In [1]:

```
world = [['red','green','green','red','red'],
         ['red','red','green','red','red'],
         ['red','red','green','green','red'],
         ['red','red','red','red','red'],
         ['red','green','green','red','red']]
```

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:

```
measurements = ['green','green','green','green','green', 'red']
motions = [[0,0], [0,1], [1,0],[1,0],[0,1], [0,1]]
```

In [10]:

```
sensor_right = 1
p_move = 1
```

In [6]:

```
x, y = np.shape(world)
estimate = np.full(np.shape(world), 1./(x*y))
estimate
```

Out[6]:

```
array([[0.04, 0.04, 0.04, 0.04, 0.04],
       [0.04, 0.04, 0.04, 0.04, 0.04],
       [0.04, 0.04, 0.04, 0.04, 0.04],
       [0.04, 0.04, 0.04, 0.04, 0.04],
       [0.04, 0.04, 0.04, 0.04, 0.04]])
```

In [7]:

```
def estimate_after_motion(estimate, motion, p_move):
    non_move_estimate = np.multiply(estimate, 1-p_move)
    for motion_dim in range(len(motion)):
        estimate = np.roll(estimate, motion[motion_dim], axis=motion_dim)
    move_estimate = np.multiply(estimate, p_move)
    return np.add(move_estimate, non_move_estimate)
```

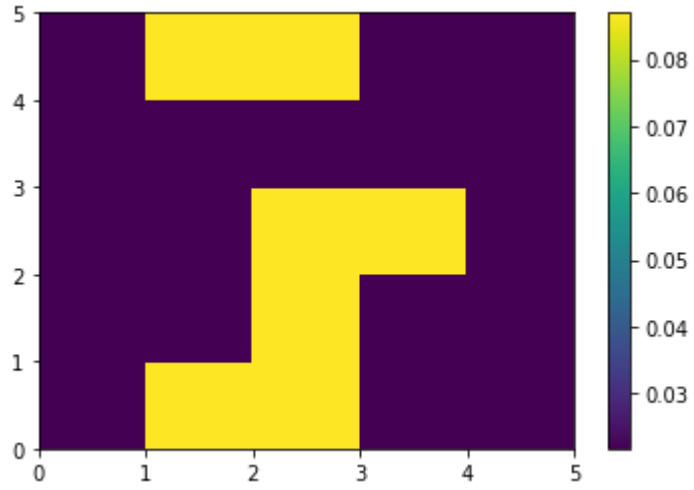
In [8]:

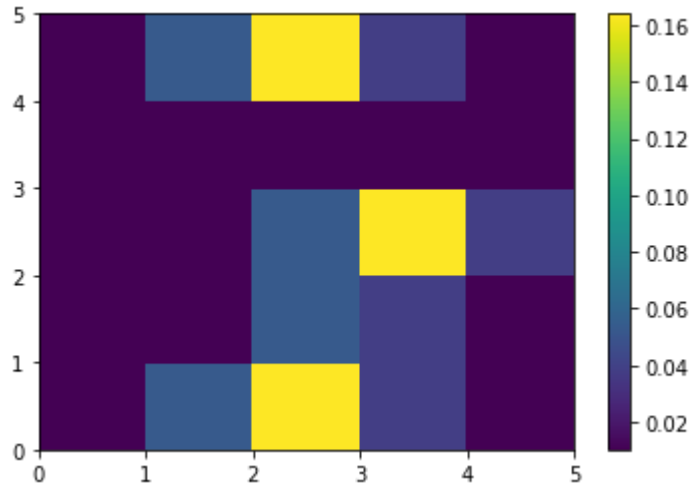
```
def estimate_after_measurement(world, estimate, measurement, sensor_reliability
):
    n_rows, n_cols = np.shape(world)
    non_normalized_estimate = estimate
    for i_row in range(n_rows):
        for i_col in range(n_cols):
            hit = (world[i_row][i_col]==measurement)
            non_normalized_estimate[i_row][i_col] = estimate[i_row][i_col]*(sens
or_reliability*hit+(1-sensor_reliability)*(1-hit))

    return np.divide(non_normalized_estimate, np.sum(non_normalized_estimate))
```

