In [1]:
```
## import packages
import numpy as np
from PIL import Image
from numpy import histogram as hist  # call hist, otherwise np.histogram
import matplotlib.pyplot as plt

import histogram_module
import dist_module
import match_module
import rpc_module
```

In [2]:
```
## Find best match (Question 3.a)

with open('model.txt') as fp:
    model_images = fp.readlines()
model_images = [x.strip() for x in model_images]

with open('query.txt') as fp:
    query_images = fp.readlines()
query_images = [x.strip() for x in query_images]

eval_dist_type = 'intersect';
eval_hist_type = 'rg';
eval_num_bins = 30;
```

3.a : Show neigbours and find best match

The show_neighbours() takes all the input images and their corresponding images. The distance formulae implemented in 2.c are used to compute the similarities in the images. The output of the function D is of size (no of model images *no of query images). In this specific example 8989*. Each cell [i,j] represents the distance between ith result image and jth query images. The best match is the one that has the least distance from the model image.

In [3]:
```
[best_match, D] = match_module.find_best_match(model_images, query_images, ev
print("Shape of D: "+ str(np.shape(D)))
print("Shape of Best Match: "+ str(np.shape(best_match)))
```

```
Shape of D: (89, 89)
Shape of Best Match: (89,)
```

In [4]:
```
print("Distance formula used : %s "% (eval_dist_type))
print("Histrograms formula used : %s "% (eval_hist_type))
print("No of bins used : %i "% (eval_num_bins))
```

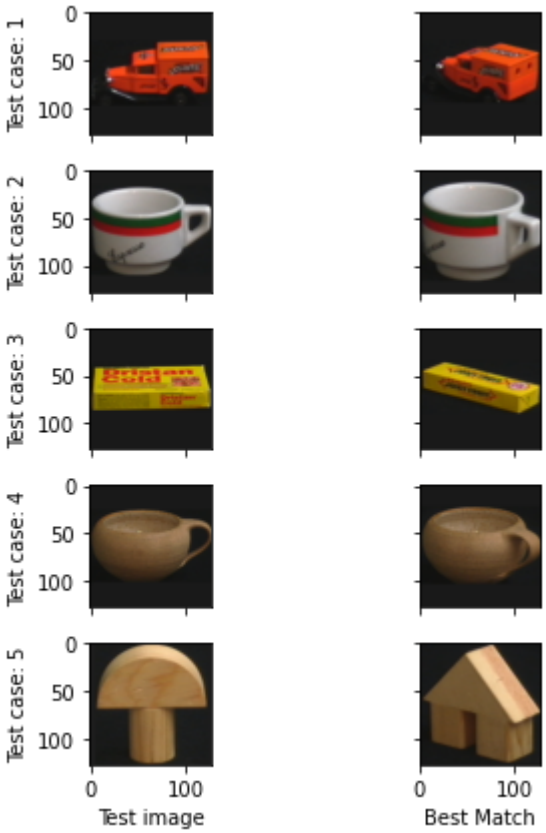
```
Distance formula used : intersect
Histrograms formula used : rg
No of bins used : 30
```

Additionally 5 sample images are taken from model_images and their best_match are plotted

In [5]:
```
print("Distance formula used : %s "% (eval_dist_type))
print("Histrograms formula used : %s "% (eval_hist_type))
print("No of bins used : %i "% (eval_num_bins))
plt.figure()
fig, axes2d = plt.subplots(nrows=5, ncols=2,sharex=True, sharey=True,figsize=
```

```
for i, row in enumerate(axes2d):
    for j, cell in enumerate(row):
        if(j==0):
            cell.imshow(np.array(Image.open(model_images[i])), vmin=0, vmax=2
            if i == len(axes2d) - 1:
                cell.set_xlabel("Test image")
        else:
            cell.imshow(np.array(Image.open(query_images[best_match[i]])), vm
            if i == len(axes2d) - 1:
                cell.set_xlabel("Best Match")
        if j == 0:
            cell.set_ylabel("Test case: {0:d}".format(i + 1))
plt.tight_layout()
plt.show()
```

```
Distance formula used : intersect
Histrograms formula used : rg
No of bins used : 30
<Figure size 432x288 with 0 Axes>
```



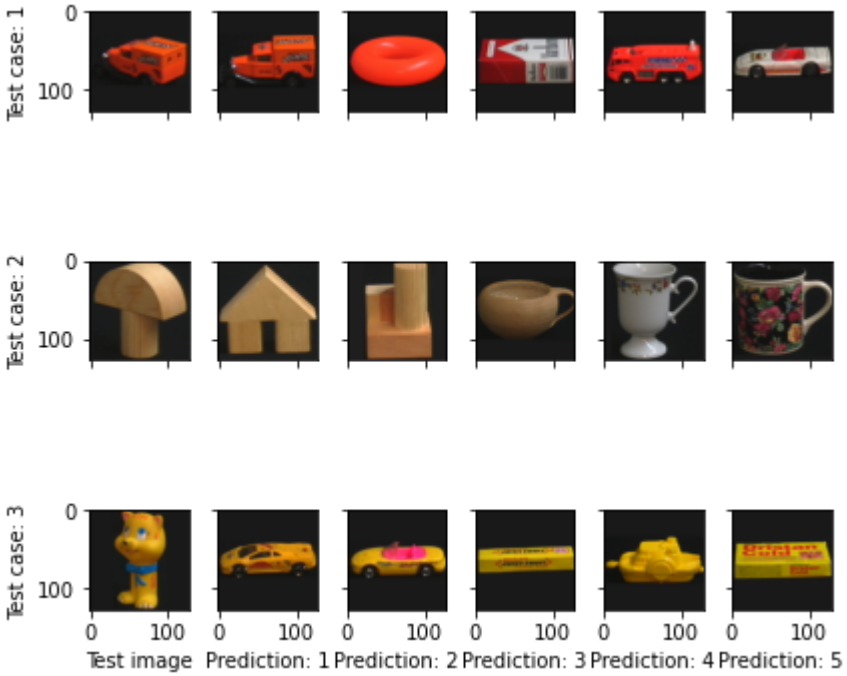
3.b Five Nearest neighbors

As required neighbours are calculated from the three given images. Using the D matrix calculated by find_match_module is used to find the distances. For each given image, top 5 candidate images based on their similiarity are plotted.

In [6]:

```
## visualize nearest neighbors (Question 3.b)
query_images_vis = [query_images[i] for i in np.array([0,4,9])]
match_module.show_neighbors(model_images, query_images_vis, eval_dist_type, e
```

```
<Figure size 432x288 with 0 Axes>
```



In [7]:
```
print("Distance formula used : %s "% (eval_dist_type))
print("Histrograms formula used : %s "% (eval_hist_type))
print("No of bins used : %i "% (eval_num_bins))
```

```
Distance formula used : intersect
Histrograms formula used : rg
No of bins used : 30
```

3.c Recognition percentage

The recognition percentage is calculated for the distance formulae and histogram used in task 3.a and 3. b

In [9]:
```
## compute recognition percentage (Question 3.c)
# import ipdb; ipdb.set_trace()
print("Distance formula used : %s \n"% (eval_dist_type))
print("Histrograms formula used : %s \n"% (eval_hist_type))
print("No of bins used : %i \n"% (eval_num_bins))
num_correct = sum( best_match == range(len(query_images)) )
print('number of correct matches: %d \n'% (num_correct))
print('total sample: %d \n'% (len(query_images)))
print('recognition percentage: %f\n'% (1.0 * num_correct / len(query_images)))
```

```
Distance formula used : intersect

Histrograms formula used : rg

No of bins used : 30

number of correct matches: 17

total sample: 89

recognition percentage: 0.191011
```

To compare the results, the nearest neigbour approach is ran for three bin sizes (5,20,40) , three distance formulae (l2, chi square and intersect) and three histograms (rgb, rg, dxdy). The results are plotted against different pairs of these parameters

In [8]:
```
print('distance functions:')
distance_types = ['l2', 'intersect', 'chi2']
print(distance_types)
print('\n')

print('histogram types:')
hist_types = [ 'rgb', 'rg', 'dxdy']
print(hist_types)
print('\n')
```

```
distance functions:
['l2', 'intersect', 'chi2']


histogram types:
['rgb', 'rg', 'dxdy']
```

In [33]:
```
bins=np.array([5,20,40])
Correct_Match_Table = np.zeros( (len(distance_types), len(hist_types),len(bin
```

