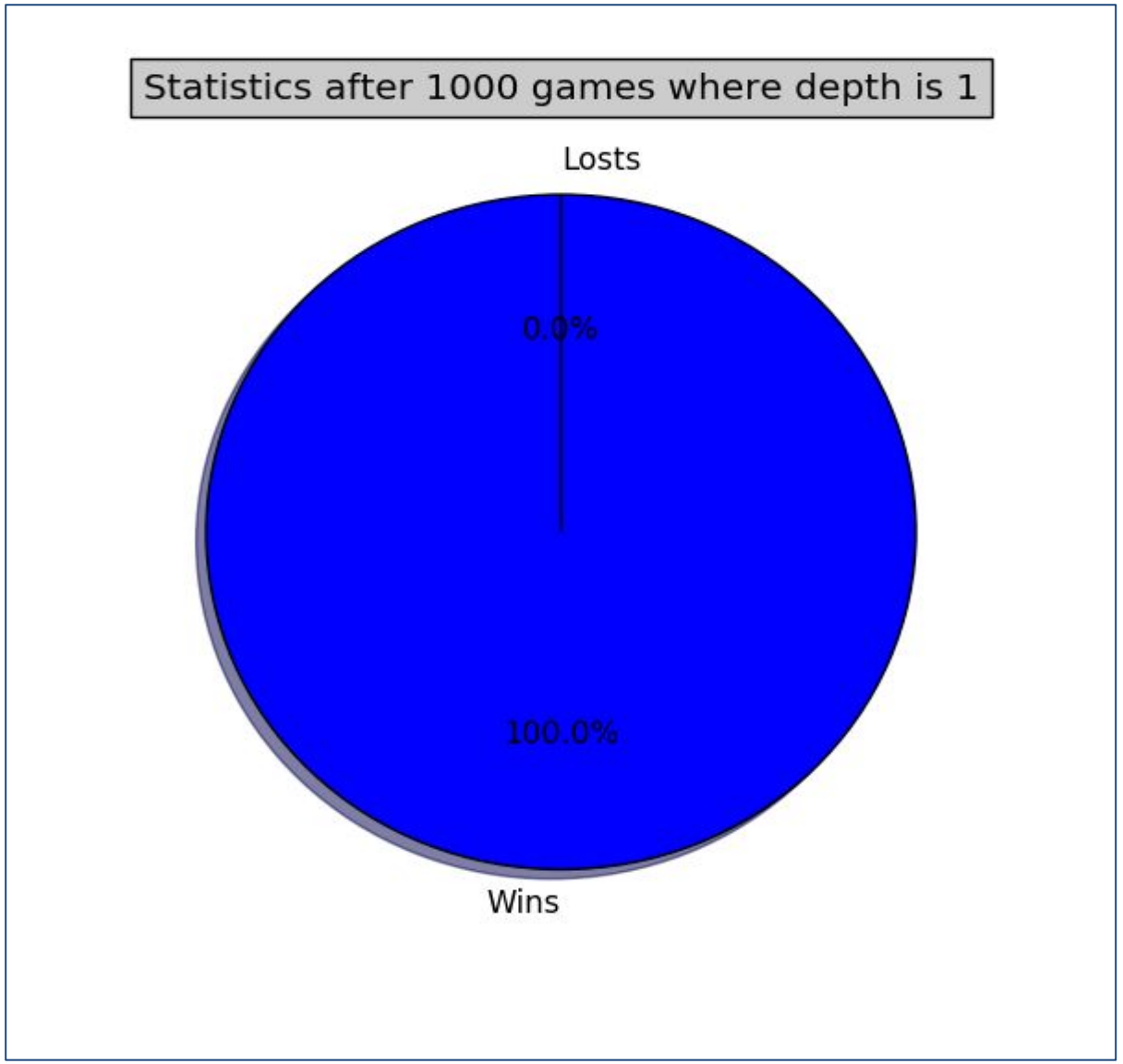
- *Connect four* for large board
- *Fuzzy-controller* for breakout
- *Self organizing maps*
- *Bayesian imitation learning*

Task 3.1: Minmax search for connect four 19X19

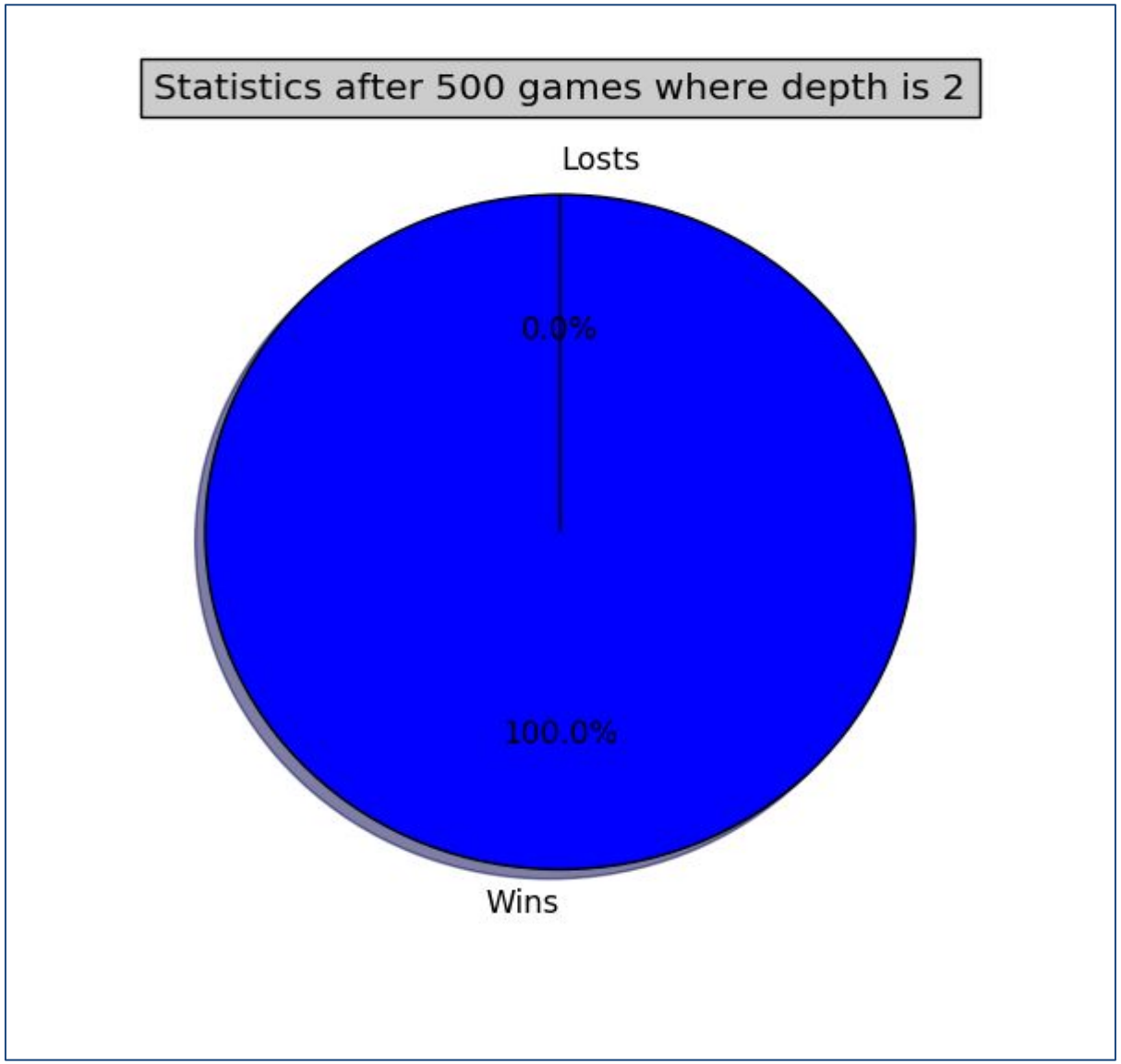
1) **Implement the depth-restricted search to find the best move**
   a) Modify the depth-parameter to find the best case
2) **Gather statistics with resized board (19x19)**
   a) Win/Loss statistics
   b) Time statistics
   c) The number of filled layers
   d) Depth-parameter statistics

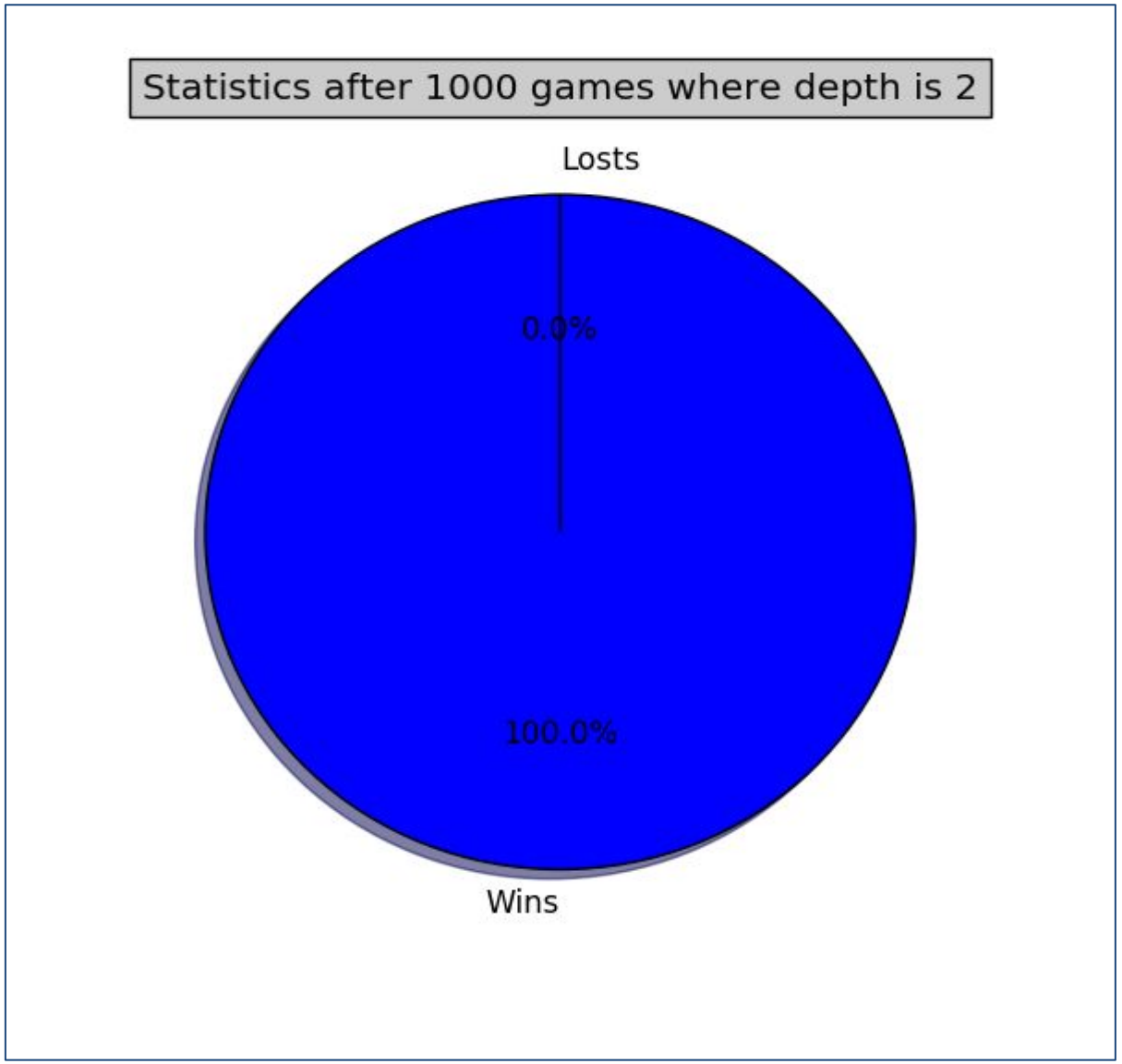
Task 3.1:Win/Loss statistics





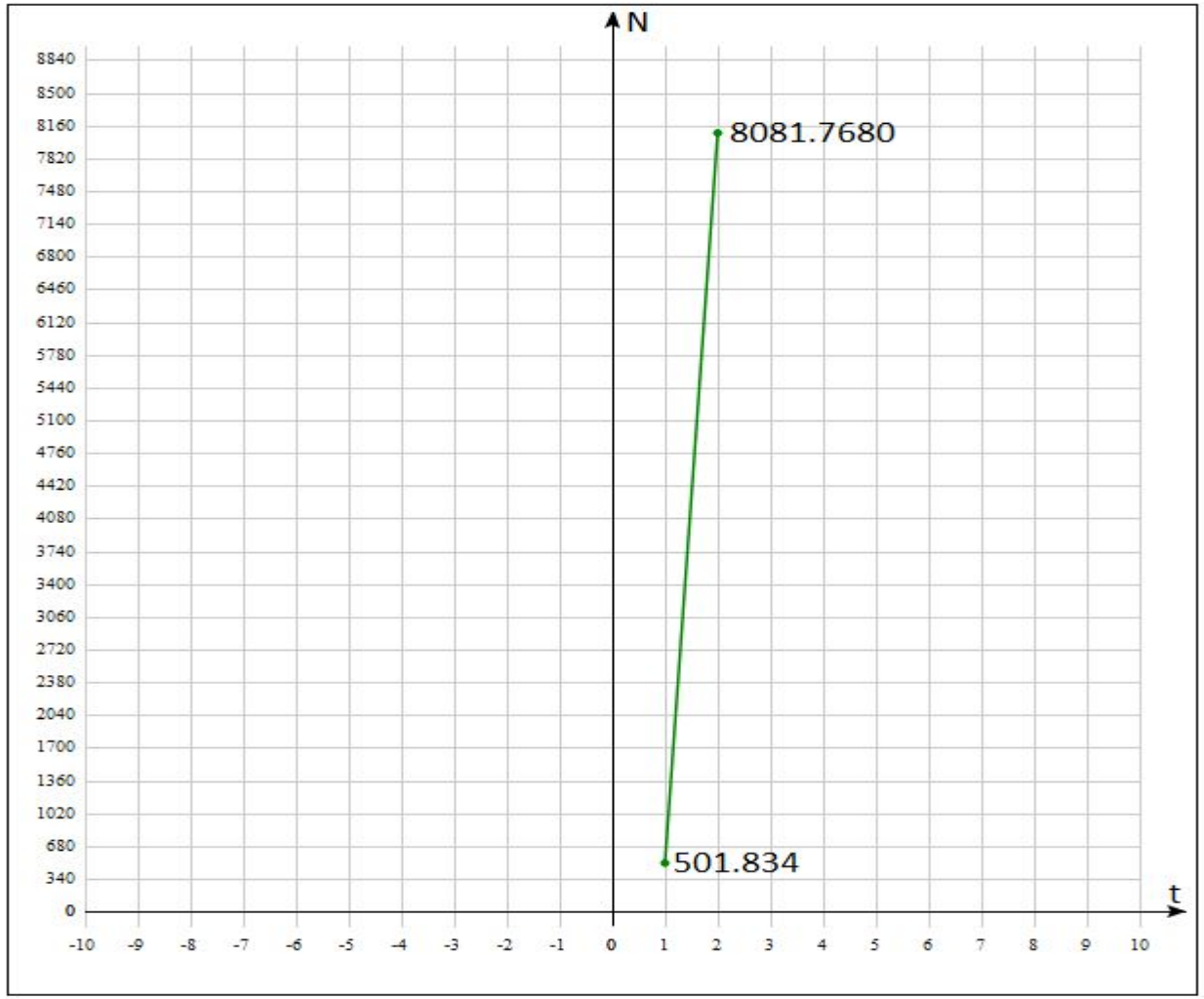
Task 3.1:Win/Loss statistics



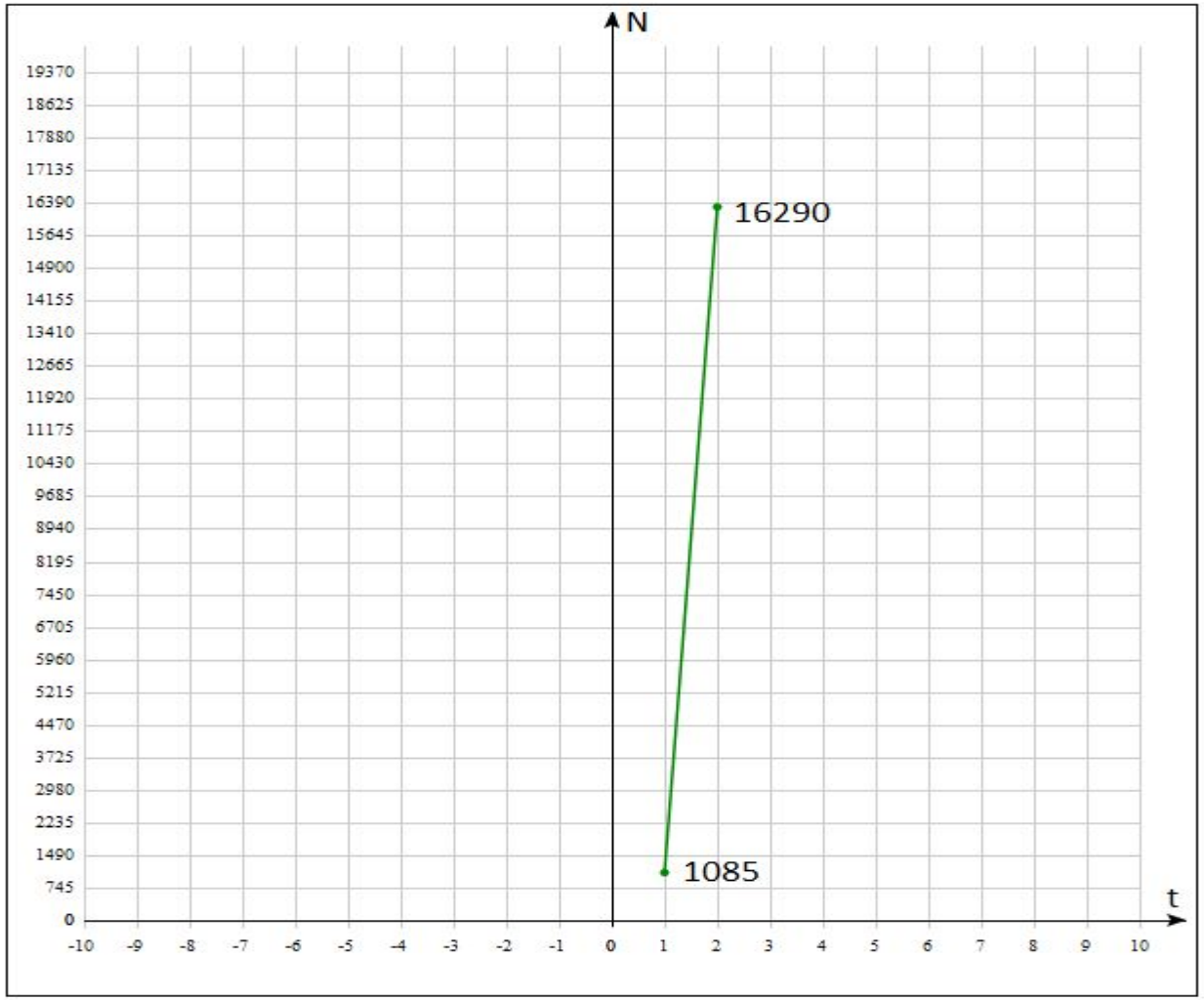


Task 3.1:Time statistics

500 samples



1000 samples



Task 3.1:The number of filled layers

500 Samples depth 1

	0	1	2	3
...				
13		1	1	
14		5	1	
15	416	19	3	
16	444	21	3	
17	466	22	4	2
18	500	80	71	62

1000 Samples depth 1

	0	1	2	3
...				
13		1		
14		12	1	
15	839	37	10	
16	898	40	14	
17	949	49	16	5
18	1000	148	134	112

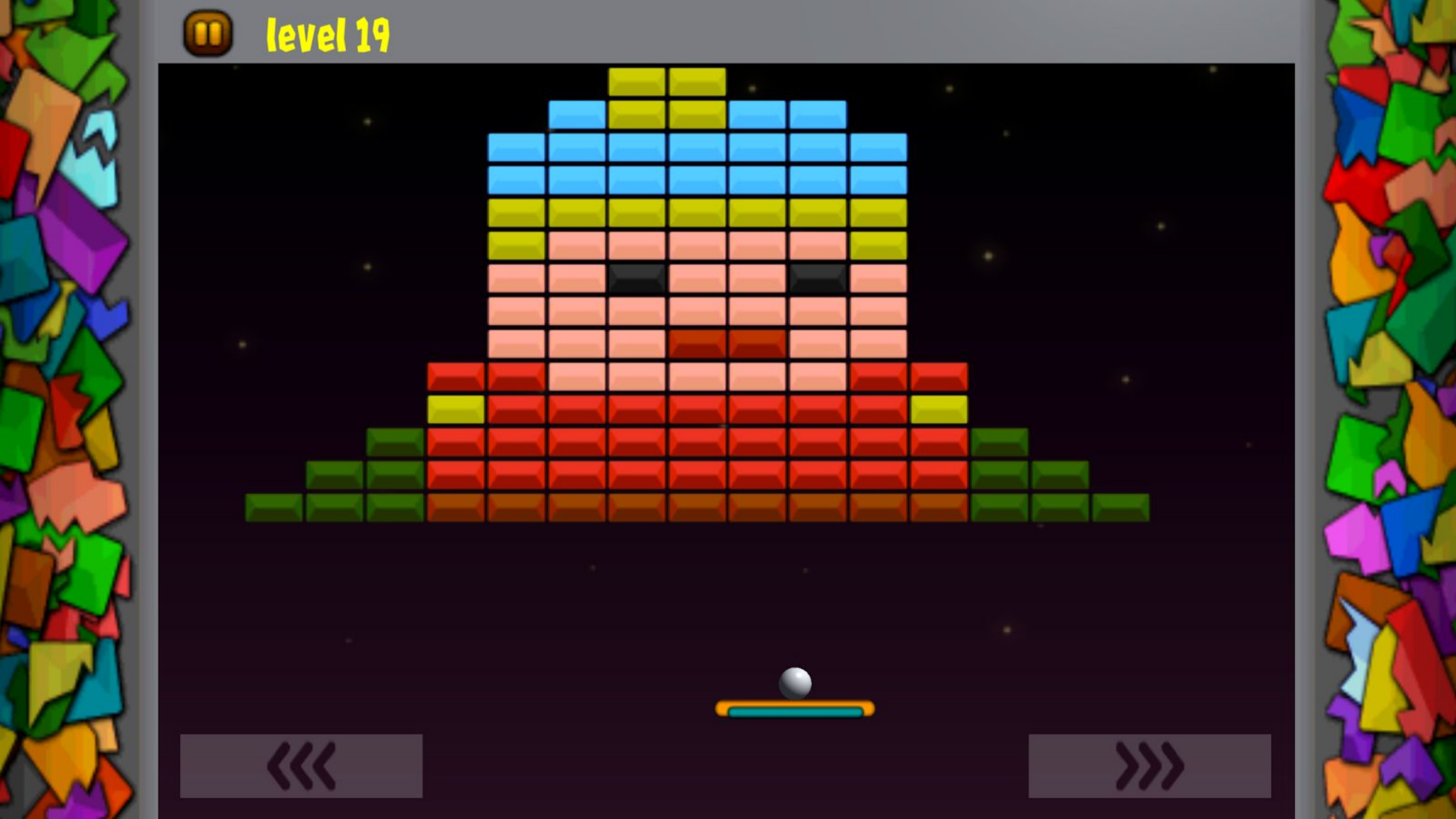
Task 3.1:The number of filled layers

500 Samples depth 2

	0	1	2	3
...				
13		3	1	
14		8	1	
15	432	17	2	
16	450	15	2	
17	473	17	1	
18	500	62	57	52

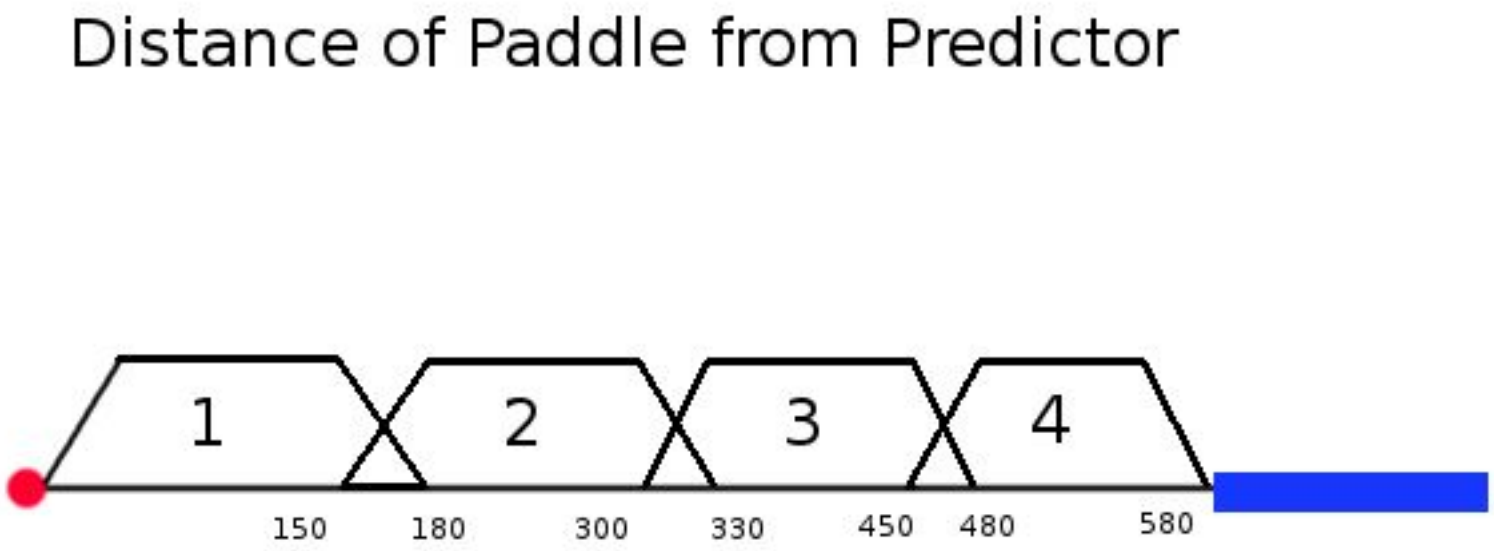
1000 Samples depth 2

	0	1	2	3
...				
13				
14		7		
15	856	30	5	
16	900	30	7	
17	949	34	12	7
18	1000	140	126	104


level 19

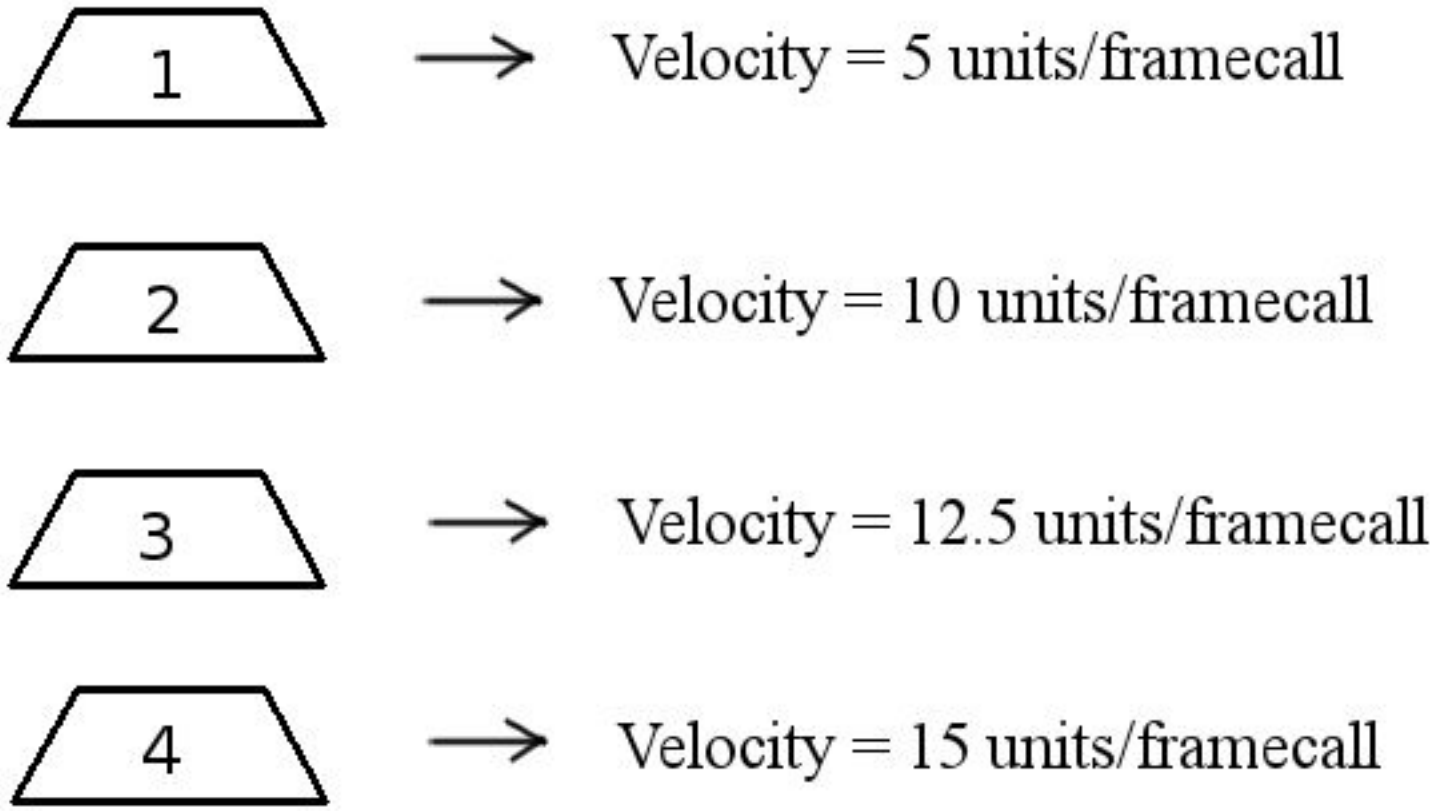
Task 3.2: Fuzzy Controller for Breakout

The maximum distance (= 580 units) between the Paddle and the Predictor ball is divided into 4 rules



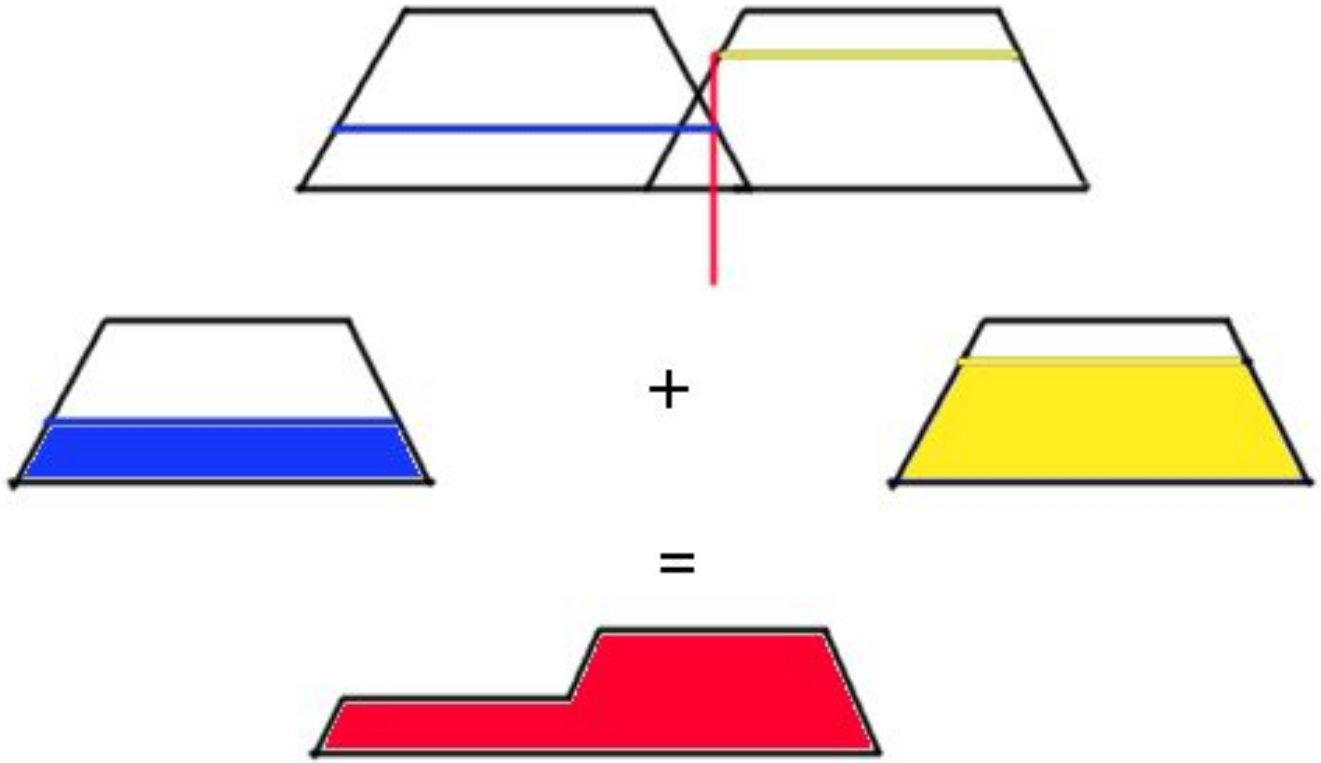
Task 3.2: Fuzzy Controller for Breakout

The rule set for the divided distance is:



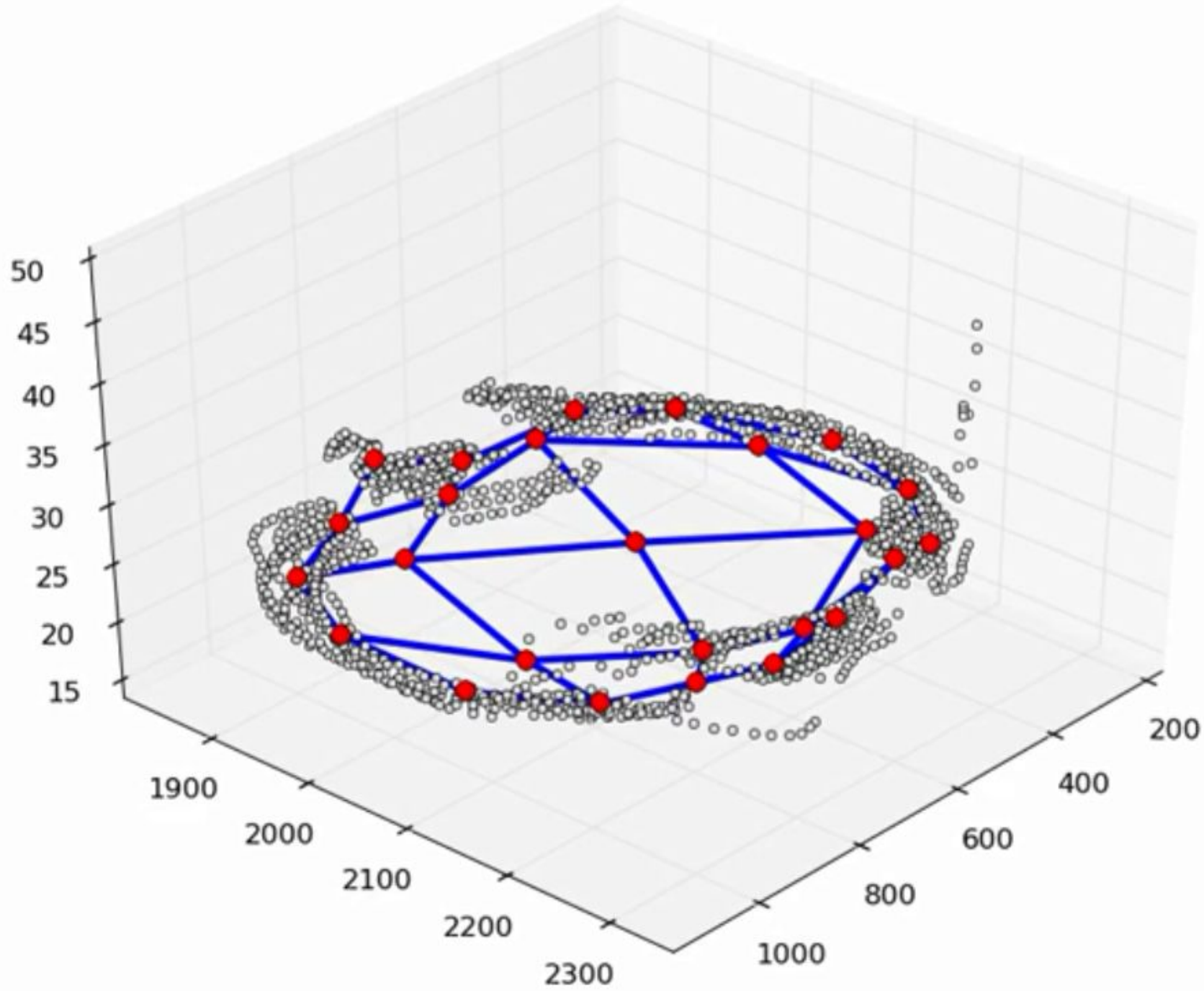
Task 3.2: Fuzzy Controller for Breakout

If the distance between the predictor ball and the paddle leads to two rules firing, then the velocity function is the weighted average:



Velocity = Weighted Average of Both Areas

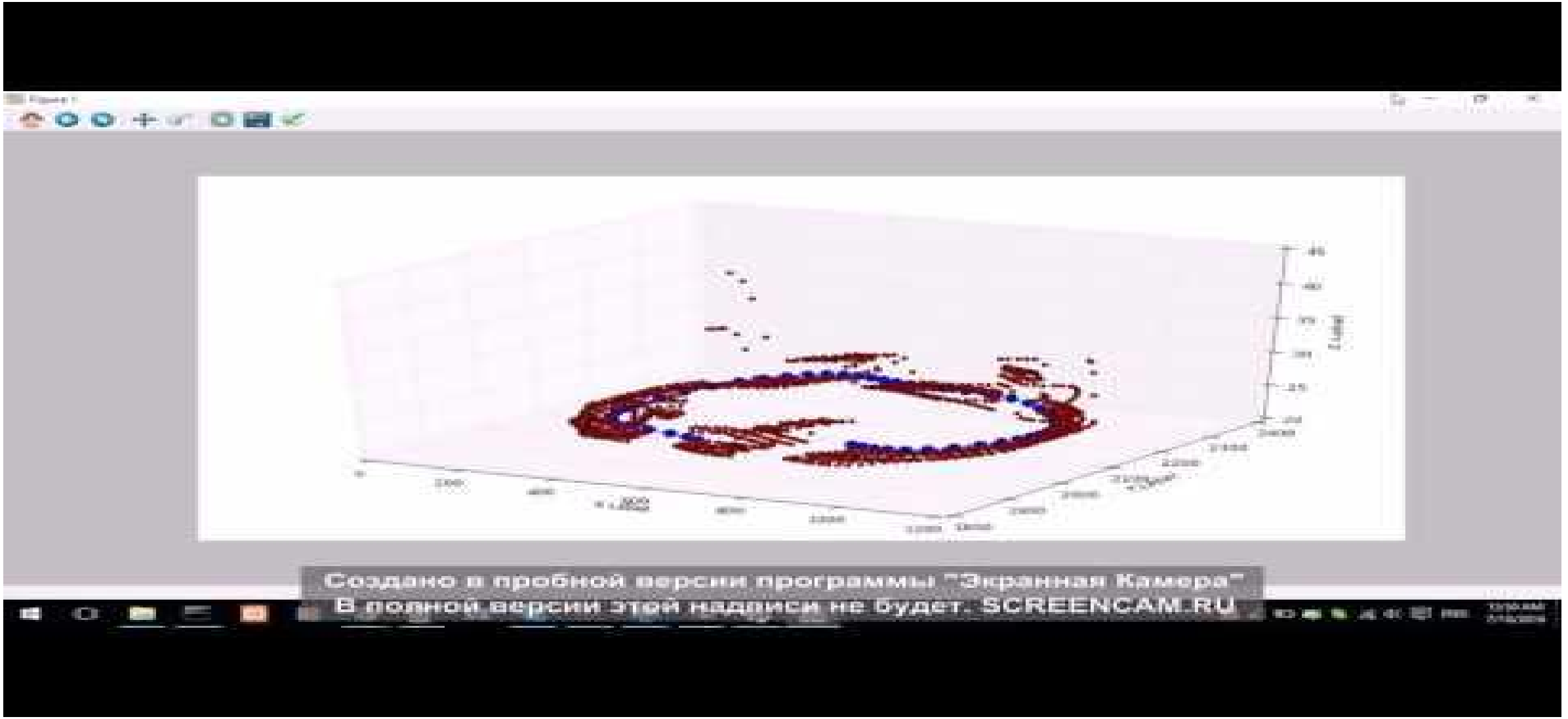
= [5*Area(1)+10*Area(2)]/[Area(1)+Area(2)]


50
45
40
35
30
25
20
15
1900
2000
2100
2200
2300
200
400
600
800
1000

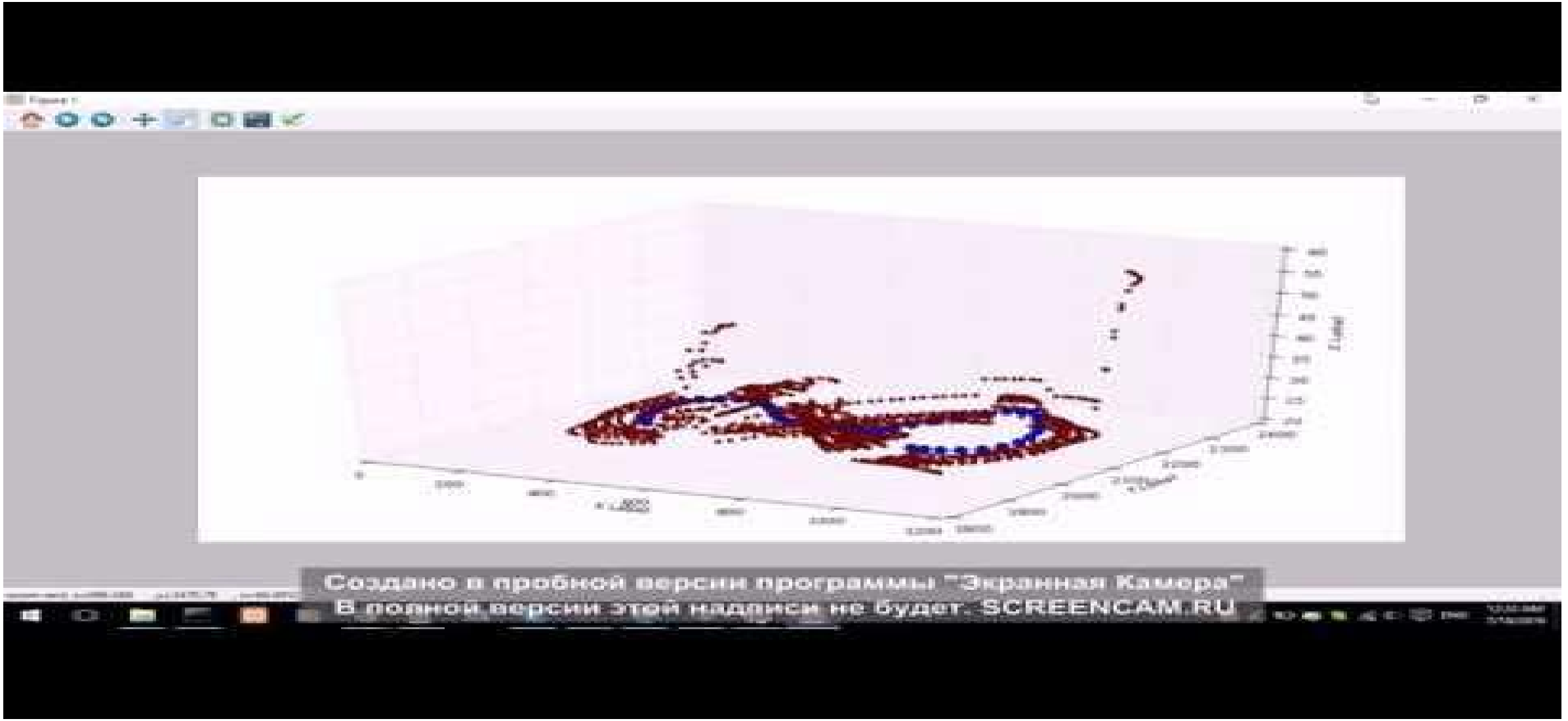
Task 3.3: Self organizing maps

1) **Fit self organizing maps to the datasets**
    a) Self organizing map algorithm
        i) Assign random values to neurons (centers)
        ii) Choose random pattern from dataset
        iii) Find the closest neuron to that pattern(custom metric)
            (1) distance[winner] = distance(neuron,pattern)
        iv) Update neurons(centers)
            (1) nrn[i] += lr(t)*e^(dist(nrn[i],pat)/2*sig(t))*(pat - nrn[i])
            (2) lr(t) = 1 - (t/T)
            (3) sig(t) = e^[-(t/T)]

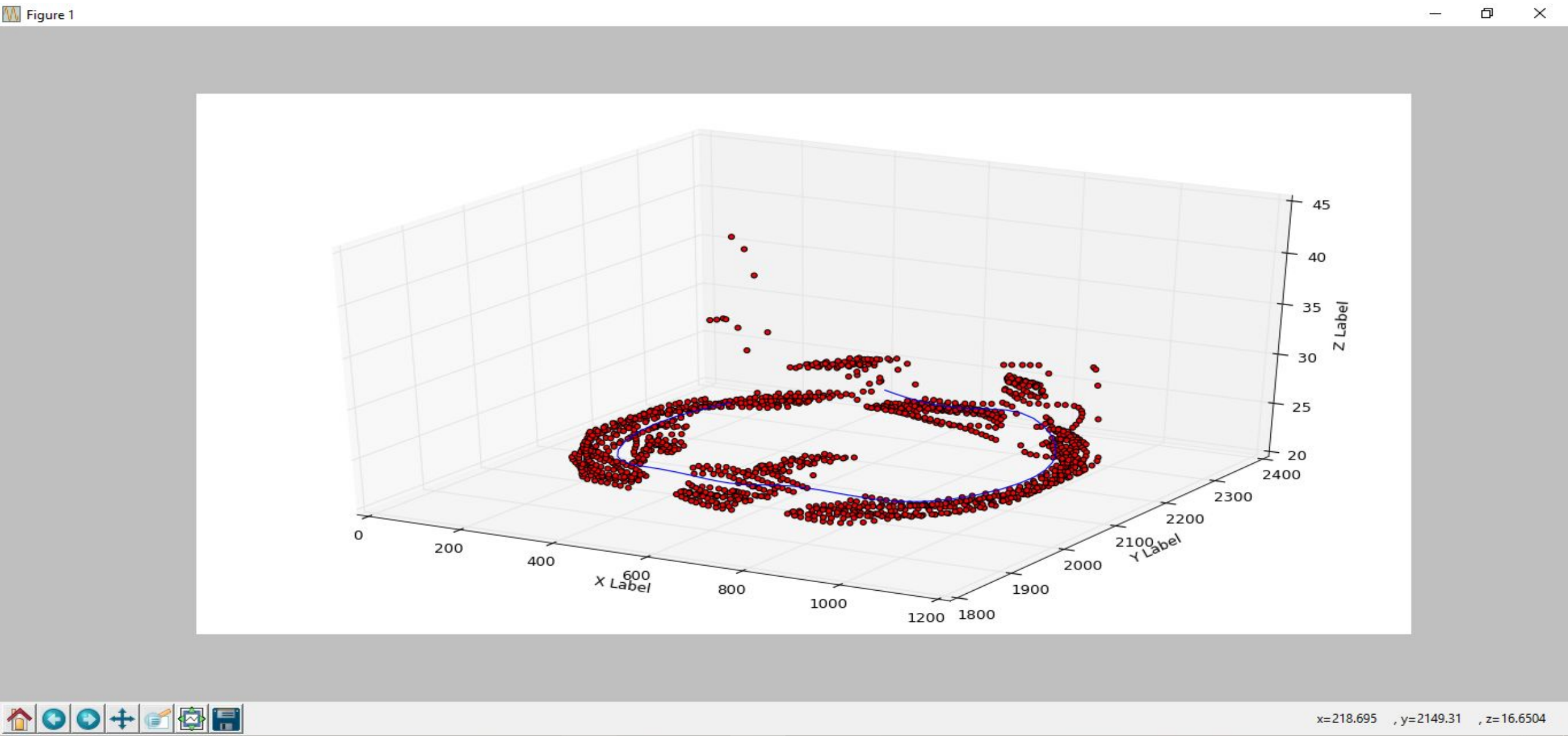
Task 3.3: Self organizing maps (circle figure)



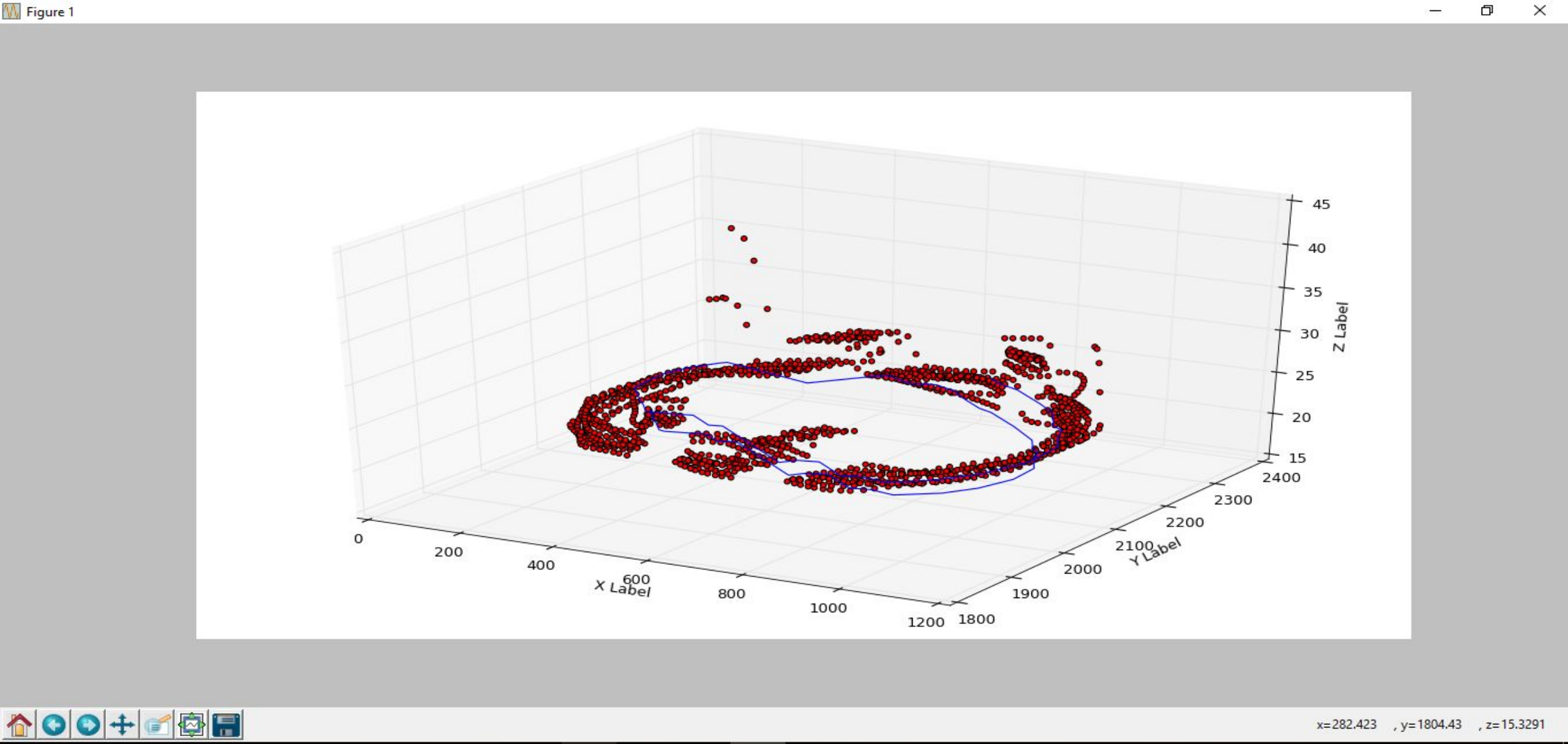
Task 3.3: Self organizing maps (figure 8)



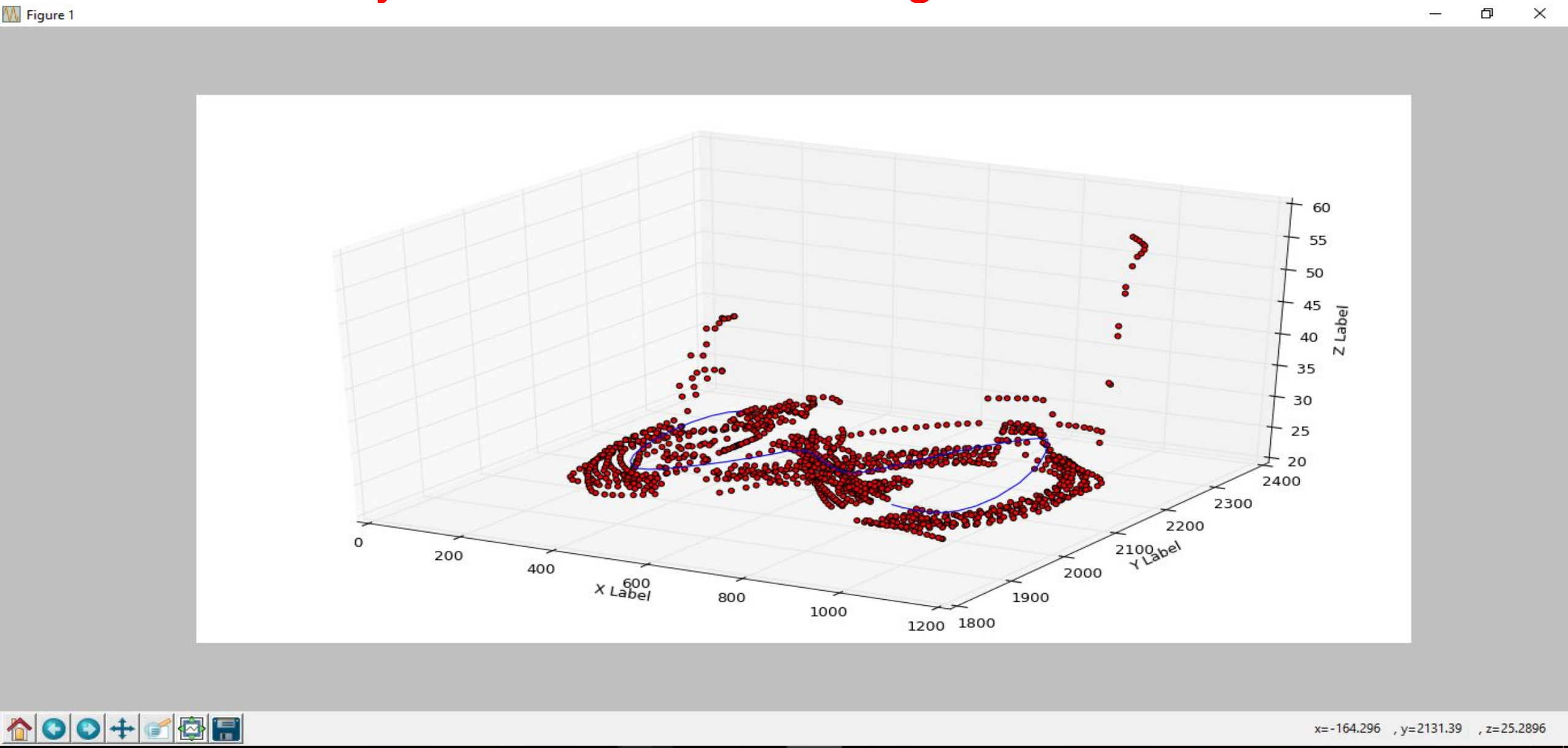
Task 3.4: Bayesian imitation learning



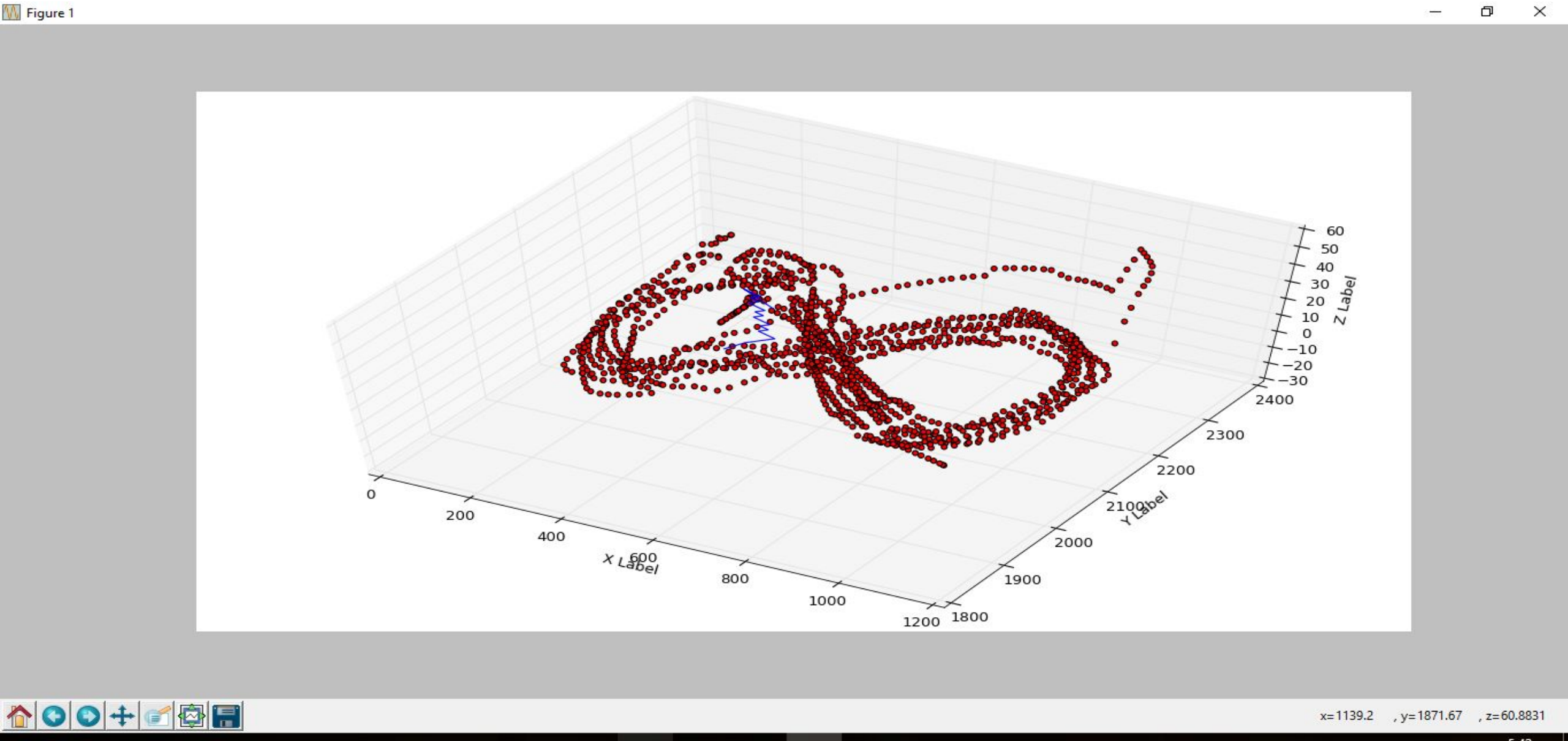
Task 3.4: Bayesian imitation learning



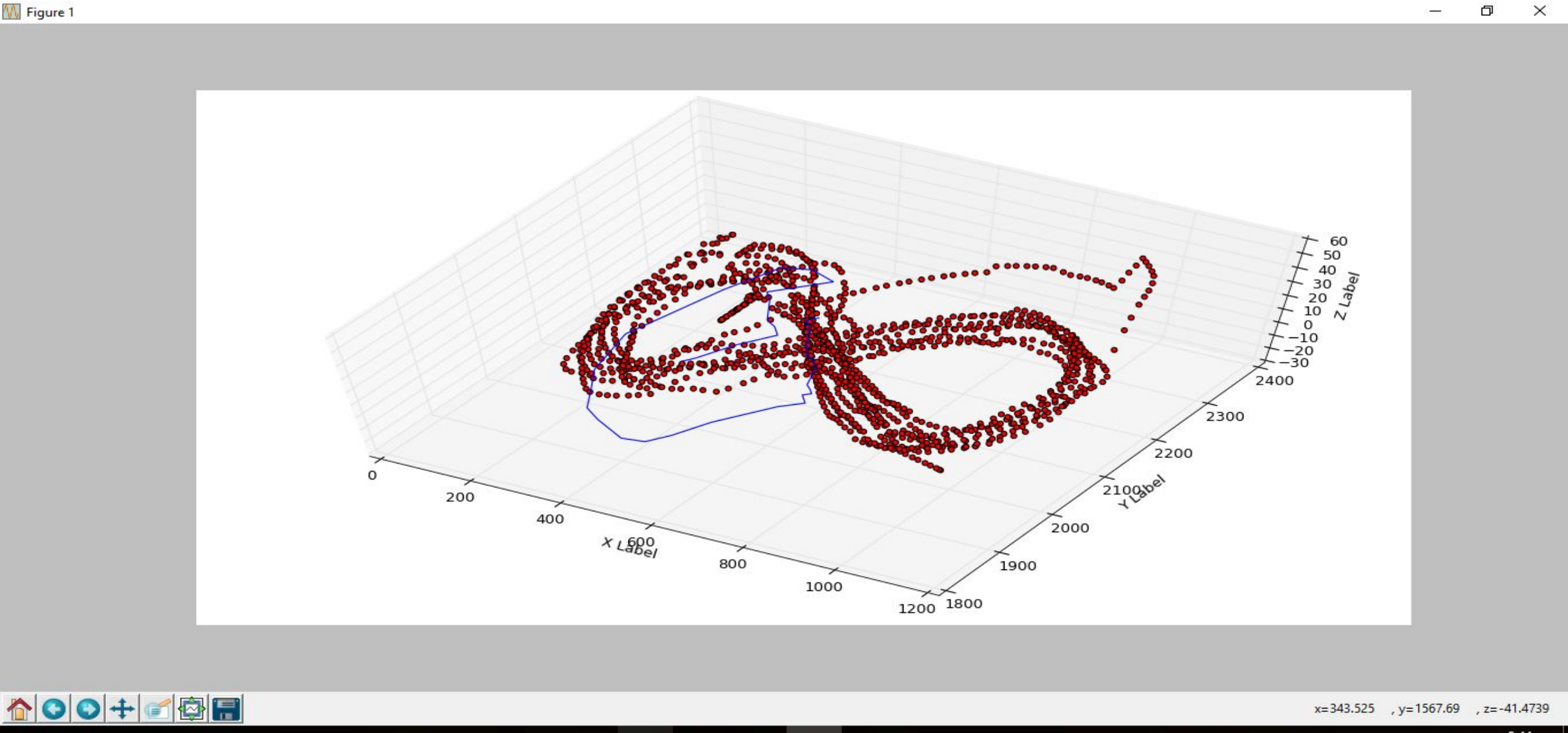
Task 3.4: Bayesian imitation learning



Task 3.4: Bayesian imitation learning



Task 3.4: Bayesian imitation learning



Thank you for attention