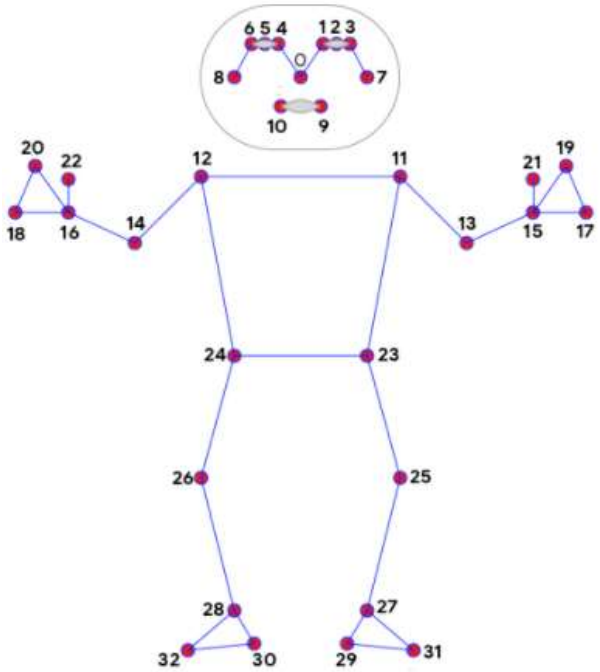
Annalysis on name calling cordinates

So the intestions are to

1. View the x,y,z differences with respect to time

2. Mesure the rate of change cordinates

Viewing the rate of change of cordinates



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Regions of intrests

0
1
2
3
4
5
6
7
8
9
10
11
12
23
24

In [ ]:

Import Depedencies

In [2]:
```
import pandas as pd
```

```
from matplotlib import pyplot as plt
```

In [3]:
```
# Obtainig data from children CSV file
sample_data_t1 = pd.read_csv('t1.csv')
```

In [4]:
```
sample_data_t1
```

Out[4]:

	x1	y1	z1	v1	x2	y2	z2	v2	x3	
0	0.510848	0.477077	-1.060948	0.999989	0.538799	0.447599	-1.028916	0.999962	0.556344	0.44
1	0.510828	0.474697	-0.873405	0.999989	0.539297	0.446556	-0.837205	0.999964	0.556925	0.44
2	0.510666	0.474901	-0.798744	0.999989	0.539291	0.448770	-0.759397	0.999966	0.556906	0.44
3	0.510511	0.476937	-0.811794	0.999988	0.539123	0.452102	-0.773349	0.999967	0.556754	0.45
4	0.510122	0.481119	-0.841685	0.999988	0.539008	0.457208	-0.802940	0.999968	0.556609	0.45
...	...	...	...	...	...	...	...	...	...	
324	0.533253	0.533242	-0.806675	0.999975	0.554080	0.507754	-0.771432	0.999939	0.570539	0.50
325	0.546073	0.530420	-0.851295	0.999977	0.563164	0.503419	-0.812670	0.999945	0.577913	0.50
326	0.558712	0.526548	-0.917379	0.999980	0.572090	0.499527	-0.877737	0.999951	0.585219	0.49
327	0.560395	0.522303	-0.906787	0.999982	0.575512	0.495566	-0.869038	0.999955	0.589245	0.49
328	0.559001	0.521114	-0.900645	0.999983	0.576437	0.494929	-0.852659	0.999959	0.590853	0.49

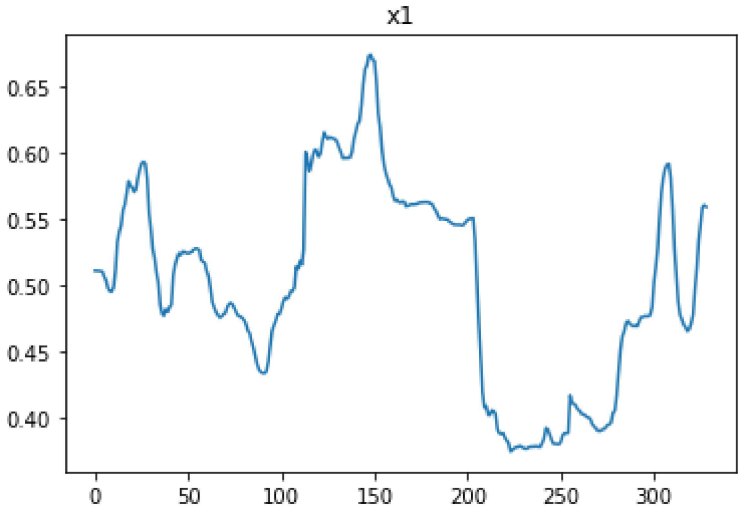
329 rows × 264 columns

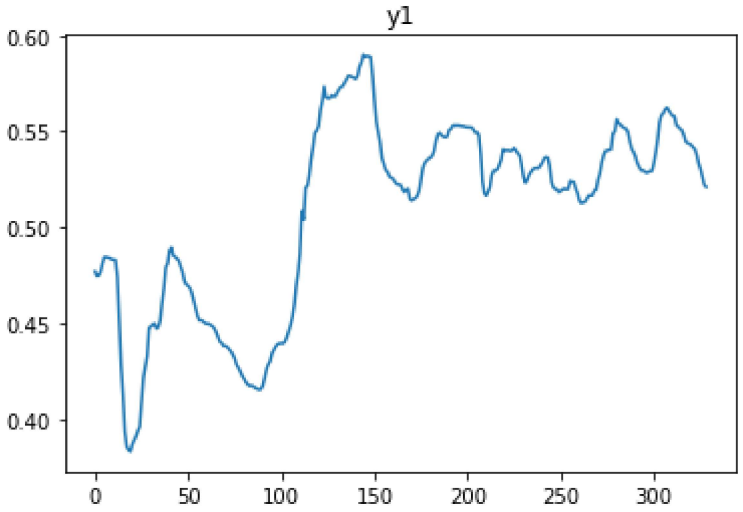
Annalysis

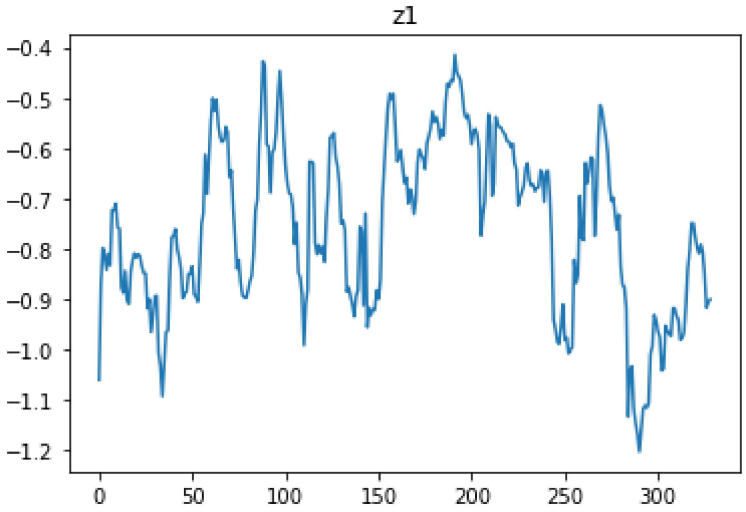
0 - Nose

xyz seperately

In [6]:
```
plt.plot(sample_data_t1.x1)
plt.title("x1")
plt.show()
plt.plot(sample_data_t1.y1)
plt.title("y1")
plt.show()
plt.plot(sample_data_t1.z1)
plt.title("z1")
plt.show()
# plt.legend(['x1', 'y1', 'z1'])
# plt.xlabel('frame')
# plt.show()
```







Measuring the rate of change of cordinates

In [9]:
```
data    = pd.DataFrame(data=sample_data_t1);
```

In [10]:
```
percentageChange_t1 = data.pct_change();
```

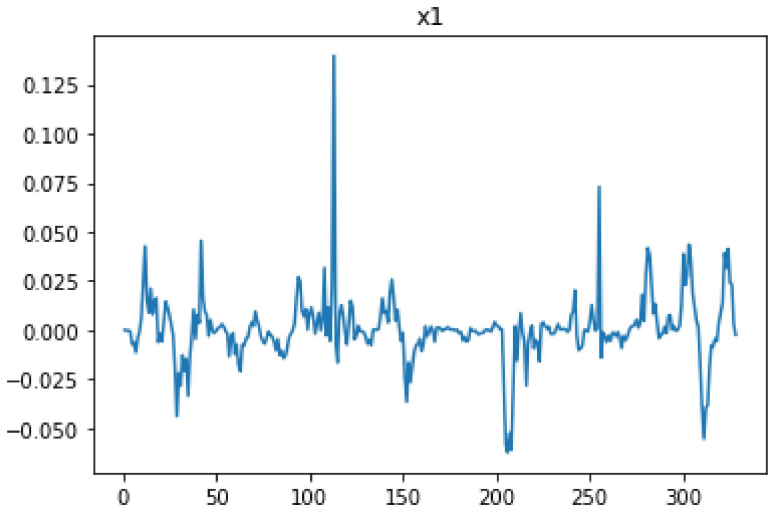
In [11]:
```
percentageChange_t1
```

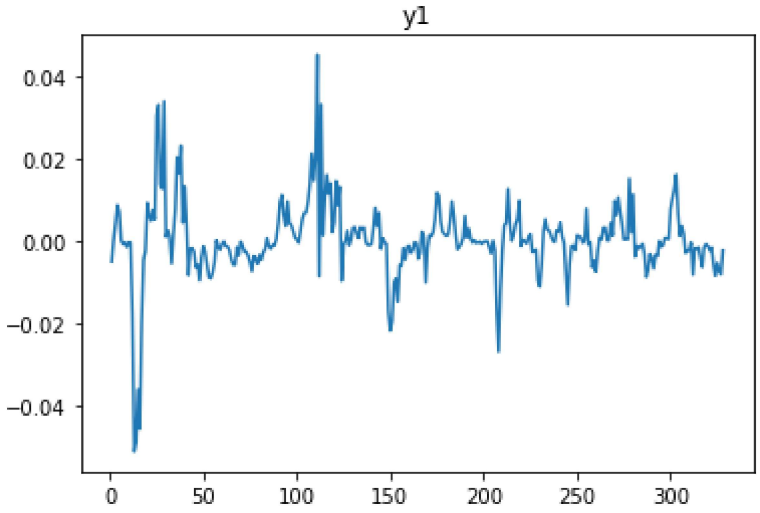
Out[11]:

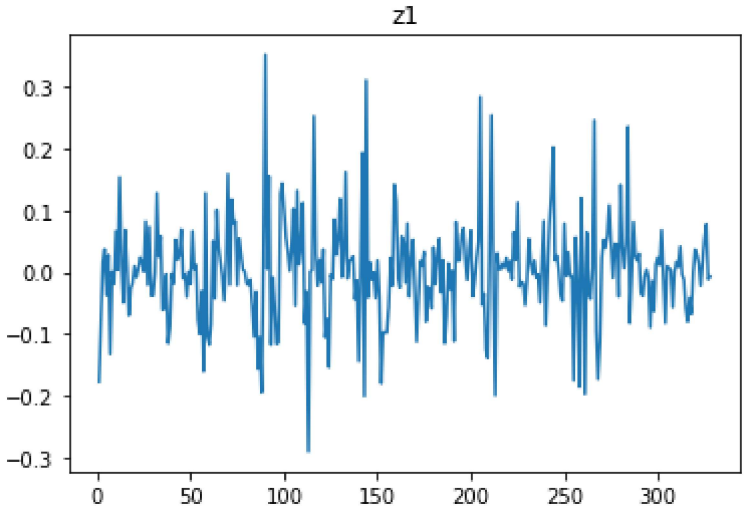
	x1	y1	z1	v1	x2	y2	z2	v2	x
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Nal
1	-0.000038	-0.004989	-0.176769	1.788160e-07	0.000923	-0.002330	-0.186323	0.000002	0.00104
2	-0.000318	0.000428	-0.085483	-2.980265e-07	-0.000010	0.004958	-0.092938	0.000002	-0.00003
3	-0.000303	0.004289	0.016339	-5.960533e-08	-0.000311	0.007424	0.018372	0.000002	-0.00027
4	-0.000762	0.008768	0.036820	-3.576320e-07	-0.000213	0.011293	0.038264	0.000001	-0.00026
...	...	...	...	...	...	...	...	...	.
324	0.041372	-0.008459	0.019160	2.622677e-06	0.030117	-0.012165	0.023726	0.000007	0.02303
325	0.024040	-0.005291	0.055314	2.443852e-06	0.016395	-0.008539	0.053456	0.000006	0.01292
326	0.023145	-0.007300	0.077627	2.205422e-06	0.015850	-0.007731	0.080066	0.000005	0.01264
327	0.003013	-0.008063	-0.011546	1.907388e-06	0.005981	-0.007929	-0.009911	0.000005	0.00687
328	-0.002487	-0.002276	-0.006773	1.668961e-06	0.001608	-0.001285	-0.018847	0.000004	0.00272

329 rows × 264 columns

In [12]:
```
plt.plot(percentageChange_t1.x1)
plt.title("x1")
plt.show()
plt.plot(percentageChange_t1.y1)
plt.title("y1")
plt.show()
plt.plot(percentageChange_t1.z1)
plt.title("z1")
plt.show()
```

