For question 3(a), we have several works to do :

First, we have to find minimum boundary rectangles in match1.gif for each object, clover, spade, steer, and airplane, respectively. We implement this operation in minbounrec.m : we use build-in functions built-in function, bwconncomp and labelmatrix, divide the whole image to four different labeled regions, and we define

these four regions as four distinct objects.

Second, we use the formulas expressed through L6-11 to L6-15 to calculate size distribution, pattern spectrum and shape complexity, and the corresponding operation shown in sizedistr.m, Pecstrum.m and shapcomplx.m, respectively. In addition, due to the results of shape complexities of these objects, we find the most complex object in match1 is airplane.

Third, we implement the method described in L6-15 to decide what is best matching objects between match1 and mathc3. These operation is realized in bestmatch1. Moreover, since we have no special concern, we set all complement to 1 in C.

However, there is a very special phenomenon that the mated results have some overlapped. In order to resolve this problem, we observe differences between match1 and match3, and difference between pecstrums of these image, we set C(1)~C(6) equal to 1, otherwise is zero, then we get a perfect matching!

In problem 3.(b), since we just have to identify four solid objects in shadow1.jpg to corresponding objects in shadow1rotated.gif, we apply the conditional dilation method we used in the first problem to remove the hollow objects in these two images:

First, from observing both hollow and solid objects, we can choose a structuring element B1 (pixels 25 X 5),through opening original images ( shadow1.gif and shadow1rotated1.gif ), and remove hollow object, which thick are less than 25 X 5, and remain some part of solid ones.

Second, through using dilating the result of first step by B2 (5x5), we can get a image only remaining these four solid objects. The reason why we choose B2 (5x5) is that a square structuring object can dilate a image in 8-direction, that is, left, right, upper, lower, and two diagonal

direction. In addition, 5x5 can reconstruct the solid objects more quickly than a 3x3 structuring element in both shadow1.gif and shadow1rotated1.gif.

After gotten solid objects, we start to match the objects in shadow1 and the objects in shadow1rotated. The process is as followed: First, as mentioned before, we use the method stated in L6-15 to find a best matched objects of objects in shadow1rotated.gif from the objects in shadow1.gif, that is, calculate each distance between every objects in these two images and find the shortest distance to decide which is best matched. Note that we set all element in C as 1. Then, we get a result that two objects in shadow1rotated.gif match to the same object in shadow1.gif. Specifically, object 2(ref2) and 3(ref3) in shadow1rotated are matched to object 4(ob4) in shadow1. In order to resolve this problem, we compare distances between ob4 and ref2 and between ob4 and ref3, and then we find the latter is shorted which means ob4 is matched with ref3 not ref2. Finally, we find only object 2 (ob2) in shadow1rotated not matched. We assume ob2 matched with ref2. In order to proof this, we compare all distances between ob2 and objects (ref1~ref4) in shadow1rotated with the distance of the best matched pairs, and then we find the distance between ob2 and ref1, ref3, and ref4, respectively, is not the shortest distance. Remove these choose, we get ob2 best matched with ref2.