What we should know about Linear Algebra

- 3D Coordinate geometry
- Vectors in 2 space and 3 space
- Dot product and cross product – definitions and uses
- Vector and matrix notation and algebra
- Properties (matrix associativity but **NOT** matrix commutativity)
- Matrix transpose and inverse – definition, use, and calculation
- Homogenous coordinates

Shopping example summary

- From Transformations lecture…
- Column vector for quantities, q: $\begin{bmatrix} 6 \\ 5 \\ 1 \\ 2 \end{bmatrix}$
- Row vector for corresponding prices at the stores (P):

store A (East Side)	[0.20 0.93 0.64 1.20]
store B (B & C)	[0.65 0.95 0.75 1.40]
store C (Store 24)	[0.95 1.10 0.90 3.50]

- Sum:

$$P(All) = \begin{bmatrix} totalCost_A \\ totalCost_B \\ totalCost_C \end{bmatrix} = \begin{bmatrix} 0.20 & 0.93 & 0.64 & 1.20 \\ 0.65 & 0.95 & 0.75 & 1.40 \\ 0.95 & 1.10 & 0.90 & 3.50 \end{bmatrix} \begin{bmatrix} 6 \\ 5 \\ 1 \\ 2 \end{bmatrix}$$

- The totalCost vector is determined by row-column multiplication where row = price, column = quantities, i.e. dot product of price row with quantities column

Calculating Cost for More than One Buyer

- For two buyers, we need matrix-matrix multiplication. The new buyer matrix will have two columns (one for each buyer)

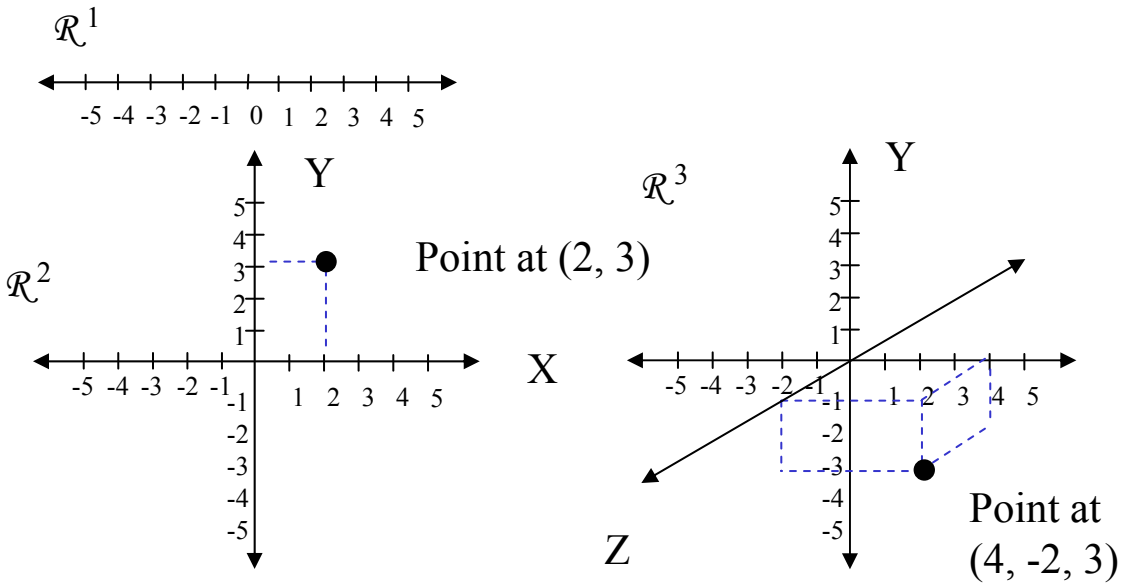
3x3 matrix 3x4 4x2

$$P(All) = \begin{bmatrix} totalCost_{1A} & totalCost_{2A} \\ totalCost_{1B} & totalCost_{2B} \\ totalCost_{1C} & totalCost_{2C} \end{bmatrix} = \begin{bmatrix} 0.20 & 0.93 & 0.64 & 1.20 \\ 0.65 & 0.95 & 0.75 & 1.40 \\ 0.95 & 1.10 & 0.90 & 3.50 \end{bmatrix} \begin{bmatrix} 6 & 3 \\ 5 & 2 \\ 1 & 8 \\ 2 & 4 \end{bmatrix}$$

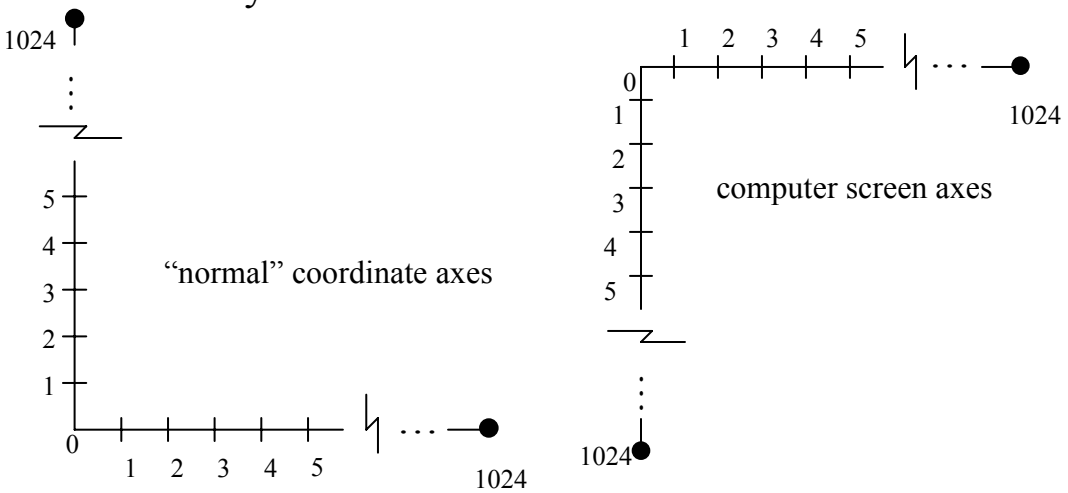
- For n different buyers, simply extend the two buyer situation. Now, we have a buyer matrix which has n columns

Cartesian Coordinate Space

- Examples: one, two and three dimensional real coordinate spaces

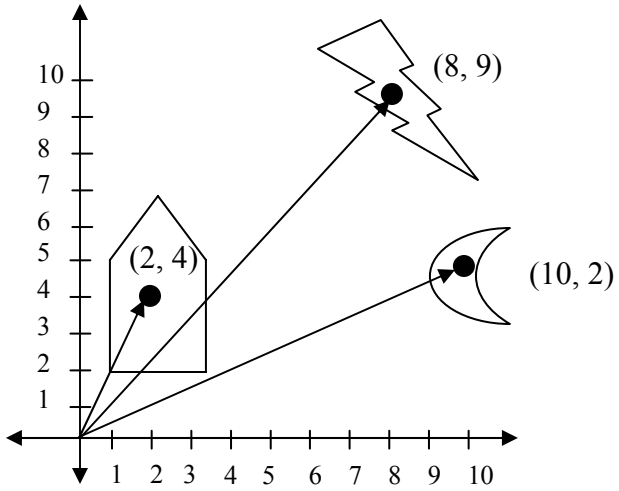


- Real numbers: between any two real numbers on an axis there exists another real number
- Compare with the computer screen, a positive integer coordinate system



Vectors & Vector Space (1/2)

- Consider all locations in relationship to one central reference point, called origin



- A vector tells you which direction to go with respect to the origin, and length of the trip
  - A vector does **not** specify where the trip begins, to continue this analogy

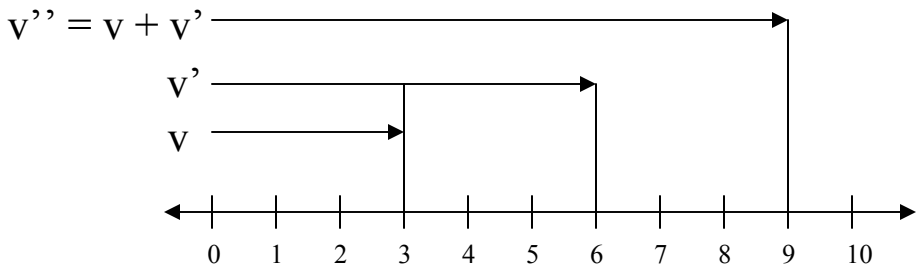
Vectors & Vector Space (2/2)

- Vectors are used extensively in computers to
  - represent positions of vertices of objects
  - determine orientation of a surface in space ("surface normal")
  - represent relative distances and orientations of lights, objects, and viewers in a 3D scene, so the rendering algorithms can create the impression of light interacting with solid and transparent objects (e.g., vectors from light sources to surfaces)

Vector addition

Vector addition in $\Re^1$

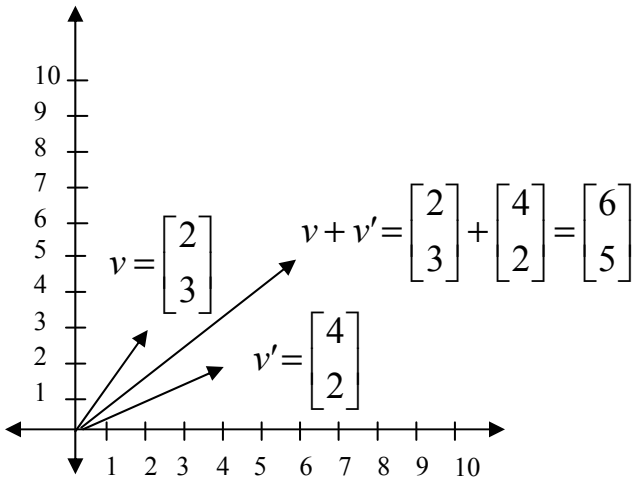
- Familiar addition of real numbers



- v = [3], v' = [6], v'' = [9]

Vector addition in $\Re^2$

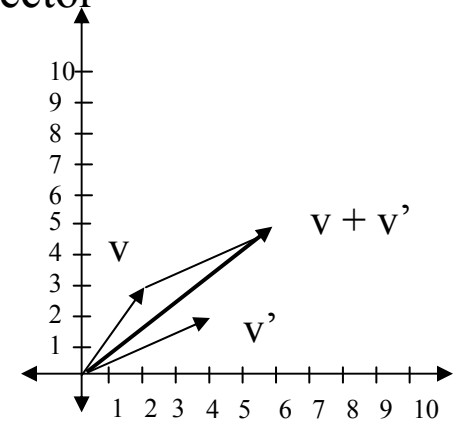
- The x and y parts of vectors can be added using addition of real numbers along each of the axes (component-wise addition)



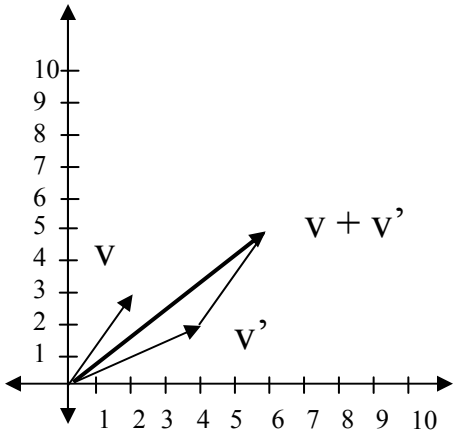
- Result, v + v', plotted in $\Re^2$ is the new vector

Adding Vectors Visually

- v’ added to v, using the parallelogram rule: take vector from the origin to v’; reposition it so that its tail is at the head of vector v; define v+v’ as the head of the new vector

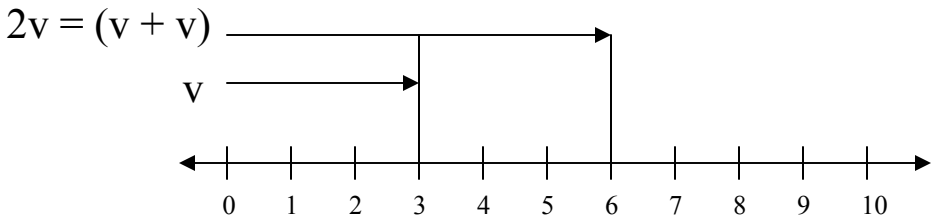


- or, equivalently, add v’ to v

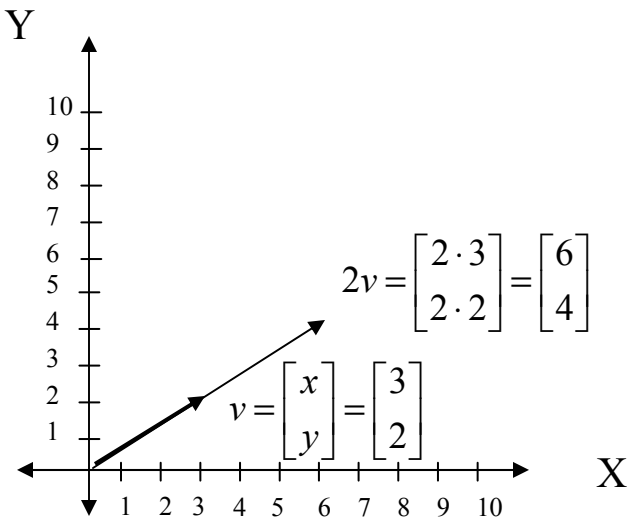


Scalar Multiplication (1/2)

- On $\Re^1$, familiar multiplication rules

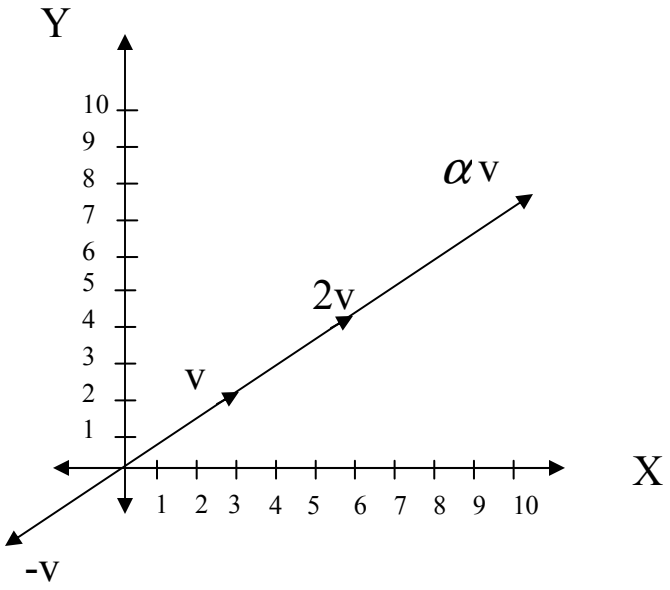


- On $\Re^2$ also



Scalar Multiplication (2/2) : Linear Dependence

- Set of all scalar multiples of a vector is a line through the origin
- Two vectors are linearly dependent when one is a multiple of the other



- Definition of dependence for three or more vectors is trickier

Basis Vectors of the plane

- The unit vectors (i.e., whose length is one) on the x and y-axes are called the standard basis vectors of the plane
- The collection of all scalar multiples of $\begin{bmatrix}1\\0\end{bmatrix}$ gives the first coordinate axis
- The collection of all scalar multiples of $\begin{bmatrix}0\\1\end{bmatrix}$ gives the second coordinate axis
- Then any vector $\begin{bmatrix}x\\y\end{bmatrix}$ can be expressed as the sum of scalar multiples of the unit vectors:

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

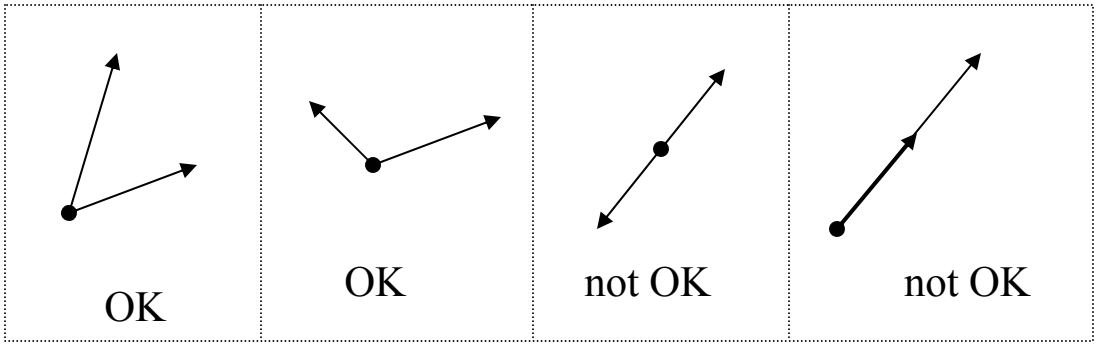
- We call these two vectors **basis** vectors for $\Re^2$ because any other vector can be expressed in terms of them
  - **This is a very important concept. Make sure that you understand it.**

Non-orthogonal Basis Vectors

- $\begin{bmatrix}1\\0\end{bmatrix}$ and $\begin{bmatrix}0\\1\end{bmatrix}$ are perpendicular. Is this necessary?
- Questions rephrased: can we make any vector $\begin{bmatrix}n\\m\end{bmatrix}$ from scalar multiples of random vectors $\begin{bmatrix}a\\b\end{bmatrix}$ and $\begin{bmatrix}c\\d\end{bmatrix}$
- Or, is there a solution to the following, where α and β are unknown:

$$\begin{bmatrix}n\\m\end{bmatrix} = \alpha\begin{bmatrix}a\\b\end{bmatrix} + \beta\begin{bmatrix}c\\d\end{bmatrix} = \begin{bmatrix}\alpha a + \beta c\\ \alpha b + \beta d\end{bmatrix}$$

  This is just a linear system of two equations. When is this unsolvable? How does that make sense geometrically?
- The vectors cannot be linearly dependent.



Uses of the dot product

- Define length or magnitude of a vector
- Normalize vectors (generate vectors whose length is 1, called unit vectors)
- Measure angles between vectors
- Determine if two vectors are perpendicular
- Find the length of a vector projected onto a coordinate axis
- There are applets for the dot product under *Applets -> Linear Algebra*

Rule for Dot product

- Also known as scalar product, or inner product. The result is a scalar (i.e., a number, not a vector)
- Defined as the sum of the pairwise multiplications
- Example:

$$\begin{bmatrix} a & b & c & d \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = ax + by + cz + dw$$

- Note, the result is not a vector of the component-wise multiplications, it is a scalar value
- Many CPUs support the dot product of floats in a MAC vector instruction: Multiply and accumulate, that takes two 1-D arrays as operands (this instruction exists because the dot product is computed so often in so many different applications)

Finding the Length of a Vector

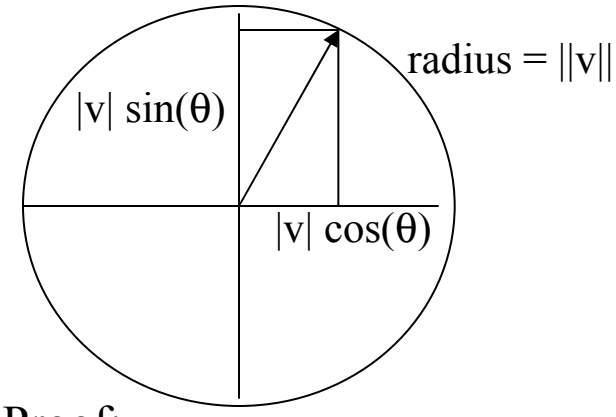
- The dot product of a vector with itself, $(v \bullet v)$, is the square of the length of the vector:

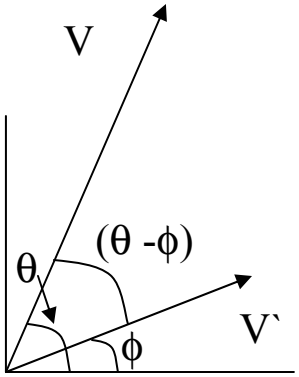
- We define the norm of a vector (i.e., its length) to be $\|v\| = \sqrt{v \bullet v}$
- Thus, $(v \bullet v) \geq 0$ for all v, with $(v \bullet v) = 0$ if and only if $v = 0$
- V is called a unit vector if $\|v\| = \sqrt{v \bullet v}$, and is denoted $\hat{v}$
- To make an arbitrary vector v into a unit vector, i.e. to "normalize" it, divide by the length (norm) of v, which is denoted ||v||. Note that if v = 0, then its unit vector is undefined. So in general (with the 0 exception) we have:

$$\hat{v} = \frac{1}{\|v\|} v$$

Finding the Angle Between Two Vectors

- The dot product of two non-zero vectors is the product of their lengths and the cosine of the angle between them: $v \cdot v' = \|v\|\|v'\|\cos(\theta - \phi)$





Proof:

$$v = [v_x v_y] = \|v\|[\cos\theta \sin\theta]$$
$$v' = [v'_x v'_y] = \|v'\|[\cos\phi \sin\phi]$$
$$v \bullet v' = \left(\|v\|\begin{bmatrix}\cos\theta \\ \sin\theta\end{bmatrix} \bullet \|v'\|\begin{bmatrix}\cos\phi \\ \sin\phi\end{bmatrix}\right) = \|v\|\|v'\|\left(\begin{bmatrix}\cos\theta \\ \sin\theta\end{bmatrix} \bullet \begin{bmatrix}\cos\phi \\ \sin\phi\end{bmatrix}\right)$$
$$= \|v\|\|v'\|(\cos\theta\cos\phi + \sin\theta\sin\varphi)$$

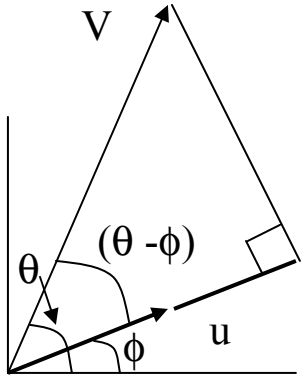
and, by a basic trigonometric identity

$$\cos\theta\cos\phi + \sin\theta\sin\phi = \cos(\theta - \phi)$$
$$\text{so, } v \bullet v' = \|v\|\|v'\|\cos(\theta - \phi)$$

More Uses of the Dot Product

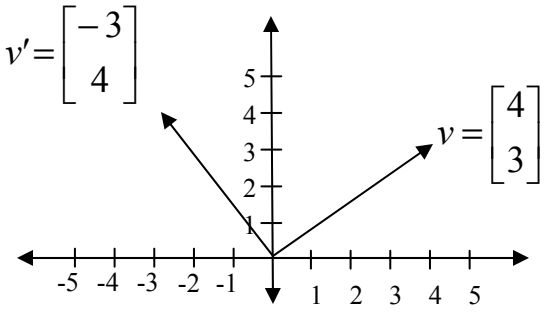
Finding the length of a projection

- If u is a unit vector, then $v \bullet u$ is the length of the projection of v onto the line containing u



Determining right angles

- Perpendicular vectors always have a dot product of 0 because the cosine of 90° is 0
- Example: $v = \begin{bmatrix} 4 \\ 3 \end{bmatrix}$ and $v' = \begin{bmatrix} -3 \\ 4 \end{bmatrix}$



$$v \bullet v' = xx' + yy' = (4 \cdot -3) + (3 \cdot 4) = 0$$

$$v \bullet v' = \|v\| \|v'\| \cos\left(\frac{\pi}{2}\right) = \|v\| \|v'\| \bullet 0 = 0$$

Cross Product

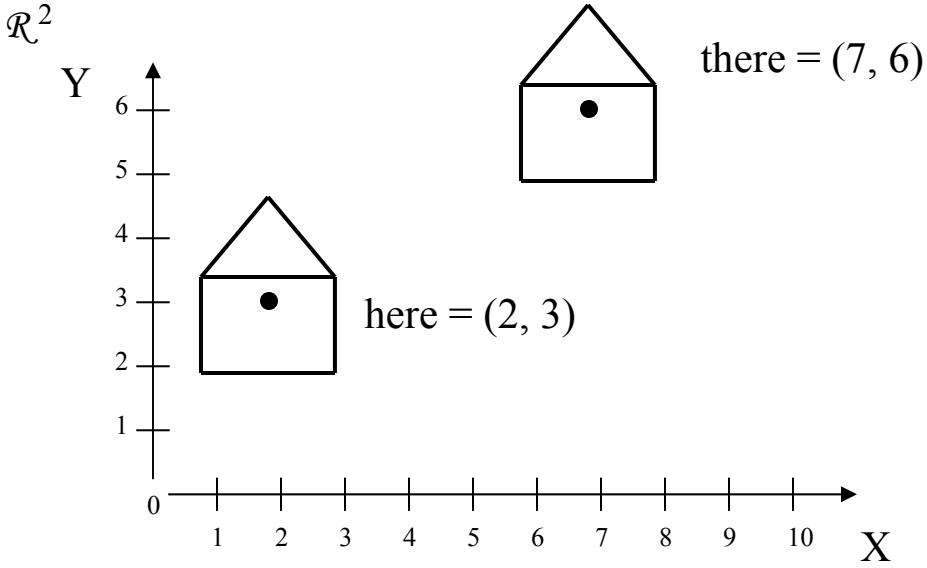
- Cross product of $\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$ and $\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$, written $\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \times \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$ is defined as: $\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \times \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}$
- The resulting vector is perpendicular to both original vectors. That is, it is normal to the plane containing the two vectors.
- Its length is equal to the area of the parallelogram formed by the two vectors
- Thus, we can write:
  $\|v_1 \times v_2\| = \|v_1\| \|v_2\| \sin\theta$, where θ is the angle between v_1 and v_2
  Note that the cross-product does not generate a normalized vector (a vector of unit length).
- An easier way to represent the math for the cross product:
- $$\vec{a} \times \vec{b} = \det\left(\begin{bmatrix} \hat{i} & \hat{j} & \hat{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix}\right)$$
  (This is for those who know how to get the determinant of a 3x3 matrix. i, j, k are the unit basis vectors.)
- Cross product follows right-hand rule, so switching order of vectors gives vector in opposite direction.
  $$\vec{a} \times \vec{b} \neq \vec{b} \times \vec{a}, \text{ in fact } (\vec{a} \times \vec{b}) = -(\vec{b} \times \vec{a})$$

Algebraic Properties of Vectors

- Commutative (vector) $v + v' = v' + v$
- Associative (vector) $(v + v') + u = v + (v' + u)$
- Additive identity There is a vector 0, such that for all v, $(v + 0) = v = (0 + v)$
- Additive inverse For any v there is a vector –v such that v + (–v) = 0
- Distributive (vector) $r(v + v') = rv + rv'$
- Distributive (scalar) $(r + s)v = rv + sv$
- Associative (scalar) $r(sv) = (rs)v$
- Multiplicative identity For any v, $1 \in \Re$, $1 \cdot v = v$

Problem: Translating Objects

- Goal: move the house from "here" to "there"



- Now that we have vectors and know how to manipulate them, this isn't hard
- We need to move the house by 5 in x and 3 in y
- So just add $\begin{bmatrix} 5 \\ 3 \end{bmatrix}$ to each vertex

What is a Matrix?

- A matrix can be easily visualized as a vector of vectors. Specifically (n) rows of (m) dimensional vectors. Intuitively, it's a 2D array of numbers
- For the purposes of this class, matrices are used to represent transformations which act on points, vectors, and other matrices
- Let's quickly recap the types of transformations we would like to be able to accomplish…

Translation

- Component-wise addition of vectors

  $v' = v + t$ where $v = \begin{bmatrix} x \\ y \end{bmatrix}, \quad v' = \begin{bmatrix} x' \\ y' \end{bmatrix}, \quad t = \begin{bmatrix} dx \\ dy \end{bmatrix}$

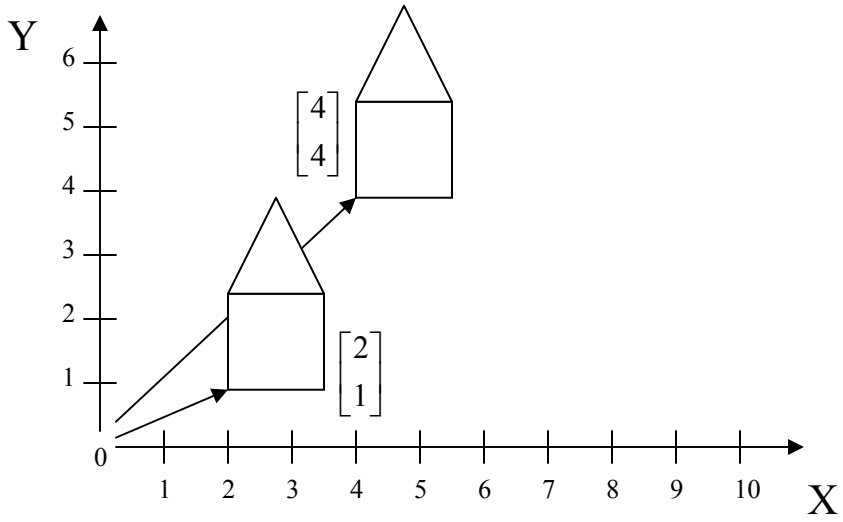
  and

  $x' = x + dx$

  $y' = y + dy$

- To move polygons: just translate vertices (vectors) and then redraw lines between them
- Preserves lengths (isometric)
- Preserves angles (conformal)

dx = 2
dy = 3



Note: House shifts position relative to origin

NB: A translation by (0,0), i.e. no translation at all, gives us the identity matrix, as it should

Scaling

- Component-wise scalar multiplication of vectors

$v' = Sv$ where $v = \begin{bmatrix} x \\ y \end{bmatrix}, \quad v' = \begin{bmatrix} x' \\ y' \end{bmatrix}$

and $S = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$

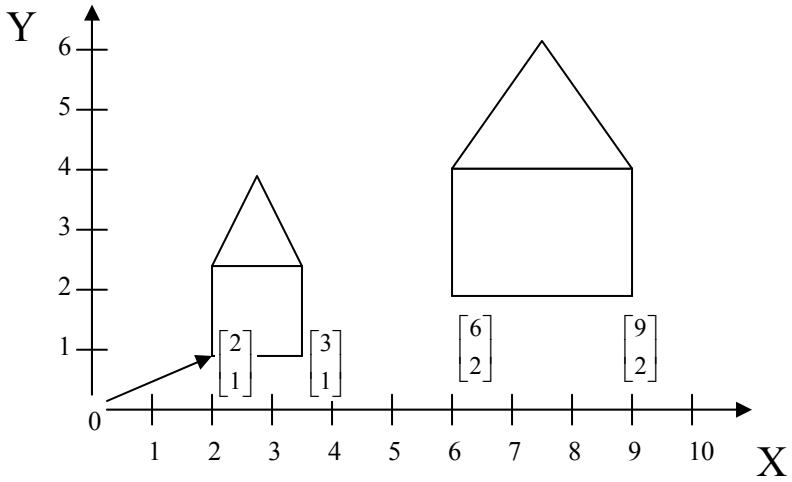
$x' = s_x x$

$y' = s_y y$

- Does not preserve lengths
- Does not preserve angles (except when scaling is uniform)

$s_x = 3$

$s_y = 2$



Note: House shifts position relative to origin

NB: A scale of 1 in both directions, i.e. no scale at all, gives us the identity matrix, as it should

Rotation

- Rotation of vectors through an angle θ

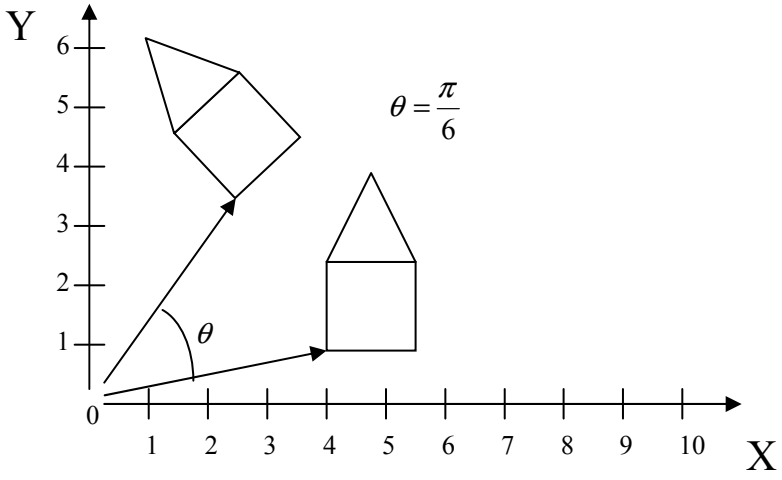
$v' = R_\theta v$ where $\quad v = \begin{bmatrix} x \\ y \end{bmatrix}, \quad v' = \begin{bmatrix} x' \\ y' \end{bmatrix}$

$$R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

and

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$

- Proof is by double angle formula
- Confused? **Look at basis vectors individually**
- Preserves lengths and angles



- Note: house shifts position relative to the origin

NB: A rotation by 0 angle, i.e. no rotation at all, gives us the identity matrix, as it should

Digression: Types of Transformations

- Projective ⊃ affine ⊃ linear
- Linear: preserves parallel lines, acts on a line to yield either another line or point. The vector [0, 0] is always transformed to [0, 0]. Examples: Scale and Rotate.
- Affine: preserves parallel lines, acts on a line to yield either another line or point. The vector [0, 0] is <u>not</u> always transformed to [0, 0]. Examples: Translate, as well as Rotate, and Scale (since all linear transformations are also affine).
- Projective: parallel lines <u>not necessarily</u> preserved, but acts on a line to yield either another line or point (not curves). Examples: you'll see a transformation in our camera model which is not Linear or Affine, but is projective. Translate, Rotate, and Scale are all Projective, since this is even more general than Affine.
- All of these transformations send lines to lines, therefore we only need to store endpoints
  - Note that we're just talking about transformations here, and not matrices specifically (yet)

Sets of Linear Equations and Matrices

- To translate, scale, and rotate vectors we need a function to give a new value of x, and a function to give a new value of y
- Examples
  - For rotation

$$x' = (x\cos\theta - y\sin\theta)$$

$$y' = (x\sin\theta + y\cos\theta)$$

  - For scaling

$$x' = s_x x$$

$$y' = s_y y$$

- These two, but not translation, are of the form

$$x' = ax + by$$

$$y' = cx + dy$$

- A transformation given by such a system of linear equations is called a linear transformation and is represented by a matrix:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Matrix-Vector Multiplication (1/2)

$$v' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \end{bmatrix} = Mv$$

$$\text{where } M = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, v = \begin{bmatrix} x \\ y \end{bmatrix}, v' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

- The new vector is the dot product of each row of the matrix with the original column vector. Thus, the kth entry of the transformed vector is the dot product of the kth row of the matrix with the original vector.
- Example: scaling the vector $\begin{bmatrix} 3 \\ 6 \end{bmatrix}$ by 7 in the x direction and 0.5 in the y direction

$$\begin{bmatrix} 7 & 0 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 3 \\ 6 \end{bmatrix} = \begin{bmatrix} (7 \cdot 3) + (0 \cdot 6) \\ (0 \cdot 3) + (0.5 \cdot 6) \end{bmatrix} = \begin{bmatrix} 21 + 0 \\ 0 + 3 \end{bmatrix} = \begin{bmatrix} 21 \\ 3 \end{bmatrix}$$

Matrix-Vector Multiplication (2/2)

- In general:

$$\begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_n \\ b_1 & b_2 & b_3 & \cdots & b_n \\ c_1 & c_2 & c_3 & \cdots & c_n \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ m_1 & m_2 & m_3 & \cdots & m_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdots \\ x_n \end{bmatrix} = \begin{bmatrix} (a_1x_1)+(a_2x_2)+(a_3x_3)+\cdots+(a_nx_n) \\ (b_1x_1)+(b_2x_2)+(b_3x_3)+\cdots+(b_nx_n) \\ (c_1x_1)+(c_2x_2)+(c_3x_3)+\cdots+(c_nx_n) \\ \cdots \\ (m_1x_1)+(m_2x_2)+(m_3x_3)+\cdots+(m_nx_n) \end{bmatrix}$$

- Can also be expressed as:

$$MX = \begin{bmatrix} \sum_{i=1}^{n} a_i x_i \\ \sum_{i=1}^{n} a_i x_i \\ \cdots \\ \sum_{i=1}^{n} m_i x_i \end{bmatrix} = \begin{bmatrix} a \bullet x \\ b \bullet x \\ \cdots \\ m \bullet x \end{bmatrix}$$

Algebraic Properties of Matrices

Properties of matrix addition

- Commutative: $A + B = B + A$
- Associative: $(A + B) + C = A + (B + C)$
- Identity: There is a matrix 0, such that, for all A, $(A + 0) = A = (0 + A)$
- Inverse: For any A, there is a –A such that $A + (-A) = 0$
- Distributive (matrix): $r(A + B) = rA + rB$
- Distributive (scalar): $(r + s)A = rA + sA$

Properties of matrix multiplication

- NOT commutative: $AB \neq BA$
- Associative (matrix): $(AB)C = A(BC)$
- Associative (scalar): $(rs)A = r(sA)$
- Distributive (vector): $A(v + v') = Av + Av'$
- Identity: There is a matrix I, such that, for all A, $AI = A = IA$
- Inverse (more later): For some A, there is a matrix A^{-1} such that: $AA^{-1} = I = A^{-1}A$

Matrix Multiplication, visually

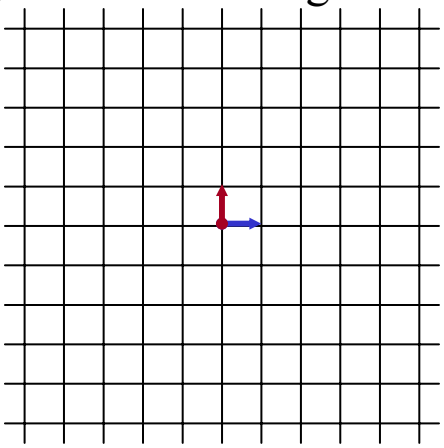
- Now that we know that matrix multiplication is distributive over vectors, we can understand linear transformations better…
- Suppose we have some matrix, like $\begin{bmatrix}1 & 1\\ 1 & 2\end{bmatrix}$, and we want to know what it will do to objects.
- First, multiply by the two unit basis vectors:

$$\begin{bmatrix}1 & 1\\ 1 & 2\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}=\begin{bmatrix}1\\ 1\end{bmatrix} \qquad \begin{bmatrix}1 & 1\\ 1 & 2\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix}=\begin{bmatrix}1\\ 2\end{bmatrix}$$

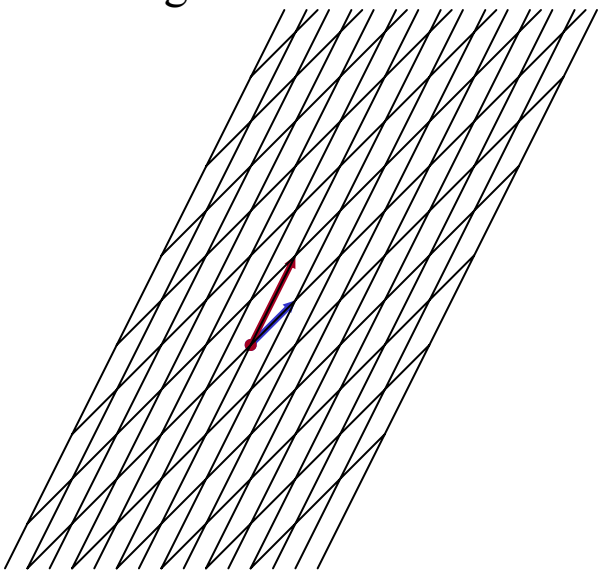
- Notice that the results are the two columns of the matrix.
- Now, since we said that matrix multiplication is distributive over vectors, we can tell where any vector will be sent by knowing where each unit basis vector is sent
- So if we have some transformation matrix, visualize what it does to an object…

Visual Matrix Multiplication, continued (2/3)

- Originally, we have a unit grid:

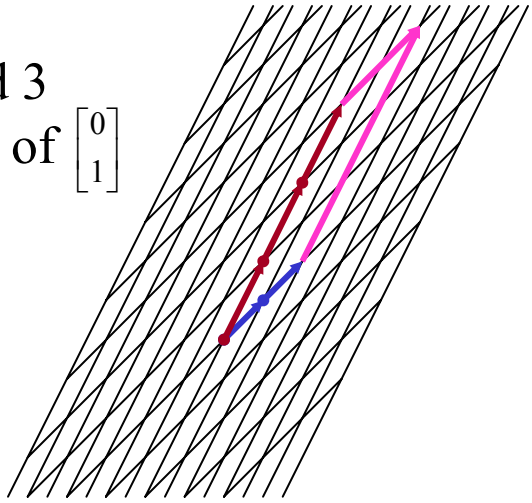


- Then we apply the transformation and set up a grid based on the images of the two unit/basis vectors:

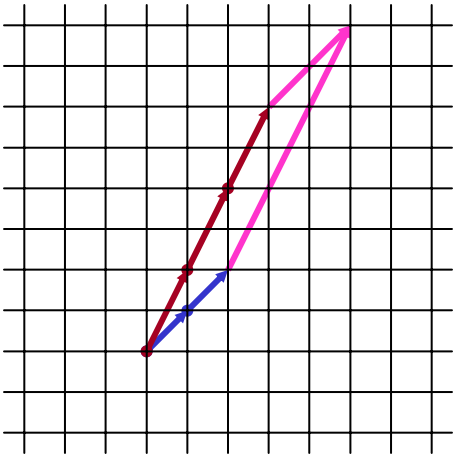


Visual Matrix Multiplication, continued (3/3)

Now, to see where $\begin{bmatrix}2\\3\end{bmatrix}$ ends up, instead of going 2 units in x and 3 in y, go 2 steps along the image of $\begin{bmatrix}0\\1\end{bmatrix}$ and 3 along the image of $\begin{bmatrix}1\\0\end{bmatrix}$



We can see what this looks like in the original coordinate system:



This example effectively illustrates the "change of basis" concept. The row vectors of the matrix define the new basis.

Matrix-Matrix Multiplication (1/2)

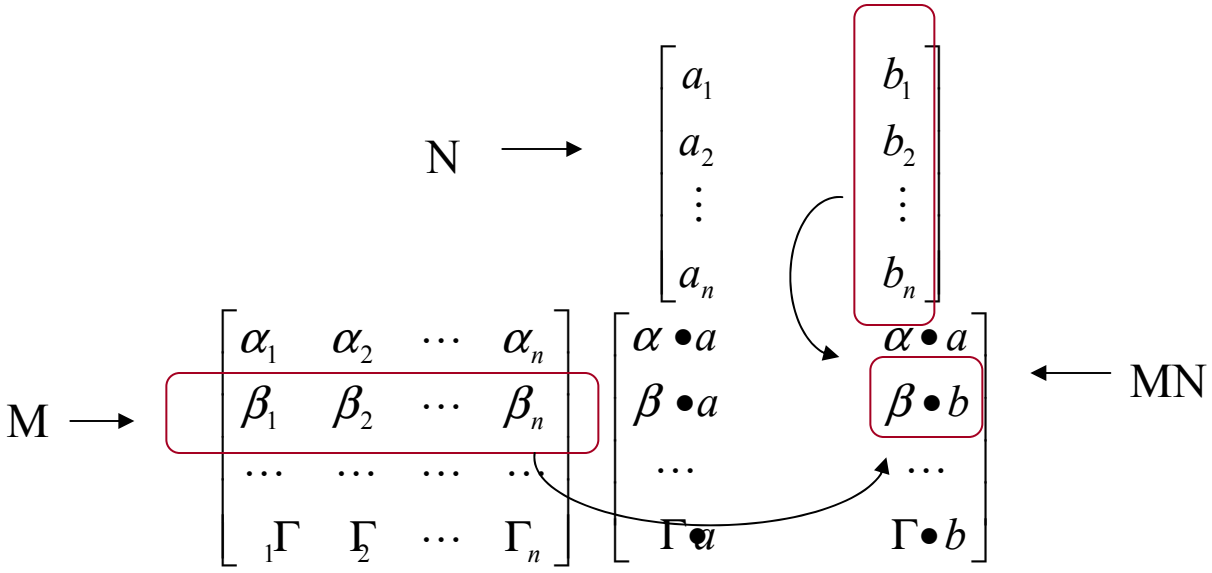
- Generally, how do we compute the product MN from matrices M and N?
- One way to think of matrix multiplication is in terms of row and column vectors: MN_{ij} = the dot product of the ith row of M and the jth column of N
- It is important to note that if M is an $m \times n$ matrix, then N must be an $n \times k$ matrix. This is because the rows of M and the columns of N must be the same size in order to compute their dot product
- So for M, an $m \times n$ matrix, and N, an $n \times k$ matrix:

$$MN = \begin{bmatrix} row_{M,1} \bullet col_{N,1} & \cdots & \cdots & row_{M,m} \bullet col_{N,k} \\ \vdots & row_{M,i} \bullet col_{N,j} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ row_{M,m} \bullet col_{N,1} & \cdots & \cdots & row_{M,m} \bullet col_{N,k} \end{bmatrix}$$

- Where $row_{A,x}$ means the xth row of A, and similarly $col_{A,x}$ means the xth column of A

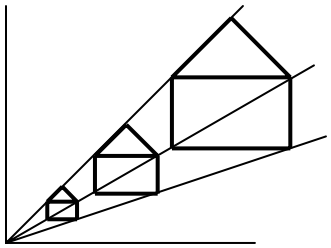
Matrix-Vector Multiplication (2/2)

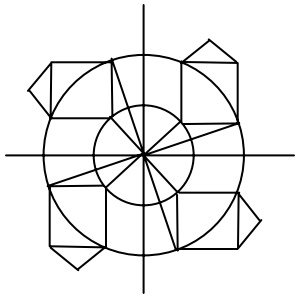
- Sometimes it can help a lot to have a picture of this process in your head
- For example, let's say that N is an nx2 matrix. Then MN can be calculated in the following manner.



Combining Transformations

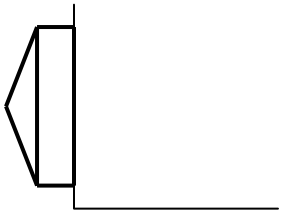
- Matrices representing transformations performed in a sequence can be composed into a single matrix
- However, there are problems with combining matrices
  - Matrix multiplication is not commutative.
    - Rotating or scaling object not centered at the origin introduces unwanted translation

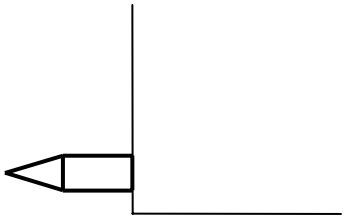




Translation induced by scaling and rotation

Rotation followed by non-uniform scale; next same scale followed by same rotation





Matrix Composition

- Rule for composing matrices

For $M_1 = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, M_2 = \begin{bmatrix} e & f \\ g & h \end{bmatrix}$ and $v = \begin{bmatrix} x \\ y \end{bmatrix}$

$$M_1(M_2(v)) = (M_1 M_2)(v)$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} e & f \\ g & h \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} ex+fy \\ gx+hy \end{bmatrix} = \begin{bmatrix} a(ex+fy)+b(gx+hy) \\ c(ex+fy)+d(gx+hy) \end{bmatrix}$$

$$v' = \begin{bmatrix} (ae+bg)x+(af+bh)y \\ (ce+dg)x+(cf+dh)y \end{bmatrix} = \begin{bmatrix} (ae+bg)(af+bh) \\ (ce+dg)(cf+dh) \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

- The product of matrices M_1M_2 is thus created from the dot products of the rows of M_1 with the columns of M_2

$$M_1 M_2 = \begin{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \bullet \begin{bmatrix} e \\ g \end{bmatrix} & \begin{bmatrix} a \\ b \end{bmatrix} \bullet \begin{bmatrix} f \\ h \end{bmatrix} \\ \begin{bmatrix} c \\ d \end{bmatrix} \bullet \begin{bmatrix} e \\ g \end{bmatrix} & \begin{bmatrix} c \\ d \end{bmatrix} \bullet \begin{bmatrix} f \\ h \end{bmatrix} \end{bmatrix}$$

Commutative and Non-Commutative Combinations of Transformations – Be Careful!

In 2D

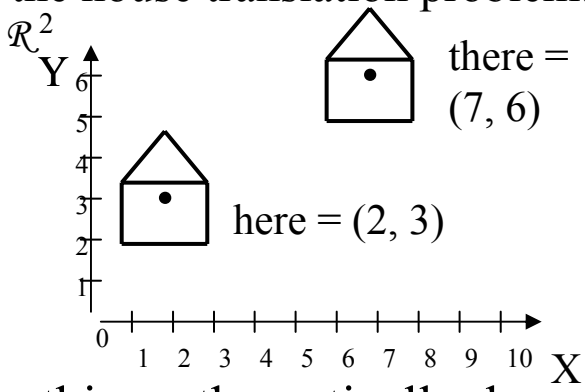
- Commutative
  - Translate, translate
  - Scale, scale
  - Rotate, rotate
  - Scale uniformly, rotate
- Non-commutative
  - Non-uniform scale, rotate
  - Translate, scale
  - Rotate, translate

In 3D

- Commutative
  - Translate, translate
  - Scale, scale
  - Scale uniformly, rotate
- Non-commutative
  - Non-uniform scale, rotate
  - Translate, scale
  - Rotate, translate
  - **Rotate, rotate**

Translation revisited

- Let's look back at the house translation problem:



- We decided to do this mathematically, by adding a vector to any vertex in the scene.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix} \quad \text{or} \quad \begin{aligned} x' &= x + dx \\ y' &= y + dy \end{aligned}$$

- Let's rewrite the math a bit…

$$\begin{aligned} x' &= 1*x + 0*y + dx*1 \\ y' &= 0*x + 1*y + dy*1 \\ 1 &= 0*x + 0*y + 1*1 \end{aligned}$$

- Hmmm, this looks familiar… $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

- This may show you more logically how we decided to use homogenous coordinates. It's a huge gain, because we can now compose translations along with other transformations in a uniform manner.

Matrix Inverse

- In the viewing lecture we will have use for M^{-1}, so here we'll talk extensively about what the inverse does. Also, how to find M^{-1}, given M (if it exists)
- Recall the definition:

$$AA^{-1} = I = A^{-1}A$$

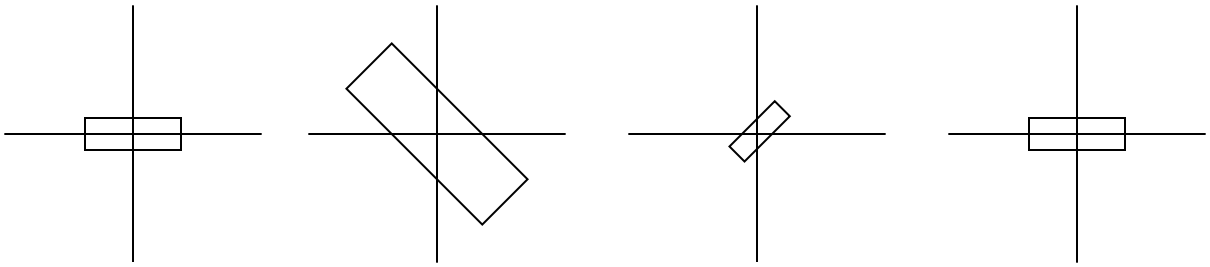
- The rule for inverting composed matrices is:

$$(AB)^{-1} = B^{-1}A^{-1}$$

- It is important to note that a matrix is not always invertible. A matrix will not be invertible if:
  - It is not a square (n x n matrix)
  - It has a row / column of all zeros
  - Any row / column is a multiple of any other row / column (called linearly dependent)
- Matrices for rotation, translation (using homogeneous coordinates), and scaling will always have inverses. This makes us happy

What Does an Inverse Do?

- The inverse A^{-1} of transformation A will "undo" the result of transforming by A
- For example: if A scales by a factor of two and rotates 135°, then A^{-1} will rotate by –135° and then scale by one half



Rectangle(B) A*B A^{-1}*B A^{-1}*A*B

What you're about to see…

- Suppose we have some matrix A that we're trying to invert.
- Let's write an obvious equation, $A = I * A$
- Now, suppose we multiply each side by some matrices, one by one, i.e. $T_2*T_1*A = T_2*T_1*I*A$
- And suppose after we've multiplied enough of these, we end up with the identity on the left side, so we have $I = T_n * \ldots * T_1*I*A$
- Then on the left we had $T_n * \ldots * T_1*A$, so $T_n * \ldots * T_1 = A^{-1}$, by the definition of A^{-1}
- But now, on the right, $(T_1*\ldots*T_1*I)=A^{-1}$, so as long as we didn't multiply by A yet, we have isolated A^{-1}
- So, to sum up:
  - Start with both A and I
  - Perform the same operations on each, until A becomes I.
  - Now, where we started with I, we are left with A^{-1}

How to Invert a Matrix (1/4)

- We're going to use Gauss-Jordan elimination
- Finding A^{-1} with Gauss-Jordan elimination is done by augmenting A with I to get [A|I], then reducing the new matrix into reduced row echelon form (rref) to get a new matrix. This new matrix will be of the form $[I|A^{-1}]$
- What does rref really mean?
- It means the matrix has the following properties:
  - If a row does not consist entirely of zeros, then the first nonzero number in the row is a 1. (Call this a leading 1)
  - If there are any rows that consist entirely of zeros, then they are grouped together at the bottom of the matrix.
  - If any two successive rows do not consist entirely of zeros, the leading 1 in the lower row occurs farther to the right than the leading 1 in the higher row
  - Each column that contains a leading 1 has zeros everywhere else.

How to Invert a Matrix (2/4)

- To transform a matrix into rref we are allowed to perform any of the three elementary row operations. These are:
  - Multiply a row by a nonzero constant
  - Interchange two rows
  - Add a multiple of one row to another row
- Notice that these can all be represented as matrices. So what we're really doing is building up the inverse bit by bit, and multiplying the matrices by the Identity as well as our original matrix just so we will still have the inverse once we are done building it up.
  - For example, the following matrix can be used to multiply the first row by 4 in a 3x3 matrix:

$$\begin{bmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

  - This matrix swaps the first 2 rows in a 3x3 matrix:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

How to Invert a Matrix (3/4)

- Note that with these operations you can:
  - Create a leading 1 by dividing by the leading constant in a row: (row op 1)
  - Move zero rows to the bottom of the matrix by interchanging rows: (row op 2)
  - Assure that zeros occur as all entries, except the leading 1, in a column by canceling out the other values with multiples of 1: (row op 3)
- It is important to note that we would have no use for the matrix at any intermediate stage. Although we are applying strictly linear transformations, they have no geometric meaning, they are just a way of achieving the end result bit by bit.

How to Invert a Matrix (4/4)

(note: these steps taken from Elementary Linear Algebra, by Howard Anton, which can be found in the SciLi)

- The steps we will actually use:
  - Step 1. Locate the leftmost column that does not consist entirely of zeros.
  - Step 2. Interchange the top row with another row, if necessary, to bring a nonzero entry to the top of the column found in Step 1.
  - Step 3. If the entry that is now at the top of the column found in Step 1 is a, multiply the first row by $1/a$ in order to introduce a leading 1.
  - Step 4. Add suitable multiples of the top row to the rows below so that all entries below the leading 1 become zeros.
  - Step 5. Now cover the top row in the matrix and begin again with Step 1 applied to the submatrix that remains. Continue in this way until the *entire* matrix is in row-echelon form. (not reduced row-echelon form)
  - Step 6. Beginning with the last nonzero row and working upward, add suitable multiples of each row to the rows above to introduce zeros above the leading 1's.

Inversion Example (1/3)

- This will be a small example because doing this by hand takes a while. Let's find A^{-1} for:

$$A = \begin{bmatrix} 11 & 13 \\ 17 & 19 \end{bmatrix}$$

- Augment this with the identity to get

$$[A \mid I] = \begin{bmatrix} 11 & 13 & 1 & 0 \\ 17 & 19 & 0 & 1 \end{bmatrix}$$

- Row operation 1, multiply row 1 by 1/11.

$$= \begin{bmatrix} 1 & \frac{13}{11} & \frac{1}{11} & 0 \\ 17 & 19 & 0 & 1 \end{bmatrix}$$

- Row operation 3, multiply row 1 by –17 and add it to row 2

$$= \begin{bmatrix} 1 & \frac{13}{11} & \frac{1}{11} & 0 \\ 0 & -\frac{12}{11} & -\frac{17}{11} & 1 \end{bmatrix}$$

Inversion Example (2/3)

- Row operation 1, multiply row 2 by –11/12

$$= \begin{bmatrix} 1 & \frac{13}{11} & \frac{1}{11} & 0 \\ 0 & 1 & \frac{17}{12} & -\frac{11}{12} \end{bmatrix}$$

- Row operation 3, multiply row 2 by –13/11 and add to row 1

$$[I \mid A^{-1}] = \begin{bmatrix} 1 & 0 & -\frac{19}{12} & \frac{13}{12} \\ 0 & 1 & \frac{17}{12} & -\frac{11}{12} \end{bmatrix}$$

- Therefore:

$$A^{-1} = \begin{bmatrix} -\frac{19}{12} & \frac{13}{12} \\ \frac{17}{12} & -\frac{11}{12} \end{bmatrix}$$

Inversion Example (3/3)

- For more detail, refer to the inversion handout or Anton's book.
- The general pattern: On the left, incrementally go from A to I

$$\begin{bmatrix} * & * & \cdots & * \\ * & * & \cdots & * \\ \vdots & \vdots & \ddots & * \\ * & * & * & * \end{bmatrix}$$

(all positions arbitrary numbers - A)

to

$$\begin{bmatrix} 1 & * & \cdots & * \\ 0 & 1 & \cdots & * \\ \vdots & \vdots & \ddots & * \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(ones along diagonal, zeroes below, arbitrary numbers above)

to

$$\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(ones along diagonal, zeroes below and above - Identity)

Addendum – Matrix Notation

- The application of matrices in the row vector notation is executed in the reverse order of applications in the column vector notation:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \leftrightarrow \begin{bmatrix} x & y & z \end{bmatrix}$$

- Column format: vector follows transformation matrix.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- Row format: vector precedes matrix and is pos-multiplied by it.

$$\begin{bmatrix} x' & y' & z' \end{bmatrix} = \begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

- By convention, we always use column vectors.

But, There's a Problem…

- Notice that

$$\begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} ax+dy+gz & bx+ey+hz & cx+fy+iz \end{bmatrix}$$

- while

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} ax+by+cz \\ dx+ey+fz \\ gx+hy+iz \end{bmatrix}$$

Solution to Notational Problem

- In order for both types of notations to yield the same result, a matrix in the row system must be the transpose of the matrix in the column system
- Transpose is defined such that each entry at (i,j) in M, is mapped to (j,i) in its transpose (which is denoted M^T). You can visualize M^T as rotating M around its main diagonal

$$M = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}, M^T = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix}$$

- Again, the two types of notation are equivalent:

$$\begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix} = \begin{bmatrix} ax & by & cz & dx & ey & fz & gx & hy & iz \end{bmatrix} \leftrightarrow$$

$$\begin{bmatrix} ax+by+cz \\ dx+ey+fz \\ gx+hy+iz \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- Different texts and graphics packages use different notations. Be careful!

Matrix Notation and Composition

- Application of matrices in row notation is reverse of application in column notation:

$$TRSv$$

Matrices applied right to left

$$vS^TR^TT^T$$

Matrices applied left to right

Summary

- We can do basic math with vectors and matrices
- We know how to compute the dot product of two vectors, and various uses for the dot product (and don't worry, you'll be using it for all these things)
- We've solved the translation problem
- We can compose many matrices so that we only have to do one matrix-vector multiplication to transform an object arbitrarily
- We can invert a matrix (for more detail, refer to *Elementary Linear Algebra*, by Howard Anton, which can be found in the Sci Li)