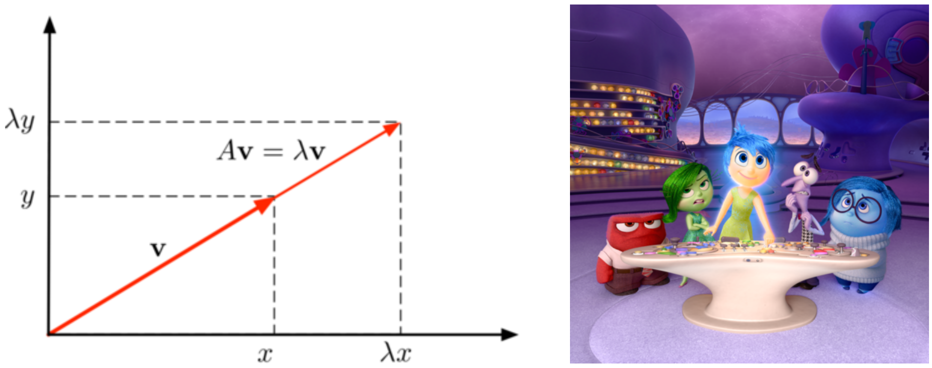
LINEAR ALGEBRA EDUCATION

Pixar's Linear Algebra

David Borland, Tim Chartier and Tabitha Peck
RENCI, The University of North Carolina at Chapel Hill, NC, 27517, USA, Borland@renci.org
Davidson College, Davidson NC, 28035, USA, tichartier@davidson.edu, tapeck@davidson.edu

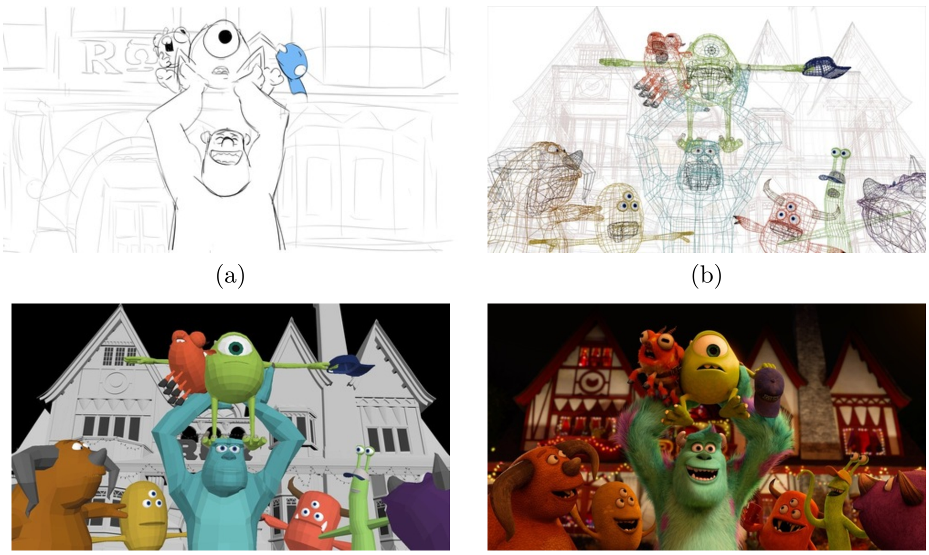
At the 2016 Academy Awards, a movie won an Oscar after dazzling millions of moviegoers with eigenvectors. Hard to believe? It is if you're imagining theaters crammed full with audiences watching the image in Figure 1(a). It's understandable when you see how eigenanalysis has helped create beloved Pixar characters like those in Figure 1(b). Indeed, essentially every frame of a Pixar film makes extensive use of linear algebra, and this includes *Inside Out*, winner of the 2016 Oscar for Best Animated Feature Film.



(a) (b)

Figure 1: Eigenvectors relate to the images in (a) and (b). Image (b) ©Disney/Pixar.

Before learning how Pixar uses eigenvectors, let's discuss the overarching steps in making computer-animated movies. Animation begins with storyboarding, and with the story come characters, as seen in Figure 2(a). The characters must be designed and then built in a computer so they can be animated. The models of the characters are created with wireframe meshes as seen in Figure 2(b), and a previsualization of the characters, using flat shading, is rendered as seen in Figure 2(c). The characters are then given textures and more detailed shading, which adds a very important element of realism to Pixar's animation, as seen in Figure 2(d).



(a) (b) (c) (d)

Figure 2: Steps in the process of making an animated film. Images ©Disney/Pixar.