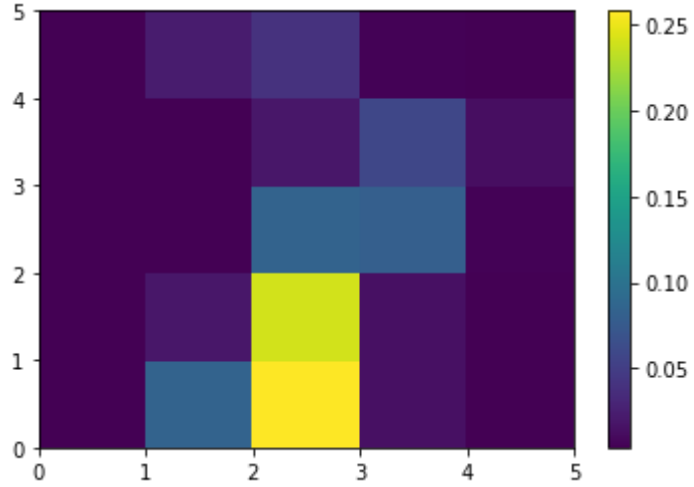
In [9]:

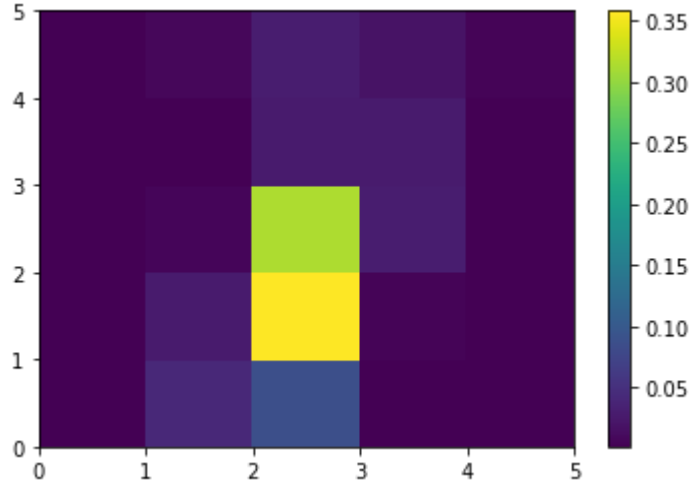
```
curr_estimate = estimate
for time_step in range(len(measurements)) :
    new_estimate_after_motion = estimate_after_motion(curr_estimate, motions[tim
e_step], p_move)
    curr_estimate = estimate_after_measurement(world, new_estimate_after_motion,
measurements[time_step], sensor_right)
    print(curr_estimate)
    plt.figure(time_step)
    plt.pcolormesh(curr_estimate)
    plt.colorbar()
```

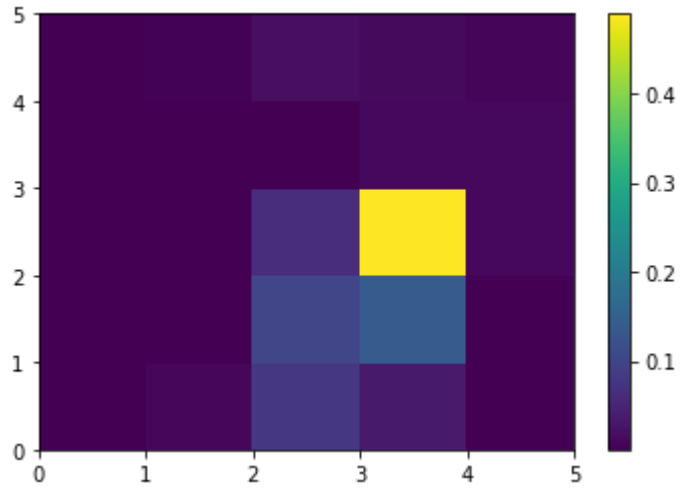
```
[[0.02173913 0.08695652 0.08695652 0.02173913 0.02173913]
 [0.02173913 0.02173913 0.08695652 0.02173913 0.02173913]
 [0.02173913 0.02173913 0.08695652 0.08695652 0.02173913]
 [0.02173913 0.02173913 0.02173913 0.02173913 0.02173913]
 [0.02173913 0.08695652 0.08695652 0.02173913 0.02173913]]
[[0.0102459  0.05327869 0.16393443 0.03790984 0.0102459 ]
 [0.0102459  0.0102459  0.05327869 0.03790984 0.0102459 ]
 [0.0102459  0.0102459  0.05327869 0.16393443 0.03790984]
 [0.0102459  0.0102459  0.0102459  0.0102459  0.0102459 ]
 [0.0102459  0.05327869 0.16393443 0.03790984 0.0102459 ]]
[[0.00403047 0.08383378 0.2579501  0.01491274 0.00403047]
 [0.00403047 0.01926565 0.24053847 0.01491274 0.00403047]
 [0.00403047 0.00403047 0.08383378 0.07948088 0.0051187 ]
 [0.00403047 0.00403047 0.01926565 0.05844182 0.01382451]
 [0.00403047 0.02289307 0.0403047  0.0051187  0.00403047]]
[[0.00140809 0.04050793 0.08673833 0.00213044 0.00140809]
 [0.00140809 0.02703251 0.3580378  0.00520993 0.00140809]
 [0.00140809 0.00619841 0.31424057 0.02986277 0.00144611]
 [0.00140809 0.00140809 0.02703251 0.02703251 0.00209242]
 [0.00140809 0.0082683  0.02986277 0.0185544  0.00448758]]
[[0.00060233 0.00909956 0.07722193 0.03348445 0.00088043]
 [0.00060233 0.00169846 0.10289141 0.13806354 0.002066  ]
 [0.00061697 0.00080725 0.06331383 0.48902658 0.0115587 ]
 [0.00086579 0.00060233 0.00169846 0.01156358 0.01049673]
 [0.0017879  0.00358316 0.01784254 0.01229054 0.00733521]]
[[9.59969192e-04 4.08719160e-04 4.47879376e-03 8.20201522e-02
  3.40294169e-02]
 [2.16132602e-03 8.01582719e-04 3.32641773e-03 1.19806038e-01
  1.40134438e-01]
 [1.17820683e-02 7.16072334e-04 1.98663296e-03 2.98041500e-02
  4.96839270e-01]
 [1.07339710e-02 9.45137626e-04 8.01582719e-04 3.02302167e-03
  1.28993796e-02]
 [7.63417377e-03 5.53784690e-04 1.40994140e-03 1.94639038e-02
  1.32800565e-02]]
```