In [11]:
```
for didx in range(len(distance_types)):
  for hidx in range(len(hist_types)):
    for x in range(len(bins)):
      print(distance_types[didx], hist_types[hidx], bins[x])
      [best_match, D] = match_module.find_best_match(model_images, query_imag
#       [best_match, D] = match_module.find_best_match(model_images, query_im
      num_correct = sum( best_match == range(len(query_images)) )
      print('number of correct matches: %d (%f)\n'% (num_correct, 1.0 * num_c
      Correct_Match_Table[didx, hidx,x] =  num_correct / len(query_images)
      print('\n')
```

```
l2 rgb 5
number of correct matches: 62 (0.696629)



l2 rgb 20
number of correct matches: 39 (0.438202)



l2 rgb 40
number of correct matches: 31 (0.348315)



l2 rg 5
number of correct matches: 31 (0.348315)



l2 rg 20
number of correct matches: 49 (0.550562)



l2 rg 40
number of correct matches: 52 (0.584270)



l2 dxdy 5
number of correct matches: 23 (0.258427)
```

```
l2 dxdy 20
number of correct matches: 21 (0.235955)
```

```
l2 dxdy 40
number of correct matches: 17 (0.191011)
```

```
intersect rgb 5
number of correct matches: 73 (0.820225)
```

```
intersect rgb 20
number of correct matches: 72 (0.808989)
```

```
intersect rgb 40
number of correct matches: 69 (0.775281)
```

```
intersect rg 5
number of correct matches: 34 (0.382022)
```

```
intersect rg 20
number of correct matches: 57 (0.640449)
```

```
intersect rg 40
number of correct matches: 64 (0.719101)
```

```
intersect dxdy 5
number of correct matches: 26 (0.292135)
```

```
intersect dxdy 20
number of correct matches: 29 (0.325843)
```

```
intersect dxdy 40
number of correct matches: 28 (0.314607)
```

```
chi2 rgb 5
number of correct matches: 66 (0.741573)
```

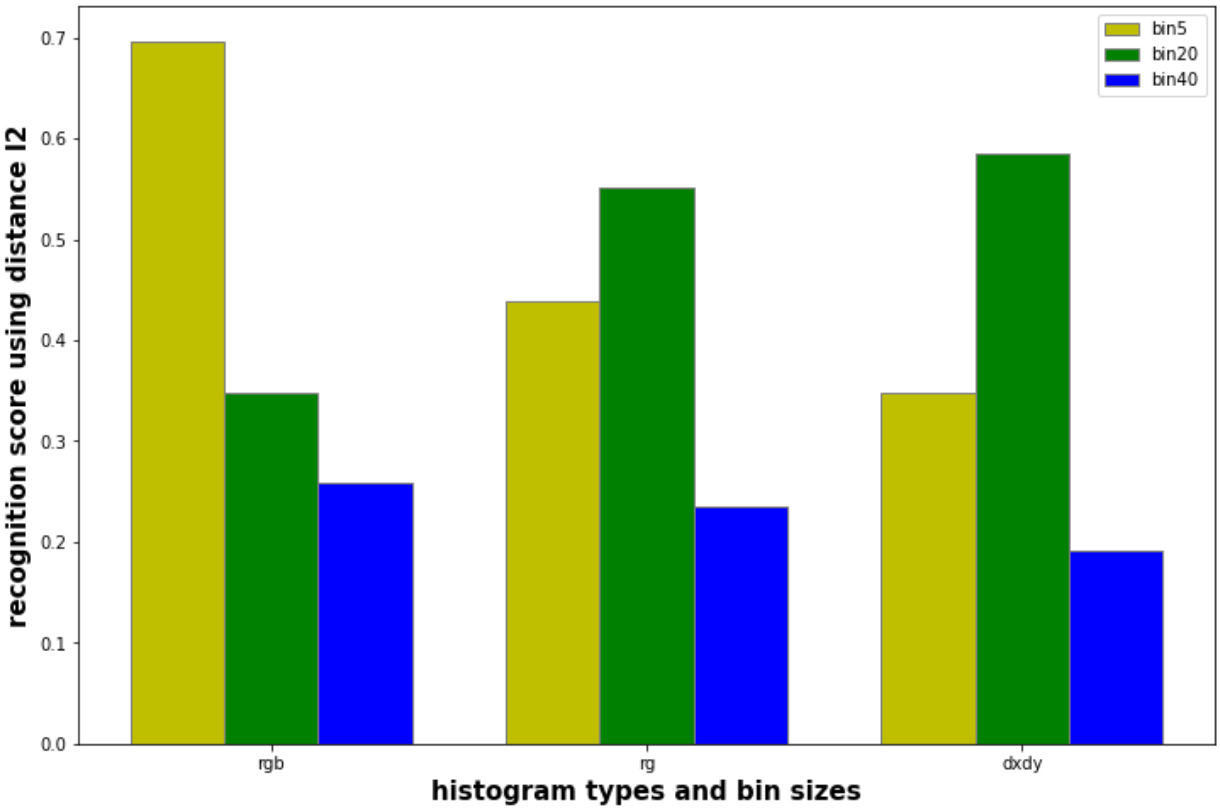
```
chi2 rgb 20
number of correct matches: 41 (0.460674)
```

