Fourier Polygons

Andrew Glassner

Microsoft Research

Polygons are everywhere. They're in our cereal, on our kitchen floors, and in the constellations in the skies. One place I didn't expect to see polygons is in the Fourier transform, but I found them there as well.

The Fourier transform is an indispensable tool in signal processing. In computer graphics, it helps us understand and cure problems as diverse as jaggies on the edge of polygons, blocky looking textures, and animated objects that appear to jump erratically as they move across the screen.

My friend and colleague Alvy Ray Smith recently wrote a memo that demonstrated a surprising interpretation of the Fourier transform. He showed how in some circumstances the Fourier transform looks like nothing more than operations on regular polygons. This column is about that fascinating insight. I'll start off with a warm-up in using complex numbers to do geometry and then move on to the Fourier series, building up to a discussion of the new interpretation. If you're unfamiliar with complex numbers, I invite you to check out the sidebar "A Quick Refresher on Complex Numbers." I'll introduce the Fourier series in the main body of the text. If it's new to you, don't worry—you'll see it's actually pretty simple by the time we get there. And the payoff is worth the journey.

Napoleon's Theorem

Let's begin with a lovely little theorem in elementary geometry:

> *Napoleon's Theorem*: Given any triangle ΔABC, erect equilateral triangles on each side (all facing inward or all facing outward), and connect the centroids of each of these triangles. The resulting triangle will be equilateral.

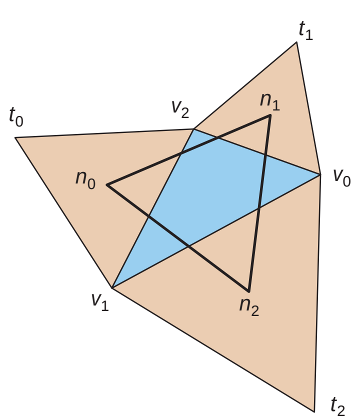
You may recall that the *centroid* of a triangle is the arithmetic average of the three vertices. Figure 1 shows an example of this theorem in action. I'll call the original three points $\mathbf{V} = (v_0, v_1, v_2)$, the three points at the tips of the new triangles $\mathbf{T}$, and the three points that make up the Napoleon triangle $\mathbf{N}$.

You can prove this theorem in lots of ways. If you like working things out for yourself you may want to take a shot at it before reading on. I've seen proofs that are strictly geometric, strictly algebraic, and various combinations of the two.

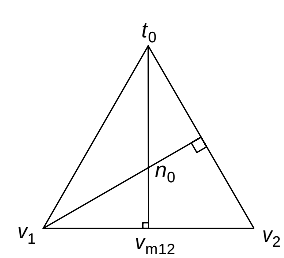
The approach I'll use here is based on representing each vertex of the triangle as a complex number. We'll carry out the construction with the complex interpretation and then prove that it's right.

Figure 2 shows the edge formed by v_1 and v_2. Although I chose this edge at random, they all work the same way. Figure 2 shows the geometry resulting from building an equilateral triangle on these points and then finding the centroid. First find point t_0 at the apex of the triangle, and drop the median down from t_0 to v_{m12}, the midpoint of v_1 and v_2. Next draw the median from v_1 to the midpoint of v_2 and t_0. The intersection of these two medians denotes the centroid, marked by the complex point n_0. It doesn't matter which two medians we use in this construction, because all three meet at n_0.

We can save time by observing that we don't have to



1 **A demonstration of Napoleon's Theorem. The original triangle, V, shown in blue, consists of points (v_0, v_1, v_2). On each edge, we build an equilateral triangle facing outwards. The new points, T, at the tip of these triangles are (t_0, t_1, t_2). The centroids of the three new triangles (n_0, n_1, n_2), are joined with heavy black lines to form the Napoleon triangle, N, which is equilateral.**



2 **Finding the centroid n_0 at the intersection of two medians.**

0272-1716/99/$10.00 © 1999 IEEE

A Quick Refresher on Complex Numbers

Complex numbers arose naturally as people thought about how to find some value of x that would satisfy $x^2 + 1 = 0$. It's a logical step in the sequence of trying to solve equations.

To begin with, consider the equation $x + 3 = 5$. Only one value of x will work (that is, only one solution exists): $x = 2$. This x is an *integer*. Now consider $3x = 5$. The solution, $x = 5/3$, is a *real* number (that is, not an integer) and a rational number, which is specified as the ratio of two integers. Getting more ambitious, suppose we want to solve $x^2 = 3$. Now the only solution is $x=\sqrt{3}$, which is also real, but *irrational*. Mathematically, irrational doesn't mean emotionally unstable, but simply nonrational (that is, $\sqrt{3}$ cannot be expressed as the ratio of two integers). Some common irrational numbers are $\sqrt{2}$, $\sqrt{3}$, e, and π.

Consider the equation $x^2 + 1 = 0$. If we just grind out a solution, we find $x=\sqrt{-1}$. Whatever this may mean, it's been given the label i, so $i=\sqrt{-1}$ (engineers often use the symbol j for this; the right interpretation is usually clear from context). The question of just what $\sqrt{-1}$ "means" has been debated for a very long time. Regardless of the philosophical interpretation, it's clearly a very convenient computational entity. We can combine a real and an imaginary number into a single new number z as $z = a + bi$, where a and b are real numbers (rational or irrational), and $i=\sqrt{-1}$. We say z is a *complex* number. Note that either part of z may be zero, so the complex numbers include all the pure reals and imaginaries.

What are the powers of i? Since anything raised to the 0 power is defined as 1, $i^0 = 1$ and anything to the first power is 1, so $i^1 = i$. By definition, $i^2 = -1$. Building on these starting points, we find $i^3 = -i$, $i^4 = 1$, $i^5 = i$, $i^6 = -1$, and so on. The pattern repeats indefinitely.

Algebra with complex numbers is straightforward. To add, just add components. If $w = a + bi$ and $z = c + di$, then $w + z = (a + c) + (b + d)i$. To multiply, treat the complex numbers as factors: $(a + bi)(c + di) = ac + bci + adi + bdi^2 = (ac - bd) + (bc + ad)i$, using $i^2 = -1$. It's natural to try to get some kind of visual take on complex numbers. Since real numbers can be plotted along an axis, we can try to put them together with *imaginary* numbers along another axis. The plus sign in the complex number $a + bi$ seems to invite the idea of replacing it with a comma, giving the 2D point (a, b).

In 1797 Caspar Wessel did just this. He placed the reals on the x-axis of a normal 2D coordinate system and the purely imaginary numbers on the y-axis, creating what was later called an Argand diagram. The complex number $z = a + bi$ is represented by the point (a, b). A purely real number r sits on the x-axis and may be thought of as $r + 0i$. A purely imaginary number such as $0 + ri$ lies on the y-axis. When a pair of axes is used this way, x identified with the reals and y with the imaginaries, it's often referred to as the *complex plane*.

Interpreting complex numbers as points on the plane gives us some more ways to understand them. We can speak of the *magnitude* of a complex number $z = a + bi$, which is simply its distance from the origin: $|z|=\sqrt{(a^2+b^2)}$and the *phase*, which is the angle it makes with the x-axis: $\theta = \tan^{-1}(b/a)$.

A remarkable property of complex numbers is revealed when we compare the series expansions for sine and cosine with the series of expansion of powers of e, Euler's constant. Many books derive these expansions. They work out to be:

$$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{3} + \frac{x^4}{4} + \frac{x^5}{5} + \frac{x^6}{6} + \cdot$$

$$\cos x = 1 - \frac{x^2}{2} + \frac{x^4}{4} - \frac{x^6}{6} + \frac{x^8}{8} - \cdots$$

$$\sin x = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} - \cdots$$

If we plug in i for x, we get the very useful identity

$$e^i = \cos 1 + i \sin 1$$

Combining this with de Moivre's theorem, which tell us $(\cos\theta + i \sin\theta) = (\cos 1 + i \sin 1)^\theta$, we find the important identity,

$$e^{i\theta} = \cos\theta + i \sin\theta$$

which relates imaginary powers of e with complex numbers or simply points on the complex plane. Notice that the magnitude of these points is 1, and the phase is simply θ. So if we know a complex number has magnitude r and phase θ, we may write it as $r(\cos\theta + i \sin\theta)$ or much more compactly as $re^{i\theta}$.

This expression, $r(\cos\theta + i \sin\theta)$, is called a *complex sinusoid*. If we plot its coordinates on the complex plane as θ moves from 0 to 2π, we get one complete circle of radius r. As θ grows beyond 2π, the sinusoid simply wraps around the circle again and again. Suppose we have two of these: $s_1 = e^{i\theta}$ and $s_3 = e^{i3\theta}$. Clearly s_3 spins around the unit circle three times faster than s_1. Just as Euclidean space builds on the idea that the x-, y-, and z-axes are orthogonal (or at right angles) to one another, so does the Fourier transform makes use of the fact that these two functions are also orthogonal (though in a more abstract way).

Writing complex numbers as points in the complex plane gives us another way to see multiplication. If $z_1 = re^{i\theta}$ and $z_2 = se^{i\varphi}$, then $z_1z_2 = (rs)e^{i(\theta+\varphi)}$. In words, this tells us that when two complex numbers are multiplied, we multiply their magnitudes and add their phases. If the magnitude of a complex number z is 1, then multiplying by z is equivalent to a rotation counterclockwise by the phase of z.

To wrap up with a bang, let's return to $e^{i\theta} = \cos\theta + i \sin\theta$ and set $\theta = \pi$. Then we arrive at one of the most remarkable expressions in all of mathematics,

$$e^{i\pi} - 1 = 0$$

which expresses a deep harmony among five fundamental constants representing number theory (0), arithmetic (1), algebra (i), trigonometry (π), and calculus (e).

actually make point t_0. Figure 3 shows the geometry of the equilateral triangle. Triangle $\Delta v_1 n_0 v_{m12}$ is a 30-60-90 triangle. If the distance $|v_1 v_2| = 1$, then $|v_1 v_{m12}| = 1/2$, and $|v_1 n_0| = 1/\sqrt{3}$. So we can find n_0 simply by taking the vector $v_2 - v_1$, rotating it 30 degrees counterclockwise, scaling it by a factor of $1/\sqrt{3}$, and adding it back to point v_1.

This is exactly the sort of geometric thing that complex numbers handle very well. If we multiply the vector $v_2 - v_1$ by a complex number $d = (1/\sqrt{3})e^{i\pi/6}$, then we're set. In symbols, $n_0 = v_1 + d(v_2 - v_1)$. This n_0 is one of the three points n_i making up the Napoleon triangle **N**.

To prove that Napoleon's Theorem is correct, we need to show that **N** is an equilateral triangle. To do that, I'll change our construction process a little (this will keep the algebra from getting too messy). We'll construct the Napoleon triangle in two steps. Starting with the original triangle **V**, create the three points **T** at the vertices of the temporary construction triangles and find the centroids of these triangles, which will give us the points **N**.

To build the set of points **T**, take each edge in turn as we did before, but rotate it by 60 degrees counterclockwise. To do this, just multiply the edge vectors by a complex number $g = e^{i\pi/3}$ representing a counterclockwise rotation of 60 degrees and add back the starting point. For example, $t_0 = v_1 + g(v_2 - v_1)$. The Napoleon points **N** are the average of the two points on each edge and the newly constructed point on **T**. For example, $n_0 = (t_0 + v_1 + v_2)/3$.

Now we're ready for the punch line. We'll prove that triangle **N** is equilateral by taking one of the edges, rotating it 120 degrees clockwise, and showing that we land exactly on top of the adjacent edge. That is, we take **N**, rotate the whole thing by 120 degrees, and get the very same triangle. Only equilateral triangles satisfy this property.

For specificity, let's follow one edge in particular—say $n_0 n_2$. Looking at Figure 1, we expect that rotating this will land us on $n_2 n_1$. We already have g lying around, which rotates by 60 degrees, so if we apply it twice, we'll get 120 degrees. Algebraically, we want to show that $g^2(n_0 - n_2) = n_2 - n_1$, or equivalently,

$$g^2(n_0 - n_2) - (n_2 - n_1) = 0$$

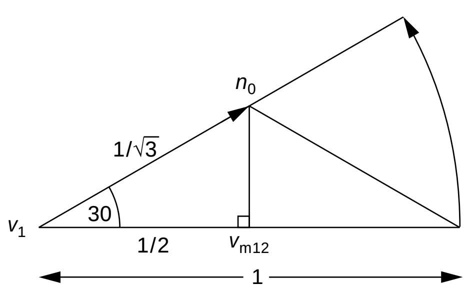
Plugging in the values derived earlier for triangles **N** (for example, $n_0 = (t_0 + v_1 + v_2)/3$) and **T** (for example, $t_0 = v_1 + g(v_2 - v_1)$) and simplifying, this becomes

$$(g^2 - g + 1)[(g-1)v_0 - (1+2g)v_1 + (2+g)v_2]/3 = 0$$

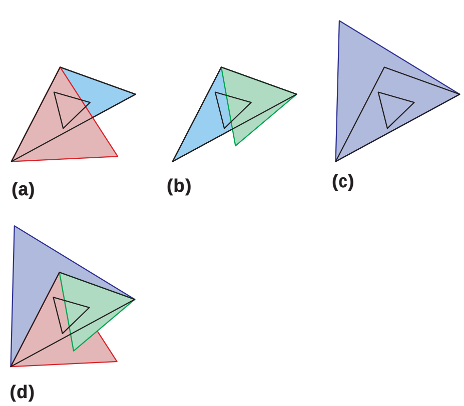
A little algebra shows that $g^2 - g + 1 = 0$, so the formula holds true. We've found that rotating this edge by 120 degrees gives us the previous edge. There was nothing special about our choice of edge, so all edges share this property, ergo **N** is an equilateral triangle. Napoleon's Theorem is indeed true!

Figure 4 shows the development of the three triangles when they point inward rather than outward and the equilateral triangle they form.

Interestingly, because of symmetry, we can tile the plane with Napoleon constructions. Figure 5 shows a tiling of the plane using the triangles of Figure 1. In Figure 6 I marked the equilateral triangles formed by the inward-pointing construction and the irregular tiling they create.



3 **To find point n_0, rotate the vector $v_2 - v_1$ counterclockwise by 30 degrees, scale it by $1/\sqrt{3}$, and then add it back into v_1.**



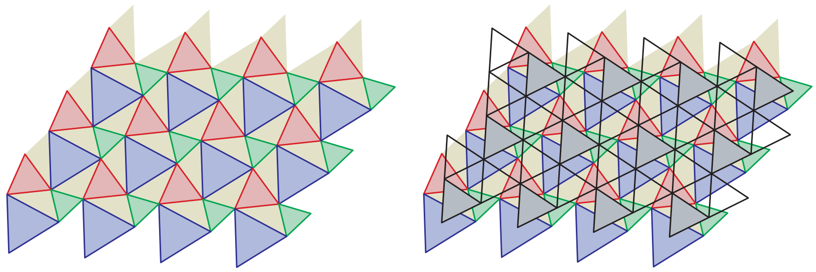
4 **Building the Napoleon triangle with inward-pointing triangles. The original triangle (in blue), the same as in Figure 1. (a) The first triangle. (b) The second triangle. (c) The third triangle. (d) The Napoleon triangle (in thick black lines) formed from the centroids of the three triangles.**

Lighting design by accident

The Fourier transform is an important tool for understanding signals of all sorts and of great value in computer graphics. I'll give a very bare-bones description of the discrete-time Fourier series, which will be just enough for the big payoff in the next section. If you're already familiar with the DTFS, you might want to just skim the next two sections.

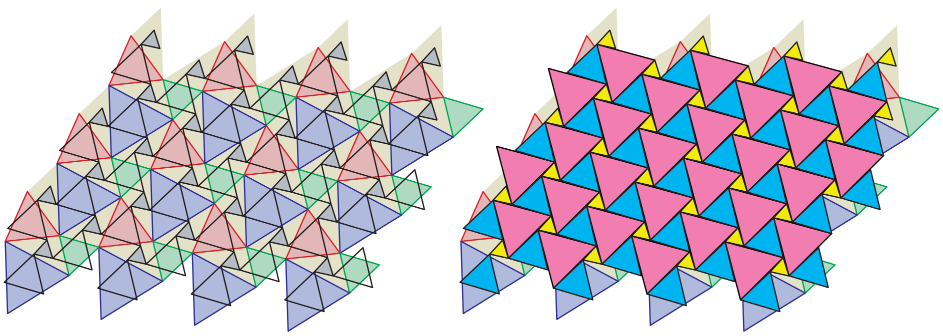
Let's start off with an analogy. Suppose you're the director of a new off-off-Broadway play. You don't have much money, so you only bought nine identical lights for your theater. Since you can't afford gels, all the lights project the same off-yellow color. The show opens next week, but you're still unhappy with the lighting for the important midnight-picnic scene.

Theatrical lights contain a built-in slot in front of the lens that holds a thin, etched metal plate, called a *gobo* or *cookie*. When the light shines through the holes in the gobo, a pattern of light and dark projects on the stage. For example, one gobo might project a shadow pattern like that of moonlight through a tree's leaves. You bought nine different gobos, and they've already been inserted in front of the nine lights. The lights have been



(a) (b)

5 Tiling with Napoleon triangles. (a) A tiling of the plane using the original triangle (yellow) and the three outward-pointing equilateral triangles built upon it (red, green, blue). (b) The tiling created by the Napoleon triangles (gray).



(a) (b)

6 The same tiling as in Figure 5, but using the Napoleon triangle built from inward-pointing triangles. (a) The tiling created by the Napoleon triangles (gray). (b) A colored version of the complete tiling.

aimed properly, but you're unhappy with the way they mix. Looking at the set, you know that some lights need to be brighter and others dimmer. The lighting board, which contains a series of sliders—one per lamp—controls this. Push the slider up, and the corresponding lamp gets brighter.

It's late at night, and everyone else has gone home. You're sitting alone in the seats staring at the set, trying to imagine which lights need to be brighter and which dimmer, when you hear a terrible crash behind you. Jerking around in your seat, you see that a great load of junk has fallen off a shelf and collapsed onto the lighting board, screwing up all the settings. But it's your lucky day, because a glance at the stage reveals that the lighting is now perfect! You only have two problems. One, you can't see the sliders on the board because the junk is obscuring them. Two, you realize you can't get the junk off without knocking all the sliders into new positions. Somehow you need to determine the positions of the sliders just from the lights themselves.

Perhaps you can eyeball their intensities. You climb up on the stage and look up at the lights, but that's hopeless—they're all much too bright to look at directly. You look at the pattern of light falling on the stage itself, but the complicated shadows cast by the gobos makes it impossible to visually distinguish how much each light contributes. You call your friend Jackie who advises you to do something strange: go through the props, get out nine napkins, and arrange them in a big three-by-three grid that covers the stage. She then advises you to use your handy light meter to measure and record how much light falls on each napkin. With those measurements safely written down, she tells you that you can now clean off the lighting board without fear, since she can recover the slider settings from knowledge of the gobo patterns and the nine readings you took.

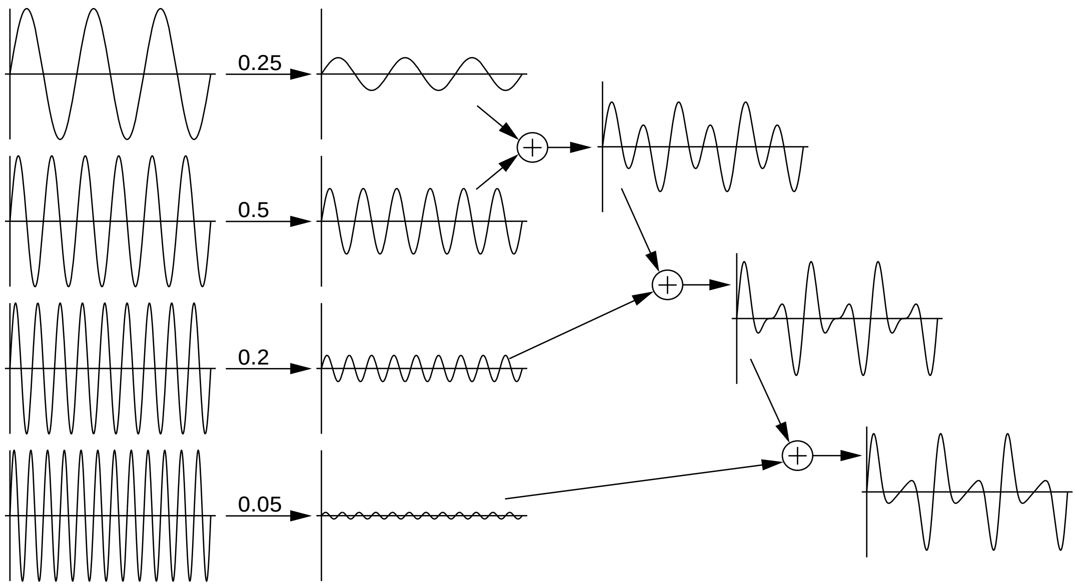
How could she possibly untangle the contributions of each of the nine lights given only nine readings throughout the stage? This is exactly the sort of job that the Fourier transform does. We change the language a bit, but the idea remains the same.

The new bits of language are the complex sinusoid (see the sidebar on "A Quick Refresher on Complex Numbers") and the idea of a *signal*. For our discussion here, a signal may be considered nothing more than an indexed list of numbers. Whether they're real, imaginary, or complex makes no difference. We'll write the signal itself in lowercase bold letters, like $\mathbf{x}$, and the elements of the signal indexed either with brackets or subscripts: $x[n]$ and x_n both refer to the nth element of the signal $\mathbf{x}$. Signals can arise in an infinite number of ways. Two of the most common are by sampling a function (for example, $x[n] = n + 2$ for $n = [0, 5]$) or else direct assignment (for example, $\mathbf{x}$ is the temperature at noon over eight successive days in May).

So in our story, the nine light readings at the napkins create a nine-element signal. We're interested in matching the lighting with the sum of nine scaled lights. In a Fourier problem, we want to match a signal with the sum of nine scaled complex sinusoids. In the story, the goal was to find the slider value for each light. In a Fourier transform, we want to find the scaling coefficient for each sinusoid.

The Fourier transform

In the one-dimensional Fourier transform, we take a set of complex sinusoids, scale them, and add them together to create a signal. If we already have a signal to start with, then we use the Fourier transform to analyze the signal and find the contribution of each sinusoid.



7 The left column shows four sine waves. From the top, they plot sin(*x*), sin(2*x*), sin(3*x*), and sin(4*x*). The middle column shows those waves after scaling by a real constant. On the right, I show the result of adding them together to build a composite real signal. The Fourier transform uses complex sinusoids, so in addition to these sine waves, it includes corresponding imaginary cosine waves.

The inverse Fourier transform turns the process around and synthesizes the signal by adding up scaled sinusoids. Figure 7 shows an example. It's a big subject that I'll cover in only the briefest way—the "Further Reading" sidebar points to some books where you can learn more.

Actually, a few different types of Fourier transform exist. In this column, I'm exclusively concerned with the DTFS. This takes as input a vector of complex samples of a signal **x**. The result is **a**, the vector of coefficients that describe the amplitude of complex sine waves of increasing frequency. By convention, the elements of **x** are written $x[n]$, while the elements of **a** are written a_k. We'll assume that we have N values of the signal **x**, and thus also N coefficients at our disposal in **a** to match **x**. The sine waves that match **x** are called the *basis functions* for the *decomposition* of **x**.

A discussion of why complex sinusoids are good basis functions would take us very far afield. I recommend consulting one of the books in the "Further Reading" sidebar if this question interests you.

Now that we've set the stage, let's look at the specifics of the transform. The sidebar on complex numbers shows that a convenient way to write the complex sine wave $\cos(\theta) + i\sin(\theta)$ is $e^{i\theta}$. Our list of sine waves begins with $\theta = 0$. That is, the first complex sine wave is $e^{i0} = 1$. This is often called the DC (direct current) value. It's not really much of a sine wave, since it's just the constant 1. We can scale this wave to add a global offset to the entire synthesized signal. The remaining waves used by the Fourier process are the complex sinusoids that have a frequency given by integer multiples of $2\pi/N$. These are $e^{i(2\pi/N)}$, $e^{i2(2\pi/N)}$, $e^{i3(2\pi/N)}$, and so on, up to $e^{i(N-1)(N-1)(2\pi/N)}$. The left column of Figure 7 shows the first few of these.

The complex coefficients a_n tell us how strongly to weight each of these waves to reconstruct the original signal **x**. The complete synthesis equation for element $x[n]$ is

$$x[n] = \frac{1}{\sqrt{\mathrm{N}}} \sum_{k\in[0,\mathrm{N})} a_k e^{ik(2\pi/\mathrm{N})n}$$

If this is new to you, don't let all the superscripts and subscripts throw you. Remember, all we're doing is adding up values of complex sinusoids to come up with a single number. In words, to find the value for element $x[n]$, we add up N different sine waves. For each wave, indexed by k that runs from 0 to $N-1$, we take the basic frequency $(2\pi/N)n$, multiply it by k, call this θ, evaluate $\cos(\theta) + i\sin(\theta)$, and scale the resulting complex number by a_k. The factor of $1/\sqrt{N}$ at the start is a necessary normalization factor, which I won't get into here.

The matching analysis equation turns the process around and computes the a_k from the signal **x**:

$$a_k = \frac{1}{\sqrt{\mathrm{N}}} \sum_{n\in[0,\mathrm{N})} x[n] e^{-ik(2\pi/\mathrm{N})n}$$

The only difference here (besides the swap of the a's and x's and their indices k and n) is that there's a minus sign in the exponent. In words, to find the kth coefficient, we step through the sine waves (this time indexed by n), take the basic frequency $-k(2\pi/N)$, multiply it by n, call this θ and evaluate $\cos(\theta) + i\sin(\theta)$, and scale this by $x[n]$.

These two equations are a matched pair. If you start with a signal **x**, analyze it to compute the vector **a**, and synthesize a new signal **x′** from that, you'll get back what you started with.

The bottom line is that the DTFS lets us take a list of complex values **x** and convert them into a list of complex coefficients **a**, and vice versa. Whether we want to think of our signal as a set of values or as a set of coefficients of complex sine waves—or both at once—makes no difference because they both describe exactly the same information. Sometimes the set of values $x[n]$ is referred to as the *signal space* representation of the signal, and the set of coefficients a_k are referred to as the *frequency space* representation.

The DTFS (and the other variants of the Fourier transform) prove useful because many operations are more convenient or meaningful on one space or the other. These two equations tell us how to easily go back and forth between the two equivalent representations. The important thing to remember is that both the values of x and the coefficients a may be complex.

Fourier polygons

What the heck has all this Fourier stuff got to do with Napoleon's Theorem? The connection, of course, is their common use of complex numbers. The point of visiting Napoleon's Theorem was to get comfortable using complex numbers to represent points in the plane and multiplying by $e^{i\theta}$ to represent a rotation by θ. The interesting thing is that this viewpoint leads to a really nice interpretation of the Fourier transform.

The DTFS discussed above is a one-dimensional kind of beast. The signal values are indexed by a single number and the coefficients are as well. Fourier transforms have been applied to two, three, and higher numbers of dimensions, and those are all useful. Usually in graphics we think of a 2D Fourier transform as operating on an image. We start with a collection of real numbers representing pixel intensities, transform those into complex amplitudes (this time of 2D complex sinusoids), and back again.

Let's reconsider the input to the DTFS. The DTFS actually operates in complex space, and it's completely appropriate to use complex numbers for the signal $x[n]$.

If we think this way, then we can consider the complex sinusoids that form the heart of the transform as complex numbers themselves. In the discussion of Napoleon's Theorem, we saw that multiplication by a complex number $se^{i\theta}$ causes a scaling by s and a counterclockwise rotation by θ. This is called a *spiral dilation*, and these are precisely the terms in the heart of the Fourier transform.

Let's write out the synthesis equation for the DTFS in matrix form. The reconstructed signal vector $\mathbf{x}$ is given by the product of a matrix $\mathbf{F}$ and the coefficient vector $\mathbf{a}$ or in symbols, $\mathbf{x} = \mathbf{Fa}$. If we write $\gamma = i2\pi/N$, the tableau form of the synthesis equation may be written as in Figure 8.

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix} = \frac{1}{\sqrt{N}} \begin{bmatrix} e^0 & e^0 & e^0 & \ldots & e^0 \\ e^0 & e^{\gamma} & e^{2\gamma} & \ldots & e^{(N-1)\gamma} \\ e^0 & e^{2\gamma} & e^{4\gamma} & \ldots & e^{2(N-1)\gamma} \\ \vdots & \vdots & \vdots & & \vdots \\ e^0 & e^{(N-1)\gamma} & e^{2(N-1)\gamma} & \ldots & e^{(N-1)(N-1)\gamma} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{N-1} \end{bmatrix}$$

8 A tableau representation of the synthesis equation for the discrete-time Fourier series. Here, $\gamma = i2\pi/N$.

Recall that $e^0 = 1$, so the top row and left column are all 1's. Notice that this matrix is symmetrical—the rows and columns can be transposed.

Now we're almost ready for the trick.

Every element of $\mathbf{F}$ is simply the complex number $\cos\theta + i \sin\theta$ for some value of θ. But each entry in the matrix also represents a point on the complex plane. And when multiplied with another complex number, it causes a spiral dilation. In fact, because the magnitude of each element of $\mathbf{F}$ is 1, they're simply rotations.

And not just any rotations, but rotations of integer multiples of the angle $2\pi/N$. Now, what do you get if you place points around the circumference of a circle, each separated by an angle $2\pi/N$? It's just a regular polygon with N sides. If $N = 3$, you get an equilateral triangle, if $N = 4$ you get a square, and so on.

Further Reading

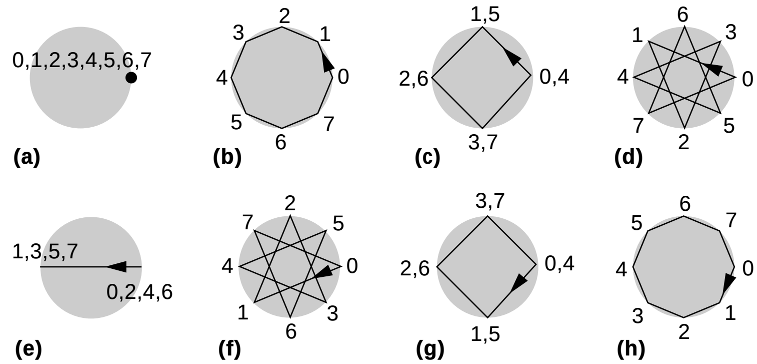
My main inspiration for this column was Alvy Ray Smith's "Eigenpolygon Decomposition of Polygons" (Technical Memo 19, Microsoft Research, Redmond, Wash., 1998), where I first saw the idea of basis polygons. Alvy describes the basis polygons as the eigenvectors of a matrix of rotations operating on polygons in the complex plane. He also provides a nice operator-based way of writing Napoleon's Theorem. Alvy credits some of his insights to *Over and Over Again* by Keng-Che Chang and Thomas W. Sederberg (Mathematical Association of America, Washington, DC, 1997).

Some online math sites feature discussions of Napoleon's Theorem. I adapted the proof given by Alexander Bogomolny for this column (http://cut-the-knot.com/proofs/napoleon_complex.html), and Scott Brodie gives a couple of other proofs (http://mirrors.org.sg./mathi/proofs/napoleon.html). You can also experiment with some interactive gadgets for playing with the construction (http://www.saltire.com/applets/advanced_geometry/napoleon_executable/napoleon.htm).

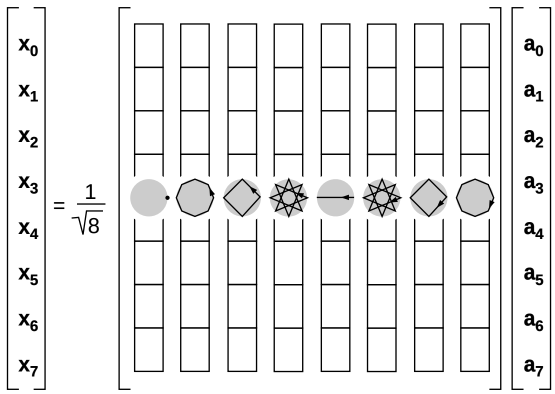
Some historical information on Napoleon's Theorem can be found in a mail thread on the Internet. I particularly recommend the postings by Michael Deakin and Julio Gonzalez Cabillon (to read the mail thread, go to http://forum.swarthmore.edu/discussions/epi-search/all.html and search for "napoleon").

The development of complex numbers in the sidebar is based on the discussion in *Introduction to Geometry* (2nd Edition) by H.S.M. Coxeter (Wiley & Sons, New York, 1969).

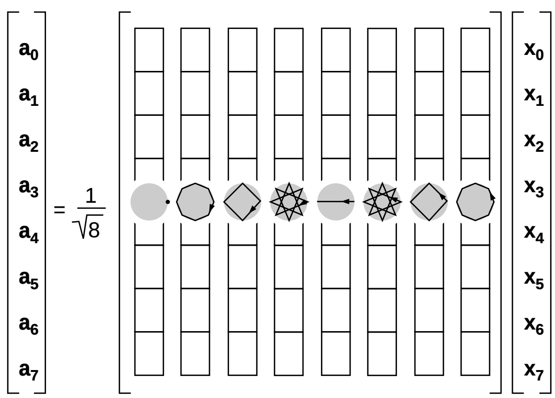
You can learn about Fourier transforms from lots of places. If you don't mind learning from a textbook, *Digital Signal Processing* by Alan V. Oppenheim, Alan S. Willsky, and Ian T. Young (Prentice Hall, Upper Saddle River, N.J., 1983) is a good book to teach yourself from. I also like *Signals and Linear Systems* (2nd Edition) by Robert A. Gabel and Richard A. Roberts (J. Wiley, New York, 1980). For a treatment of signal processing and Fourier transforms specifically targeted to the computer graphics community, I humbly offer my own book, *Principles of Digital Image Synthesis* (Morgan-Kaufmann, San Francisco, 1995).



9 The eight basis polygons, plotted in the unit circle. The points are numbered in the order they are visited. (a) The degenerate octagon, where all 8 points land on (1, 0). (b) The counterclockwise octagon. (c) The octagon consists of points that form two overlapping diamonds. (d) An 8-pointed star. (e) The octagon consists of alternating visits to (1, 0) and (–1, 0). (f) Figure 9d traversed clockwise. (g) Figure 9c traversed clockwise. (h) Figure 9a traversed clockwise.



10 A tableau form of the Fourier transform. Each basis octagon is represented by a column of the matrix.



11 A tableau form of the inverse Fourier transform. Each basis octagon is represented by a column of the matrix.

Now we're ready for the big climax. The insight we've been building up to can be expressed in two different, but equivalent, ways:

1. The rows and columns of the Fourier matrix are regular polygons in the complex plane.
2. Any N-sided polygon may be described as a weighted sum of N regular polygons.

Let's see why these statements are true and why they're so cool.

We'll start by looking at the columns of **F** and plot them in the complex plane as a sequence of points. For specificity, I'll select $N = 8$, so $\gamma = i2\pi/8 = i\pi/4$, and multiplication by e^{γ} is a counterclockwise rotation of 45 degrees.

The left-most column of **F** is simply the complex point (1, 0) repeated eight times. It will be helpful to think of this as a really degenerate octagon, where all the vertices are coincident. Figure 9a shows the results.

The second column is the sequence of points $[e^0, e^{i\pi/4}, e^{i2\pi/4}, e^{i3\pi/4}, e^{i4\pi/4}, e^{i5\pi/4}, e^{i6\pi/4}, e^{i7\pi/4}]$. If we plot these, as in Figure 9b, we get a regular octagon inscribed in a circle of radius 1. The octagon starts at (1, 0) and proceeds counterclockwise.

The third column doubles the angles, giving us the points $[e^0, e^{i2\pi/4}, e^{i4\pi/4}, e^{i6\pi/4}, e^{i8\pi/4}, e^{i10\pi/4}, e^{i12\pi/4}, e^{i14\pi/4}]$. If we remember that $e^{in\pi} = 1$ for any integer n, these points are $[e^0, e^{i\pi/2}, e^{i\pi}, e^{i3\pi/2}, e^{i2\pi}, e^{i\pi/2}, e^{i\pi}, e^{i3\pi/2}]$. In words, the eight points form a square (or diamond) that wraps around itself twice, as in Figure 9c. This is still an octagon, but a very strange one. One way of thinking about this is that we're going around the octagon, but skipping one point each time we move.

Following the pattern, the fourth column yields a star octagon, as in Figure 9d. You can think of this as our original octagon, except we skip over two points on each move. It's still an octagon, but it's a pretty twisted one.

The fifth column simplifies to the two points (1, 0) and (–1, 0) in alternation. Figure 9e plots this sequence of points. Though it looks like a line, this is again a distorted octagon.

The remaining polygons are repeats of their predecessors, taken in the opposite order and traversed in the opposite direction. The sixth column yields Figure 9f, where we have a star octagon that looks like Figure 9d, but is visited clockwise rather than counterclockwise. The remaining two columns repeat the diamond and regular octagon, except these two are also traversed clockwise.

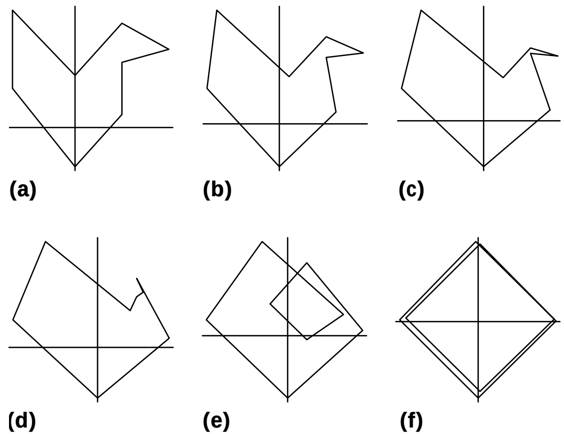
Another way of writing the matrix tableau then is as in Figure 10, where each column is indicated as a series of points around the indicated polygon.

Recall that I spoke of the Fourier transform as decomposing an input signal into a sum of basis functions, which were complex sinusoids. In the new interpretation, we can say that the Fourier transform decomposes a polygon into a sum of basis polygons, themselves simply regular polygons (though some are degenerate). Then the vector **a** represents the version of the input polygon in basis space. Converting **a** back to **x** means simply weighting the basis polygons and adding them back together again.

To see this in action, consider how to compute $x[2]$. We form the dot product of the column vector **a** and row 3 of the matrix. This row represents the third vertex of each basis polygon. Thus we find the position of each vertex in the output polygon as a weighted sum of the corresponding vertices in the basis polygons.

To see how symmetrical this all is, Figure 11 shows the analysis form that computes **a** from **x**. Notice that the only change in the matrix is that now the basis polygons are traversed in the opposite order.

Figure 12 shows an example of applying this interpretation. Here I made a little cat's head out of eight points. The pieces of the figure show what happens when I break the figure into its basis polygons, crank up the diamond component, and resynthesize the figure. In Figure 12b, I scaled up the diamond by 1.5 and later figures show higher amplitudes. You can clearly see the cat's head being pulled towards the double-wrapped diamond as that component of the shape dominates.



12 **The transformation of a cat. If you use your imagination, the octagon in (a) is something like a cat's head. Each figure shows the result of amplifying the counterclockwise "diamond" component of the Fourier transform (as shown in Figure 9c) and leaving other components unchanged. (a) The original shape. (b) Amplification of the diamond component by 1.5, (c) by 2, (d) by 3, (e) by 8, and (f) by 100.**

Note that Figure 12 isn't a "morph." In that technique, you have two line drawings representing the start and end states, and you move the points in some way from their start locations to the end locations. Rather, I have only one shape throughout—the cat's head. All that's happening is that the diamond-shaped component of the Fourier transform is being scaled up with respect to the other components so that when I reconstruct the shape, the diamond predominates. In other words, the cat's head had a diamond shape already in it (as do most octagons). All I've done here is emphasize its diamond nature over its other constituent shapes.

I think that this is a wonderful way to look at polygons. The fact that it's the Fourier representation that makes this all work out is natural and, like all the best insights, obvious in retrospect. ■

Readers may contact Glassner at Microsoft Research, e-mail glassner@microsoft.com.