# CS 7960, Advanced Image Processing
# Spring 2010, Prof. Guido Gerig

## Project 3: Contour Description by Fourier-Harmonics

| | |
|---|---|
| Out: | Thursday Feb-25-2010 |
| Due: | Thursday Mar-11-2010 |
| Office hours: | Tue 1pm to 3pm, please contact me in advance or for other arrangements. |

**Required Readings:** Papers and Materials Lecture Elliptic Harmonics

## Contour Description by Elliptic Harmonics

You need to implement the shape representation of a contour with Fourier harmonics, and to make the representation invariant to translation, start-point selection and rotation.

### Summary Elliptic Harmonics

In the complex notation $z(u)$ is represented as a series of complex exponentials.

$$z(s) = \sum_{n=-\inf}^{\inf} z_n e^{\frac{2\pi jns}{L}} = z_0 + \sum_{n=1}^{\inf} (z_n e^{\frac{2\pi jns}{L}} + z_{-n} e^{-\frac{2\pi jns}{L}})$$

where the complex coefficient $z_n$ can be expressed in polar notation, i.e.

$$z_n = r_n e^{j\psi_n} \ ,$$

with $r_n \in R, r_n \geq 0$, and $\psi \in R$. The terms $e^{\frac{2\pi jns}{L}}$ describe rotations as a funct. of arclength $s$. We have demonstrated that the terms $(z_n e^{\frac{2\pi jns}{L}} + z_{-n} e^{-\frac{2\pi jns}{L}})$ form ellipses that are traversed $n$ times while traversing the figure from 0 to $L$. Remember the demonstration with pairs of phasors of different length $|z_n|$ rotating clockwise and counterclockwise, and resulting vector being the sum of the clockwise phasor $z_n = \{a, b\}$ and the counterclockwise phasor $z_{-n} = \{c, d\}$. There is a close relationship between the complex and real notation (see the document *Kelemen-EllipticHarmonicsOnly.pdf*), i.e. they can be converted into each other.
In the complex notation of Fourier coefficients real and imaginary parts of $z_n$ correspond to the $x$ and $y$ coordinates

$$\begin{pmatrix} x \\ y \end{pmatrix}_n = \begin{pmatrix} a_n & b_n \\ c_n & d_n \end{pmatrix} \begin{pmatrix} \sin \frac{2\pi ns}{L} \\ \cos \frac{2\pi ns}{L} \end{pmatrix} \ ,$$

where the real valued coefficients $a_n$, $b_n$, $c_n$, and $d_n$ are defined as follows.

$$
\begin{aligned}
a_n &= \mathrm{Re} z_n + \mathrm{Re} z_{-n} \\
b_n &= -\mathrm{Im} z_n + \mathrm{Im} z_{-n} \\
c_n &= \mathrm{Im} z_n + \mathrm{Im} z_{-n} \\
d_n &= \mathrm{Re} z_n - \mathrm{Re} z_{-n}
\end{aligned}
$$

The procedure will have three major components:

- Calculate the set of coefficients from a contour description given as a chain code.
- Make the coefficients invariant to translation, rotation and scaling.
- Reconstruct the contours from the invariant coefficients.

Some sample curves represented as chain codes are given below (taken from Kuhl and Giardina article):

$$
\begin{aligned}
V_{\text{vase}} &= 0005676644422123 \\
V_{\text{house}} &= 1117220666766644444422 2 \\
V_{\text{cat}} &= 5412343001010007711075454506541344446 \\
V_{\text{duck}} &= 0031070454764457153450413314206 00
\end{aligned}
$$

You can use other objects for your experiments, but you would need to write a program to derive the chain code from the boundary of a binary object (code with 4 or 8 neighbors would work) or from edge detection derived in the anisotropic diffusion project.

## Instructions:

- Basic procedure:
  - Implement a program to calculate the Fourier harmonics from a chain code (best is to follow Kuhl and Giardina). Maximum order of 20 should be enough for most applications.
  - Transform the set of coefficients to a descriptor that is invariant to translation, start-point location, object rotation and scaling. Please note that there is a simple way to transform coefficients between real and complex notations, so you can choose to do this via the real notation or the complex notation.
  - Write a program to reconstruct and plot the contour from the set of coefficients. Input parameters are the number of harmonics (order) and eventually the sampling of the boundary length (step-size) and image size.
- Analysis of contours:
  - Verify by reconstruction that your invarance calculation is correct, display a figure before and after the invariance transformation.
  - Calculate the pairwise difference between sets of coefficients of different objects (after invariance normalization), a difference measure discussed in the course is the sum of the squared differences of components of coefficient vectors.

You should write up a report summarizing your procedure and discussing your results. The report should be **written in html** and **accessible to the instructor via a web-browser**, if a web-system is not available you can create a pdf file.

- Short description of method to calculate coefficients and transformation to invarient descriptors.
- Description of method for contour reconstruction.
- Application to contours given above, or eventually to other contours that you create by yourself.
- Description and illustration of success of invariance transformation.
- List or table of pairwise shape differences between the 4 objects given above, or between other objects that you generated.
- Discussion of success, possible problems, and quality of contour descriptions. Where would you see potential application areas.
- Creative thoughts: Could this technique be used to create an average shape of a given shape class?
- Commentary about any issues that arose, ways for alternative implementations, potential improvements etc.

### Additional Exploratory Analysis: Bonus

Would you have more capacity, here are some ideas for further exploration:

- 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?
- Create contours of own images, e.g. similar shapes of given shape classes. These could be used to test classification between groups of shapes (this requires a procedure to calculate the chain code from your own images). E.g., you could show that same type shapes are more similar to their templates (average shapes) than to others. A shape would thus be assigned to its closest template shape.
- Could you think about eventual extensions, e.g. choice of different methods to determine startpoint or object rotation?

### More figures for testing

Circle: {0, 0, 0, 7, 0, 0, 0, 0, 7, 0, 7, 0, 7, 0, 7, 7, 0, 7, 7, 7, 7, 7, 6, 7, 6, 7, 6, 6, 7, 6, 6, 6, 6, 6, 7, 5, 6, 6, 6, 6, 6, 5, 6, 6, 5, 6, 5, 6, 5, 5, 5, 5, 5, 5, 5, 5, 5, 4, 5, 4, 4, 5, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4, 3, 4, 3, 3, 4, 3, 3, 3, 3, 3, 3, 3, 3, 2, 3, 2, 2, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1}

Ellipse: {0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 7, 0, 0, 7, 0, 0, 7, 0, 7, 7, 7, 7, 7, 7, 6, 7, 5, 6, 6, 5, 5, 5, 5, 5, 5, 4, 5, 4, 4, 5, 4, 4, 5, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4, 3, 4, 4, 3, 4, 3, 4, 3, 3, 3, 3, 3, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1}