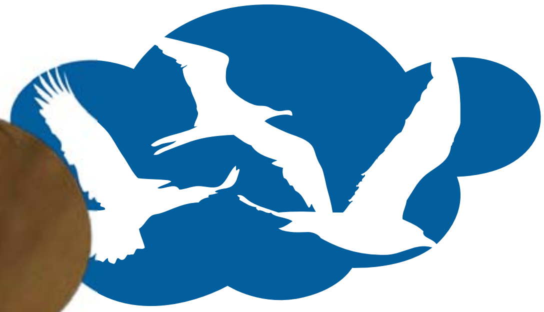
# Brad Pitt gets Angelina Jolie and I get... Math?



By
P. H. Trinh
Oxford University

## Seventh Grade Blues

Back in the seventh grade, one of the girls told me I looked like Keanu Reeves. Seriously, I was hanging upside-down on the jungle gym, minding my own business, and she just walked over and told me like it was no biggie.

Then she giggled like a moron and ran away.

Now while this singular moment of brazen flattery would become the highlight of my paste-eating academic career, I was also torn. On one hand, I wondered how *anyone* could confuse Keanu (black shades, gothic trench coat, totally awesome) with me (pubescent, angst-ridden, gawky)? Was this all some awfully cruel and sadistic joke girls liked to play on unsuspecting boys?

On the other hand, maybe — *maybe* she was on to something. Maybe somewhere — *somehow*, behind all that bad acne and ruffled hair, my hidden Keanu-like features beckoned faintly, like some distant lighthouse obscured by fog.

Today, however, thanks to the latest advances in facial recognition, I no longer have to wonder: she was right.

> **A more computationally efficient way would be to examine these faces as a statistical whole rather than as the sum of their parts.**

## Who in the Land is Fairest of All?

MyHeritage.com is an internet-based company that offers you the chance to see which celebrity you most resemble. Remember how in Snow White, the queen has a magical mirror which provides her with uninhibited flattery? This is the same, but like, tons better.

After a free signup, you upload a large-ish jpeg of your mug, then let the software crank away. My personal resemblance results were: Brad Pitt (71%), Keanu Reeves (63%), Luke Perry (63%), and Matt Damon (63%).

Brad Pitt? *Really*? Matt Damon? *Really*? Who *wouldathunk*? But y'know, as I gaze into the mirror... well... yes, I see it now. Definitely. We're practically brothers!

How does it all work? Is this actual science or just deceptive flattery? To understand how facial recognition works, we're going to have to delve into the mathematics behind the algorithm.

## Recognizing Faces

Suppose we were given someone's picture. How might we go about identifying that person from a large database of faces?

> **How might we go about identifying that person from a large database of faces?**

One way we can go about it is by identifying the characteristics of the subject — perhaps the person has small lips, or a pointed chin, or distinctive eyes. From here, we then consult the database, going from picture to picture, each time isolating the features of the faces and

> **One way we can go about it is by identifying the characteristics of the subject**

checking for a match.

Whilst this might work, it would also be a lot of work; algorithms would need to be defined to analyse each desired feature. Imagine having to do this for each of the thousands of faces that stream pass the gates at a football match or in a busy airport.

As stored in a computer, a picture is nothing more than a great big grid of dots (or pixels).

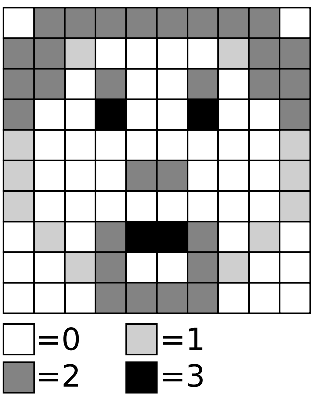
Now in the abstract theory of linear algebra, these grids of pixels are called 'vectors'.

A more computationally efficient way would be to examine these faces as a statistical *whole* rather than as the sum of their parts. This is similar to the difference between identifying a city by its landmarks and identifying the same city by the density of its roads, the clusters and heights of its buildings, its downtown areas, and so on.

## A Picture is Worth a Thousand Digits

*Snap!* But what are pictures, *really?*

As stored in a computer, a picture is nothing more than a great big grid of dots (or pixels). If the picture is greyscale, each pixel is associated with a number from 0 to 255 representing its brightness, from pitch black (0) to pure white (255).




| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 0 |
| 2 | 2 | 1 | 0 | 0 | 0 | 0 | 1 | 2 | 2 |
| 2 | 2 | 0 | 2 | 0 | 0 | 2 | 0 | 2 | 2 |
| 2 | 0 | 0 | 3 | 0 | 0 | 3 | 0 | 0 | 2 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 2 | 3 | 3 | 2 | 0 | 1 | 0 |
| 0 | 0 | 1 | 2 | 0 | 0 | 2 | 1 | 0 | 0 |
| 0 | 0 | 0 | 2 | 2 | 2 | 2 | 0 | 0 | 0 |

Figure 1: A picture is nothing more than a large grid of numbers.

Now in the abstract theory of ***linear algebra***, these grids of pixels are called ***vectors***. You've probably encountered vectors before in Physics class and in fact, these 'face vectors' are quite similar.

Like vectors representing force or motion, these new 'face vectors' have a ***magnitude*** (an overall brightness), as well as a ***direction***. Moreover, they can be added, subtracted, multiplied, and manipulated like most other mathematical quantities — the only difference is that they inhabit some higher-dimensional ***face space***, rather than the two or three dimensional physical world we live in.

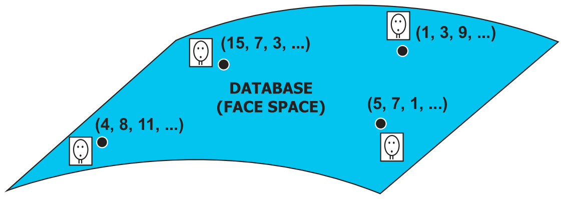



Figure 2: Faces can be identified as coordinates in a higher dimensional plane.

## What's Your Eigenface Basis?

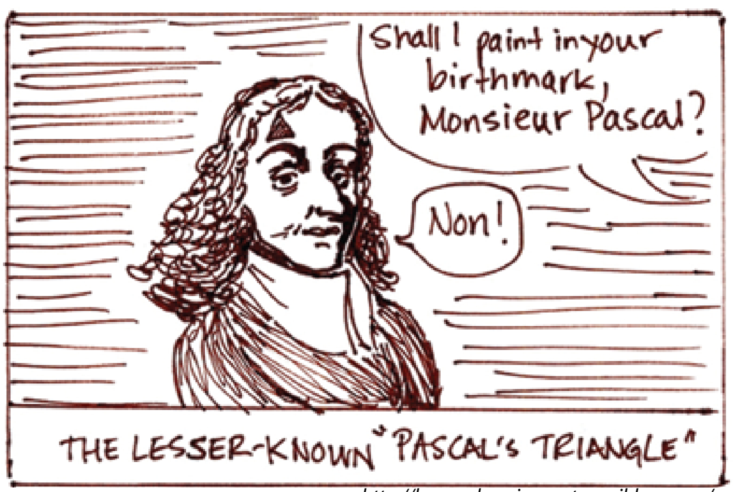
However, face spaces are complicated affairs — they're high dimensional boxes stuffed with a large number of faces, each face containing thousands of pixels.

...we look to construct a small group of pictures representing the general facial patterns of the database. This small but crucial group is called the eigenface basis.

It would thus be foolish to try and compare each face pixel by pixel; instead we look to construct a small group of pictures representing the general facial patterns of the database. This small but crucial group is called the ***eigenface basis***.

Think of how, when we analyse the motion of a ball flying through the air, we break the motion into its horizontal and vertical components. These two components provide a fundamental basis capable of describing any arbitrary motion.

Similarly, once the eigenface basis is found using linear algebra, each face in the database can then be expressed using certain percentages of each of




http://brownsharpie.courtneygibbons.org/

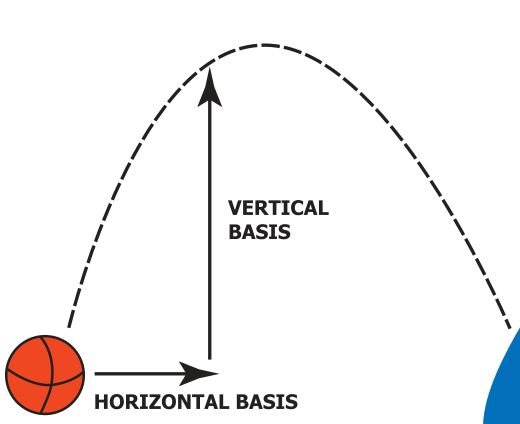



Figure 3: Vectors can be decomposed into basis elements.

the building blocks. For example, we may say that a picture is composed of 10% of the first eigenface, 25% of the second, 4% of the third, and so on.

The beauty of this treatment is that even in a large database, each unique face can be expressed very simply using its eigenface decomposition. We no longer have to express each face using thousands of pixels; now, like a simple recipe in which the eigenfaces are the key ingredients, the entire database can be reconstructed as it was before.

**In optimal conditions (with good lighting, a representative database, front-facing pictures, etc.), a simple eigenface routine might produce accurate readings of up to 90%.**

**Unfortunately, real life is never that simple, and one must contend with a multitude of 'noisy' factors.**

## A Problem of Distance

Now imagine each face in the database, represented in terms of its eigenface percentages, akin to coordinates lying in some higher-dimensional plane. Our test subject (which may or may not lie in the database) is then projected onto this plane by expressing it in terms of the eigenface components.

Now the problem of recognising the subject becomes as simple as finding the shortest distance (or closest match) between our subject and the faces in the database, a process aided enormously by the fact that each face is now represented by only a handful of eigenface components.

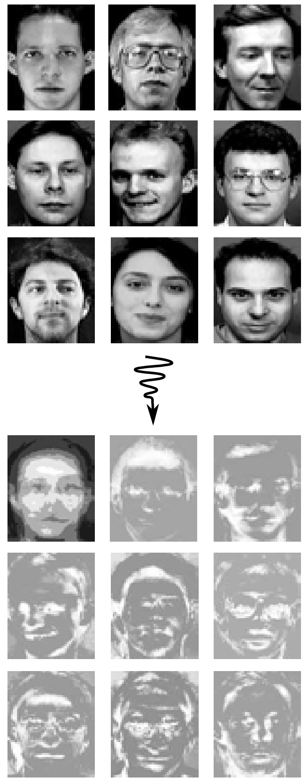


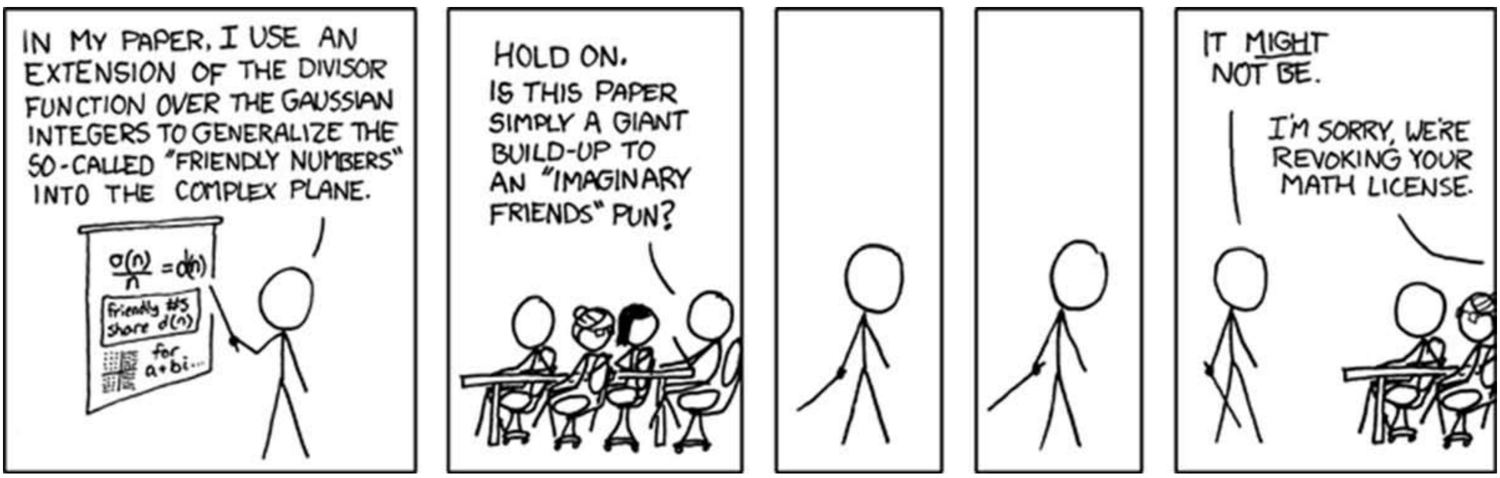
Figure 4: A database of faces can be used to construct an eigenface basis. Afterwards, a face under scrutiny can be decomposed into different percentages of each eigenface (Faces courtesy of AT&T Cambridge)

## The Future and You

But really, just how accurate are these eigenface algorithms?

In optimal conditions (with good lighting, a representative database, front-facing pictures, etc.), a simple eigenface routine might produce accurate readings of up to 90%.

Unfortunately, real life is never that simple, and one must contend with a multitude of 'noisy' factors. These include variance in pose (person facing at an angle), obstructions (sunglasses or other people), resolution, lighting, and so on. Despite this, however, the science of facial recognition has steadily improved to the point where today, it is becoming a standard for many military, security, and commercial applications.




http://xkcd.com/

But I digress. You see, the whole point of this article was that the inner workings of facial recognition is nothing but science and mathematics. That's right. It's not half-drunken, hand-wavy speculation that I resemble Brad Pitt and Keanu Reeves. I really, *really* do. It's backed up with science and everything.

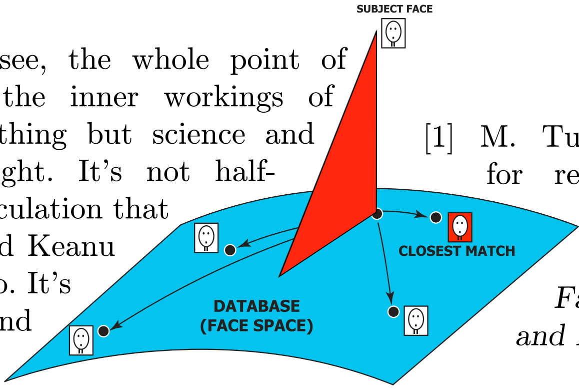


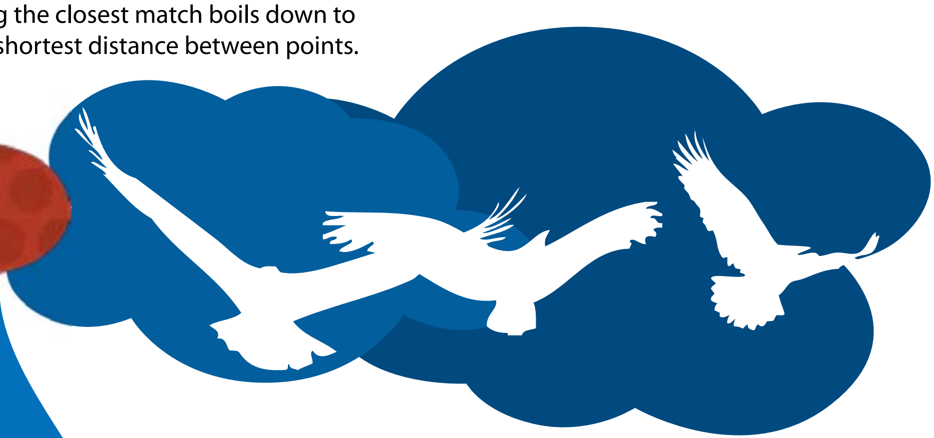

Figure 5: Finding the closest match boils down to calculating the shortest distance between points.

## FURTHER READING

[1] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience (1991) 3 (1)*, 71-86.

[2] W, Zhao and R. Chellappa. *Face Processing: Advanced Modeling and Methods*, Academic Press, 2006.

# HEXAGONAL CODES



By
Anthony Quas
and
Peter J. Dukes
University of Victoria

The idea for this article arose from a simple magic trick based on error-correcting codes. This activity has been used in *Math Mania*, an outreach event in which students and staff members at the University of Victoria visit local elementary schools for hands-on math activities with students.

In the trick, the young student is presented with a rectangular array of cups, some face up and some face down. While the magician looks away, the subject is asked to invert (i.e. flip) any cup. Looking back at the cups, the magician then is able to determine which cup the student inverted, without any memorizing! To do this trick, the cups are initially put into a configuration in which each row and each column contains an even number of cups that are the right way up, as in Figure 1. After a cup is inverted, the column in which it sits contains an odd number of cups the right way up (there is either one more or one less than before); and similarly the row in which it sits contains an odd number of cups the right way up. This allows the magician to figure out which row and column the cup that the student inverted is in. Of course this is enough to figure out which cup was inverted.

Figure 1: Rectangular array of cups used in the magic trick.

In this article, we will discuss the connections between this magic trick and error-correcting codes, and investigate an extension where hexagonal arrays replace the rectangular array of cups.

It is shown that the hexagonal pattern corresponds to a more robust code (and hence a flashier magic trick!) More on this later; we now comment on codes and their uses.

In a basic model of a communication system there is a source that produces some sort of data that is to be sent to a receiver. A typical first step in this process is for the data to be transformed into binary digits (***bits*** - 0s and 1s), with strings of several bits representing one piece of data. If we want to use strings of $k$ bits to represent pieces of data then there are $2^k$ different strings. This means that if there are $N$ different pieces of data, then we need to choose $k$ with $2^k \geq N$ so that all of the possibilities can be encoded. For example, if the data that we want to transmit are letters of the alphabet, then since $2^4 < 26 \leq 2^5$ we need to use