Pixar used eigenvectors in the 1997 short film *Geri's Game*, which demonstrates their subdivision technique that continues to be used in Pixar films today. Subdivision smooths a wireframe mesh by subdividing each polygonal face of the mesh into smaller and smaller faces, giving the appearance of a smooth surface to provide greater realism. To understand subdivision, let's see how it works in 2D.

We'll begin by creating a square, as seen in Figure 3(a). Then, for each line segment, we find the midpoint. This step is called *split* and is seen in Figure 3(b). Next, we replace the original vertices of our square with points found by a weighted *average*. To begin, we'll use a 1-1 weighted average, also known as the 1-1 rule. That is, our original vertices and midpoints are each given the same weight. In the clockwise direction, we replace each vertex of our square with the average of its value and that of the next midpoint. This creates a polygon with eight points as seen in Figure 3(c). We can repeat this process. We *split* the new line segments to find the midpoints, then in the clockwise direction we replace the vertices of our current polygon with an *average* of the vertex and the next midpoint. Each *split* and *average* produces a new *subdivision*.

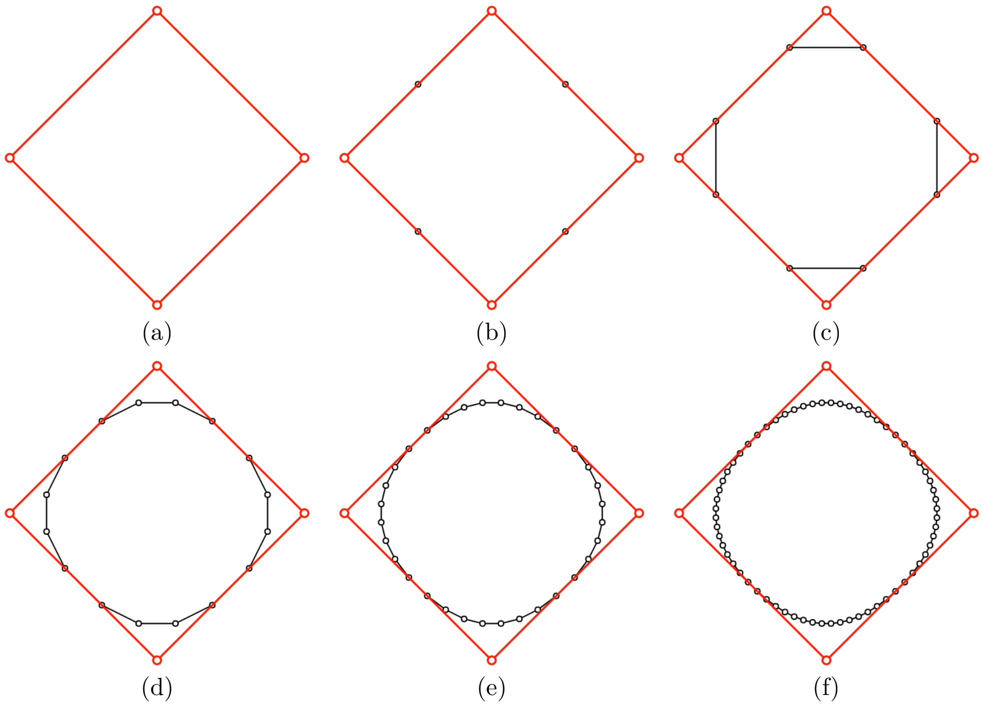


Figure 3: Starting square for subdivision (a), splitting step (b), and averaging step (c). Additional subdivision iterations produce (d), (e) and (f).

If we continue to loop through this process, we get a smoother and smoother polygon, as seen in Figure 3(d), (e) and (f), which depict successive iterates of this algorithm. In the limit, we have the smooth curve we are looking for. What can we say about the curve that would be produced after infinitely many steps? For one thing, two consecutive points get closer and closer in each iteration. So, in their limit, they meet. If you look carefully, we know at every step where two points will meet. Specifically, they'll meet at the midpoint of a line segment on the current polygon.

So we know, at every step, points that lie on that final, smooth curve. The first step gives us the midpoints of the four line segments of our polygon. At the second step, we have eight midpoints, four of which are the original four midpoints, giving us an additional four points on the final curve.

In this previous example, we used the 1-1 rule, and we have yet to use eigenanalysis. We'll get to eigenvectors in a moment, but first let's look at a 1-2-1 weighted average, a.k.a., the 1-2-1 rule. That is, for each vertex, let's take the average of the two adjacent midpoints, one in each of the clockwise and counterclockwise locations, along with two parts of the location of the vertex itself.