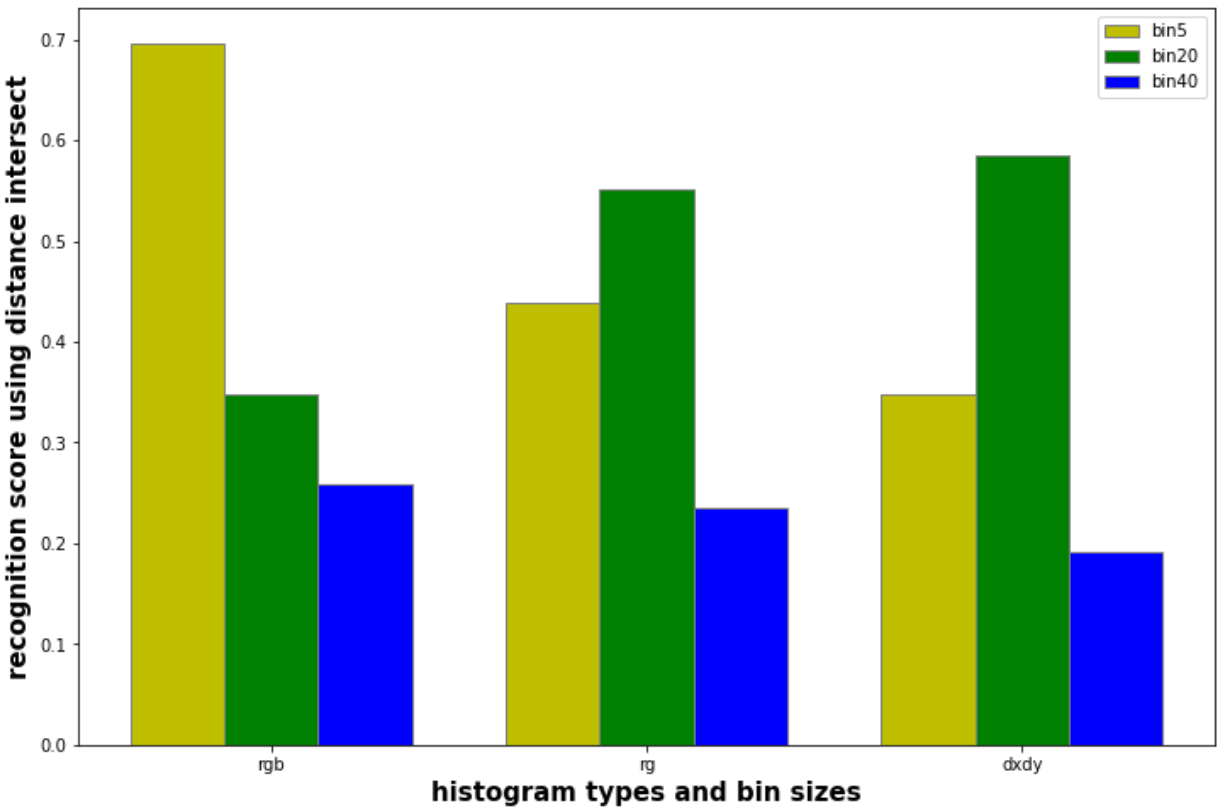
```
chi2 rgb 40
```

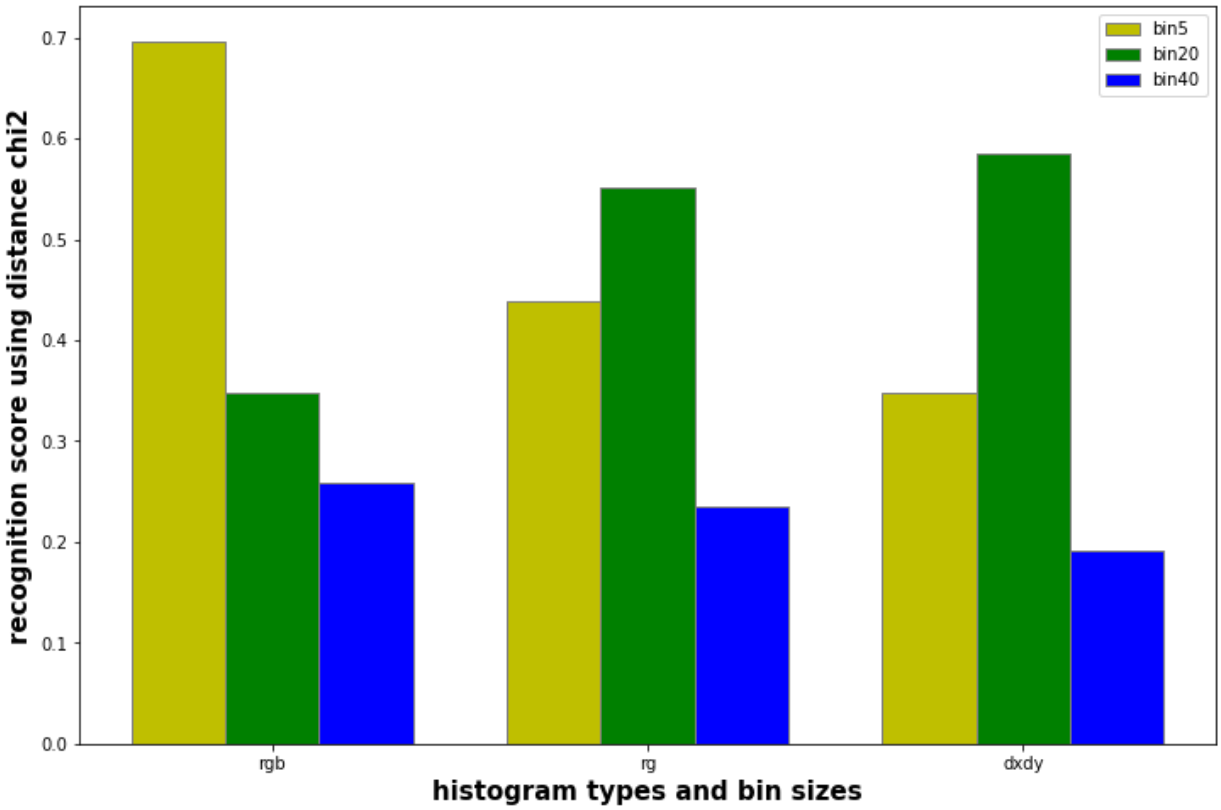
```
number of correct matches: 31 (0.348315)



chi2 rg 5
number of correct matches: 33 (0.370787)



chi2 rg 20
number of correct matches: 52 (0.584270)



chi2 rg 40
number of correct matches: 56 (0.629213)



chi2 dxdy 5
number of correct matches: 24 (0.269663)



chi2 dxdy 20
number of correct matches: 23 (0.258427)



chi2 dxdy 40
number of correct matches: 17 (0.191011)
```

In [40]:

```
for i in range(0,len(Correct_Match_Table[0])):
    # set width of bar
    barWidth = 0.25
    fig = plt.subplots(figsize =(12, 8))
    data=Correct_Match_Table[i]
    # Set position of bar on X axis
    br1 = np.arange(len(IT))
    br2 = [x + barWidth for x in br1]
    br3 = [x + barWidth for x in br2]

    # Make the plot
    plt.bar(br1, data[0], color ='y', width = barWidth,
            edgecolor ='grey', label ='bin'+str(bins[0]))
    plt.bar(br2, data[1], color ='g', width = barWidth,
            edgecolor ='grey', label ='bin'+str(bins[1]))
    plt.bar(br3, data[2], color ='b', width = barWidth,
            edgecolor ='grey', label ='bin'+str(bins[2]))

    # Adding Xticks
    plt.xlabel('histogram types and bin sizes', fontweight ='bold', fontsize
    plt.ylabel('recognition score using distance '+str(distance_types[i]), fo
    plt.xticks([r + barWidth for r in range(len(IT))],
            ['rgb', 'rg', 'dxdy'])

    plt.legend()
    plt.show()
```


bin5
bin20
bin40
recognition score using distance l2
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
rgb
rg
dxdy
histogram types and bin sizes


bin5
bin20
bin40
recognition score using distance intersect
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
rgb
rg
dxdy
histogram types and bin sizes


bin5
bin20
bin40
recognition score using distance chi2
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
rgb
rg
dxdy
histogram types and bin sizes

In [ ]: