Stephanie Orgill
Linear Algebra

Image Manipulation via Matrices

Understanding the makeup of a digital image is an essential part to manipulating that image by changing, or *transforming*, its matrix. The matrix equation A**x** = **b** is at the center of matrix transformations. Think of matrix A as an object that is acting on vector **x** by multiplication, producing a new vector A**x**. By thinking of the matrix equation in this way, we can see that finding a solution to the equation is the same as finding all vectors **x** in R^n which can be transformed into vector **b**. The transformation from **x**⟶A**x** follows a set of rules which preserve the operations of vector addition and scalar multiplication:

1. A(**u** + **v**) = A**u** + A**v**
2. A(*c***u**) = *c*A**u**

With these rules in mind, we can begin to use linear transformations to translate, shear, scale, rotate, and reflect images in 2 -D space.

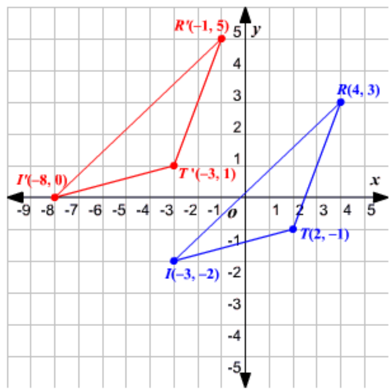
Transformations in 2-D Space

Translating

A transformation that occurs when an image is moved from one location to another without changing size, orientation, or shape is a translation. Translation is done purely from vector addition. We say that a vector **x** is translated by a vector **v** to **x + v** since the act of adding **v** to **x** is to move **x** in a direction parallel to the line through both **p** and 0. As an example, let's take the image of triangle with vertices at (2, -1), (4, 3), and (-3, -2). Say we wish to translate this image 5 units left and 2 units up. Graphically, this means we need to subtract 5 from each x value and add 2 to each y value using a transformation matrix. In matrix form, this looks like:

$$\begin{bmatrix} 2 & 4 & -3 \\ -1 & 3 & -2 \end{bmatrix} + \begin{bmatrix} -5 & -5 & -5 \\ 2 & 2 & 2 \end{bmatrix} = \begin{bmatrix} -3 & -1 & -8 \\ 1 & 5 & 0 \end{bmatrix}$$

This translation visually results in the below blue triangle sliding left and up to form the red triangle:

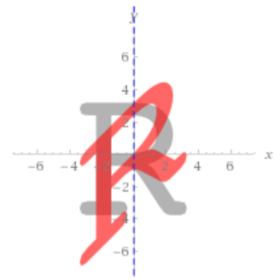


Shearing

A transformation T from $R^2 \longrightarrow R^2$ defined by T(**x**) = A**x** is called a shear transformation. Shearing a matrix causes each point, or each m_{ij} , to be displaced in a fixed direction by an amount proportional to its distance from a line parallel to that direction. Visually, shearing turns a square image into a parallelogram. Shearing can occur along the vertical or horizontal axes. The matrix to compute a horizontal shear by a scalar k is $\begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix}$, while the matrix to compute a vertical shear is $\begin{bmatrix} 1 & 0 \\ k & 1 \end{bmatrix}$. Take, for example, the following vertical shear: $(x,y) \longrightarrow (x, x+y)$. In matrix form, this becomes:

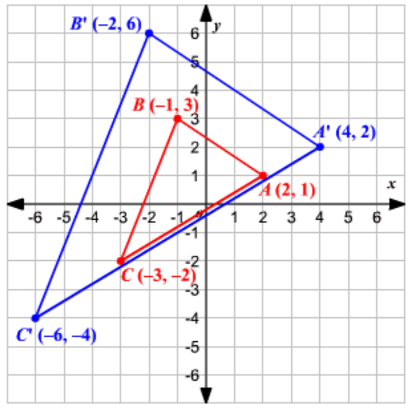
$$\begin{bmatrix} x \\ y \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ x+y \end{bmatrix}$$

Shown in grey is the original image, which the red sheared transformation above it:

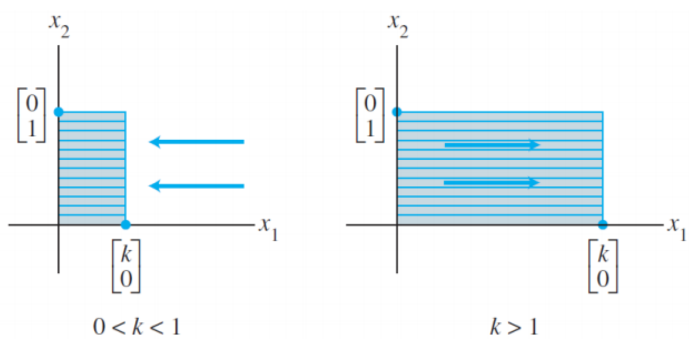


Scaling

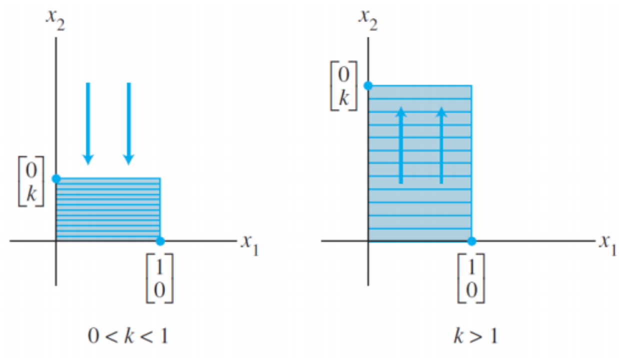
There are two main types of scaling: contraction and dilation. Contraction occurs when a matrix transformation T is multiplied by a scalar value r such that T(**x**) = r**x** when 0 ≤ r ≤ 1**.** This causes the vectors to become smaller but retain their location and direction. For the image, this means that the image becomes smaller but otherwise remains the same. Dilation occurs when T is multiplied by a scalar k such that T(**x**) = k**x** when k > 1. Similarly to contraction, dilation causes the vectors to retain their location and direction, but grow larger. For the image, this means that the image scales larger. As a simple example, take a 2x3 matrix defined as $\begin{bmatrix} 4 & -2 & -6 \\ 2 & 6 & -4 \end{bmatrix}$. This forms the image of a triangle with vertices at (4, 2), (-2, 6), and (-6, -4). This triangle can be either dilated or contracted by multiplying it by a scalar. Let's uniformly contract the image by multiplying it by k = 1/2 to get: $\begin{bmatrix} 2 & -1 & -3 \\ 1 & 3 & -2 \end{bmatrix}$ This creates the same image, but scaled down to half the size.



Contracting and dilating images can be made more specific, as well. If you do not want the image to contract or dilate uniformly and instead want to contract or dilate it only in one direction, the appropriate matrix can be applied. To contract or dilate horizontally, multiply the image matrix by $\begin{bmatrix} k & 0 \\ 0 & 1 \end{bmatrix}$. To contract or dilate vertically, multiply the image matrix by $\begin{bmatrix} 1 & 0 \\ 0 & k \end{bmatrix}$. It is easy to see that this is an identity matrix with only one element scaled by k. See below for a visual example of horizontal contraction and dilation:



As well as a visual example of vertical contraction and dilation:

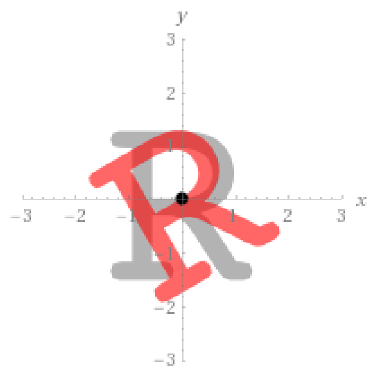


Rotating

Images can be rotated around a central point (for example, the origin) through an angle θ, with a counterclockwise rotation for a positive angle. While a rotation may not intuitively seem linear, it is since it obeys the linear transformation rules for scalar multiplication and vector addition. The transformation matrix for this calculation is as follows: $\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}$. As a basic example, we can rotate an image counterclockwise 30 degrees around the xy axis. $\cos(30) = \frac{\sqrt{3}}{2}$ and $\sin(30) = \frac{1}{2}$. The transformation of such a matrix is as follows: (x,y) ⟶ ($\frac{\sqrt{3}}{2}x - \frac{y}{2}, \frac{x}{2} + \frac{\sqrt{3}}{2}y$). In matrix form, this is:

$$\binom{x}{y} \longrightarrow \begin{bmatrix}\frac{\sqrt{3}}{2} & -\frac{1}{2}\\ \frac{1}{2} & \frac{\sqrt{3}}{2}\end{bmatrix} * \begin{bmatrix}x\\y\end{bmatrix} = \begin{bmatrix}\frac{\sqrt{3}}{2}x - \frac{y}{2}\\ \frac{x}{2} + \frac{\sqrt{3}}{2}y\end{bmatrix}$$

Below is an image that has been rotated by 30 degrees (in red) with the original image in grey.



Reflecting

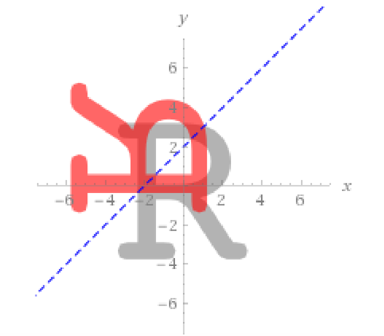
Another common type of linear transformation is reflection, and it is what it sounds like: flipping the image over a line so that it looks like a mirror image of the original image. There are many types of reflection transformation, so here are a few common ones (presuming the original image starts in quadrant one):

- Reflection through the x-axis: $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
- Reflection through the y-axis: $\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$
- Reflection through the x = y line: $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
- Reflection through x = -y line: $\begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$
- Reflection through the origin: $\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$

As an example of reflection, here is a transformation matrix that will flip an image cross the line (y = x) and translated by 2 units. As noted above, we use the matrix $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.

$$\begin{bmatrix} x \\ y \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} -2 \\ +2 \end{bmatrix} = \begin{bmatrix} y - 2 \\ x + 2 \end{bmatrix}$$

Here is a visual example of the above calculation:



Transformations in 3-D space

While the examples above are simple by design to help with clean explanation, the same logic applies to images in 3-D space. The main difference is that transformations in 3-D space use 4x4 matrices rather than 2x2. The use of 4x4 matrix may seem odd, since we are moving objects in 3-D space, but it's for a good reason: certain types of transformations only require 3x3 coordinate systems (such as rotation and scaling), while other transformations (like translations) require a fourth dimension. This fourth dimension, w, is "projective space" and coordinates within this space are called homogeneous coordinates. Homogenous coordinates, often depicted as 4-D vectors $\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$, are used in computer science as well as robotics and mechatronics to program movements through 3-D space.

To better understand projective space, imagine a 2-D image being projected onto a screen. The x and y dimensions are the horizontal and vertical lengths of the screen, and the w dimension can be viewed as the distance from the projector to the screen. The value of w affects the scale of the image; as the value of w grows (the distance from the projector to the screen grows), the image grows larger, while if w gets smaller, the image also gets smaller. The value of w works the same way in 3-D space; when w increases, the coordinates expand and when w decreases, the coordinates shrink closer together. Generally, w will be set equal to 1 when taking a 3-D space with vector $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$ to 4-D space with vector $\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$ so that the object remains the same size. Thus, when w=1, it has no effect on the x, y, and z values. Since we will need to be able to perform multiple types of transformations on objects in 3-D space, we use 4x4 matrices for all transformations – even the transformations that only require 3 dimensions to compute, like rotations. To emphasize how essential it is to use 4x4 matrices for movement in 3-D space, we must remember that a 4x4 matrix cannot be multiplied against a 3-D vector by the laws of matrix multiplication. So all 3-D vectors must be described in four dimensions in order to transform it in 3-D space using a 4x4 transformation matrix. To summarize, homogenous coordinates for 3-D space have an extra fourth dimension w, which scales the x, y, and z dimensions and allows for translations and perspective projection transformations of objects in 3-D space.

As a demonstration of transformations in 3-D space, imagine that we want to transform a cube in 3-D space. The cube can be translated by applying the below matrix, with values in the x, y, and z spots to indicate how many units to move and in what direction along the x, y, and z axes:

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The cube can be scaled by applying the below matrix, with values in the x, y, and z spots to indicate the scaling factor in each direction along each axis. If the scaling value of x, y, and z are the same, then the scaling of the cube is uniform and the cube retains its shape but changes size. Otherwise the scaling is not uniform and the object is deformed. The matrix is as follows:

$$\begin{bmatrix} x & 0 & 0 & 0 \\ 0 & y & 0 & 0 \\ 0 & 0 & z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The cube can be sheared by applying the below matrix, with values in the x, y, and z spots to indicate in which direction the cube will be sheared. As an example, a shear in the direction of the x-axis has the following matrix:

$$\begin{bmatrix} 1 & \cot(x) & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The cube can be rotated by applying the below matrix, with angle values in the x, y, and z spots to indicate around which axis the cube will be shared. For example, a rotation about the z-axis has a transformation matrix like this:

$$\begin{bmatrix} \cos(z) & \sin(z) & 0 & 0 \\ -\sin(z) & cos(z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

While a rotation about the x-axis has the following transformation matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(x) & \sin(x) & 0 \\ 0 & -\sin(x) & \cos(x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

And a rotation about the y-axis has the following transformation matrix:

$$\begin{bmatrix} \cos(y) & \sin(y) & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(y) & \cos(y) & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For example, let's say an object is located along the vector **x** = [4 3 2 1], and we want to translate it by -2 in the x direction, 3 in the y direction, and 1 in the z direction. Furthermore, we want to rotate it by 45 degrees along the z axis and 30 degrees along the y axis. Lastly, we want to shear it in the x-direction by 45 degrees. The combined transformation matrix (T) is as follows, in order of translation, z rotation, y rotation, and shear: T =

$$\begin{bmatrix} 1 & 0 & 0 & -2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(45) & \sin(45) & 0 & 0 \\ -\sin(45) & cos(45) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(30) & \sin(30) & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(30) & \cos(30) & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & \cot(45) & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T = $\begin{bmatrix} \frac{\sqrt{3}}{2\sqrt{2}} & \frac{3+\sqrt{3}}{2\sqrt{2}} & 0 & -2 \\ \frac{\sqrt{3}}{2\sqrt{2}} & \frac{1-\sqrt{3}}{2\sqrt{2}} & 0 & 3 \\ -\frac{1}{2} & \frac{\sqrt{3-1}}{2} & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

T**x** = $\begin{bmatrix} \frac{\sqrt{3}}{2\sqrt{2}} & \frac{3+\sqrt{3}}{2\sqrt{2}} & 0 & -2 \\ \frac{\sqrt{3}}{2\sqrt{2}} & \frac{1-\sqrt{3}}{2\sqrt{2}} & 0 & 3 \\ -\frac{1}{2} & \frac{\sqrt{3-1}}{2} & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \\ 2 \\ 1 \end{bmatrix}$ =

Multiplying the location vector **x** by T will result in x being rotated, translated, and sheared as indicated above.

Conclusion

Linear transformations can seem simple on the surface, especially when applied to simple 2-D images. But when applied to 3-D images and moving images, linear transformations are an essential part of the process. The matrix operations that control the linear transformation is simple, consistent, and reliable. From capturing an object on film perfectly such that the perspective is just right for the viewer, to programming an object in 3-D space to move (such as a robot), linear transformations make up the math behind it all.

References

Coordinate Systems and Transformations. www.uio.no/studier/emner/matnat/ifi/INF3320/h03/undervisningsmateriale/lecture3.pdf.

"Explaining Homogeneous Coordinates & Projective Geometry." *Explaining Homogeneous Coordinates & Projective Geometry - Tom Dalling*, www.tomdalling.com/blog/modern-opengl/explaining-homogenous-coordinates-and-projective-geometry/.

Kay, Jennifer. "Introduction to Homogeneous Transformations & Robot Kinematics." *Elvis.rowan.edu*, elvis.rowan.edu/~kay/papers/kinematics.pdf.

Lay, David C. *Linear Algebra and Its Applications.* 5th Ed. Boston: pearson/Addison-Wesley, 2006. Print.

Linear Transformations in Graphics. graphics.ics.uci.edu/ICS6N/NewLectures/App4.pdf.

"Matrices and Digital Images." *Matrices and Digital Images*, www.uff.br/cdme/matrix/matrix-html/matrix-en.html.

"Matrix Transformations | Linear Algebra | Math." *Khan Academy*, Khan Academy, www.khanacademy.org/math/linear-algebra/matrix-transformations.

"Wolfram|Alpha Examples: Geometric Transformations." *Wolfram|Alpha*, www.wolframalpha.com/examples/mathematics/geometry/geometric-transformations/.