With this new rule, can we find a point on the final curve? With our 1-1 rule, we saw this would be the midpoint. This isn't as easily seen with the 1-2-1 rule, at least until you see how to use eigenvectors.

Let's look at a segment of the curve seen in Figure 4(a). Applying the 1-2-1 rule of subdivision creates the points A_1,

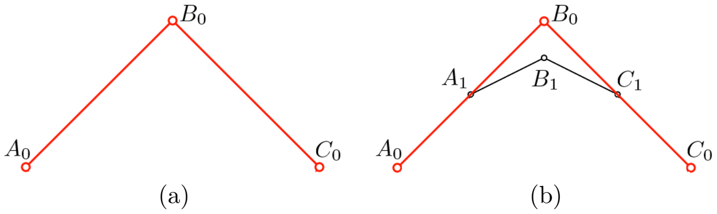


Figure 4: Steps of subdivision using the 1-2-1 rule.

B_1, and C_1 as seen in Figure 4(b). Points A_1 and C_1 are obtained by *splitting* and are defined in Equations 1 and 2.

$$A_1 = (A_0 + B_0)/2 \tag{1}$$

$$C_1 = (B_0 + C_0)/2 \tag{2}$$

What about B_1? This is created with our weighted average, in this case the 1-2-1 rule. That is, B_1 is created as an average of 1 part A_1, 2 parts B_0, and 1 part C_1, which equals $(A_1 + 2B_0 + C_1)/4$, or

$$B_1 = \frac{(A_0 + B_0)/2 + 2B_0 + (B_0 + C_0)/2}{4} = \frac{A_0 + 6B_0 + C_0}{8}. \tag{3}$$

Notice that Equations 1, 2, and 3 define A_1, B_1, and C_1 in terms of A_0, B_0, and C_0.

Using our 1-2-1 rule and the Equations 1, 2, and 3, let's find A_n, B_n, and C_n with matrices. We'll compute

$$\begin{bmatrix} A_n \\ B_n \\ C_n \end{bmatrix} = P \begin{bmatrix} A_{n-1} \\ B_{n-1} \\ C_{n-1} \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/8 & 6/8 & 1/8 \\ 0 & 1/2 & 1/2 \end{bmatrix} \begin{bmatrix} A_{n-1} \\ B_{n-1} \\ C_{n-1} \end{bmatrix}.$$

Notice, for example, that this computes A_n as an average of A_{n-1} and B_{n-1}.

We see that $P\,[A_0, B_0, C_0]^T = [A_1, B_1, C_1]^T$ and $P\,[A_1, B_1, C_1]^T = [A_2, B_2, C_2]^T$. Through substitution, we have $[A_2, B_2, C_2]^T = P^2\,[A_0, B_0, C_0]^T$. Continuing the pattern, you see that $[A_n, B_n, C_n]^T = P^n\,[A_0, B_0, C_0]^T$.

We want to know where A_n, B_n, and C_n converge as n gets bigger and bigger. First, note that as n increases, A_n, B_n, and C_n get closer together. That is, A_n, B_n, and C_n are going to converge to the same point in their limit. But where will that point be?

Here is where eigenanalysis comes in. Our matrix P can be diagonalized, such that $P = R\Lambda L$, where the columns of R are the right eigenvectors of P, the rows of L are the left eigenvectors of P, and Λ is a diagonal matrix of the eigenvalues of P. In particular,

$$R = \begin{bmatrix} 1 & -1 & 2 \\ 1 & 0 & -1 \\ 1 & 1 & 2 \end{bmatrix}, \quad \Lambda = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix}, \quad \text{and} \quad L = \begin{bmatrix} 1/6 & 4/6 & 1/6 \\ -1/2 & 0 & 1/2 \\ 1/6 & -2/6 & 1/6 \end{bmatrix}.$$

Remember that we derived $[A_n, B_n, C_n]^T = P^n\,[A_0, B_0, C_0]^T$, but what we are interested in is what happens as $n \to \infty$. First, note that $R = L^{-1}$. Then, if we solve for P^2, we have

$$P^2 = (R\Lambda L)^2 = R\Lambda L R\Lambda L = R\Lambda^2 L.$$

Following the same pattern, we see that $P^n = R\Lambda^n L$, which can be rewritten as

$$P^n = R\Lambda^n L = R \begin{bmatrix} 1^n & 0 & 0 \\ 0 & 1/2^n & 0 \\ 0 & 0 & 1/4^n \end{bmatrix} L.$$

So, as $n \to \infty$,

$$P^n = R\Lambda^n L \;\to\; R \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} L.$$

Using our 1-2-1 rule, as $n \to \infty$,

$$P^n \to \begin{bmatrix} 1/6 & 4/6 & 1/6 \\ 1/6 & 4/6 & 1/6 \\ 1/6 & 4/6 & 1/6 \end{bmatrix}.$$

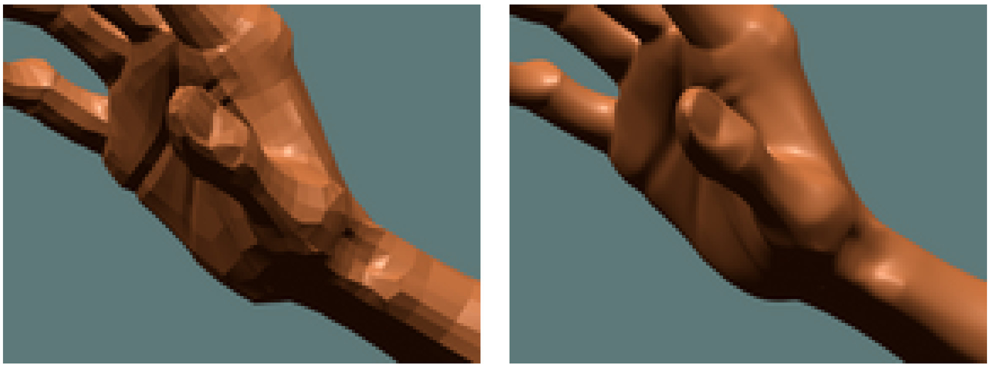
This is just a matrix of the left eigenvector of P with the largest eigenvalue! This eigenvector becomes the coefficients of a weighted average to find values on the final, smooth curve. To see how, consider $(A_1 + 4B_1 + C_1)/6$. Using Equations 1, 2, and 3 and making substitutions, we get

$$\frac{A_1 + 4B_1 + C_1}{6} = \frac{(A_0 + B_0)/2 + (A_0 + 6B_0 + C_0)/2 + (C_0 + B_0)/2}{6} = \frac{A_0 + 4B_0 + C_0}{6}.$$

So, $(A_1 + 4B_1 + C_1)/6 = (A_0 + 4B_0 + C_0)/6$. This implies that $(A_2 + 4B_2 + C_2)/6 = (A_0 + 4B_0 + C_0)/6$, which further implies $(A_n + 4B_n + C_n)/6 = (A_0 + 4B_0 + C_0)/6$.

Let's call A_∞ the point that A_n is converging to as n grows. Our work shows that $(A_\infty + 4B_\infty + C_\infty)/6 = (A_0 + 4B_0 + C_0)/6$. Now, remember, A_∞ and C_∞ are converging to B_∞ since the points get closer and closer to each other with every step. So, $(B_\infty + 4B_\infty + B_\infty)/6 = (A_0 + 4B_0 + C_0)/6$, or, $B_\infty = (A_0 + 4B_0 + C_0)/6$. In other words, we know in one step what point will lie on the final, smooth curve. For the 1-1 rule we saw earlier, this was the midpoint. Now we have a formula for the 1-2-1 rule, which came from computing an eigenvector.

This is how Pixar creates surfaces for its movies. In what parts of a Pixar film do we see the result of eigenanalysis? Every character you've seen since *Geri's Game* has been a product of eigenanalysis. This significant step in animation changed how characters were designed. Initial wireframes, like that of Geri's hand in Figure 5(a), which is analogous to the images in Figure 2(b) and (c), could be smoothed through subdivision. Applying subdivision to Geri's hand in Figure 5(a) produces the image in (b).



(a) (b)

Figure 5: The hand of Geri in Pixar's animated short *Geri's Game*, before (a) and after (b) subdivision. Images ©Disney/Pixar.

Grab a Pixar movie like *The Incredibles*, *Finding Nemo*, *Up*, *Brave*, *Toy Story 3*, or *Monsters University*. Or go to the theater and see one of the recent releases. Enjoy the film and, of course, enjoy seeing eigenvectors, or at least the result of eigenanalysis on the screen! Want to try it yourself? Visit `http://math365.org/lifeislinear/Subdivision/Subdivision.html` and you can experiment with subdivision, and even animate your shapes and become, in a sense, your own animation studio.