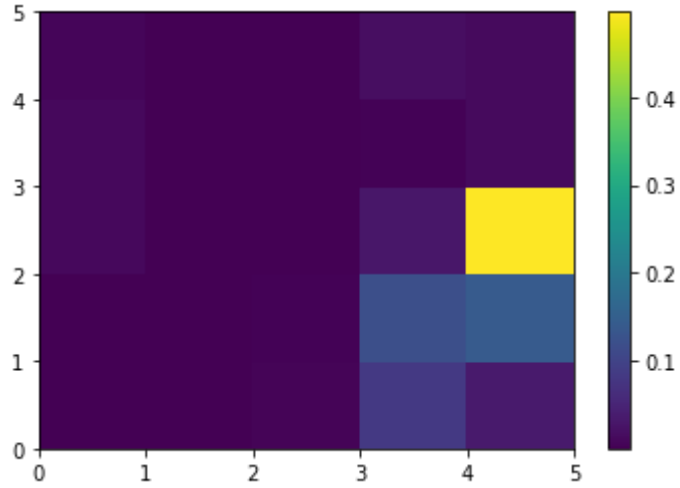



0 1 2 3 4 5
0 1 2 3 4 5
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02


0 1 2 3 4 5
0 1 2 3 4 5
0.25
0.20
0.15
0.10
0.05


0 1 2 3 4 5
0 1 2 3 4 5
0.35
0.30
0.25
0.20
0.15
0.10
0.05


5
4
3
2
1
0
0
1
2
3
4
5
0.4
0.3
0.2
0.1


5
4
3
2
1
0
0
1
2
3
4
5
0.4
0.3
0.2
0.1

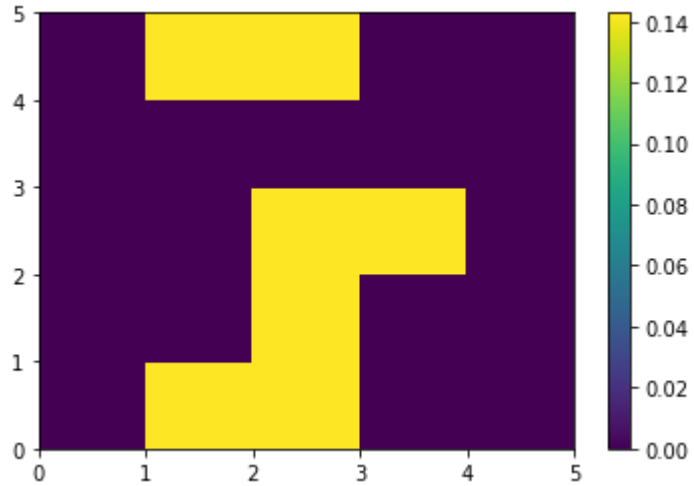
In [12]:

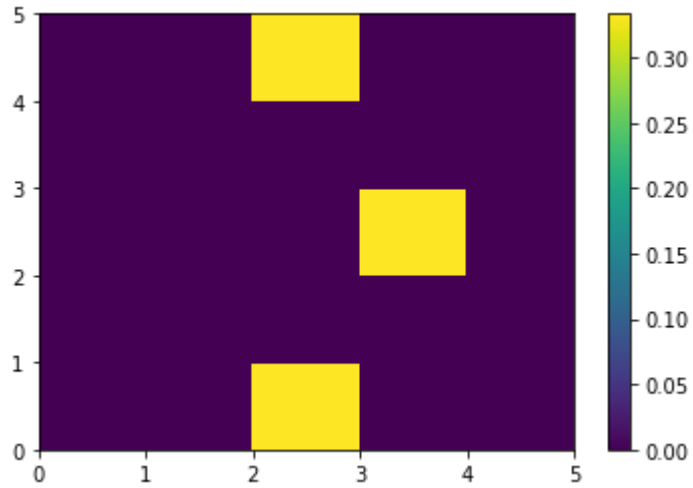
```
curr_estimate = estimate
for time_step in range(len(measurements)) :
    new_estimate_after_motion = estimate_after_motion(curr_estimate, motions[tim
e_step], p_move)
    curr_estimate = estimate_after_measurement(world, new_estimate_after_motion,
measurements[time_step], sensor_right)
    print(curr_estimate)
    plt.figure(time_step)
    plt.pcolormesh(curr_estimate)
    plt.colorbar()
curr_estimate
```

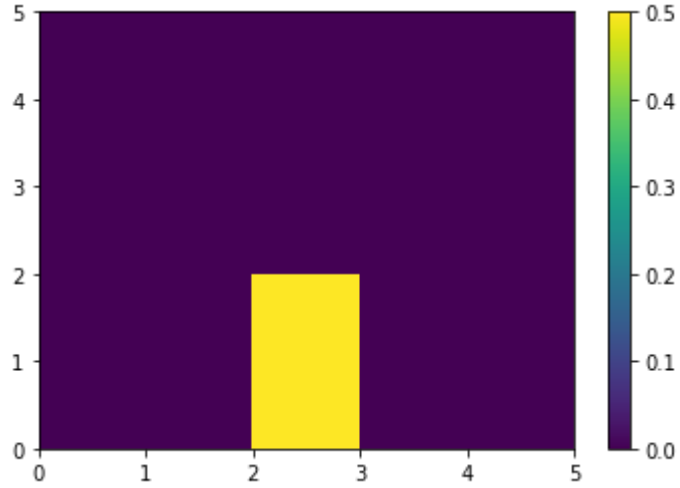
```
[[0.         0.14285714 0.14285714 0.         0.        ]
 [0.         0.         0.14285714 0.         0.        ]
 [0.         0.         0.14285714 0.14285714 0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.14285714 0.14285714 0.         0.        ]]
[[0.         0.         0.33333333 0.         0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.33333333 0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.33333333 0.         0.        ]]
[[0.  0.  0.5 0.  0. ]
 [0.  0.  0.5 0.  0. ]
 [0.  0.  0.  0.  0. ]
 [0.  0.  0.  0.  0. ]
 [0.  0.  0.  0.  0. ]]
[[0.  0.  0.  0.  0. ]
 [0.  0.  0.5 0.  0. ]
 [0.  0.  0.5 0.  0. ]
 [0.  0.  0.  0.  0. ]
 [0.  0.  0.  0.  0. ]]
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
```

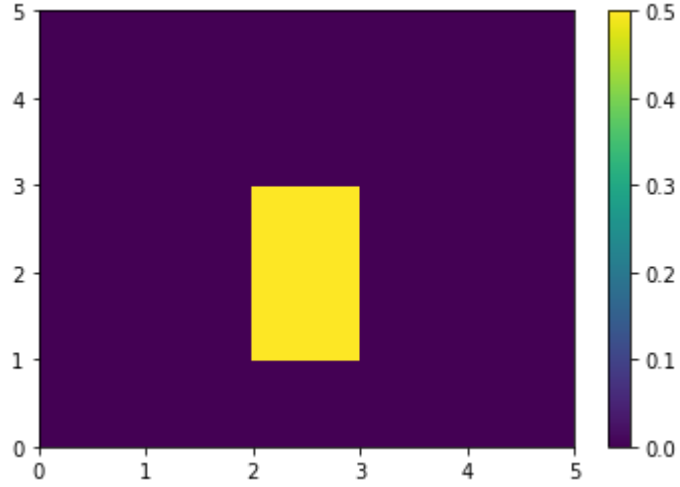
Out[12]:

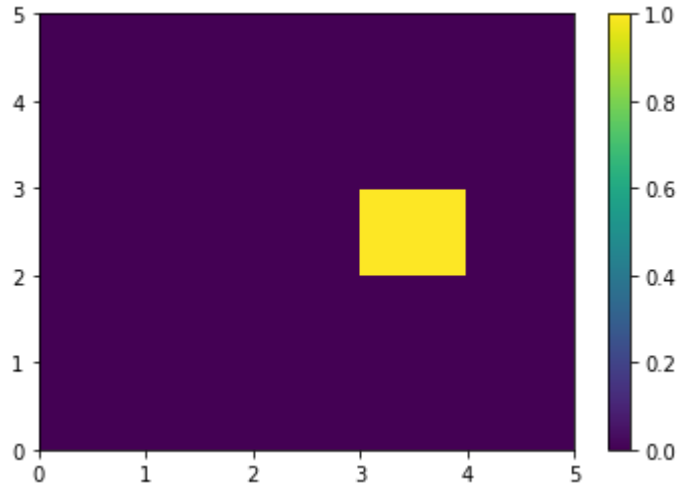
```
array([[0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 1.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.]])
```

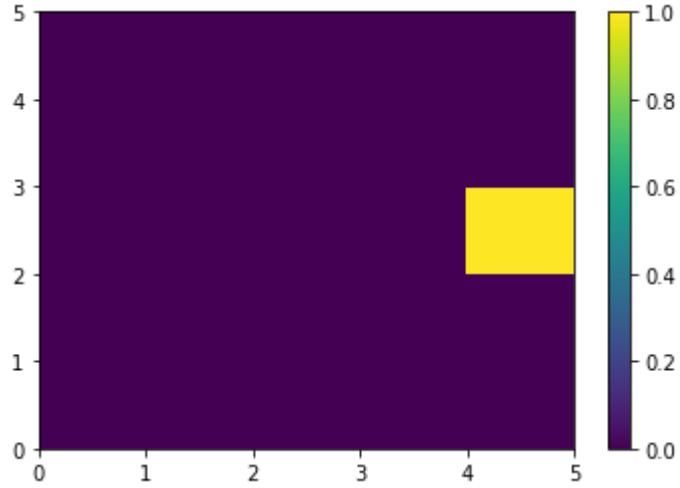



0 1 2 3 4 5
0.30
0.25
0.20
0.15
0.10
0.05
0.00


0 1 2 3 4 5
0.5
0.4
0.3
0.2
0.1
0.0


0 1 2 3 4 5
0.5
0.4
0.3
0.2
0.1
0.0




In [ ]:

In [ ]:

In [ ]:

In [ ]: