

# CS 7690, Advanced Image Processing

## Project3 Contour Description by Fourier-Harmonics

### Xiang Hao

#### Method to calculate coefficients and transformation to invariant descriptors

First of all, project the chain code on x and y. After this step, I got three arrays: x, y, and t.

Secondly, calculation of coefficients. From x, and y, I can get the z, which is just  $x + i*y$ , then I just used the formula:  $Z_n = \frac{L}{4 * \pi^2 * n^2} * \sum_{k=0}^M \frac{\Delta z_p}{|\Delta s|} * \exp(-2 \pi j n \frac{s_{k+1}}{L}) - \exp(-2 \pi j n \frac{s_k}{L})$  to compute the coefficients. Since the formula cannot compute  $Z_0$ , so I used another formula:

$$Z_0 = \frac{\sum_k x}{k} + i \frac{\sum_k y}{k} \quad \text{to compute } z_0, k \text{ is the number of the links.}$$

Thirdly, transform the coefficient to invariant descriptors.

- From the magnitude and the angle of  $z(1)$  and  $z(-1)$ , say  $\text{mag}(1)$ ,  $\text{mag}(-1)$ ,  $\text{angle}(1)$ ,  $\text{angle}(-1)$ .
- Compute  $\theta = 0.5 * (\text{angle}(-1) - \text{angle}(1))$ ,  $\phi = 0.5 * (\text{angle}(-1) + \text{angle}(1))$
- Use the formula:  $Z_n \frac{\exp(j(n\theta - \phi))}{|Z_1| + |Z_{(-1)}|}$ ,  $Z_0 = 0$  to get the invariant descriptors.

#### Method for contour reconstruction.

Suppose the invariant descriptors are  $Z_n$ ,

- Use the formula to compute the  $Z(s) = \sum_{n=-N}^N Z_n \exp(2 \pi j n \frac{s}{L})$ .
- Then the real part of  $z(s)$ , which is an array, is the x coordinate of each point, the imaging part of  $z(s)$  is the y coordinate of each point.
- 
- Use plot function to plot the contour, such as  $\text{plot}(x, y)$ .

#### Application and Illustration

In this section, I will illustrate the invariant coefficients. The invariant includes scaling, rotation, start points and translation. Since the chain code is invariant to the translation (Suppose we translate an contour by 10 units, the chain code will not change), so it is obvious that the coefficients are invariant to translation, and I did not show pictures about translation.

Here are some terms I will use in the following.

X and y projection: project the chain code onto the x and y direction; use the x and y coordinates to reconstruct the image.

Variant coefficient: compute the coefficient from the x and y projection. Reconstruct the image from the set of coefficient.

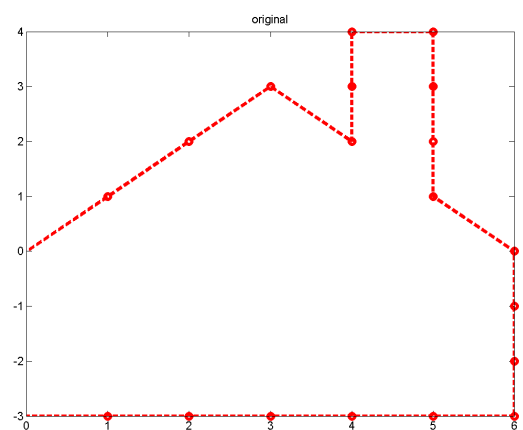
The first invariant coefficient: make the coefficient invariant and reconstruct the image from the invariant coefficient.

The other invariant coefficient: since there are two possible classifications, so we need compute two

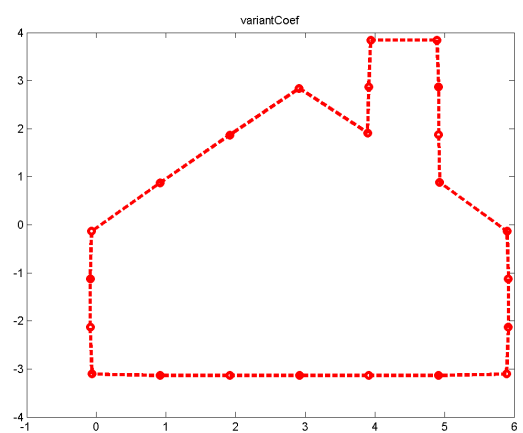
sets of coefficients to represent the shape. This is the other set.

### An illustration of the above terms.

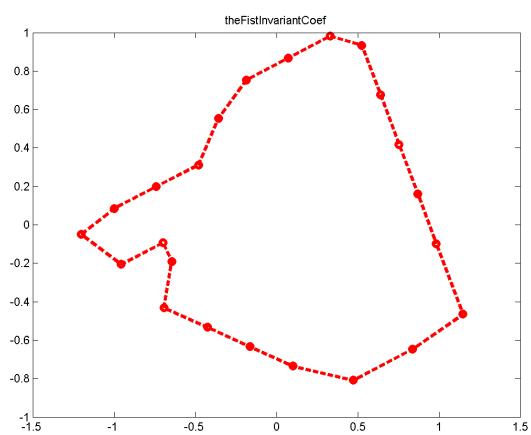
X, y projection



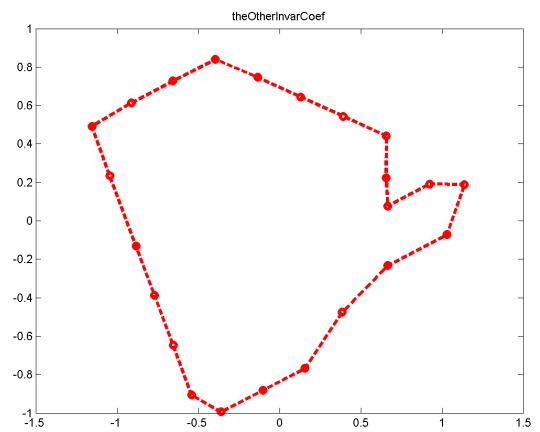
Variant coefficient



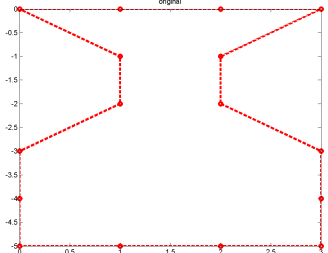
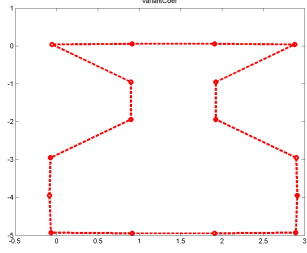
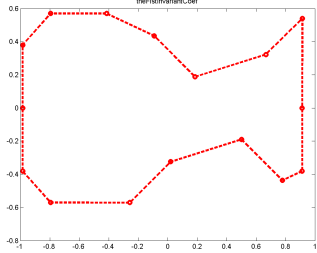
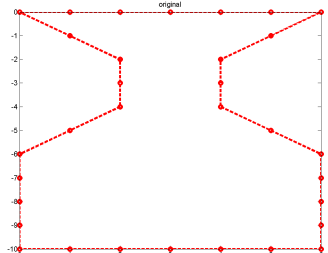
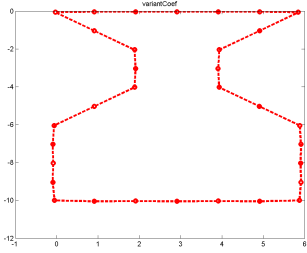
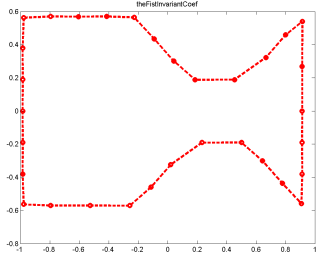
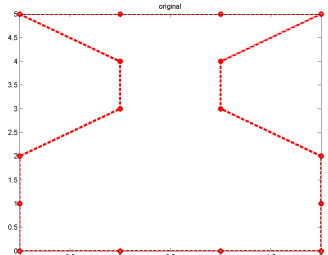
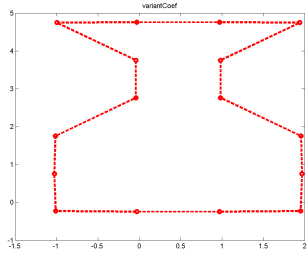
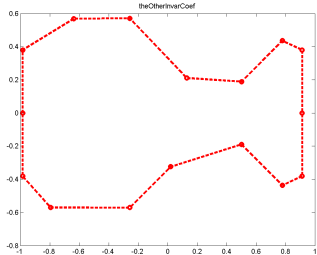
### The first invariant coefficient

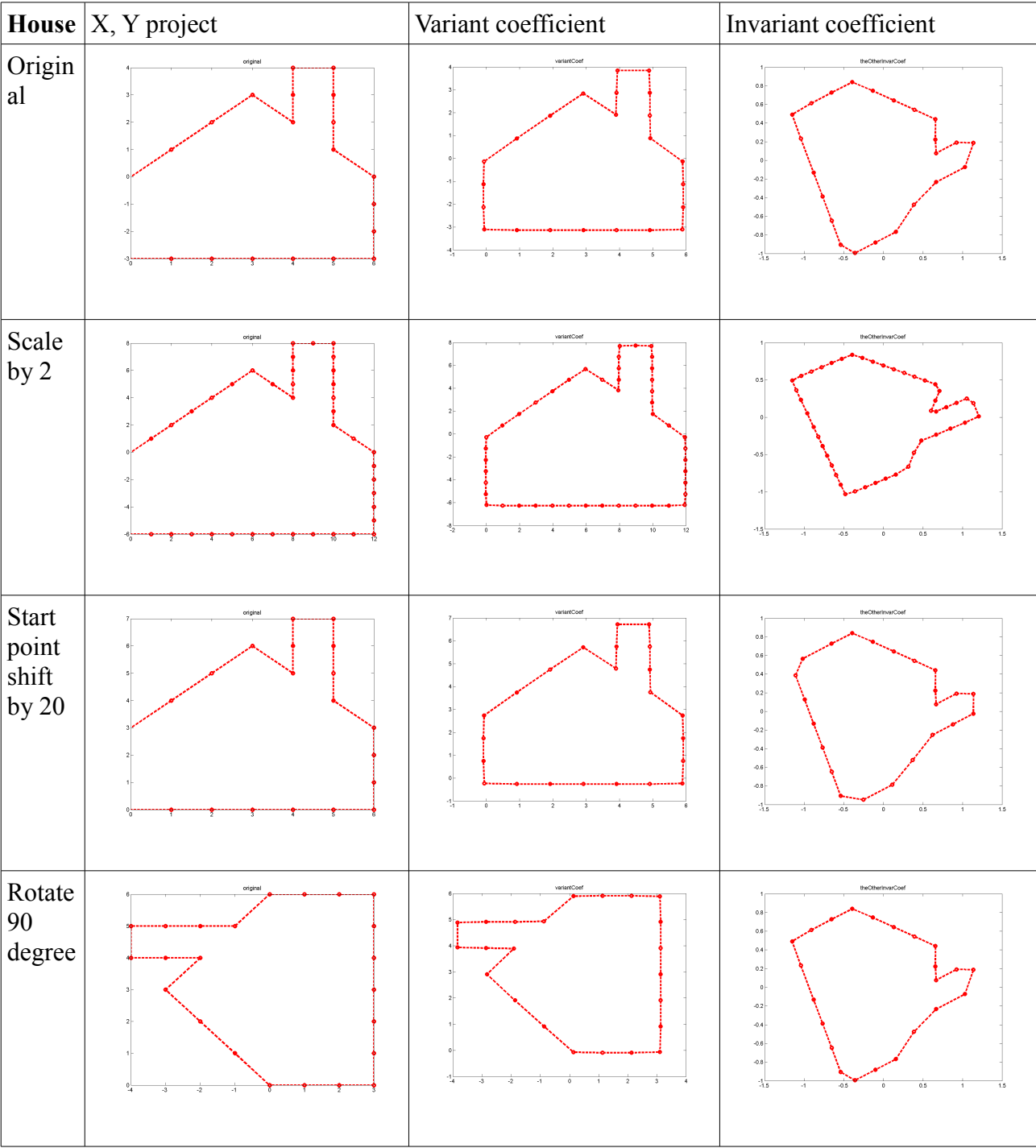
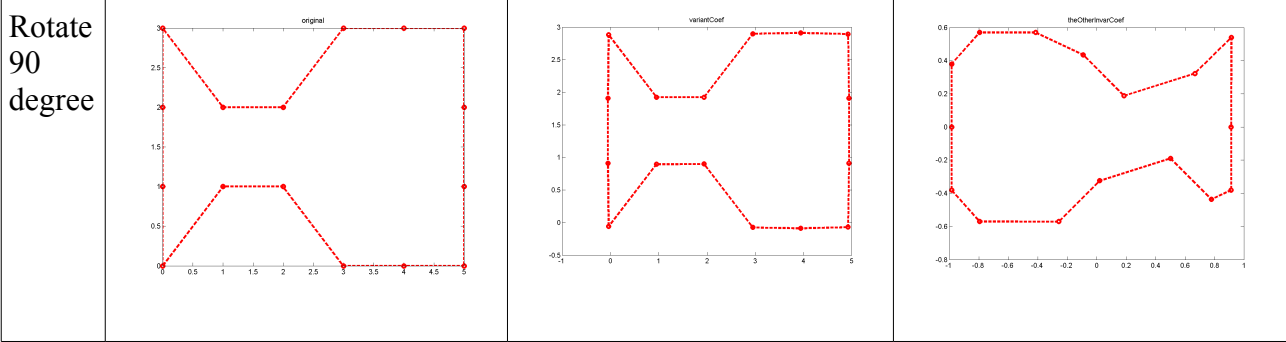


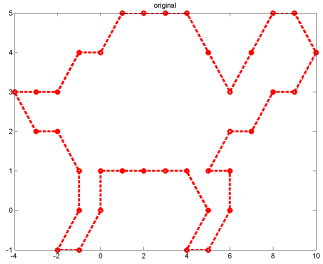
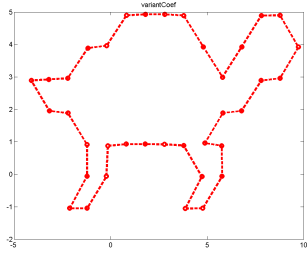
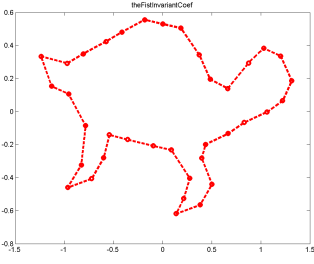
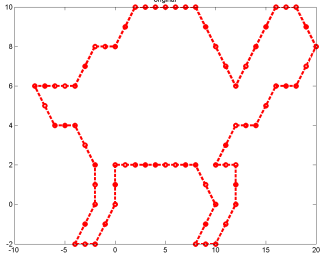
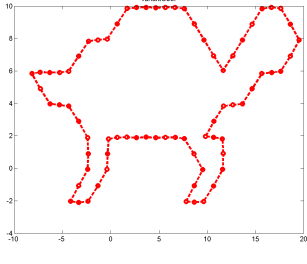
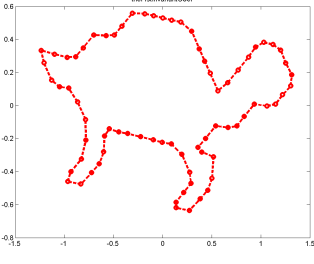
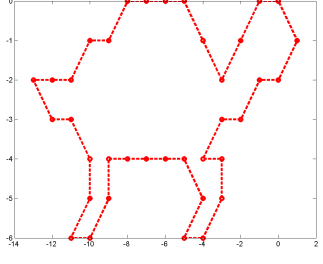
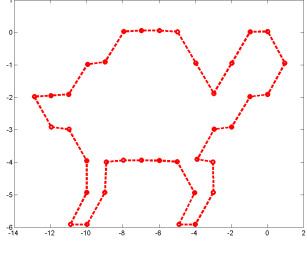
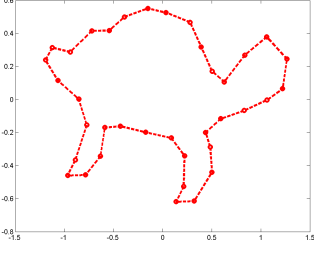
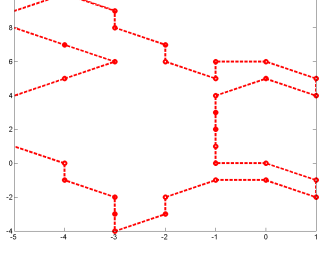
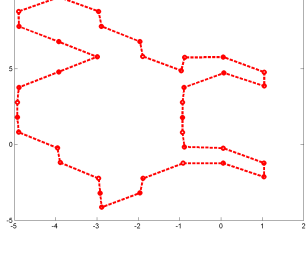
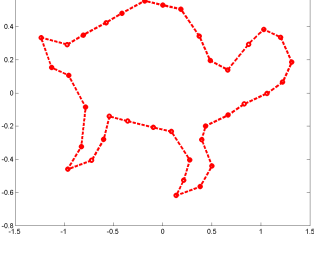
### The other invariant coefficient



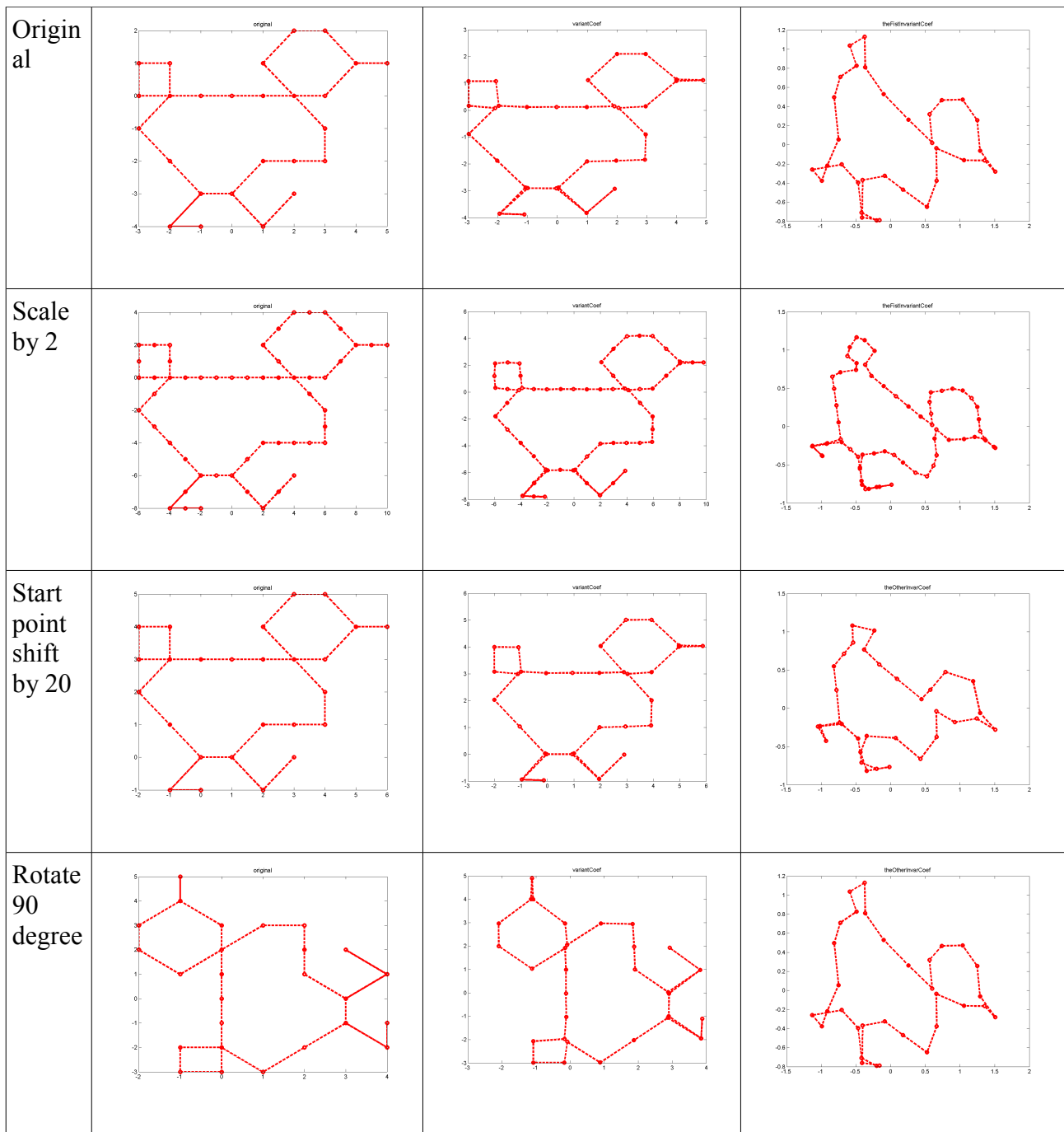
### Application of each object:

Vase	X, Y project	Variant coefficient	Invariant coefficient
Original			
Scale by 2			
Start point shift by 10			



Cat	X, Y project	Variant coefficient	Invariant coefficient
Original			
Scale by 2			
Start point shift by 20			
Rotate 90 degree			

Duck	X, Y project	Variant coefficient	Invariant coefficient
------	--------------	---------------------	-----------------------



## Pairwise shape differences between shapes

	Vase	House	Cat	Duck
Vase	0	0.1521 or 0.0970	0.0815 or 0.0735	0.1328 or 0.1950
House	0.1521 or 0.0970	0	0.1669 or 0.1439	0.1866 or 0.1241
Cat	0.0815 or 0.0735	0.1669 or 0.1439	0	0.1651 or 0.1631
Duck	0.1328 or 0.1950	0.1866 or 0.1241	0.1651 or 0.1631	0

Since each shape two possible coefficient sets, so each pair of objects have two distances.

From the above distance matrix. We can see that after making the coefficient invariant, the differences of the each two objects are quite small, that is because:

- 1) the distance is the distance of the magnitude, it does not include the distance of the phase
- 2) the above shape do not have high frequency, which means the low frequency coefficients have high magnitude comparing to the high frequency coefficient.
- 3) We do a normalization, so the magnitude of each coefficient is not large

### **Discussion of success, possible problems, and quality of contour description**

From the illustration above, we can see that the Fourier descriptor can represent the shapes, and the descriptors are invariant to scaling, rotation, startint points and translation.

However, we can also see some problems:

The Fourier descriptors do not reconstruct the vase object very well. This problem is caused by lack of links. I mean there are not enough links for the vase shape.( we have 16here). This problem can be easily solved by increasing the number of the links. As we saw, when I scale the vase by 2, the number of links are doubled, and we can reconstruct the shape much better.

In my program, the maximum order of the coefficient is 50. Even I increase the order to 10000, the result is not improved a lot. Which mean, the first several coefficients contain most information of the shape. This can be good, since we only need the first coefficients to reconstruct the shape, but this can also be bad, since we need to tons of coefficient in order to get every detail of the shapes.

From the distance matrix. We can see that after making the coefficient invariant, the differences of the each two objects are quite small. This is not good, if we want to classification since it not easy to tell the differences of two different shapes if their distance is quite small.

### **Creative thoughts: could this technique be used to create an average shape of a given shape class**

First, we need to know how to create the average of a shape class.

This is the way I thought:

- Compute the coefficients for each shape in the class.
- Make each coefficient invariant.
- Take the average of the invariant coefficient of difference shapes, which is the average.

Does it work?

Let think about an example, if we have only have N same shapes in the shape class. Go through the above process, we will get the average coefficients, which is obvious the same as the coefficients of each shape. So at least, in this case, it works. For other cases, if the shapes are quite similar with each other, we can guess it also works.

But, there may be some constraints:

For simple shapes, such as circles, this technique definitely can be used to compute the average shape.

For complex shapes, if the shapes have very very high frequency, we need to compute a very high order of coefficient for each shape, which may not be efficient.

### **Issues, alternative implementations, and potential improvements**

**Issues:**

Need enough links for each shape if we want to reconstruct the shape quite well.  
 Need a lot of coefficients in order to reconstruct the shape perfectly.  
 Each shape have two possible descriptors.  
 May not be a good way to do classification.

**Potential improvements:**

Given a chain codes, first scale it, so we have enough links in the chain codes. Then do the same process as we did above.

When we make the coefficients invariant to translation, we make  $Z_0 = 0$ . I am confused about this. Since the chain code is already invariant to translation, we do not need to set  $Z_0 = 0$  if we want to make the coefficients invariant to translation. When we set  $Z_0 = 0$ , we are moving the center of the first ellipse to (0, 0). Actually, I think the  $z_0$  is a useful in some case because it is kind of the center of the shape. Two different shapes may have different centers.

**Additional Exploratory Analysis: Bonus**

• Below are chain codes of a discrete circle and a discrete ellipse. First, what do you expect if you would derive descriptors from these two figures? (Hint: think about the parametric form of circles and ellipses and its relationship to the Fourier transform). Apply your procedure to derive sets of coefficients, what do you see? Could you imagine what happens and why it happens?

If the contour is just a ellipse, which means the contour has frequency 1 and it can be represented by a single ellipse. So,  $Z_n$ , will be 0, where  $\text{abs}(n) > 1$ . In our case, since the ellipse is discrete, which is not a perfect ellipse, so for  $\text{abs}(n) > 1$ ,  $Z_n$  may not be 0, but they should be very small.

If the contour is just a circle, which means the contour also has frequency 1 and it can be represented by a single circle, which is a special case of ellipse. So,  $Z_n$ , will be 0, where  $\text{abs}(n) > 1$ . In addition, since we only need a circle, not a ellipse, so one of the  $Z_1$  and  $Z(-1)$  will be 0. In our case, since the circle is discrete, which is not a perfect circle, so for  $\text{abs}(n) > 1$ ,  $Z_n$  may not be 0, but they should be very small.

Here, I copy the a part of the coefficients of the ellipse and the circles, the following results are corresponding to my guess.

Ellipse:

Columns 46 through 50

0.0148 - 0.0002i -0.0017 + 0.0017i 0.0564 + 0.0015i -0.0021 + 0.0027i **0.7972 - 0.0000i**

Columns 51 through 55

0 **0.2028 + 0.0000i** 0.0009 - 0.0052i 0.0101 + 0.0016i 0.0007 + 0.0020i



Columns 56 through 60

$0.0069 + 0.0011i$   $0.0019 - 0.0029i$   $0.0038 + 0.0004i$   $0.0005 + 0.0021i$   $0.0030 - 0.0005i$

Circle:

Columns 46 through 50

$-0.0007 + 0.0032i$   $0.0021 - 0.0002i$   $-0.0002 - 0.0010i$   $-0.0081 - 0.0014i$   **$0.9968 - 0.0000i$**

Columns 51 through 55

$0$   **$0.0032 + 0.0000i$**   $-0.0006 + 0.0006i$   $-0.0054 - 0.0023i$   $-0.0007 - 0.0003i$

Columns 56 through 60

$0.0005 - 0.0019i$   $0.0017 - 0.0019i$   $-0.0049 + 0.0026i$   $0.0023 - 0.0021i$   $0.0004 - 0.0011i$