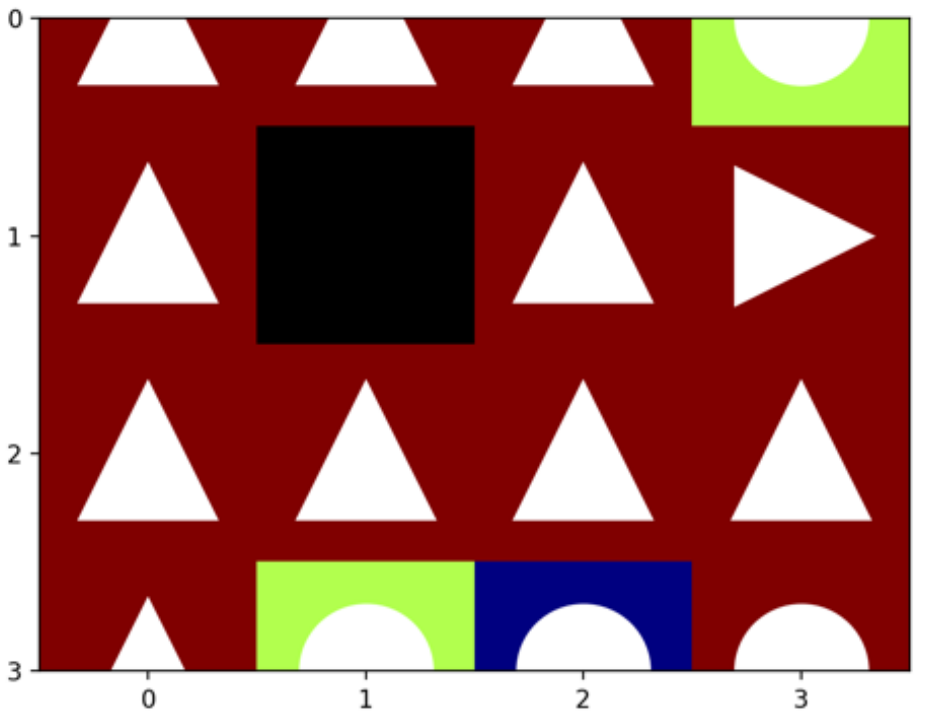
Report

- **With VI and PI algorithms with r:0, d:1 and p:1.**

Optimal Policy(R: Right, D: Down, Up: U, Left: L)

U	U	U	1
U	O	U	R
U	U	U	U
U	1	-10	10



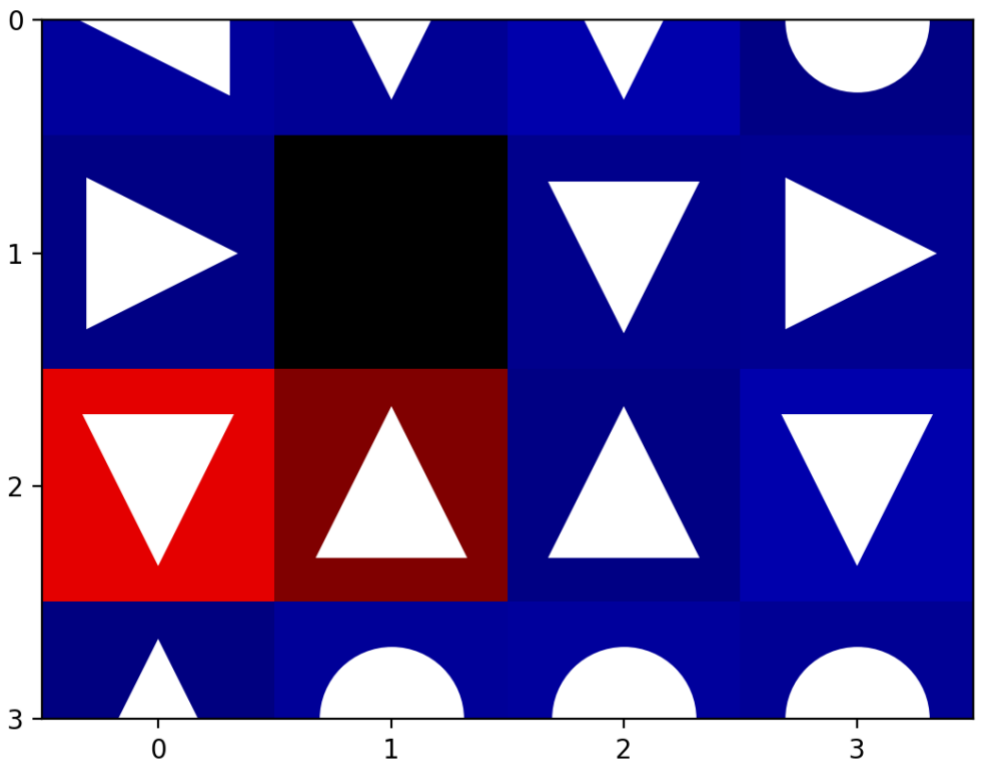
- **With e:0, a:0.1 and N:1000, after Q-learning**

Q Values

	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10
up	-0.00370	-0.42667	-0.19263	-0.45323	-0.23899	-0.71781	-0.00677	0.16465	0.07629	0.07656	0.03329
right	-0.15827	0.26821	0.47766	-0.25395	0.18048	0.11087	1.12323	0.06534	-0.56466	0.47398	-0.54096
down	-0.08628	-0.45804	0.03112	0.12528	-0.29533	0.20481	-0.42635	1.24910	-0.15884	-0.04111	-0.17039
left	0.33702	0.07212	0.12953	-0.10945	-0.30821	-0.17928	-0.35318	0.16465	-0.15488	0.04569	0.00715

Optimal Policy (R: Right, D: Down, Up: U, Left: L)

L	**D**	**D**	**1**
R	**O**	**D**	**R**
D	**U**	**U**	**D**
U	**1**	**-10**	**10**



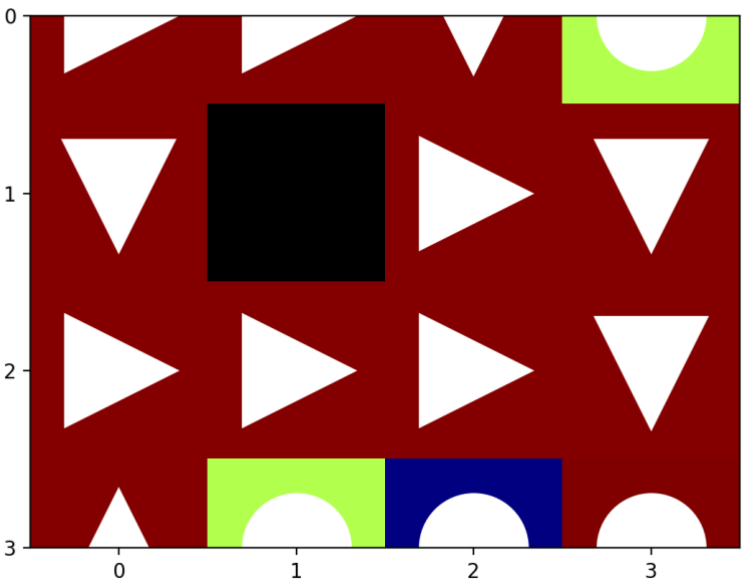
Increasing N does not change value because it is already converge approximately at 20th iteration.

a. r:-0.01.

For VI and PI policy and values are change it is getting better that r = 0. Because after that for each action it is getting -0.01 reward, because of this it need to reach finish point with less action.

Optimal Policy (R: Right, D: Down, Up: U, Left: L)

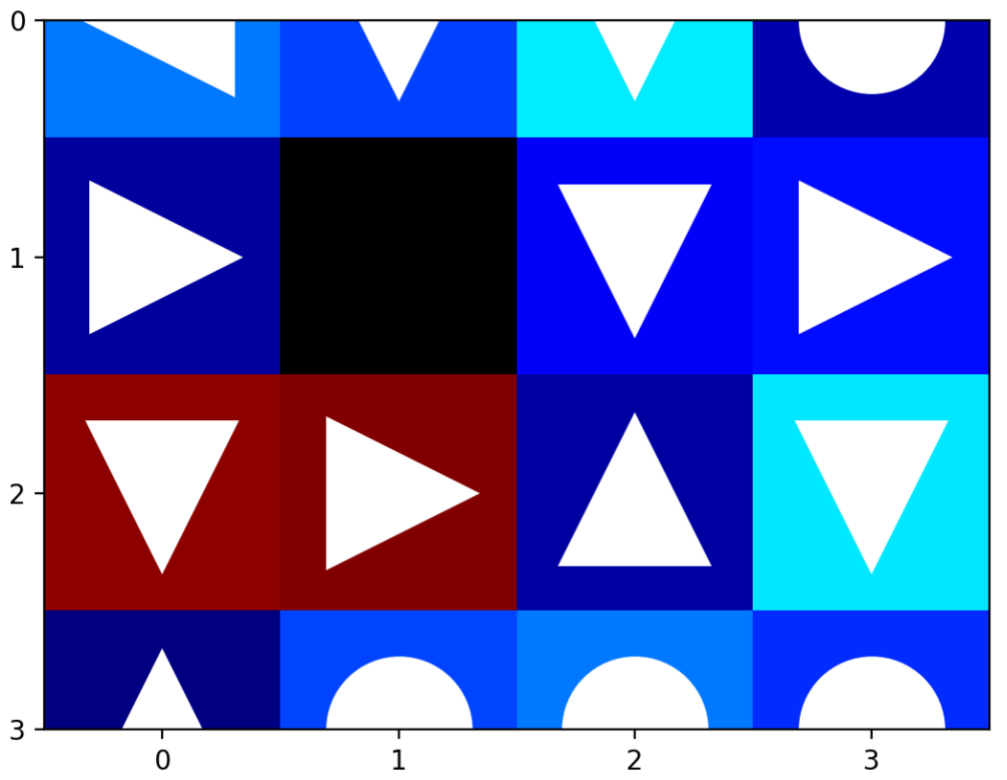
R	**R**	**D**	**1**
D	**O**	**R**	**D**
R	**R**	**R**	**D**
U	**1**	**-10**	**10**



For Q-Learning values converge later.(Approximately = 40)
And Optimal Policy change because of reward is does not change (discount factor = 1) so after many iteration reward will greater than gained reward so it must reach finish point sooner.

Optimal Policy (R: Right, D: Down, Up: U, Left: L)

L	D	D	1
R	O	D	R
D	R	U	D
U	1	-10	10



b. Update the discount factor as d:0.2.

It does not change optimal policy but it change convergence point it getting sooner because since the whole algorithm is about

making decisions where the outcome partly depends on random inputs which can drift away over time,

invalidating initial decision, it makes sense to prefer decisions which a better as short-term solutions.

c. Change the discount factor back to d:1. Update rewards as r:5. Again comment on the results of VI and PI.

- For VI getting if we say reward for each action is 5 and never decrease, it want to take highest reward and some finish reward is already fewer than action reward so it want to make action instead of reaching finish.

  Optimal Policy is same as r = 1 because of same reason.

- For PI, policy does not updated after actions with same reason with VI.

  It is same with how it was created.

d. VI and PI with d:1, r:-0.01 and p:0.5.

With this changes agent can move different directions with probabilities

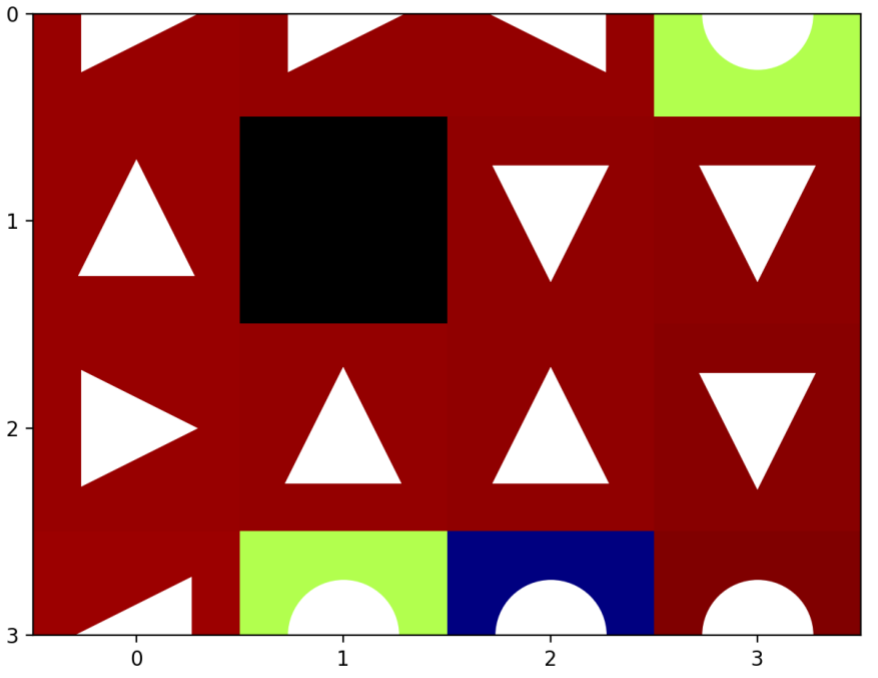
up: 0.5 ,

left and right: 0.25 ;

so it changes optimal policy.

Optimal Policy (R: Right, D: Down, Up: U, Left: L)

R	**R**	**L**	**1**
U	**O**	**D**	**D**
R	**U**	**U**	**D**
L	**1**	**-10**	**10**



e. d:0.9, r:-0.01, p:0.8 for VI and PI.

Utility Values For VI

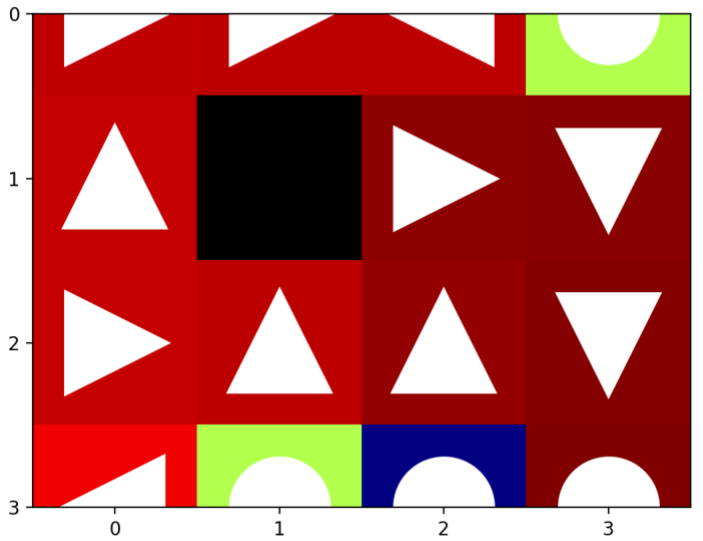
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010
-0.020	-0.020	0.788	-0.020	-0.020	0.788	-0.020	0.788	-0.020	-0.020	-0.020
-0.030	0.616	0.867	-0.030	0.697	6.457	0.697	0.786	5.378	-0.030	0.616
0.477	0.807	0.946	0.542	5.780	7.735	0.832	4.471	6.089	0.477	0.807
0.737	0.908	4.795	0.764	6.881	8.867	3.716	5.408	7.093	0.737	0.908
0.867	4.008	5.686	3.115	8.272	9.196	4.490	6.305	7.310	0.867	4.008
3.594	5.340	7.109	4.205	8.647	9.459	5.662	6.568	8.204	3.594	5.340
5.042	6.745	7.541	5.360	9.089	9.550	6.065	7.310	8.533	5.042	6.745
6.426	7.372	8.036	5.914	9.237	9.677	6.876	7.647	8.970	6.426	7.372
7.122	7.893	8.217	6.673	9.432	9.738	7.244	8.031	9.127	7.122	7.893
7.684	8.142	8.425	7.120	9.515	9.802	7.695	8.194	9.326	7.684	8.142
7.984	8.358	8.516	7.570	9.607	9.833	7.907	8.370	9.410	7.984	8.358
8.232	8.475	8.611	7.891	9.649	9.863	8.133	8.455	9.504	8.232	8.475
8.382	8.574	8.657	8.154	9.692	9.879	8.254	8.539	9.547	8.382	8.574
8.503	8.630	8.701	8.326	9.714	9.893	8.367	8.601	9.591	8.503	8.630
8.577	8.677	8.736	8.458	9.734	9.901	8.446	8.667	9.615	8.577	8.677
8.635	8.714	8.779	8.543	9.746	9.908	8.522	8.729	9.638	8.635	8.714
8.679	8.756	8.814	8.607	9.758	9.912	8.589	8.789	9.654	8.679	8.756
8.723	8.792	8.852	8.655	9.766	9.916	8.651	8.846	9.670	8.723	8.792
8.761	8.830	8.885	8.699	9.775	9.918	8.709	8.899	9.683	8.761	8.830

Utility Values For PI

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010	-0.010
-0.030	0.616	0.867	-0.030	0.690	0.867	0.616	0.581	0.698	-0.030	0.616
0.731	0.908	0.969	0.769	5.983	8.356	0.923	4.650	6.689	0.731	0.908
0.922	4.153	6.458	3.274	8.607	9.448	5.140	6.461	8.483	0.922	4.153
5.628	7.100	8.034	5.674	9.352	9.743	7.039	7.644	9.218	5.628	7.100
7.615	8.183	8.516	7.433	9.616	9.854	7.950	8.293	9.502	7.615	8.183
8.329	8.558	8.686	8.221	9.714	9.896	8.345	8.592	9.614	8.329	8.558
8.590	8.694	8.790	8.524	9.756	9.913	8.526	8.731	9.662	8.590	8.694
8.721	8.782	8.872	8.680	9.778	9.921	8.673	8.859	9.693	8.721	8.782
8.811	8.863	8.947	8.779	9.793	9.925	8.789	8.969	9.717	8.811	8.863
8.890	8.938	9.015	8.861	9.805	9.929	8.894	9.064	9.736	8.890	8.938
8.961	9.006	9.078	8.934	9.816	9.932	8.986	9.145	9.753	8.961	9.006
9.028	9.068	9.134	9.005	9.826	9.935	9.067	9.216	9.769	9.028	9.068
9.088	9.124	9.186	9.081	9.835	9.938	9.140	9.278	9.782	9.088	9.124
9.144	9.176	9.234	9.150	9.843	9.940	9.205	9.332	9.794	9.144	9.176
9.194	9.223	9.277	9.213	9.850	9.942	9.262	9.380	9.804	9.194	9.223
9.240	9.266	9.316	9.268	9.856	9.944	9.313	9.422	9.814	9.240	9.266
9.282	9.305	9.352	9.317	9.862	9.945	9.359	9.460	9.823	9.282	9.305
9.326	9.341	9.385	9.360	9.868	9.947	9.399	9.493	9.830	9.326	9.341
9.366	9.374	9.415	9.399	9.872	9.948	9.435	9.522	9.837	9.366	9.374

Final Policy (R: Right, D: Down, Up: U, Left: L)

R	**R**	**L**	**1**
U	**O**	**R**	**D**
R	**U**	**U**	**D**
L	**1**	**-10**	**10**



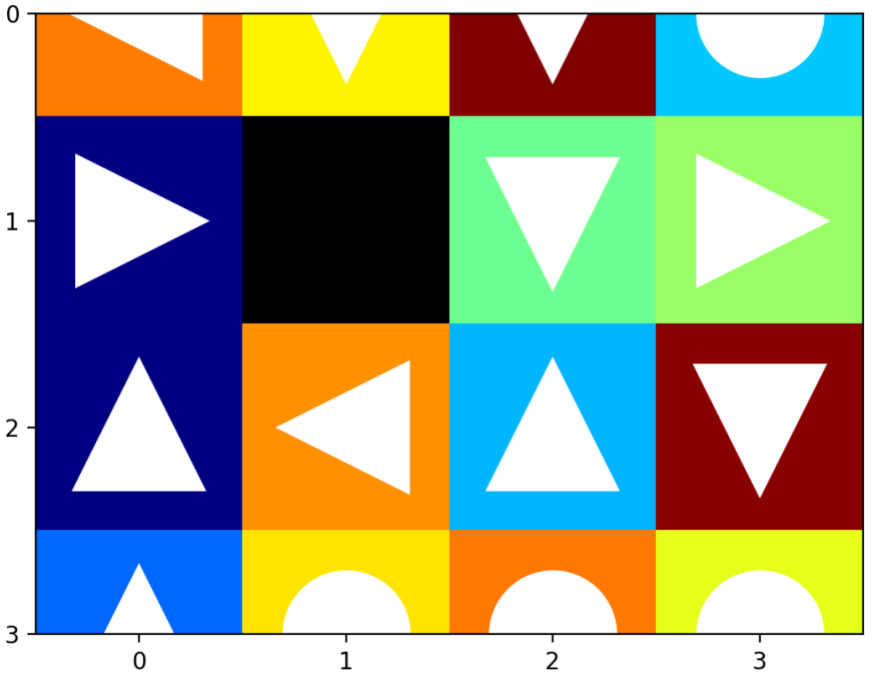
For Q-Learning

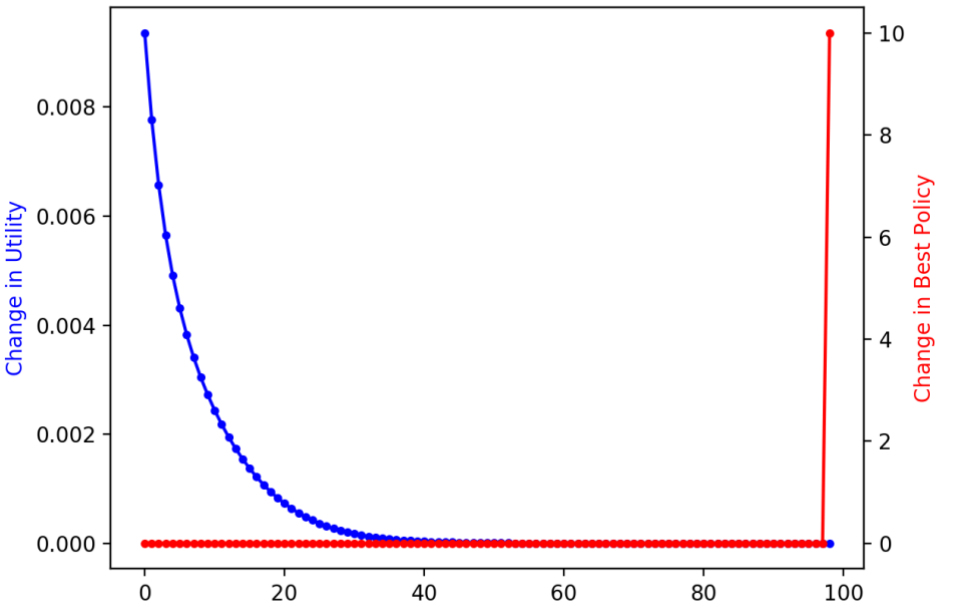
Q-Values

	s0	**s1**	**s2**	**s3**	**s4**	**s5**	**s6**	**s7**	**s8**	**s9**	**s10**
up	-0.00370	-0.42667	-0.19263	-0.45323	-0.23899	-0.71781	0.13327	0.17465	0.07629	0.07656	0.03329
right	-0.15827	0.26821	0.47766	-0.25395	0.18048	0.11087	1.11388	0.06534	-0.56466	0.47398	-0.54096
down	-0.08628	-0.45804	0.03112	0.06033	-0.29533	0.20481	-0.42635	1.24907	-0.15884	-0.04111	-0.17039
left	0.33702	0.07212	0.12953	-0.10945	-0.30821	-0.17928	-0.35318	0.17607	-0.15488	0.04569	0.00715

Final Policy (R: Right, D: Down, Up: U, Left: L)

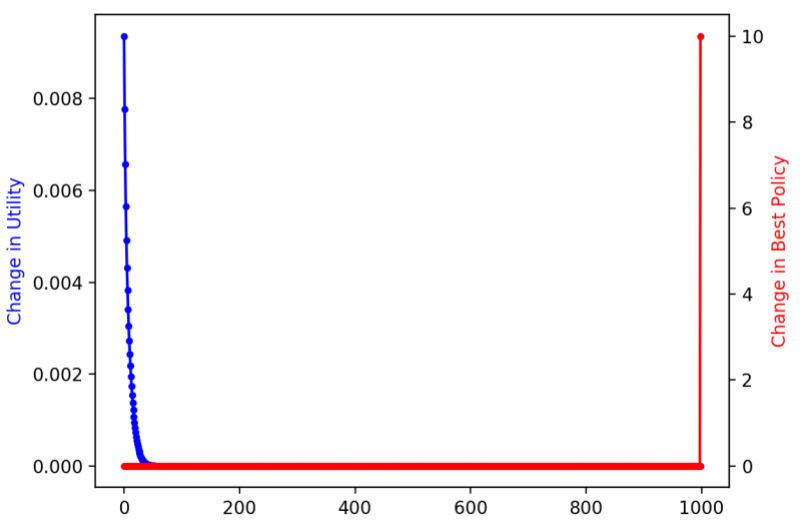
L	**D**	**D**	**1**
R	**O**	**D**	**R**
U	**L**	**U**	**D**
U	**1**	**-10**	**10**





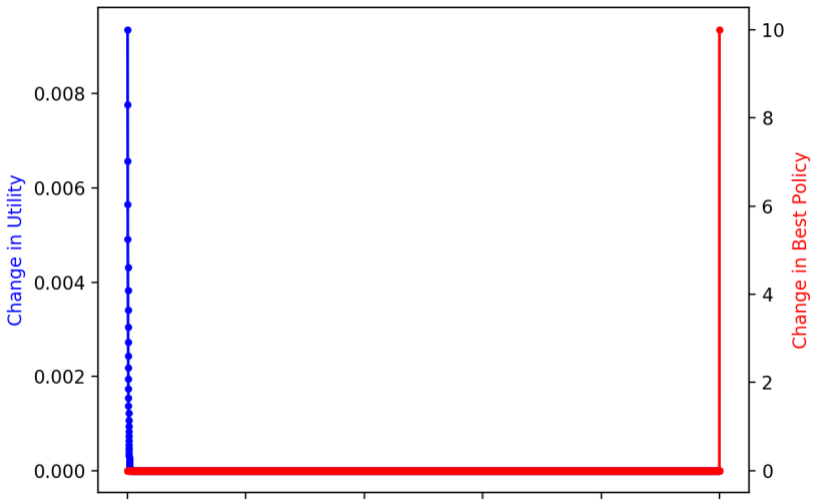
i. For result of Q-learning even if the values in calculations change, end of calculations it will converge very close values with N=100 result , after 1000 iteration. Q values(nearly) and Policy same with N = 100.

	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10
up	-0.00370	-0.42667	-0.19263	-0.45323	-0.23899	-0.71781	0.13327	0.17465	0.07629	0.07656	0.03329
right	-0.15827	0.26821	0.47766	-0.25395	0.18048	0.11087	1.11419	0.06534	-0.56466	0.47398	-0.54096
down	-0.08628	-0.45804	0.03112	0.06033	-0.29533	0.20481	-0.42635	1.24910	-0.15884	-0.04111	-0.17039
left	0.33702	0.07212	0.12953	-0.10945	-0.30821	-0.17928	-0.35318	0.17607	-0.15488	0.04569	0.00715



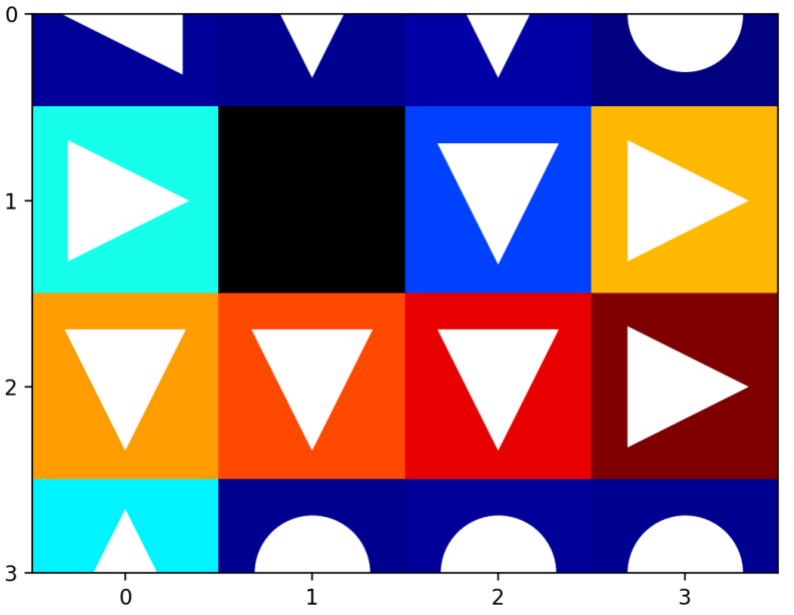
ii. If N = 10000 nothing will change in policy but some value can change with very small amount because it is already converged.

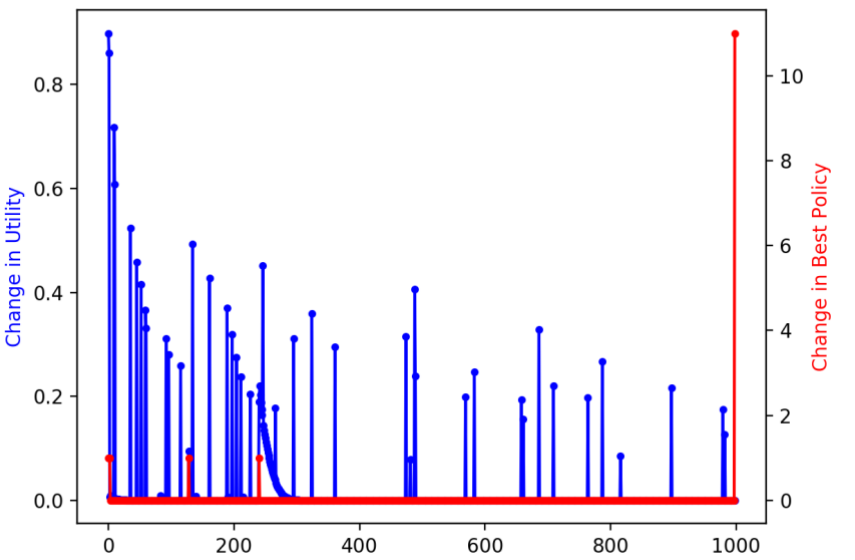
	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10
up	-0.00370	-0.42667	-0.19263	-0.45323	-0.23899	-0.71781	0.13327	0.17465	0.07629	0.07656	0.03329
right	-0.15827	0.26821	0.47766	-0.25395	0.18048	0.11087	1.11419	0.06534	-0.56466	0.47398	-0.54096
down	-0.08628	-0.45804	0.03112	0.06033	-0.29533	0.20481	-0.42635	1.24910	-0.15884	-0.04111	-0.17039
left	0.33702	0.07212	0.12953	-0.10945	-0.30821	-0.17928	-0.35318	0.17607	-0.15488	0.04569	0.00715



iii. If we change e = 0.1 it can do some random actions so values can not be best actions value. Results are not so good. If exploration probability is useful when grid is so large but if grid is small it cause policy problems.

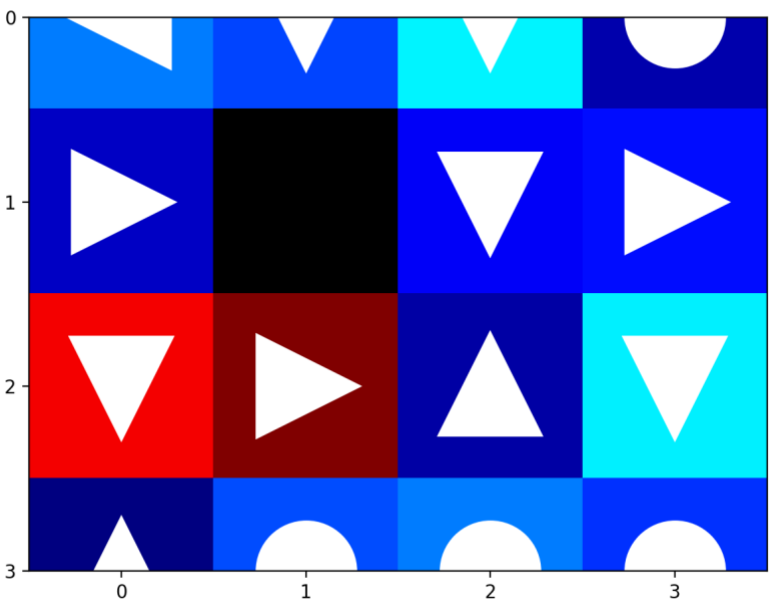
	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10
up	-0.00370	-0.42667	-0.19263	-0.45323	-0.23899	-0.71781	1.44353	5.42635	0.54967	2.71718	3.81407
right	-0.15827	0.26821	0.47766	-0.02318	2.02005	0.11722	7.42535	8.26150	9.19056	9.18434	-0.36196
down	-0.08628	-0.45804	0.03112	4.11050	-0.25807	7.13819	0.99193	1.24910	-6.37063	10.22284	-0.17039
left	0.33702	0.07212	0.12953	-0.10945	-0.30821	-0.17928	-0.35318	0.21155	-0.15488	0.04569	0.00715

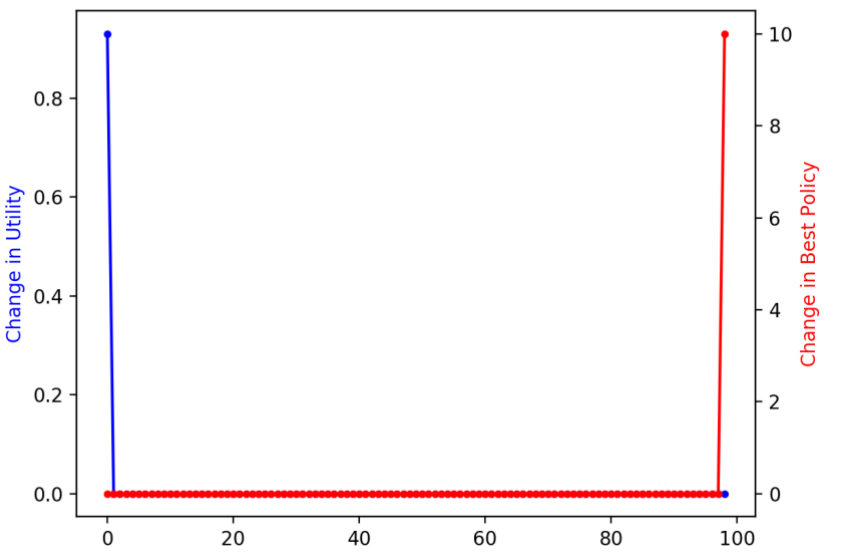




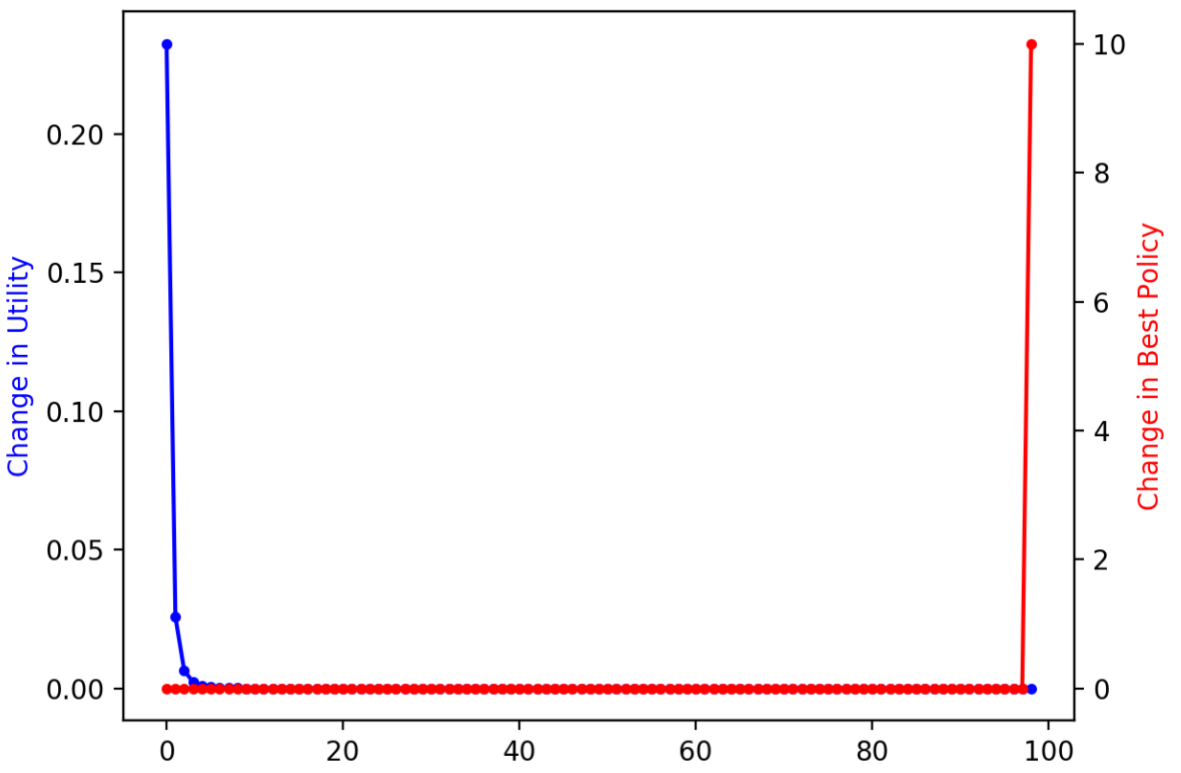
iv. If we increase learning rate = 1 the new infromation calculated by algrithm totaly overrides the old information so we can not do anything about old explorations. But it directly converge the result values and policy.

	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10
up	-0.00370	-0.42667	-0.19263	-0.45323	-0.23899	-0.71781	-0.01677	0.17618	0.07629	0.07656	0.03329
right	-0.15827	0.26821	0.47766	-0.25395	0.18048	0.11087	1.11419	0.06534	-0.56466	0.47398	-0.54096
down	-0.08628	-0.45804	0.03112	0.11528	-0.29533	0.20481	-0.42635	1.24910	-0.15884	-0.04111	-0.17039
left	0.33702	0.07212	0.12953	-0.10945	-0.30821	-0.17928	-0.35318	0.17618	-0.15488	0.04569	0.00715



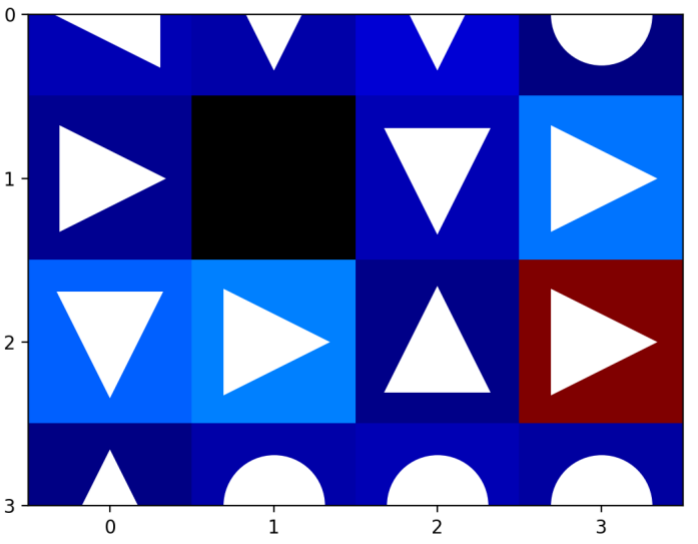


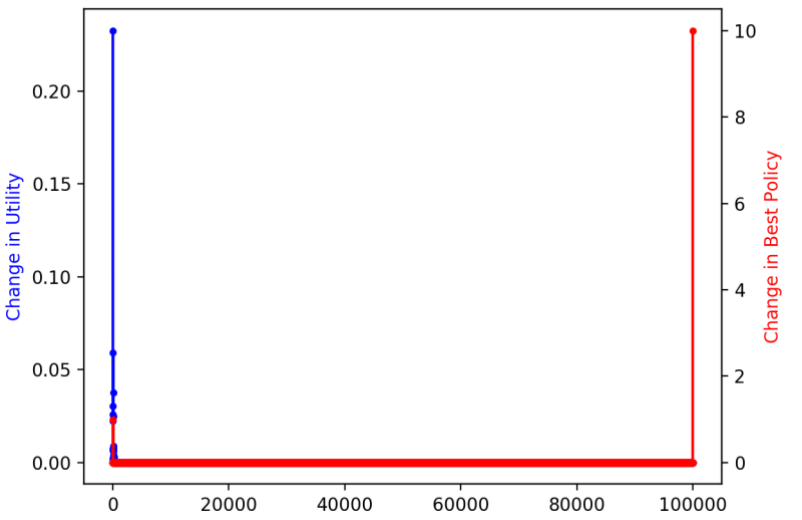
v. **When we take learning rate 1/count, for each pair count increase 1 it reduce learning rate and each iteration value of new information will decrease override size on old information will decrease so it a good combination of new learning and known information.**



vi. **If we increase N = 100000 with e : 0.1 and same learning rate with above the result more accurate than just e:0.1 because we have a good leaning rate and some e:0.1 with 100000 iteration the result is converge in 100000 iteration.**

	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10
up	-0.00370	-0.42667	-0.19263	-0.45304	-0.23662	-0.70753	0.09960	0.23986	0.14155	0.08560	0.10152
right	-0.15818	0.26824	0.47996	-0.25275	0.39154	0.13467	1.11418	0.07559	-0.40209	0.37021	-0.51180
down	-0.08627	-0.45804	0.03112	0.24771	-0.29404	1.42128	-0.38513	1.24910	-0.20780	5.08639	-0.17029
left	0.33681	0.07212	0.12953	-0.10945	-0.30821	-0.17928	-0.35318	0.17618	-0.15488	0.04569	0.00715





vii. I think is the best parameters is e = 0 and a = 0.1 for this grid world, but if we have larger number of statement with different parameters or values etc. The best parameters would be last one.