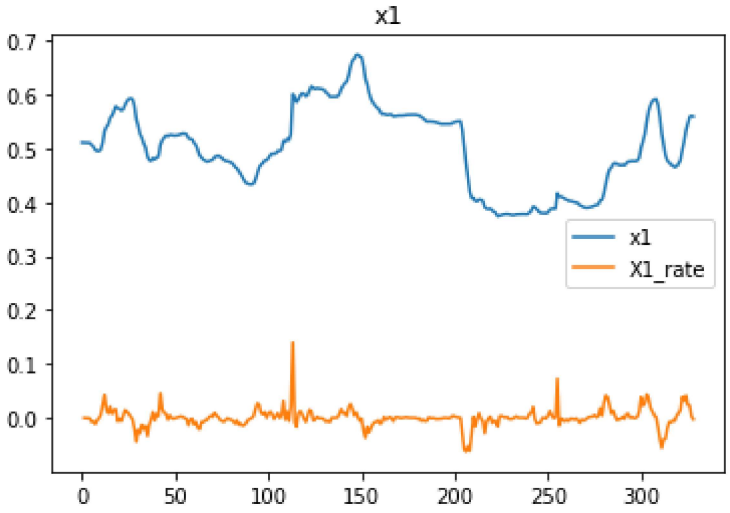


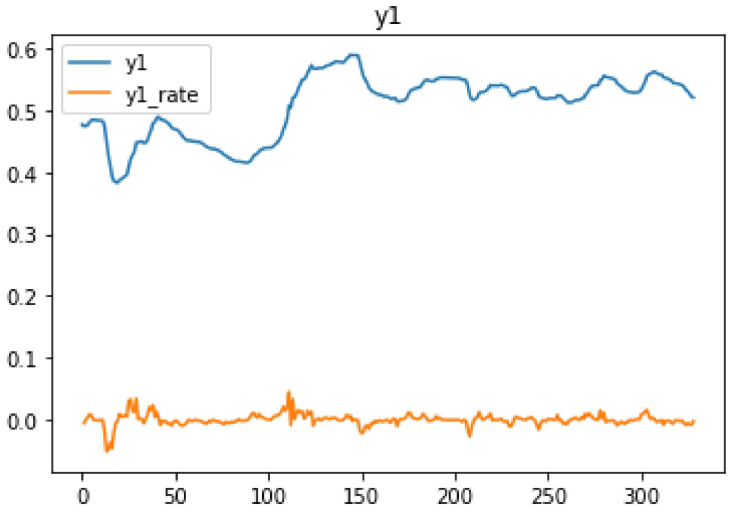


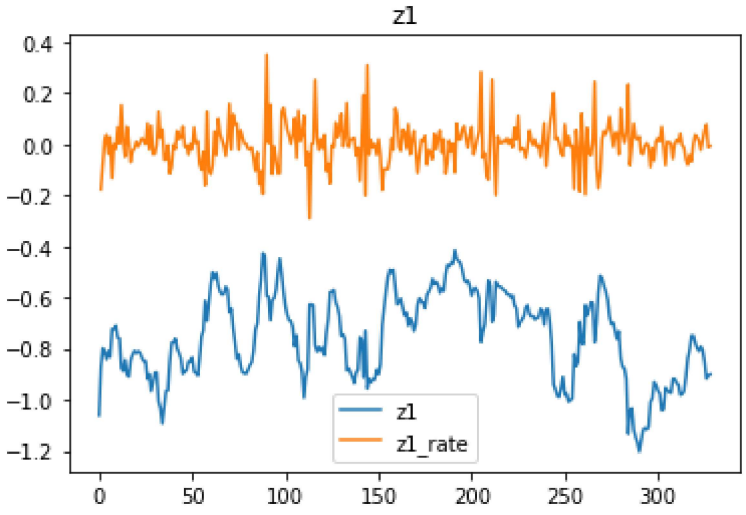


In [25]:

```
plt.plot(sample_data_t1.x1)
plt.plot(percentageChange_t1.x1)
plt.legend(["x1","X1_rate"])
plt.title("x1")
plt.show()


plt.plot(sample_data_t1.y1)
plt.plot(percentageChange_t1.y1)
plt.legend(["y1","y1_rate"])
plt.title("y1")
plt.show()


plt.plot(sample_data_t1.z1)
plt.plot(percentageChange_t1.z1)
plt.legend(["z1","z1_rate"])
plt.title("z1")
plt.show()
```







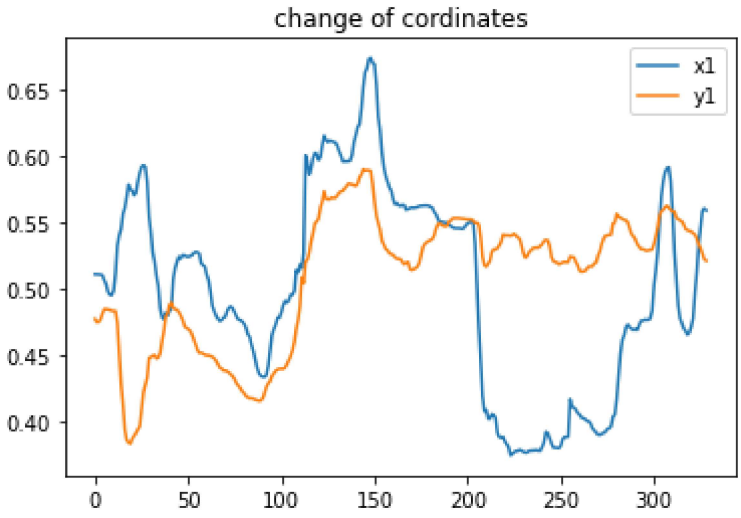
so there is spike visible in the rate of change of x1 cordinates

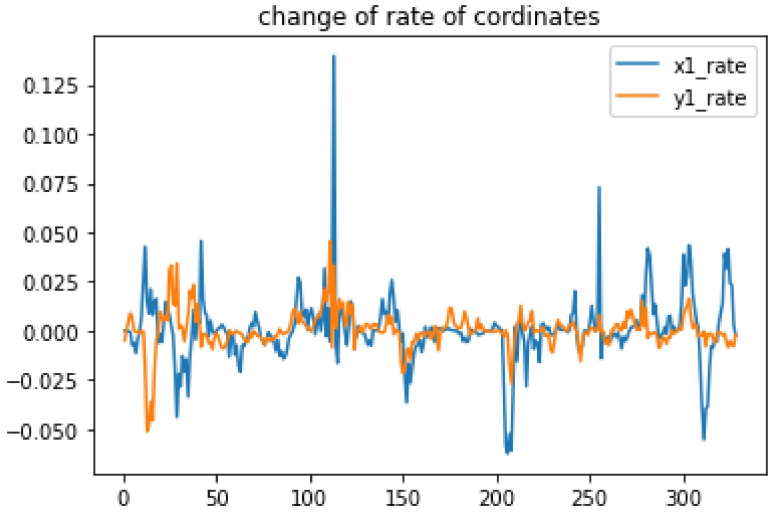
xyz in the same thing

In [24]:

```
plt.plot(sample_data_t1.x1)
plt.plot(sample_data_t1.y1)
plt.legend(['x1','y1'])
#plt.plot(sample_data_t1.z1)

plt.title("change of cordinates")
plt.show()

plt.plot(percentageChange_t1.x1)
```

```
plt.plot(percentageChange_t1.y1)
plt.legend(['x1_rate','y1_rate'])
#plt.plot(percentageChange_t1.z1)

plt.title("change of rate of cordinates")
plt.show()
```





Note : Nose

- z is too noisy to obtain the cordinates

- it seems x and y are correlated with each other for some good extent. Thus do a correlation annalysis and try to iddentify the correlation. If the correlation is high better to go with x cordinates as it seems that x cordinates are more sensitive. Thus it might produce some healthy results

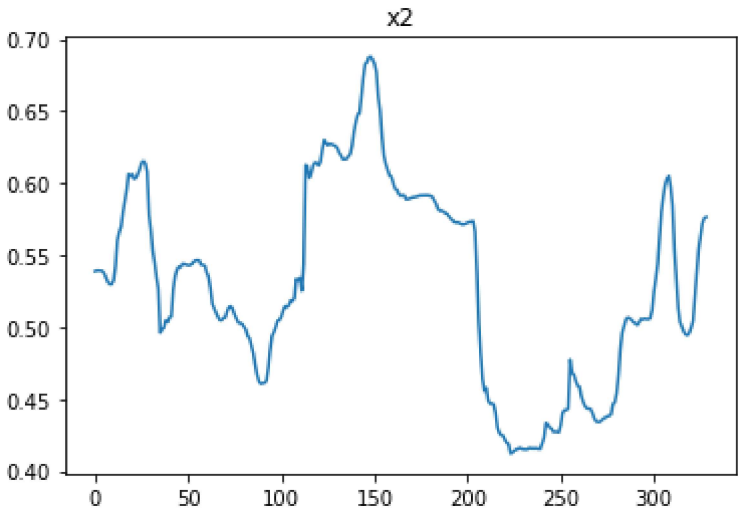
- It would be intresting to see what will happen if PCA is applied to the x and y cordinates. Try that as well
-

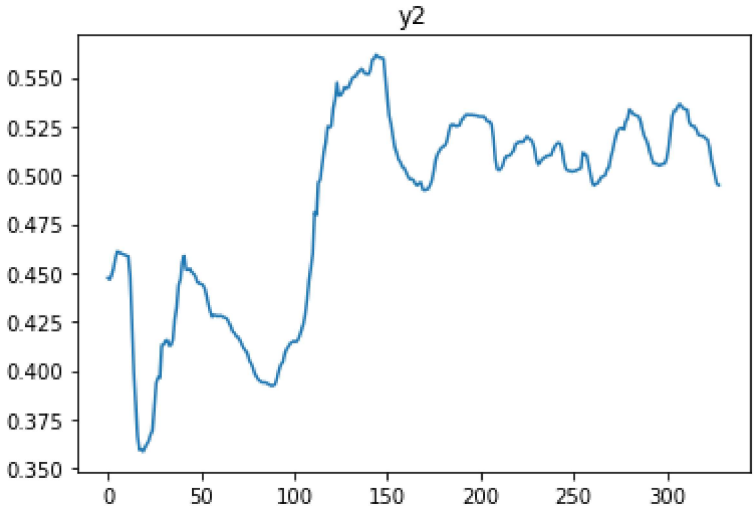
1 - Left eye inner

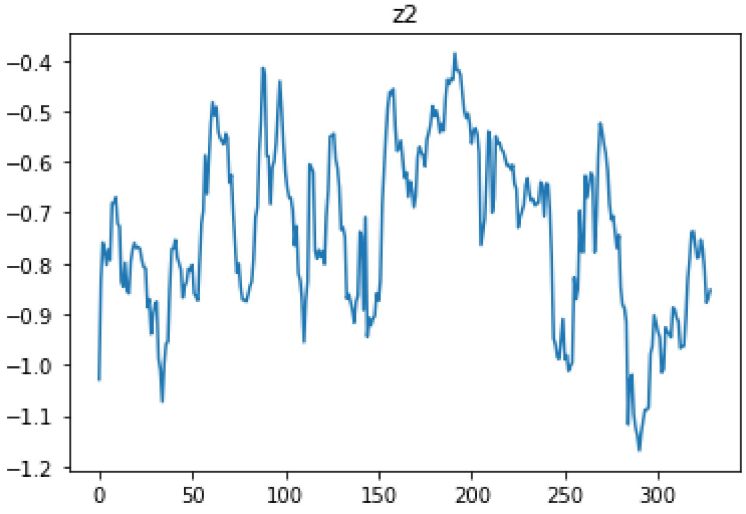
x,y,z seperately

In [27]:
```
plt.plot(sample_data_t1.x2)
```

```
plt.title("x2")
plt.show()
plt.plot(sample_data_t1.y2)
plt.title("y2")
plt.show()
plt.plot(sample_data_t1.z2)
plt.title("z2")
plt.show()
#plt.legend(['x1', 'y1', 'z1'])
# plt.xlabel('frame')
# plt.show()
```

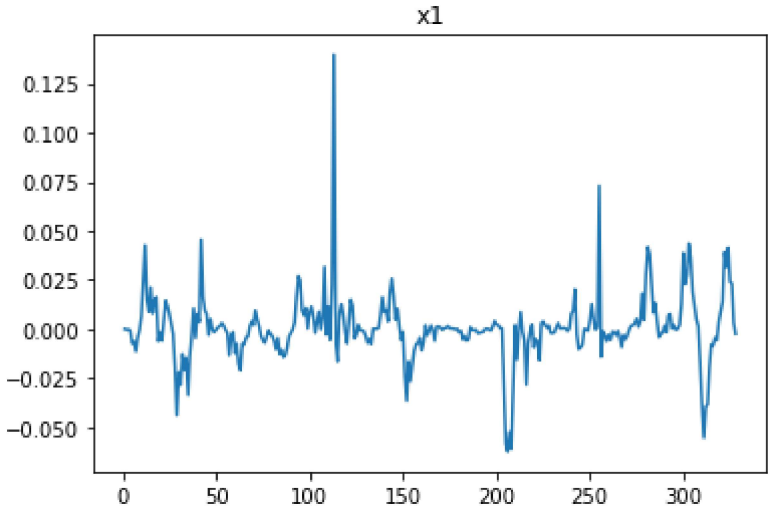


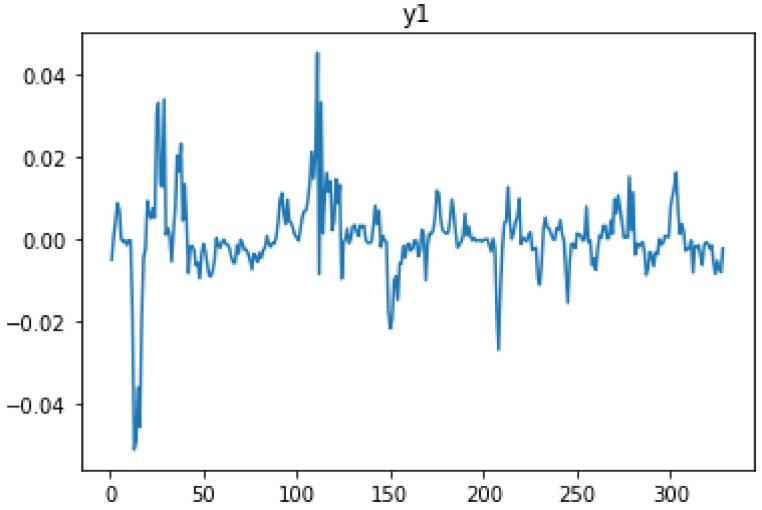


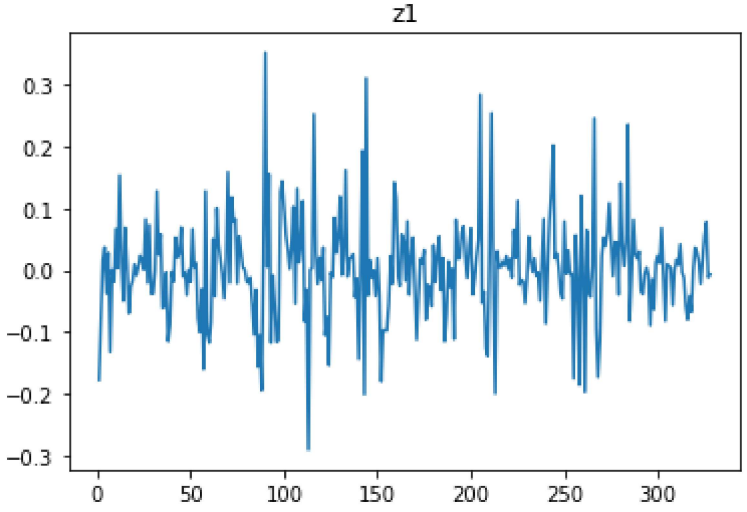


rate of change of x,y,z

In [28]:
```
plt.plot(percentageChange_t1.x1)
plt.title("x1")
plt.show()
plt.plot(percentageChange_t1.y1)
plt.title("y1")
plt.show()
plt.plot(percentageChange_t1.z1)
plt.title("z1")
plt.show()
```

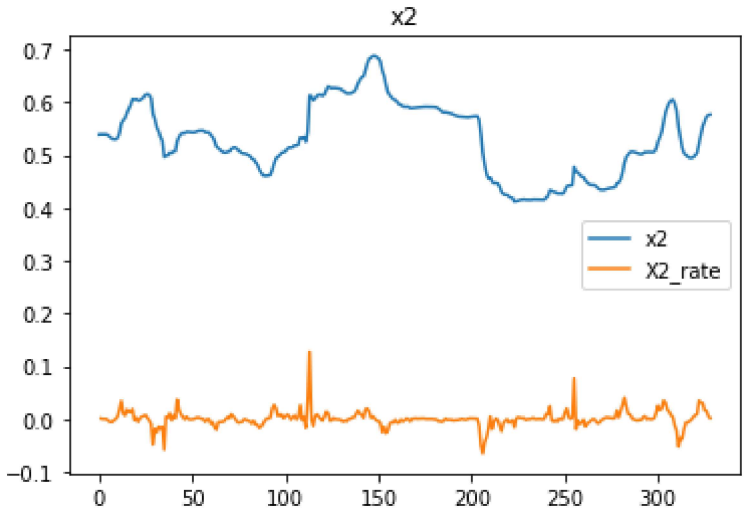


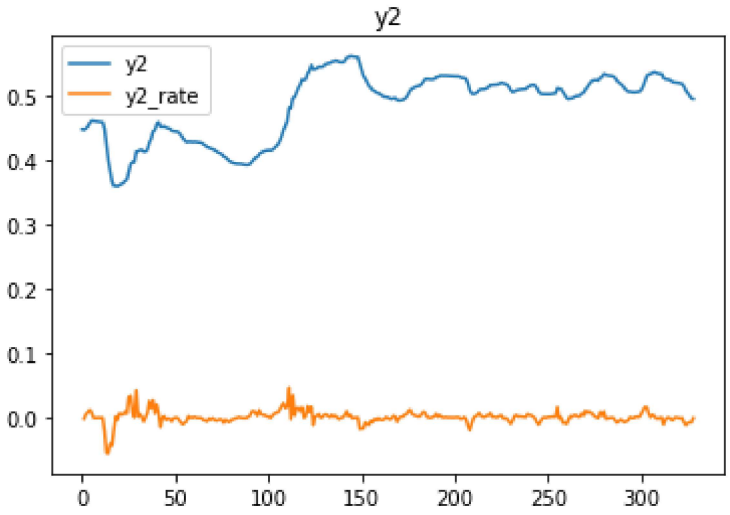


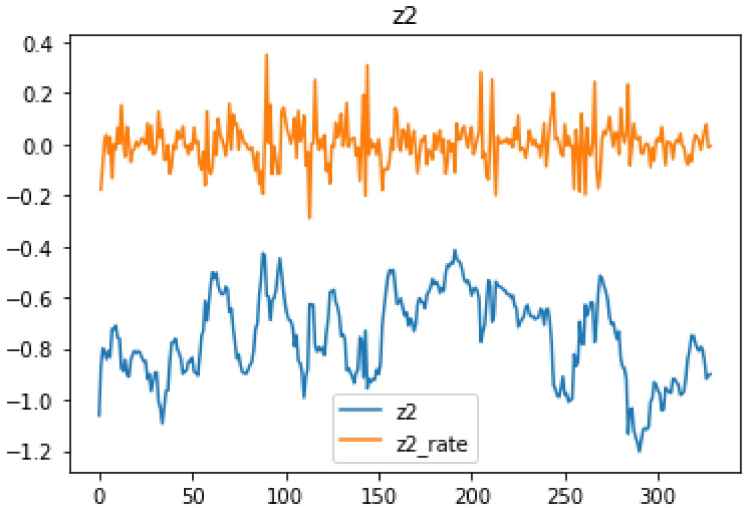


in the same plot (change and the rate of change)

In [30]:
```
plt.plot(sample_data_t1.x2)
plt.plot(percentageChange_t1.x2)
plt.legend(["x2","X2_rate"])
plt.title("x2")
plt.show()


plt.plot(sample_data_t1.y2)
plt.plot(percentageChange_t1.y2)
plt.legend(["y2","y2_rate"])
plt.title("y2")
plt.show()


plt.plot(sample_data_t1.z1)
plt.plot(percentageChange_t1.z1)
plt.legend(["z2","z2_rate"])
plt.title("z2")
plt.show()
```



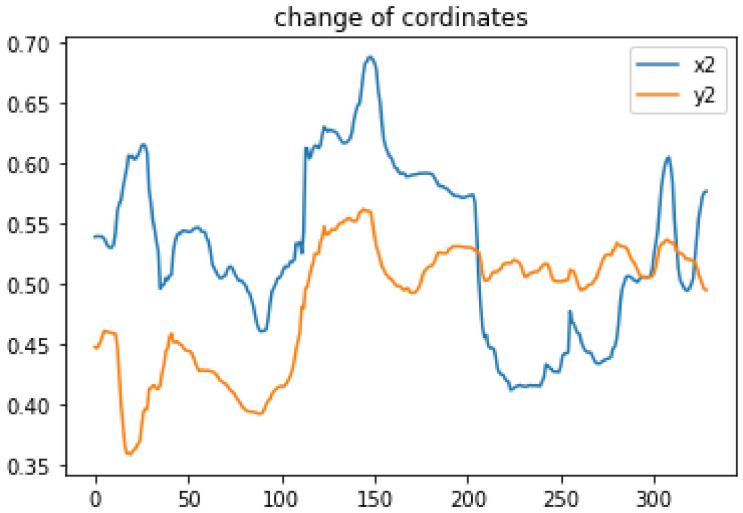


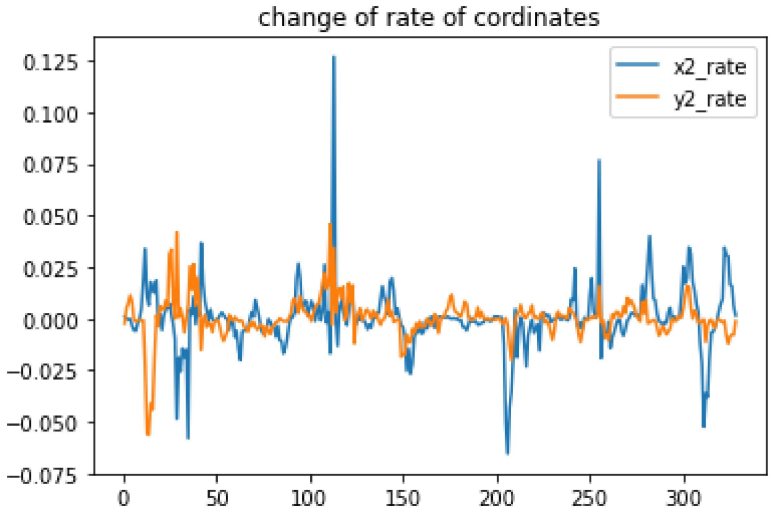


rate of chnage and the change in the same plot as x,y,z

In [31]:

```
plt.plot(sample_data_t1.x2)
plt.plot(sample_data_t1.y2)
plt.legend(['x2','y2'])
#plt.plot(sample_data_t1.z1)

plt.title("change of cordinates")
plt.show()

plt.plot(percentageChange_t1.x2)
plt.plot(percentageChange_t1.y2)
plt.legend(['x2_rate','y2_rate'])
#plt.plot(percentageChange_t1.z1)

plt.title("change of rate of cordinates")
plt.show()
```





Note : Left eye inner

Same observations made on nose can be made on this as well

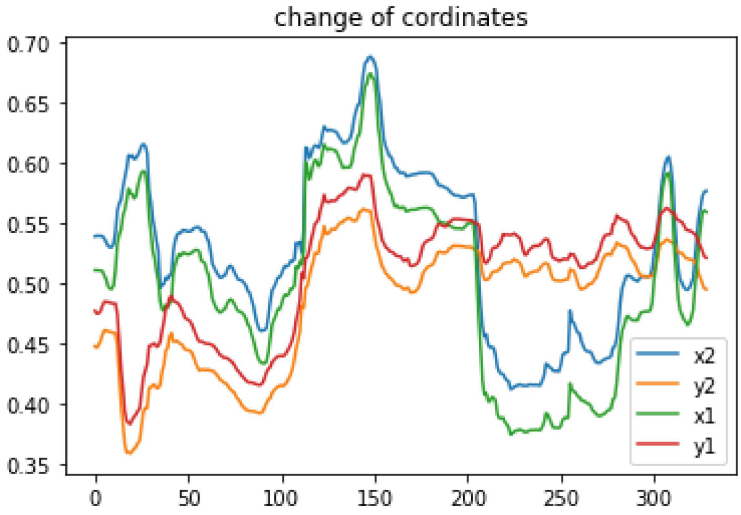
it would be intresting to allign the x cordindates of nose and left eye inner

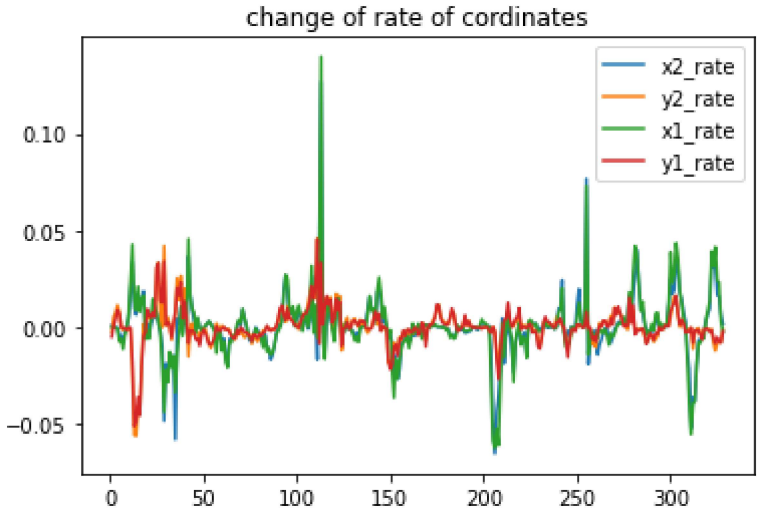
lets do that now

R1- Alligning nose and left inner eye cordinates

In [34]:

```
plt.plot(sample_data_t1.x2)
plt.plot(sample_data_t1.y2)
plt.plot(sample_data_t1.x1)
plt.plot(sample_data_t1.y1)
plt.legend(['x2','y2','x1','y1'])
#plt.plot(sample_data_t1.z1)

plt.title("change of cordinates")
plt.show()

plt.plot(percentageChange_t1.x2)
plt.plot(percentageChange_t1.y2)
plt.plot(percentageChange_t1.x1)
plt.plot(percentageChange_t1.y1)
plt.legend(['x2_rate','y2_rate','x1_rate','y1_rate'])
#plt.plot(percentageChange_t1.z1)

plt.title("change of rate of cordinates")
plt.show()
```





R1 - Note

Cordinates seems correlated

Nose seems to have a good deviation. Btw annalysing individual points may not be usefull . First of all see whether the cordinates are corrleated . If that is the case, no point of annalysing individual points

R2 - Annalysing cordiantes of ROI

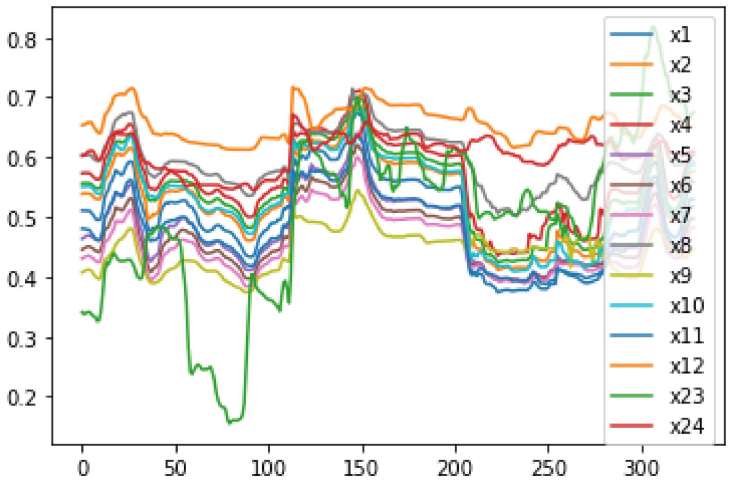
x

In [ ]:

In [36]:
```
plt.plot(sample_data_t1.x1)
plt.plot(sample_data_t1.x2)
plt.plot(sample_data_t1.x3)
plt.plot(sample_data_t1.x4)
plt.plot(sample_data_t1.x5)
plt.plot(sample_data_t1.x6)
plt.plot(sample_data_t1.x7)
plt.plot(sample_data_t1.x8)
plt.plot(sample_data_t1.x9)
plt.plot(sample_data_t1.x10)
plt.plot(sample_data_t1.x11)
plt.plot(sample_data_t1.x12)
```

```
plt.plot(sample_data_t1.x23)
plt.plot(sample_data_t1.x24)
plt.legend(['x1','x2','x3','x4','x5','x6','x7','x8','x9','x10','x11','x12','x23','x2
#plt.plot(sample_data_t1.z1)
```

Out[36]: <matplotlib.legend.Legend at 0x20eb04604c8>

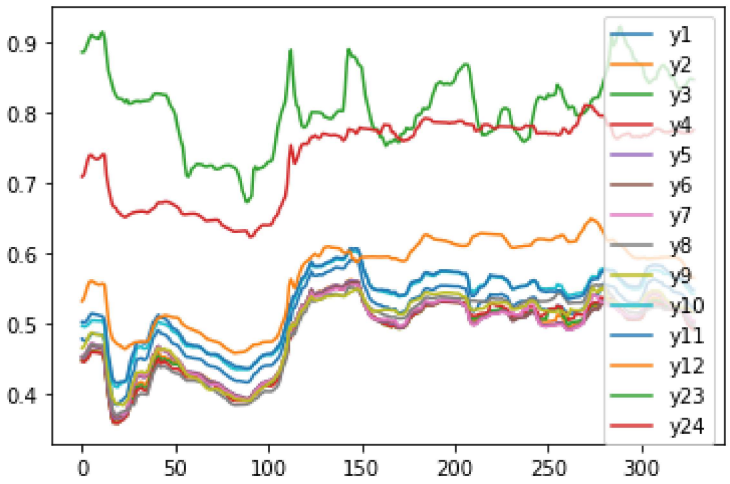


y cordinates

In [37]:

```
plt.plot(sample_data_t1.y1)
plt.plot(sample_data_t1.y2)
plt.plot(sample_data_t1.y3)
plt.plot(sample_data_t1.y4)
plt.plot(sample_data_t1.y5)
plt.plot(sample_data_t1.y6)
plt.plot(sample_data_t1.y7)
plt.plot(sample_data_t1.y8)
plt.plot(sample_data_t1.y9)
plt.plot(sample_data_t1.y10)
plt.plot(sample_data_t1.y11)
plt.plot(sample_data_t1.y12)
plt.plot(sample_data_t1.y23)
plt.plot(sample_data_t1.y24)
plt.legend(['y1','y2','y3','y4','y5','y6','y7','y8','y9','y10','y11','y12','y23','y2
#plt.plot(sample_data_t1.z1)
```

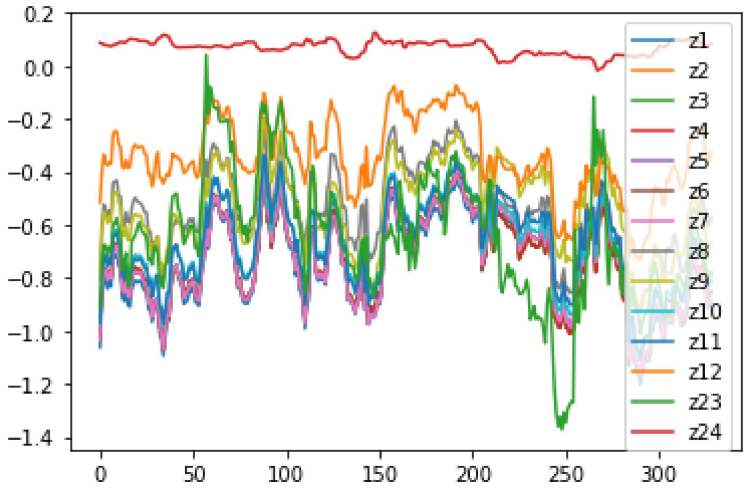
Out[37]: <matplotlib.legend.Legend at 0x20eb048e688>



Z cordinates

In [38]:
```
plt.plot(sample_data_t1.z1)
plt.plot(sample_data_t1.z2)
plt.plot(sample_data_t1.z3)
plt.plot(sample_data_t1.z4)
plt.plot(sample_data_t1.z5)
plt.plot(sample_data_t1.z6)
plt.plot(sample_data_t1.z7)
plt.plot(sample_data_t1.z8)
plt.plot(sample_data_t1.z9)
plt.plot(sample_data_t1.z10)
plt.plot(sample_data_t1.z11)
plt.plot(sample_data_t1.z12)
plt.plot(sample_data_t1.z23)
plt.plot(sample_data_t1.z24)
plt.legend(['z1','z2','z3','z4','z5','z6','z7','z8','z9','z10','z11','z12','z23','z2
#plt.plot(sample_data_t1.z1)
```

Out[38]:
```
<matplotlib.legend.Legend at 0x20eb041b988>
```



R2 : Note

Seems most of the cordiantes are correlated Do a correlation annalysis on X cordinates and see whether there is any changes

23, and 24 are showing some serious divisions. This might be becuase of issues in predictions

R3 - Corrleation matrix

For x cordinates

In [49]:
```
## Import dependencies
import pandas as pd
import seaborn as sn
import matplotlib.pyplot as plt
```

In [50]:
```
# obtain the data from the x matrix
t1_x_corr = pd.read_csv('t1_x_corr.csv')
```

In [51]:
```
t1_x_corr
```

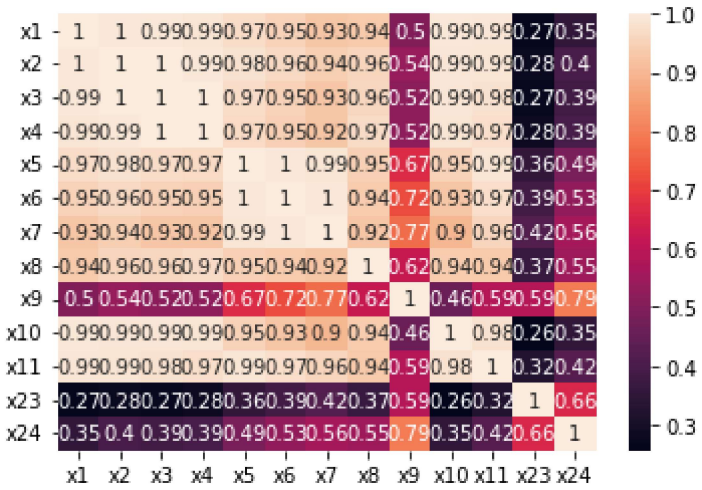
Out[51]:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x
0	0.510848	0.538799	0.556344	0.573132	0.463667	0.445807	0.430604	0.603595	0.407653	0.5507
1	0.510828	0.539297	0.556925	0.573757	0.465914	0.447802	0.432727	0.604306	0.409432	0.5509
2	0.510666	0.539291	0.556906	0.573700	0.467854	0.449528	0.434126	0.604298	0.410990	0.5509
3	0.510511	0.539123	0.556754	0.573499	0.469166	0.450825	0.435197	0.604293	0.411987	0.5512
4	0.510122	0.539008	0.556609	0.573243	0.469106	0.450802	0.435111	0.604001	0.411626	0.5515
...	...	...	...	...	...	...	...	...	...	
324	0.533253	0.554080	0.570539	0.583086	0.492414	0.475461	0.459676	0.593443	0.432815	0.5691
325	0.546073	0.563164	0.577913	0.589904	0.502134	0.485282	0.471097	0.595247	0.435985	0.5772
326	0.558712	0.572090	0.585219	0.596923	0.514783	0.494508	0.479365	0.598862	0.440348	0.5871
327	0.560395	0.575512	0.589245	0.601762	0.518200	0.497345	0.481924	0.604753	0.443579	0.5887
328	0.559001	0.576437	0.590853	0.604080	0.517706	0.497127	0.482038	0.608805	0.446233	0.5869

329 rows × 13 columns

In [52]:
```
corrMatrix = t1_x_corr.corr()
sn.heatmap(corrMatrix, annot=True)
plt.show()
```



R3 - Note

Most of the varaibles are corrleated with each other. but 3 of the variables are showing some deviations. can understand why 23 and 24 are not corrleated with each other. Most probably because of issues with estimatations. But it is intresting see why 9 (left mouth is not correlated).

Obtain the corrleation matrix for y cordinates and see

In [54]:
```
# obtain the data from the x matrix
t1_y_corr = pd.read_csv('t1_y_corr.csv')
```

In [55]:
```
t1_y_corr
```

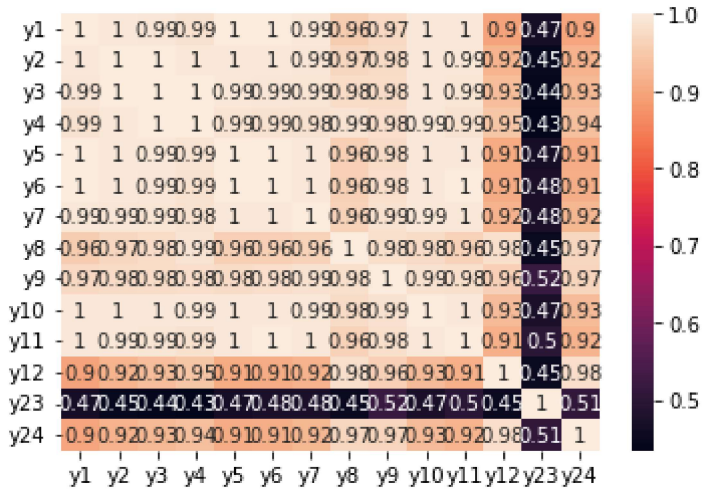
Out[55]:

	y1	y2	y3	y4	y5	y6	y7	y8	y9	y
0	0.477077	0.447599	0.446548	0.445532	0.449818	0.451001	0.452257	0.452284	0.464465	0.495
1	0.474697	0.446556	0.445295	0.444398	0.450304	0.452165	0.453941	0.452491	0.466420	0.494
2	0.474901	0.448770	0.447526	0.446594	0.453209	0.455208	0.457267	0.456890	0.471929	0.495
3	0.476937	0.452102	0.450804	0.449922	0.457626	0.460033	0.462638	0.461478	0.478325	0.497
4	0.481119	0.457208	0.455830	0.454937	0.463612	0.466025	0.468598	0.466345	0.483886	0.500
...	...	...	...	...	...	...	...	...	...	
324	0.533242	0.507754	0.505637	0.503854	0.511573	0.512274	0.512985	0.507515	0.520423	0.551
325	0.530420	0.503419	0.500809	0.498467	0.508781	0.509949	0.511259	0.502284	0.520517	0.548
326	0.526548	0.499527	0.497232	0.495064	0.505323	0.506954	0.508630	0.499884	0.520000	0.545
327	0.522303	0.495566	0.493463	0.491459	0.501037	0.502576	0.504099	0.497448	0.516707	0.542
328	0.521114	0.494929	0.492750	0.490670	0.500393	0.501762	0.503135	0.497176	0.515102	0.541

329 rows × 14 columns

In [56]:

```
corrMatrix_t1_y = t1_y_corr.corr()
sn.heatmap(corrMatrix_t1_y , annot=True)
plt.show()
```



R3 - Note

So when it comes to y cordinates mouth does not shows a corrleation.

Idiot, it has to show some corrleation nea. Mokada wela thiyennea

In [ ]: