Advanced Games Engine Creation

3D Maths Answers

1. Vector operations

a) Find the vector AB between the point A(4, -2, 8) and B(-1, 4, 6) (-5, 6, -2)
b) Add the vectors (-7, 3, 2) and (-2, -1, 4) (-9, 2, 6)
c) Multiply the vector (2, -5, 3) by -2 (-4, 10, -6)
d) Find the magnitude of the vector (2, 1, 3) 3.74
e) Normalise the vector (3, 4, 0) (0.6, 0.8, 0)

2. Multiplying vectors by matrices - Use pen and paper to perform the following calculations.

a) $\begin{bmatrix} 3 & 1 & 6 \\ 0 & 2 & 1 \\ 1 & 0 & 5 \end{bmatrix} \begin{bmatrix} 4 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 32 \\ 7 \\ 19 \end{bmatrix}$

b) $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 0 \\ 1 \end{bmatrix}$

c) $\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 6 \\ 0 \\ 1 \end{bmatrix}$

d) $\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 15 \\ 0 \\ 1 \end{bmatrix}$

e) $\begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \\ 1 \\ 1 \end{bmatrix}$

f) $\begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 0 \\ 1 \end{bmatrix}$

What is the effect of each matrix on the 3D vector in questions b)-f) (ignore the final 1, it will be explained next week) Try drawing a sketch of the vector before and after the transformation.

b) identity – no change c) uniform scaling by a factor of 2 d) non-uniform scaling (x2 in x, x5 in y) e) translation by the vector (3, 2, 1) f) rotation by 90 degrees about the z axis.

3. Multiplying matrices:

Use pen and paper to perform the following calculations:

a) $\begin{bmatrix} 2 & -1 & 6 \\ 3 & 5 & 2 \\ 6 & 4 & 1 \end{bmatrix}\begin{bmatrix} 1 & -1 & 3 \\ 4 & 0 & 2 \\ 2 & 1 & -1 \end{bmatrix}=\begin{bmatrix} 10 & 4 & -2 \\ 27 & -1 & 17 \\ 24 & -5 & 25 \end{bmatrix}$

b) $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 4 & 2 & 2 \\ 3 & 2 & -2 & 8 \\ 0 & 9 & 1 & 1 \\ 1 & -3 & 6 & 2 \end{bmatrix}=\begin{bmatrix} 1 & 4 & 2 & 2 \\ 3 & 2 & -2 & 8 \\ 0 & 9 & 1 & 1 \\ 1 & -3 & 6 & 2 \end{bmatrix}$

c) $\begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 2 & 0 & 0 & 3 \\ 0 & 5 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

d) $\begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 3 \\ 0 \\ 1 \end{bmatrix}=\begin{bmatrix} 2 & 0 & 0 & 3 \\ 0 & 5 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 3 \\ 0 \\ 1 \end{bmatrix}=\begin{bmatrix} 5 \\ 17 \\ 1 \\ 1 \end{bmatrix}$

e) $\begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 0 & -5 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Next week we will discuss the how 3D transformations can be expressed by matrix operations such as the ones in questions 2 and 3.

4. Implement a 3D Vector struct

Partial implementation:

```
struct Vector3D
{
	float x;
	float y;
	float z;

	Vector3D()
	{
		x = 0.0f;
		y = 0.0f;
		z = 0.0f;
	}

	Vector3D(float initialX, float initialY, float initialZ)
```

```
	{
		x = initialX;
		y = initialY;
		z = initialZ;
	}

	float Magnitude() {
		return sqrt(x*x + y*y + z*z);
	}

	Vector3D  Normalise() {
		float magnitude = Magnitude();
		Vector3D unit = { x / magnitude, y / magnitude, z /
magnitude };
		return unit;
	}

	Vector3D  operator+(const Vector3D b) {
		Vector3D result;
		result.x = x + b.x;
		result.y = y + b.y;
		result.z = z + b.z;
		return result;
	}

	Vector3D& operator += (const Vector3D b) {
		x = x + b.x;
		y = y + b.y;
		z = z + b.z;
		return *this;
	}
};

// Test class

#include <iostream>
#include "Commons.h"
using namespace::std;

void display(const Vector3D v) {
	cout << "[" << v.x << "," << v.y << "," << v.z << "]" <<
endl;
}

int main(void) {
	char x;
	Vector3D v1(2.0f, 3.0f, 4.0f);
	display(v1);
	cout << "Magnitude: " << v1.Magnitude() << endl;
```

```
        Vector3D  vn = v1.Normalise();
        display(vn);
        cout << "Magnitude of vn: " << vn.Magnitude() << endl;

        Vector3D v2(1.0f, 2.0f, -3.0f);
        Vector3D v3 = v1 + v2;
        cout << "Sum of v1 and v2 is ";
        display(v3);
        display(v1 += v2);
        cout << "v1 is now ";
        display(v1);
        cout << "Enter any char to exit";
        cin >> x;
}
```