2D Geometrical Transformations

- Start by considering 2D objects.
- Later we will consider scan conversion: $objects \rightarrow primitives \rightarrow pixels$.
- This section discusses the methods we use to model 2D objects and transformations of those objects: $objects \rightarrow objects$.
- In 2D, possible primitives are points, lines and polygons, which can be combined to give higher level objects.
- In 2D it is generally easy to see how to represent certain objects.
- To understand transformations we need a review of basic vector algebra!

Vectors

- A vector is an n-tuple of numbers:

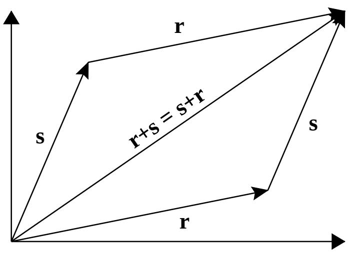
$$\boldsymbol{r} = \begin{bmatrix} 2 \\ 3 \end{bmatrix} .$$

- Vectors are denoted by lower case bold letters.
- The rule for the addition is:

$$\boldsymbol{r} + \boldsymbol{s} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} + \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} r_1 + s_1 \\ r_2 + s_2 \\ r_3 + s_3 \end{bmatrix} .$$

- Vector addition is commutative: $\boldsymbol{r} + \boldsymbol{s} = \boldsymbol{s} + \boldsymbol{r}$.

Vector addition



Vectors – multiplication

- Scalar multiplication is given by:

$$a\boldsymbol{r} = a \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} ar_1 \\ ar_2 \\ ar_3 \end{bmatrix} .$$

- Given two vectors $\boldsymbol{r}$ and $\boldsymbol{s}$ we define their dot product (sometimes called their inner product) to be:

$$\boldsymbol{r} \cdot \boldsymbol{s} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} \cdot \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = r_1 s_1 + r_2 s_2 + r_3 s_3 .$$

- The length of a vector $\boldsymbol{r}$, denoted $\|\boldsymbol{r}\|$, is given by the square root of the dot product of the vector with itself: $\sqrt{\boldsymbol{r} \cdot \boldsymbol{r}}$ (cf. Pythagoras).

Vectors – angles

- Dot product can be used to generate unit length vectors: $\boldsymbol{r}/\|\boldsymbol{r}\|$.
- The angle between two vectors is:

$$\theta = \cos^{-1} \left(\frac{\boldsymbol{r} \cdot \boldsymbol{s}}{\|\boldsymbol{r}\|\|\boldsymbol{s}\|} \right) .$$

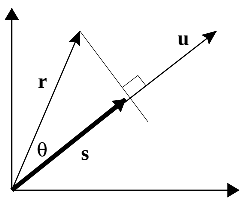
- This is very useful when computing lighting where we general only need:

$$\cos(\theta) = \left(\frac{\boldsymbol{r} \cdot \boldsymbol{s}}{\|\boldsymbol{r}\|\|\boldsymbol{s}\|} \right) .$$

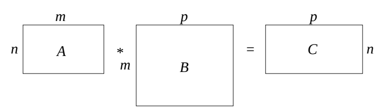
Vectors – projection

- We can also project one vector, $\boldsymbol{r}$, onto another unit vector, $\boldsymbol{u}$.
- The length of the projected vector $\boldsymbol{s}$ is:

$$\|\boldsymbol{s}\| = \|\boldsymbol{r}\| \cos(\theta) = \|\boldsymbol{r}\| \left(\frac{\boldsymbol{r} \cdot \boldsymbol{u}}{\|\boldsymbol{r}\|\|\boldsymbol{u}\|} \right) = \boldsymbol{r} \cdot \boldsymbol{u} .$$



Matrices



- A matrix is a rectangular array of numbers.
- A general matrix will be represented by an upper case letter:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} .$$

- The element on the ith row and jth column is denoted by $a_{i,j}$. Note that we start indexing at 1, whereas C indexes arrays from 0.

Matrices – multiplication

- Given two matrices, A and B if we want to multiply B by A (that is form AB) then if A is $(n \times m)$, B must be $(m \times p)$.
- This produces a result, $C = AB$, which is $(n \times p)$, with elements:

$$c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj}$$

- Basically we multiply the first row of A with the first column of B and put this in the $c_{1,1}$ element of C. And so on ....

Matrices – C code

```
/* Define the structure to hold a 3 by 3 matrix. */
 typedef struct Matrix3struct { double el[3][3]} Matrix3;
/* Declare the function  to multiply C = AB */
void MatMult3( Matrix3* a, Matrix3* b, Matrix3* c)
{
    int i,j,k;
    for (i = 0; i < 3; i++) {     /* - zero indexing */
       for (j = 0; j < 3; j++) {
         /* Recall we use the -> operator to access an element of */
         /* structure when we have a pointer to that structure. */
         c->el[i][j] = 0.0;      /* Make sure C is initialised. */
          for (k = 0; k < 3; k++) {
            c->el[i][j] += a->el[i][k]*b->el[k][j];
         }
      }
   }
 return (c);
}
```

Matrices – basics

- Unlike scalar multiplication, $AB \neq BA$.
- Matrix multiplication distributes over addition: $A(B+C) = AB + AC$
- Identity matrix for multiplication is denoted I.
- The transpose of a matrix, A, which is either denoted A^T or A' is obtained by swapping the rows and columns of the matrix. Thus:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \end{bmatrix} \quad \Rightarrow \quad A' = \begin{bmatrix} a_{1,1} & a_{2,1} \\ a_{1,2} & a_{2,2} \\ a_{1,3} & a_{2,3} \end{bmatrix}.$$

Summary

- Having finished this lecture you should:
  - be able to perfrom basic operation on vectors: addition, subtraction, multiplication (scalar and vector), compute the length of a vector, dot product and angle between two vectors;
  - be able to multiply matrices (work with them);
  - that is all.
- This basic vector and matrix algebra is key to much of computer graphics.
- There are many other vector and matrix operators that have not been introduced here, however we shall deal with these as they are required.