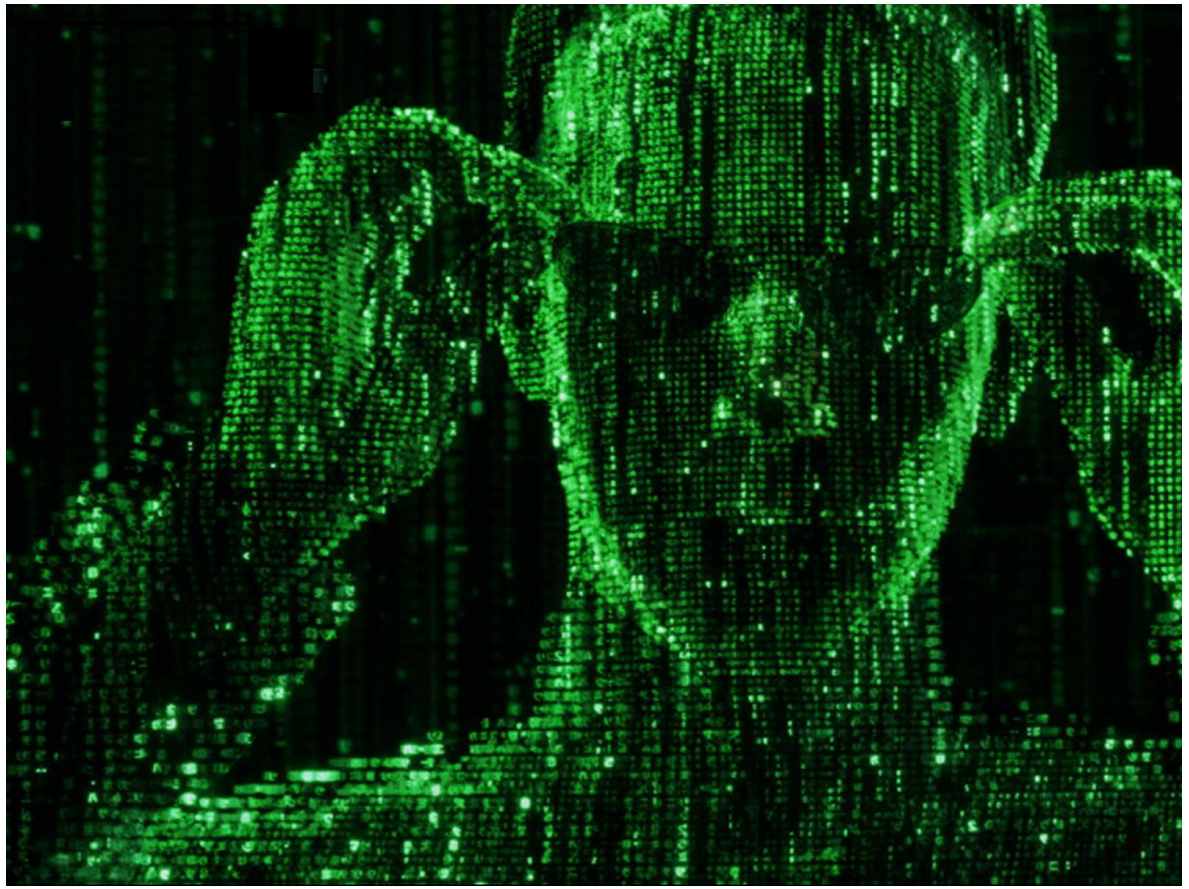
Introduction to Shader development

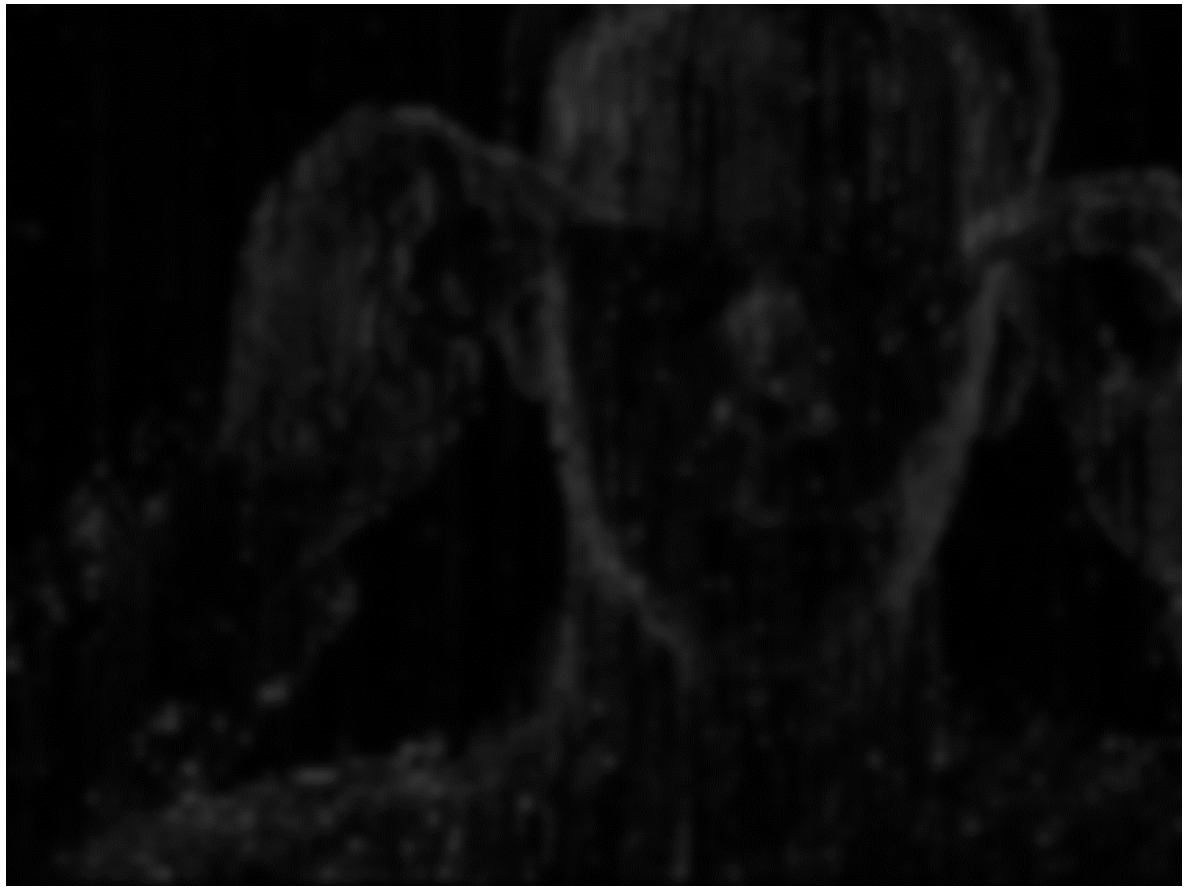
h_da WS2020/21
Paul Nasdalack

info@paul-nasdalack.com
@littleBugHunter

Vectors and Matrices



Vectors and Matrices



Vectors and Matrices

A **vector space** (also called a **linear space**) is a collection of objects called **vectors**, which may be added together andmultiplied ("scaled") by numbers, called *scalars* in this context. Scalars are often taken to be real numbers, but there are also vector spaces with scalar multiplication by complex numbers, rational numbers, or generally any field. The operations of vector addition and scalar multiplication must satisfy certain requirements, called *axioms*, listed below.

(https://en.wikipedia.org/wiki/Vector_space)

Vectors and Matrices

What are Vectors to us?

What are Vectors to us?

- A Vector is:
  - Multiple numbers, that somehow belong together
- Popular Vectors are:
  - Positions (2 or 3 Numbers XY[Z])
  - Colors (3 or 4 Numbers RGB[A])
  - Directions (2 or 3 Numbers XY[Z])
- Basically everything in Shaders is represented by a Vector or a Scalar

What are Vectors to us?

Scalar?

What are Vectors to us?

Scalar?

Just a fancy term for single Number

What are Vectors to us?

What can we do with Vectors?

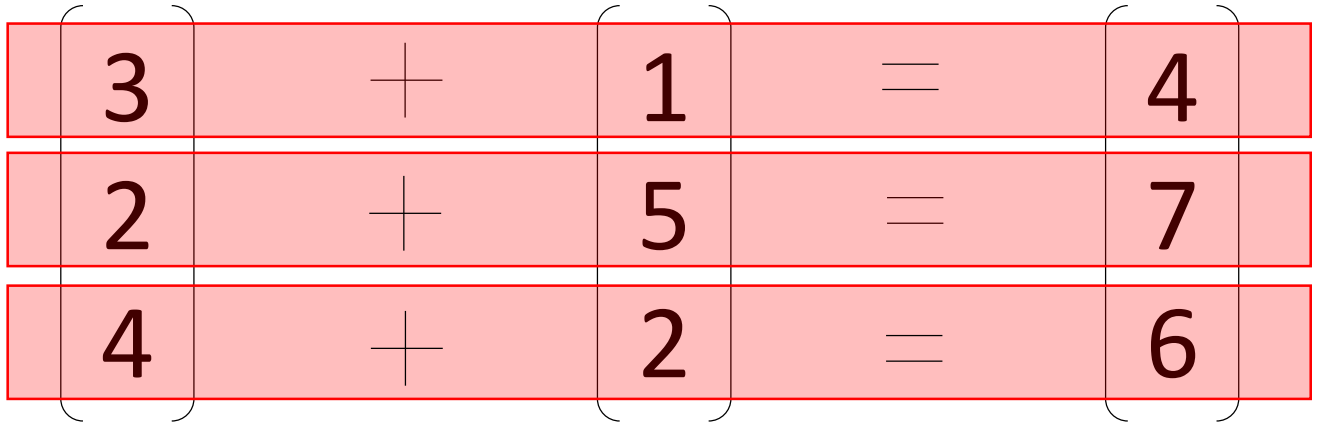
What can we do with Vectors?

- Add/Substract
- Multiply
  - Dot product
  - Cross product
  - Scalar product
- Divide???

Add/Substract

$$\begin{pmatrix} 3 \\ 2 \\ 4 \end{pmatrix} + \begin{pmatrix} 1 \\ 5 \\ 2 \end{pmatrix} = \begin{pmatrix} 4 \\ 7 \\ 6 \end{pmatrix}$$

Add/Substract



What are Vectors to us?

- Vector

HLSL/CG Syntax:

```
float a = 3.41; //normal scalar
float3 b = float3(0.2,0.32,0.4) //vector
```

Add/Substract

HLSL/CG Syntax:

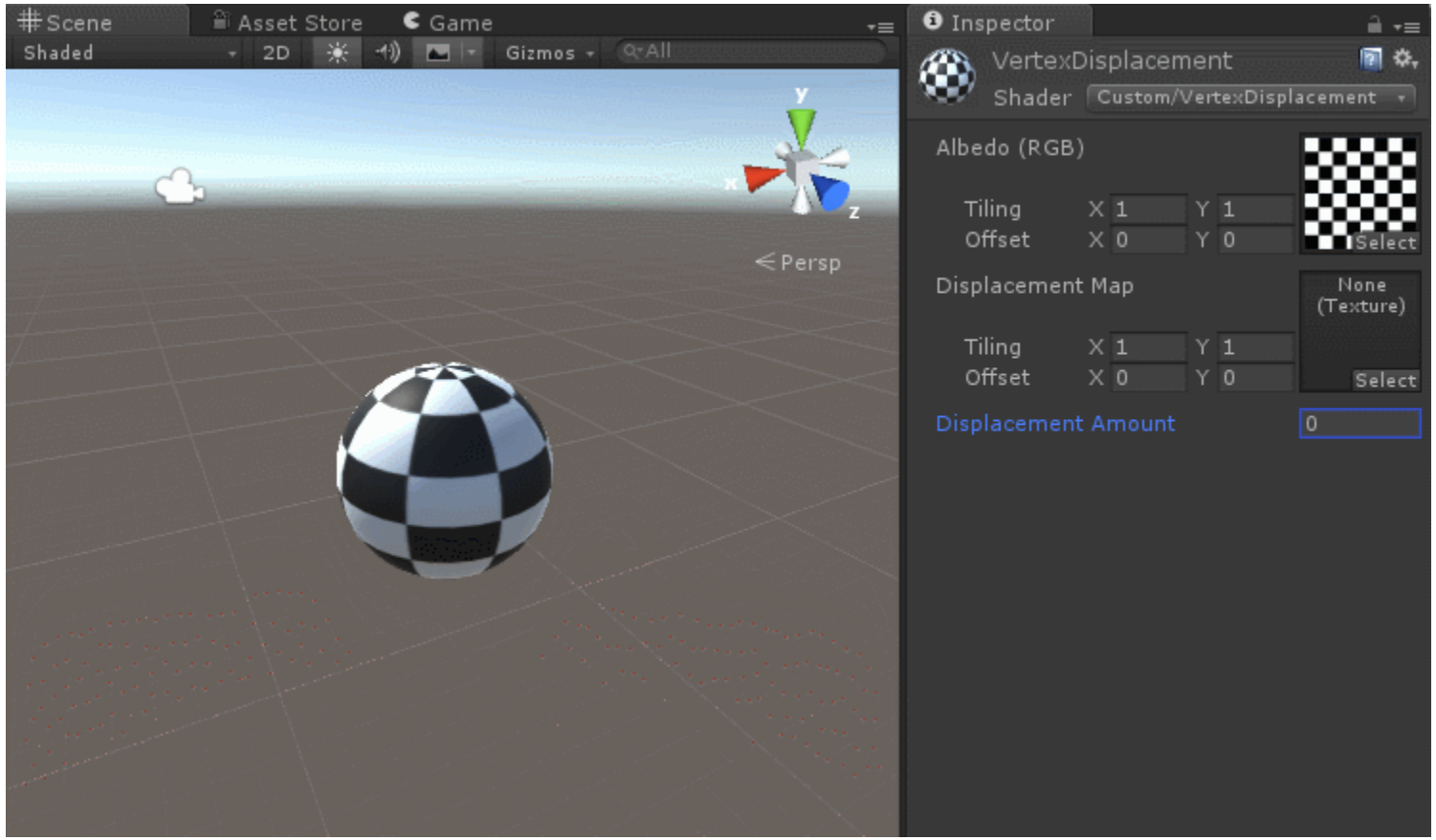
```
a+b
a-b
```

Add/Substract

What can we do with Add/Substract?

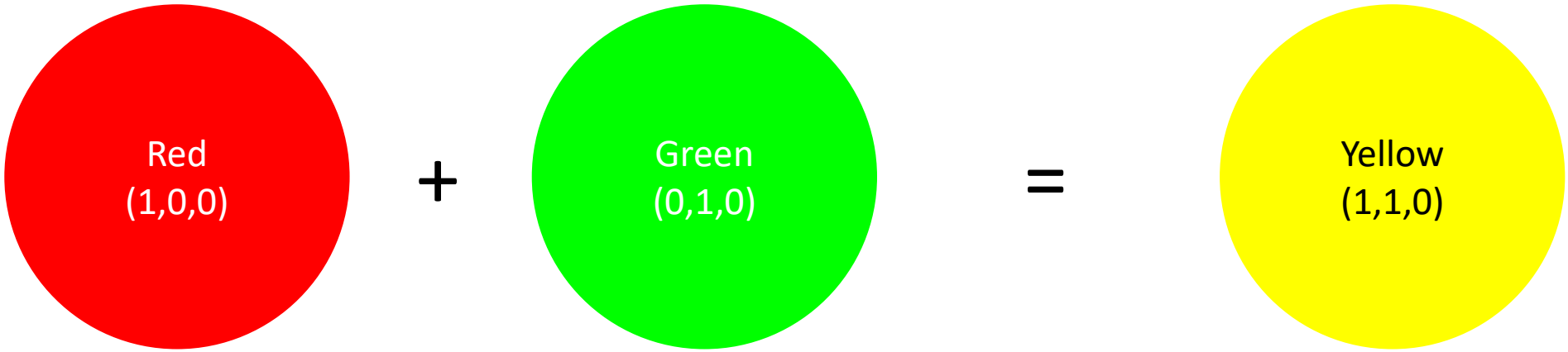
What can we do with Add/Substract?

Move Positions around (aka Vertex Displacement)



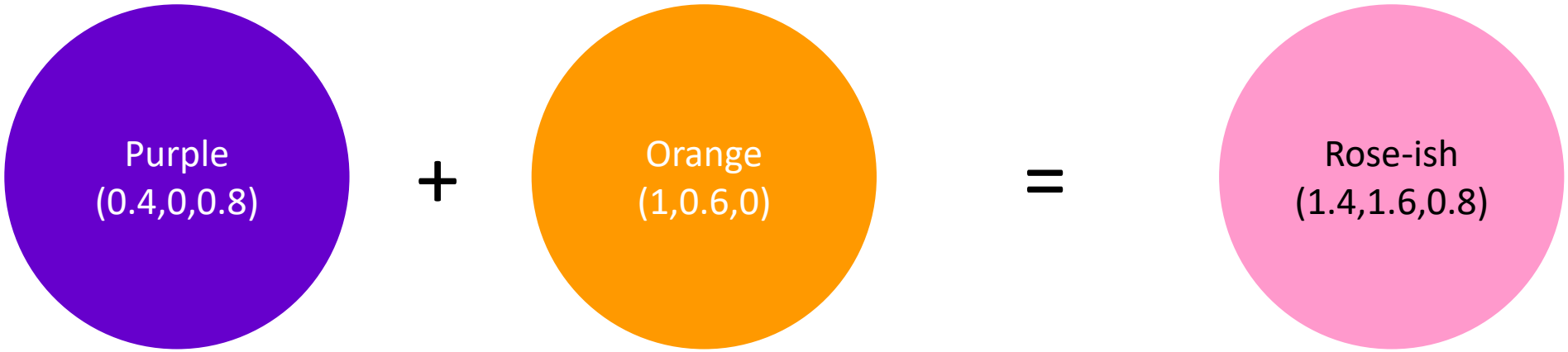
What can we do with Add/Substract?

Add Colors together



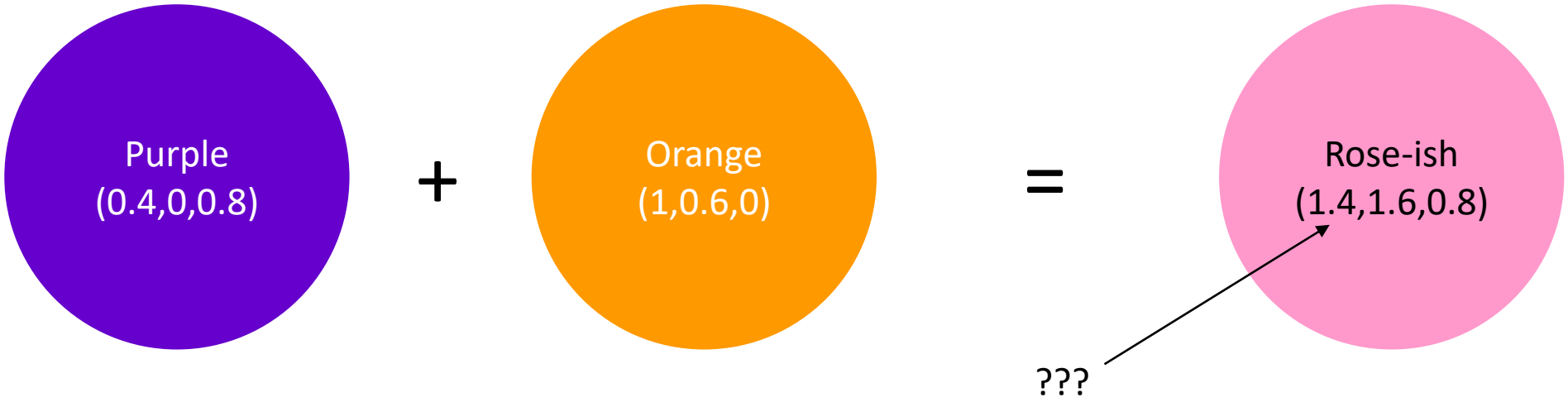
What can we do with Add/Substract?

Add Colors together



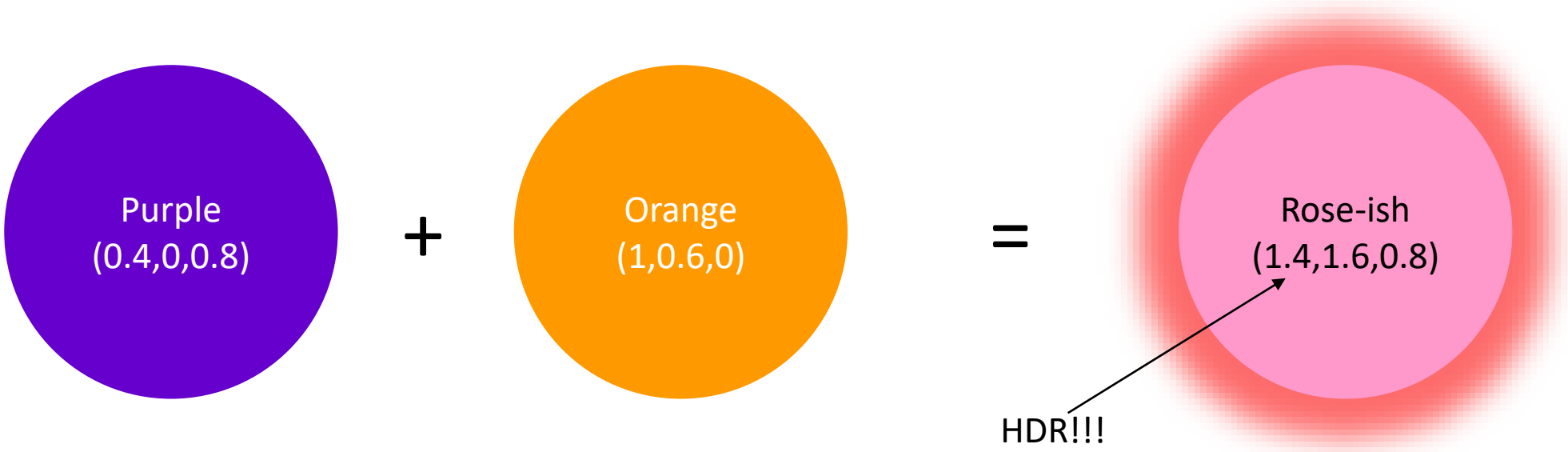
What can we do with Add/Substract?

Add Colors together



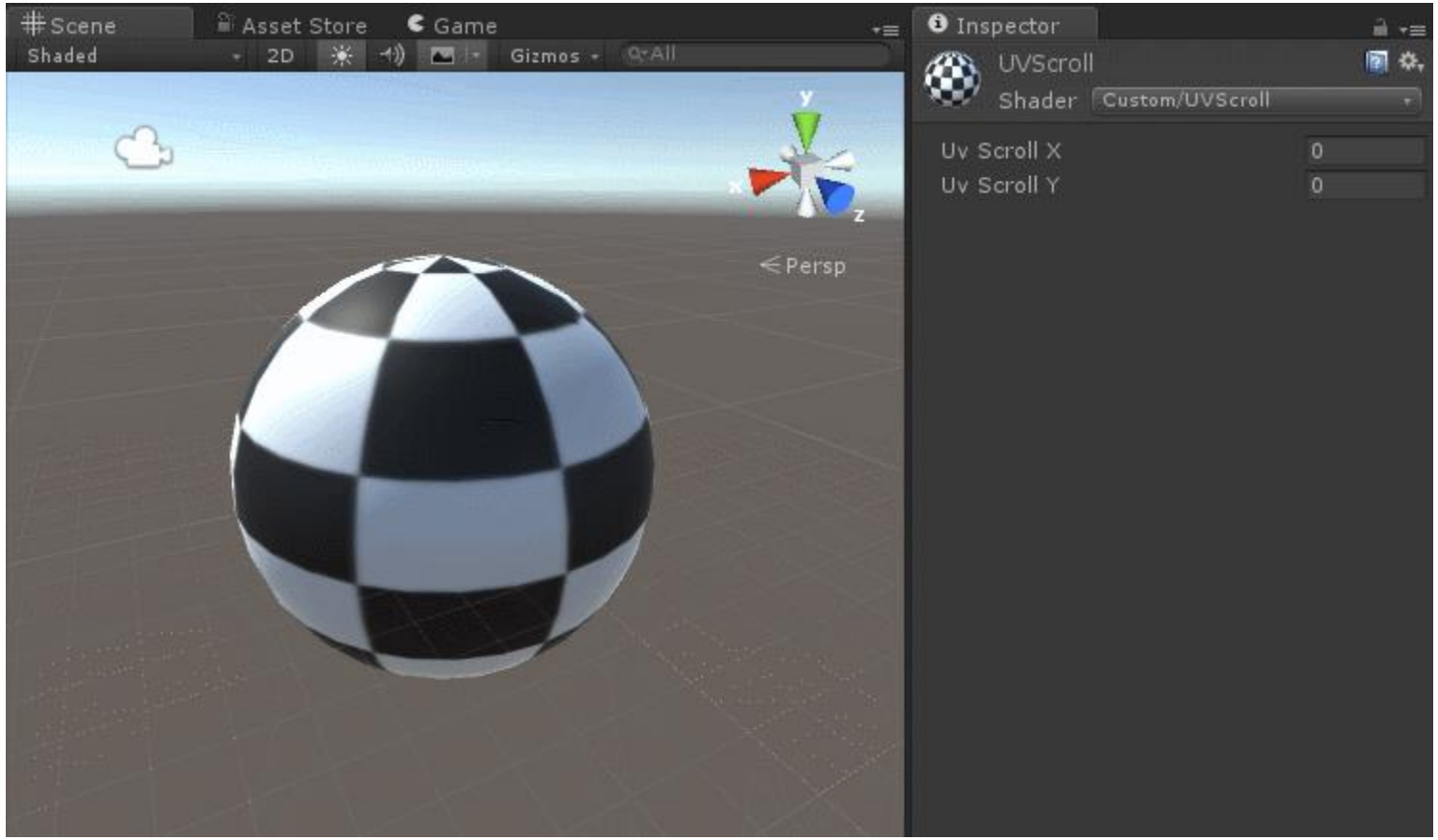
What can we do with Add/Substract?

Add Colors together



What can we do with Add/Substract?

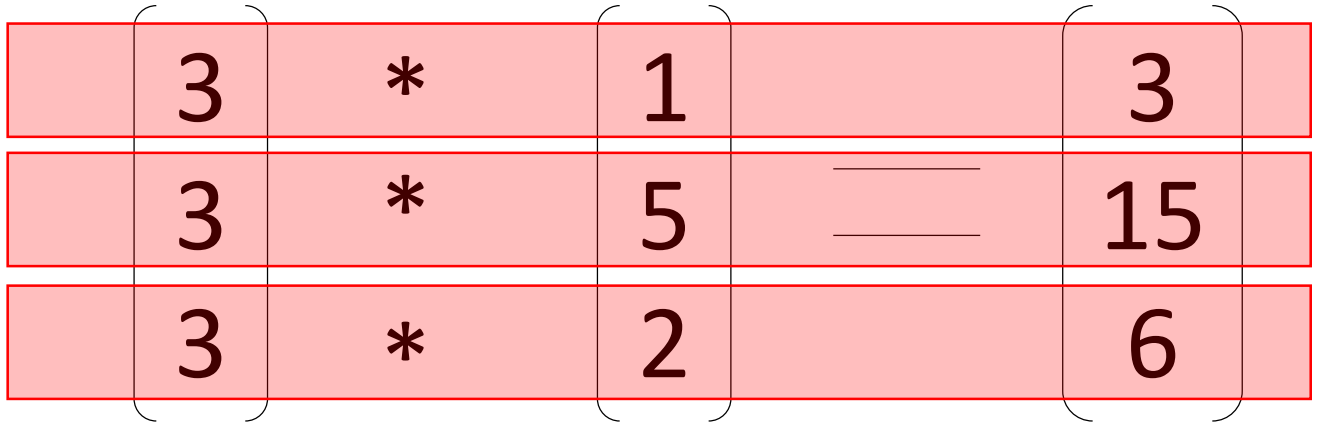
Scroll UVs around



Multiplication (Scalar Product)

$$3 \quad * \quad \begin{pmatrix} 1 \\ 5 \\ 2 \end{pmatrix} = \begin{pmatrix} 3 \\ 15 \\ 6 \end{pmatrix}$$

Multiplication (Scalar Product)



Multiplication (Scalar Product)

$$\begin{pmatrix} 2 \\ 4 \\ 5 \end{pmatrix} * \begin{pmatrix} 1 \\ 5 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 20 \\ 10 \end{pmatrix}$$

DISCLAIMER!
This is not very mathematical, but it‘s just too darn useful, so we‘ve built it into Shaders anyway

Multiplication (Scalar Product)

HLSL/CG Syntax:

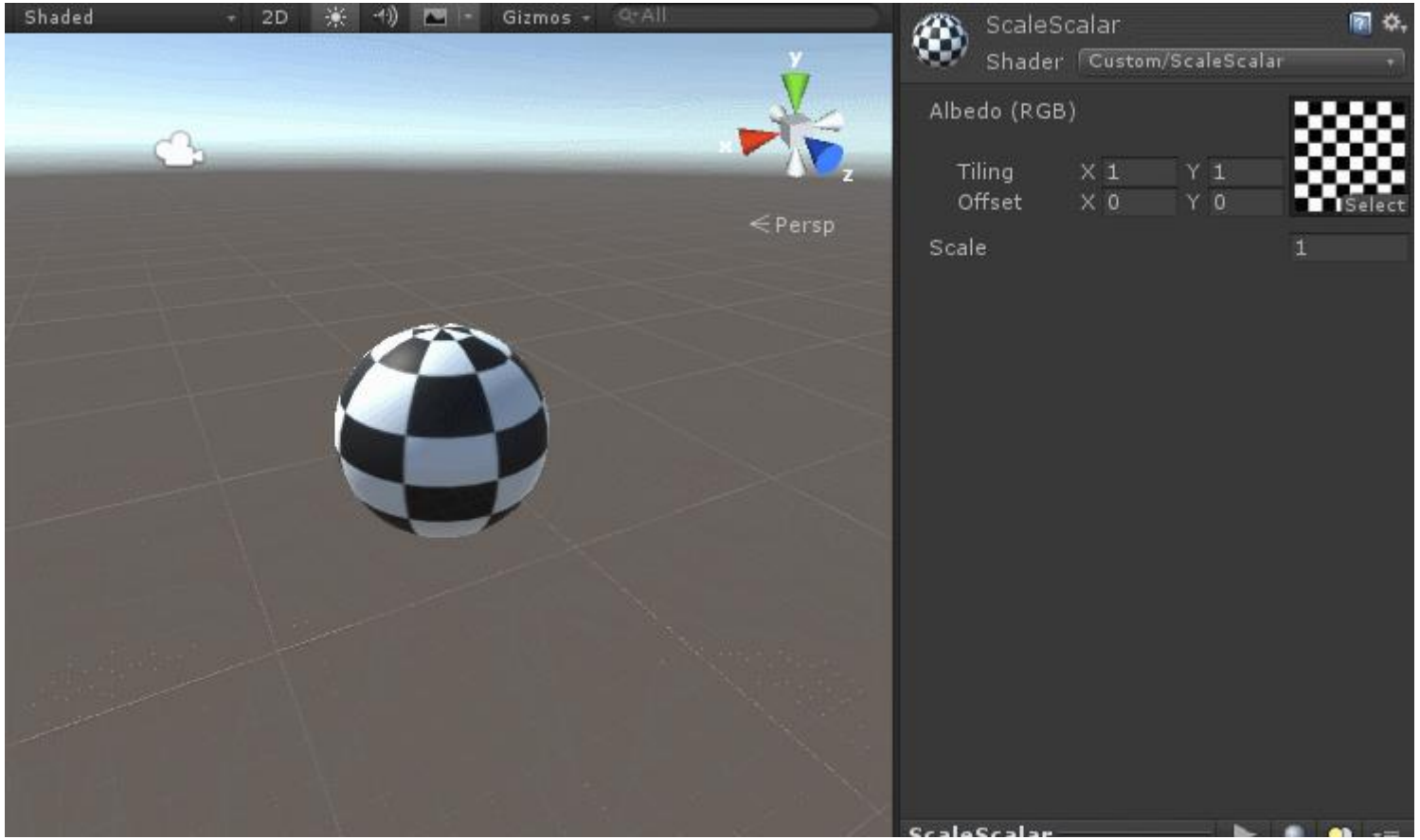
```
a*b
3*b
```

Multiplication (Scalar Product)

What can we do with Scalar Products?

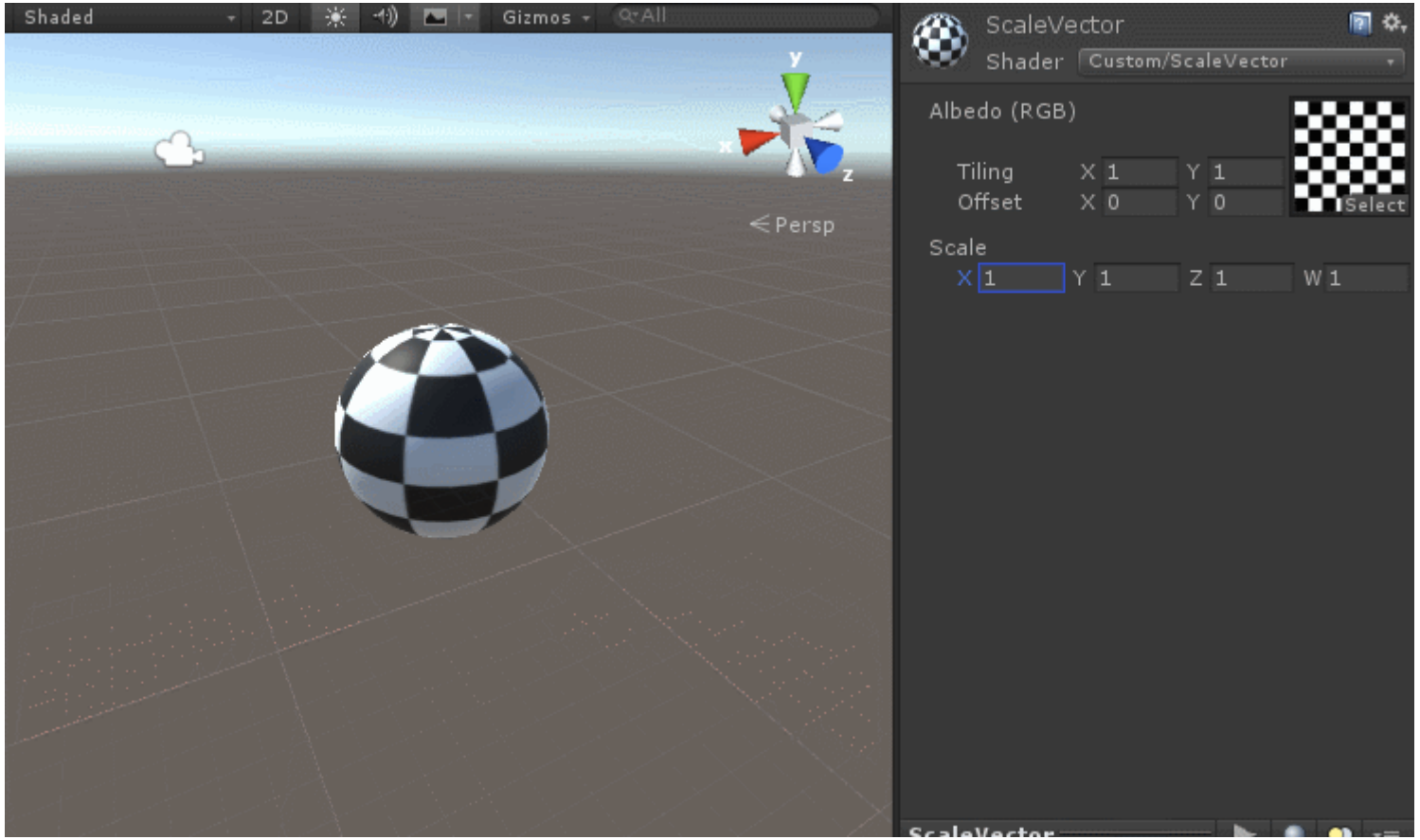
What can we do with Scalar Products?

Scale things! (using a Scalar)



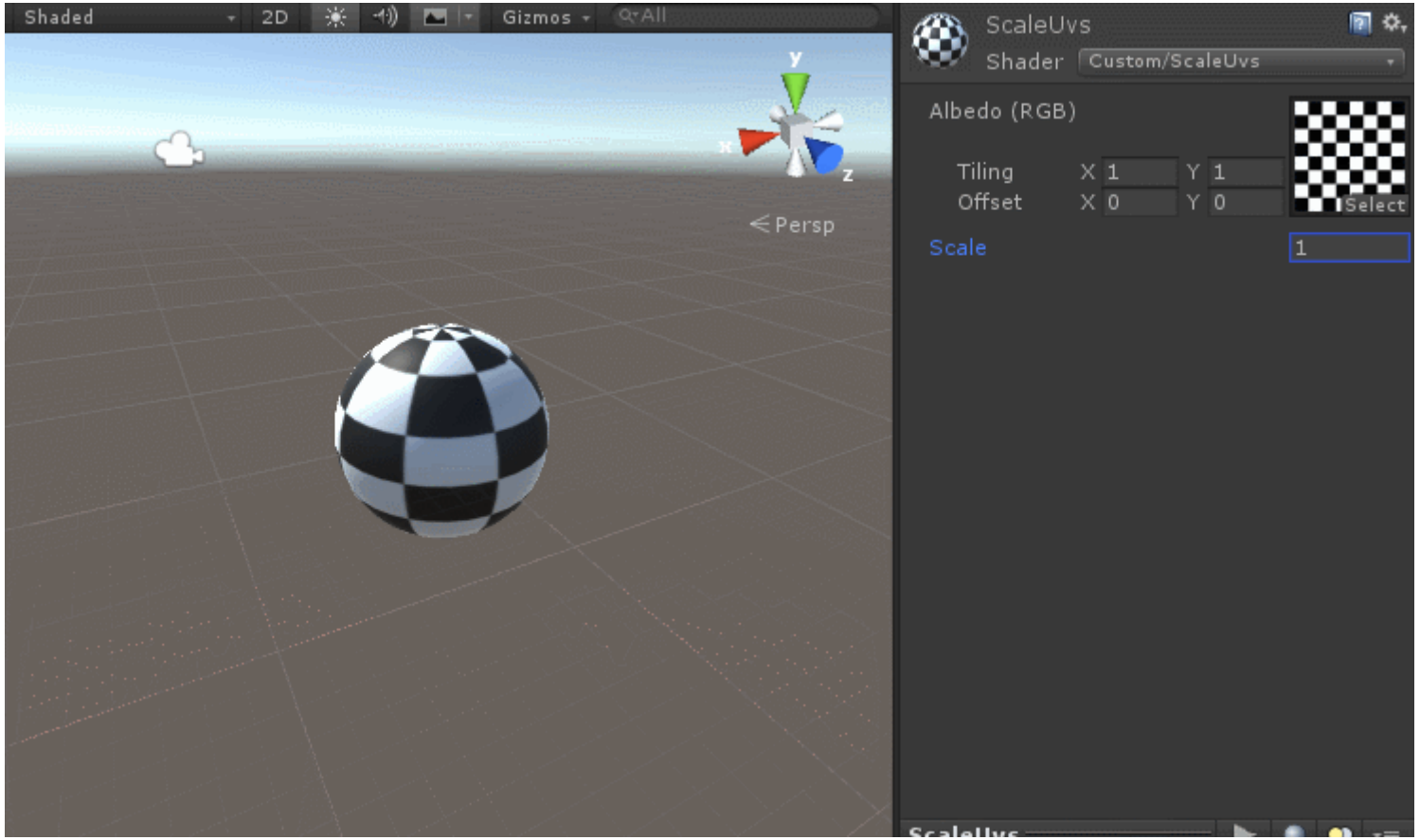
What can we do with Scalar Products?

Scale things! (using a Vector)



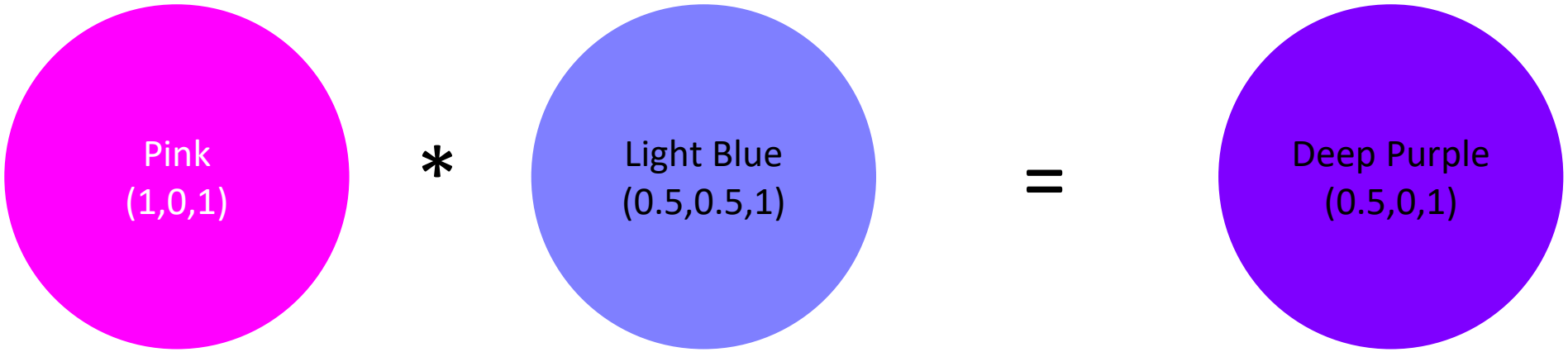
What can we do with Scalar Products?

Scale things! (UVs)



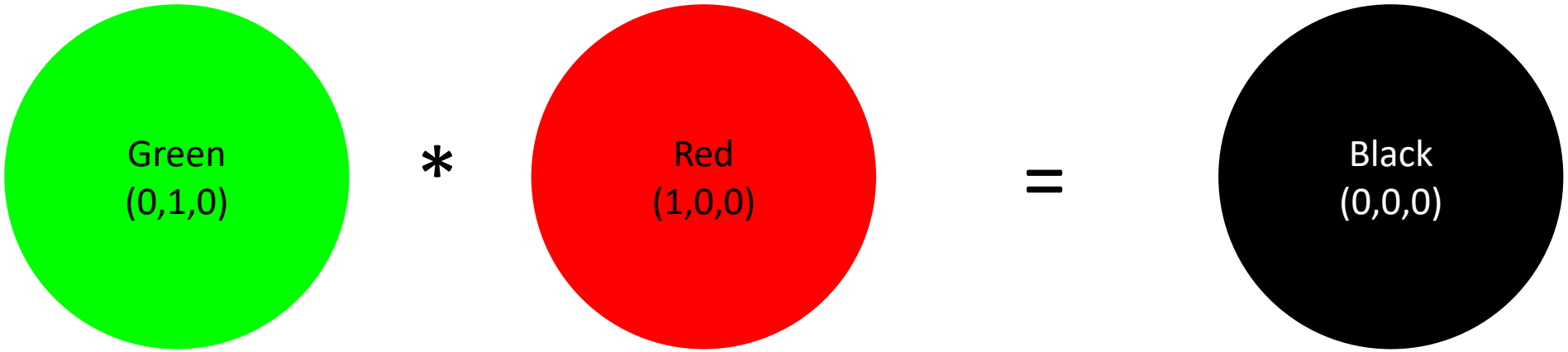
What can we do with Scalar Products?

Multiply Colors



What can we do with Scalar Products?

Multiply Colors

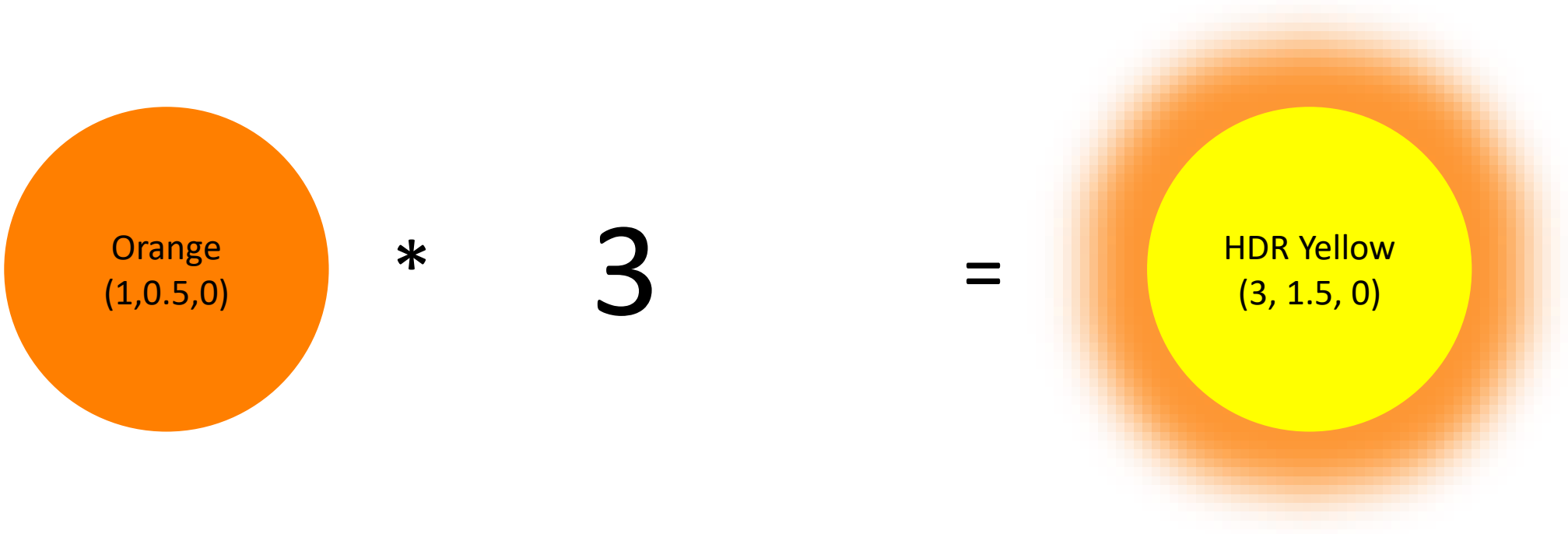


What can we do with Scalar Products?

Multiply Colors

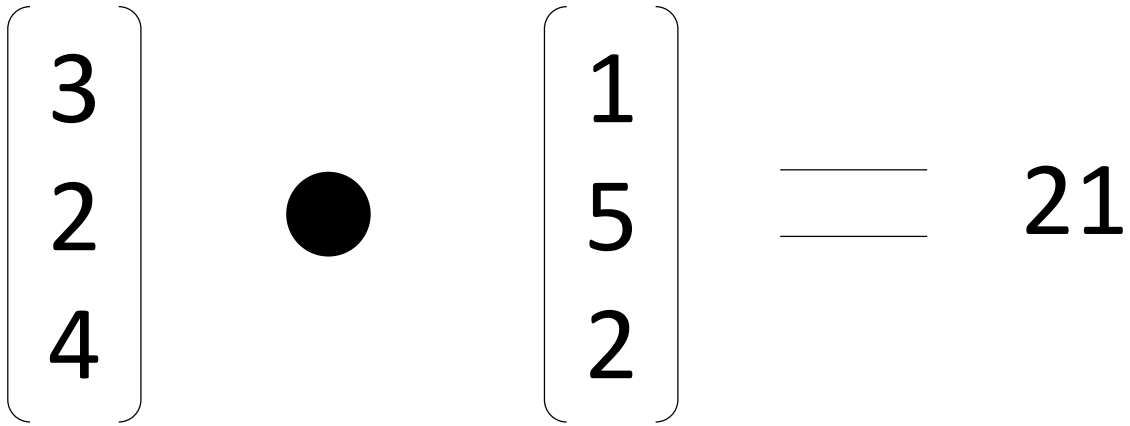
What can we do with Scalar Products?

Multiply Colors



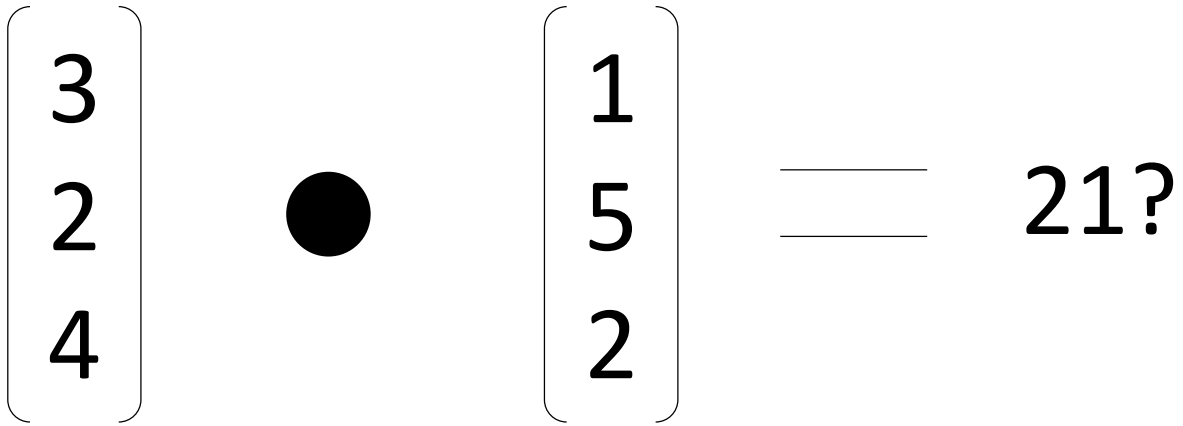
Multiplication (Dot Product)

(aka: the proper way to multiply Vectors)



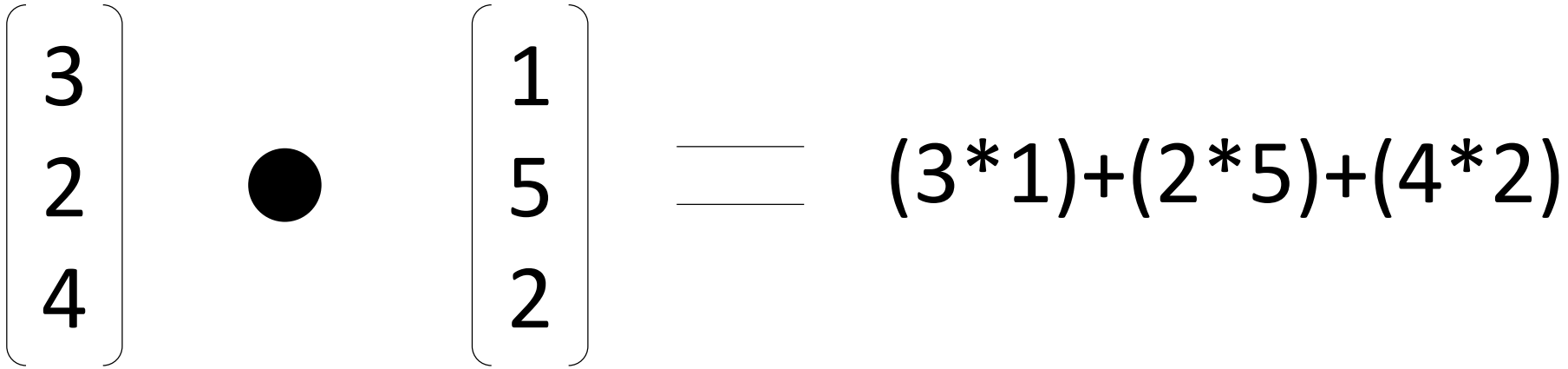
Multiplication (Dot Product)

(aka: the proper way to multiply Vectors)



Multiplication (Dot Product)

(aka: the proper way to multiply Vectors)



Multiplication (Dot Product)

(aka: the proper way to multiply Vectors)

HLSL/CG Syntax:

```
dot(a, b);
```

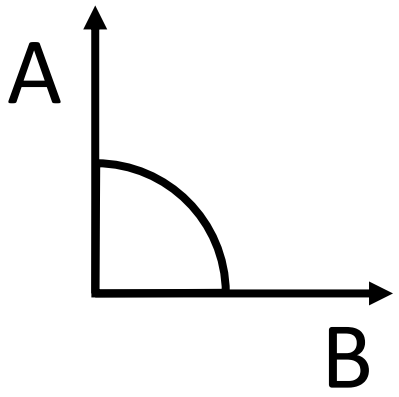
Multiplication (Dot Product)

(aka: the proper way to multiply Vectors)

What can we do with Dot Products?

What can we do with Dot Products?

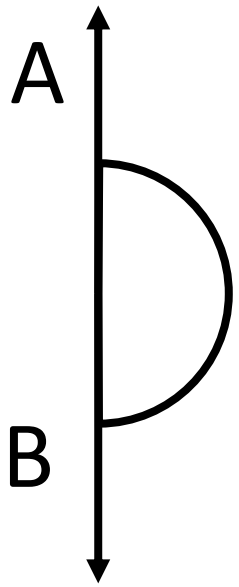
Tell the angle between two (normalized) Vectors



0

What can we do with Dot Products?

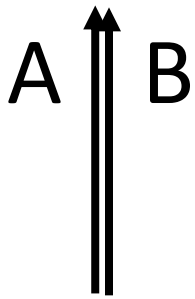
Tell the angle between two (normalized) Vectors



-1

What can we do with Dot Products?

Tell the angle between two (normalized) Vectors

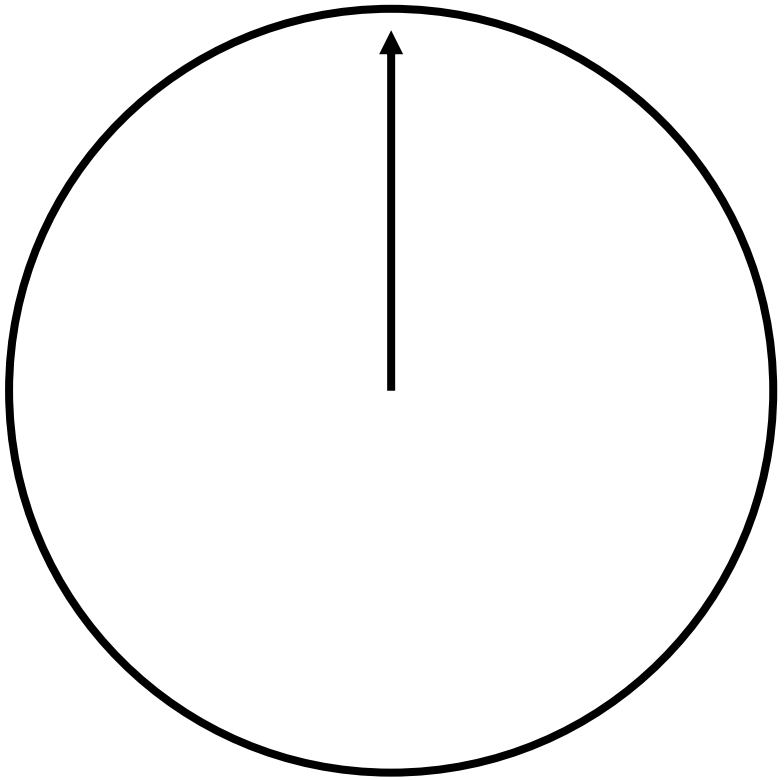


1

What can we do with Dot Products?

Normalized Vector?

A Vector with a length of 1 (usually directional Vectors are normalized)



What can we do with Dot Products?

Normalized Vector?

HLSL/CG Syntax:

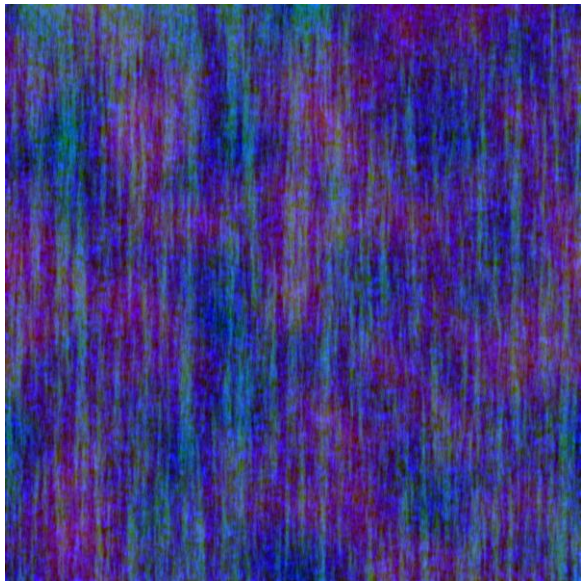
```
normalize(a);
```

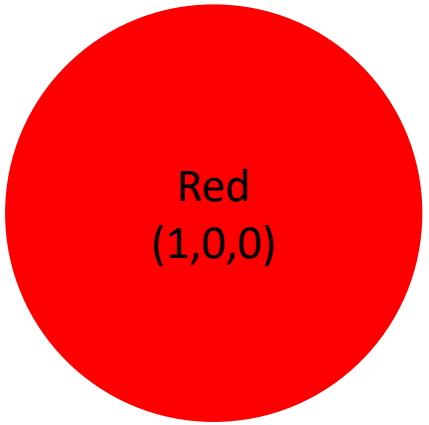
What can we do with Dot Products?

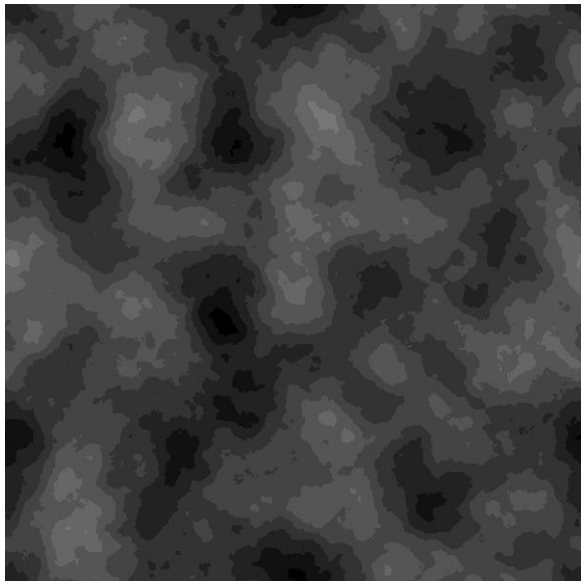
Back to Dot Products!

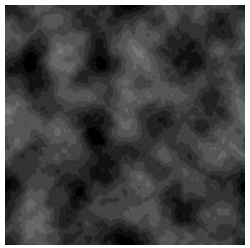
What can we do with Dot Products?

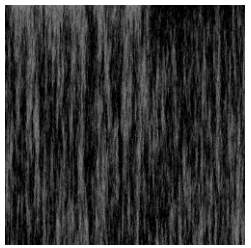
Colormask Textures

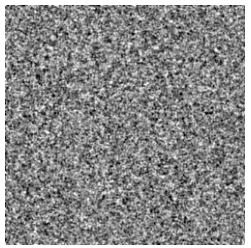






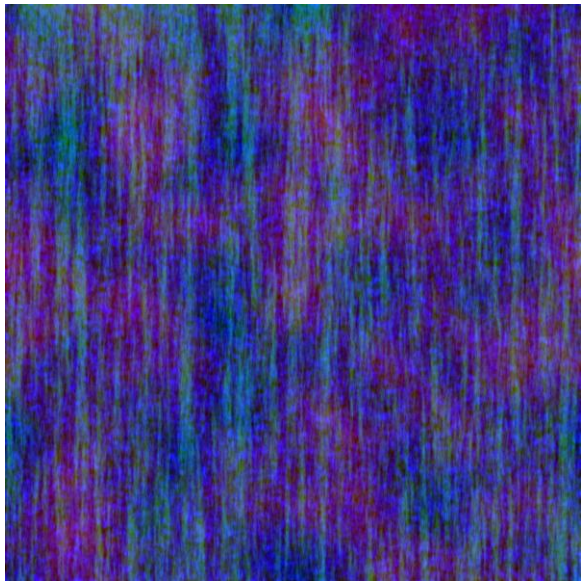


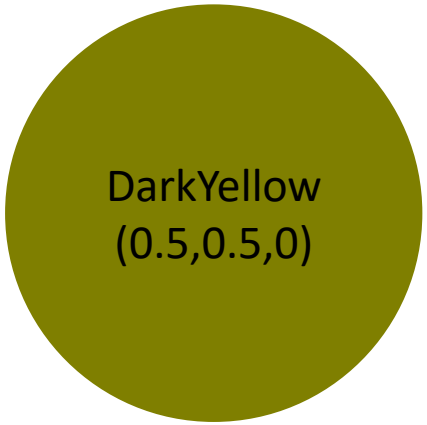


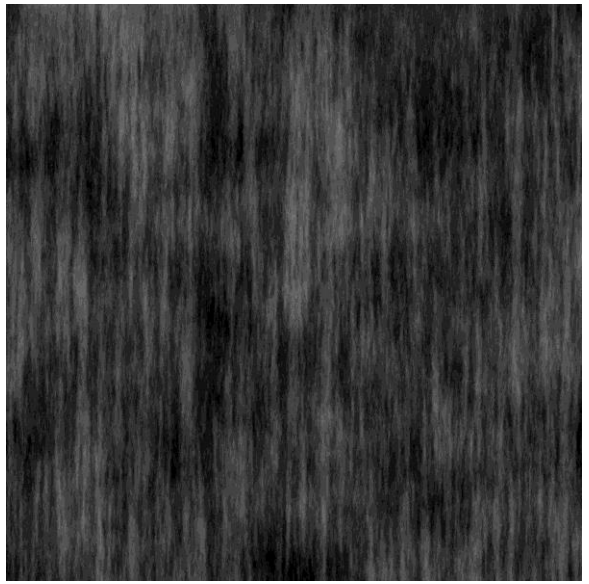


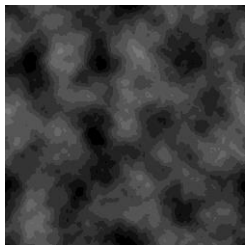
What can we do with Dot Products?

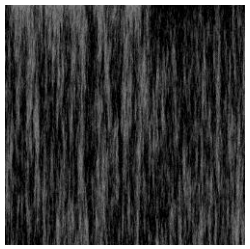
Colormask Textures

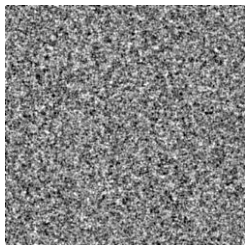








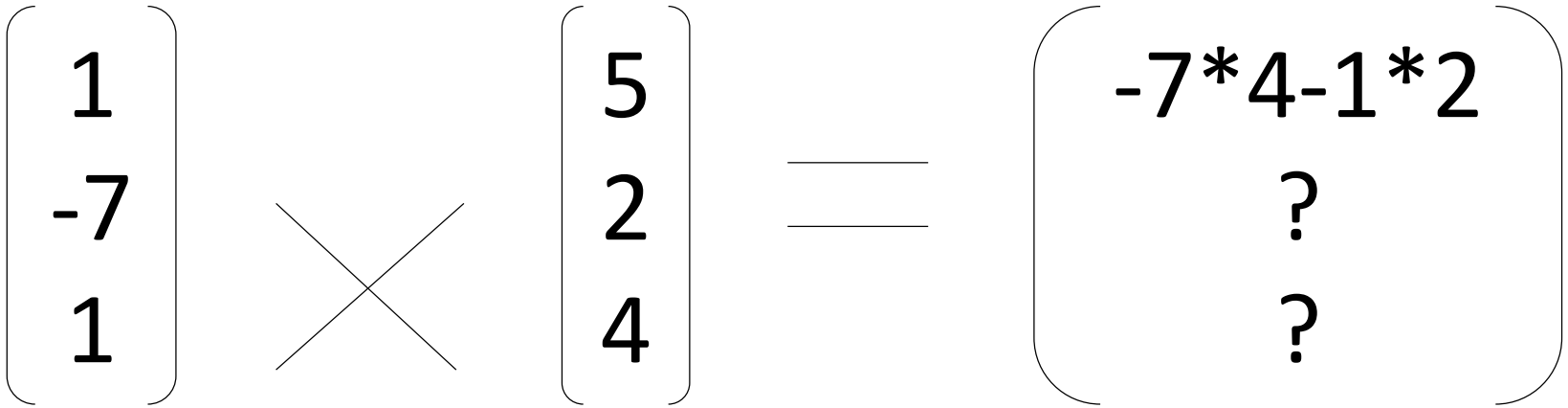




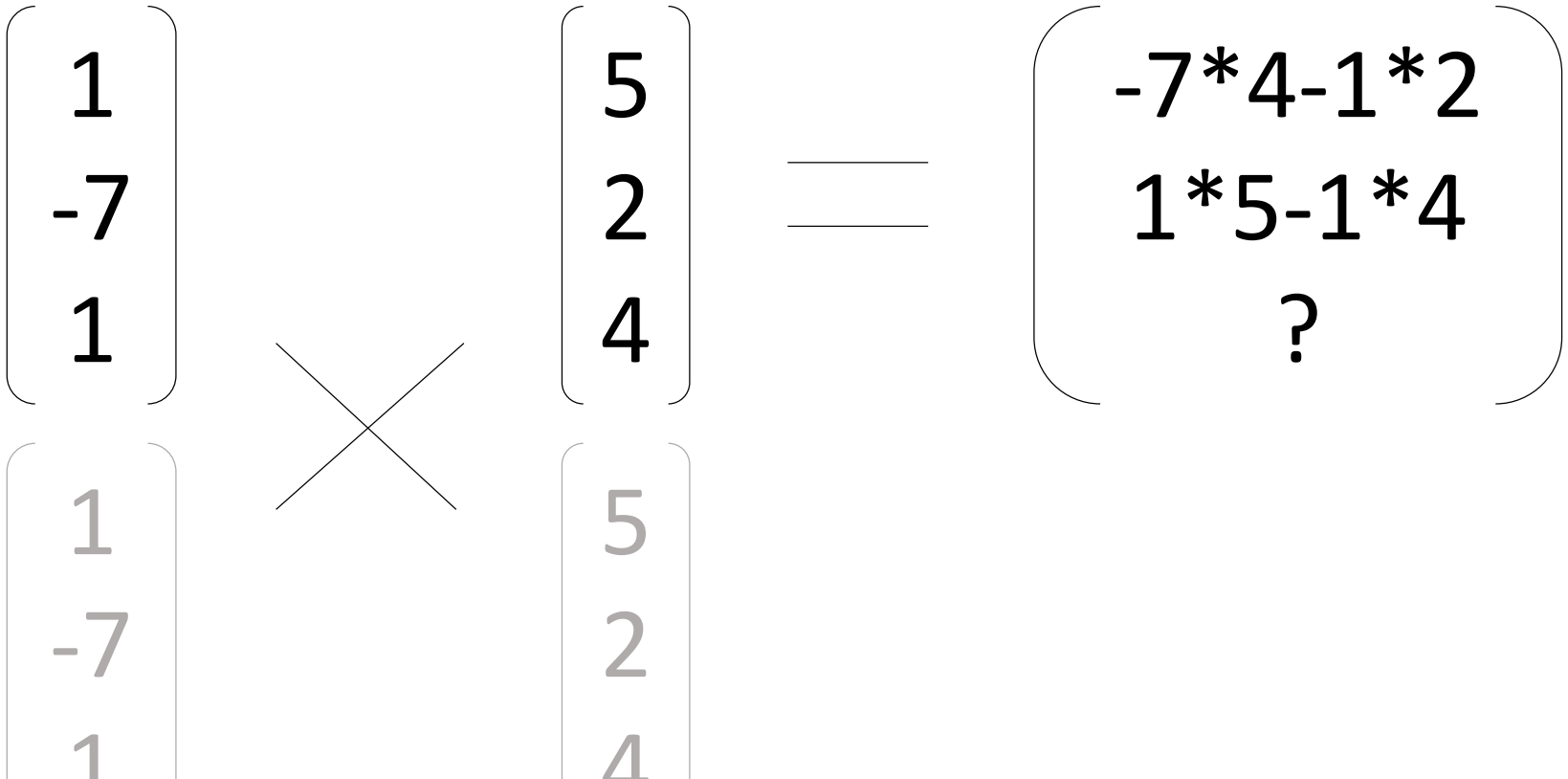
Multiplication (Cross Product)

$$\begin{pmatrix} 1 \\ -7 \\ 1 \end{pmatrix} \times \begin{pmatrix} 5 \\ 2 \\ 4 \end{pmatrix} = \begin{pmatrix} -30 \\ 1 \\ 37 \end{pmatrix}$$

Multiplication (Cross Product)



Multiplication (Cross Product)



Multiplication (Cross Product)

$$\begin{pmatrix} 1 \\ -7 \\ 1 \end{pmatrix} \times \begin{pmatrix} 5 \\ 2 \\ 4 \end{pmatrix} = \begin{pmatrix} -7*4-1*2 \\ 1*5-1*4 \\ 1*2- -7*5 \end{pmatrix}$$

Multiplication (Cross Product)

$$\begin{pmatrix} 1 \\ -7 \\ 1 \end{pmatrix} \times \begin{pmatrix} 5 \\ 2 \\ 4 \end{pmatrix} = \begin{pmatrix} -30 \\ 1 \\ 37 \end{pmatrix}$$

Multiplication (Cross Product)

HLSL/CG Syntax:

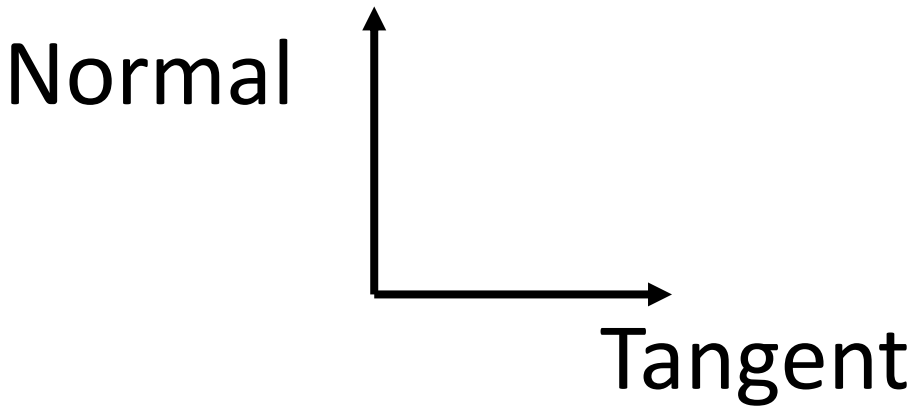
```
cross(a, b);
```

Multiplication (Cross Product)

What can we do with Cross Products?

What can we do with Cross Products?

Encode the Bitangent



What can we do with Cross Products?

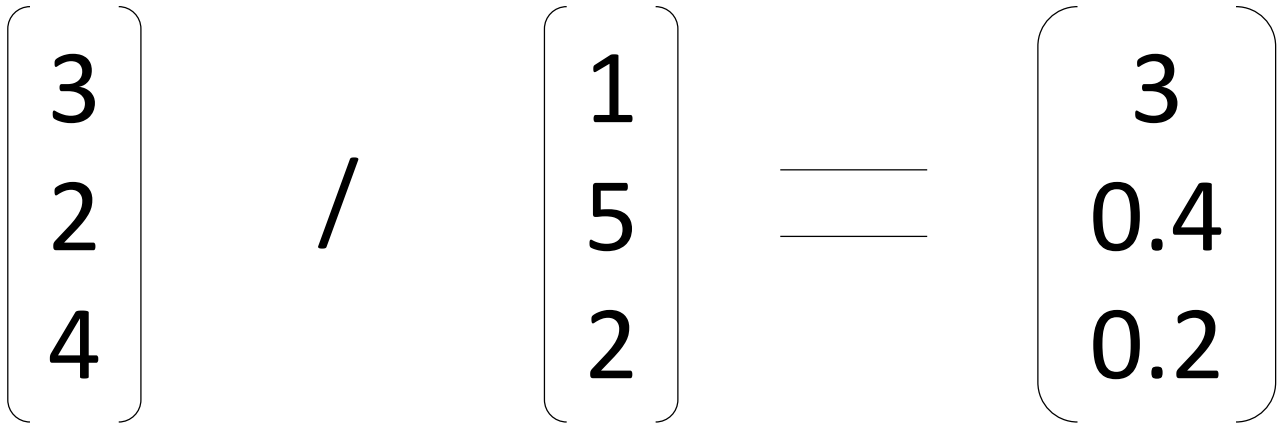
Encode the Bitangent

Division?

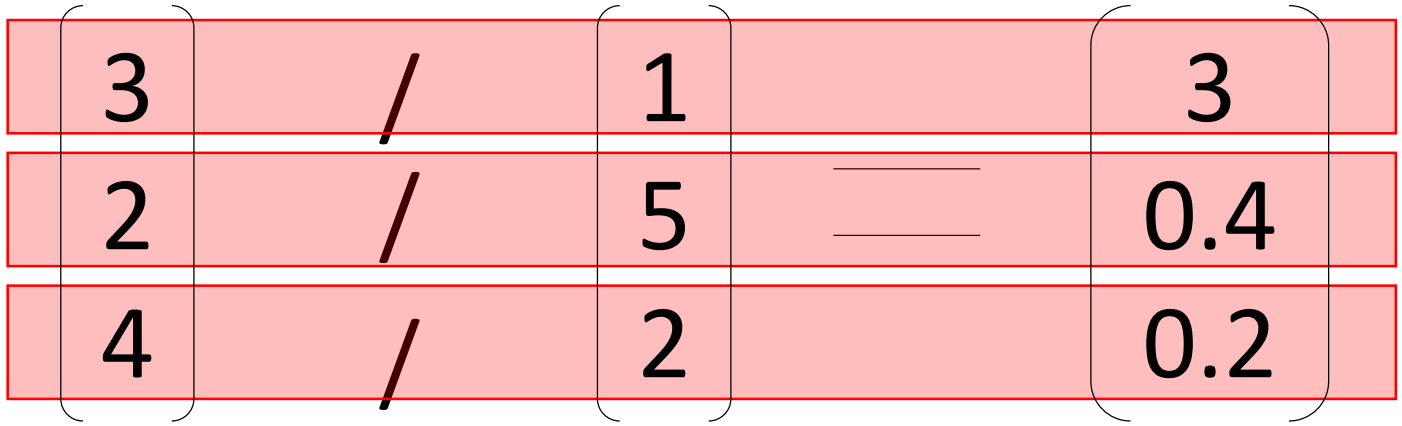
$$\begin{pmatrix} 3 \\ 2 \\ 4 \end{pmatrix} \; / \; \begin{pmatrix} 1 \\ 5 \\ 2 \end{pmatrix} = \; ?$$

DISCLAIMER!
This is not very mathematical, but it‘s just too darn useful, so we‘ve built it into Shaders anyway

Division



Division



DISCLAIMER!
This is not very mathematical, but it‘s just too darn useful, so we‘ve built it into Shaders anyway

Division

HLSL/CG Syntax:

```
a/b;
```

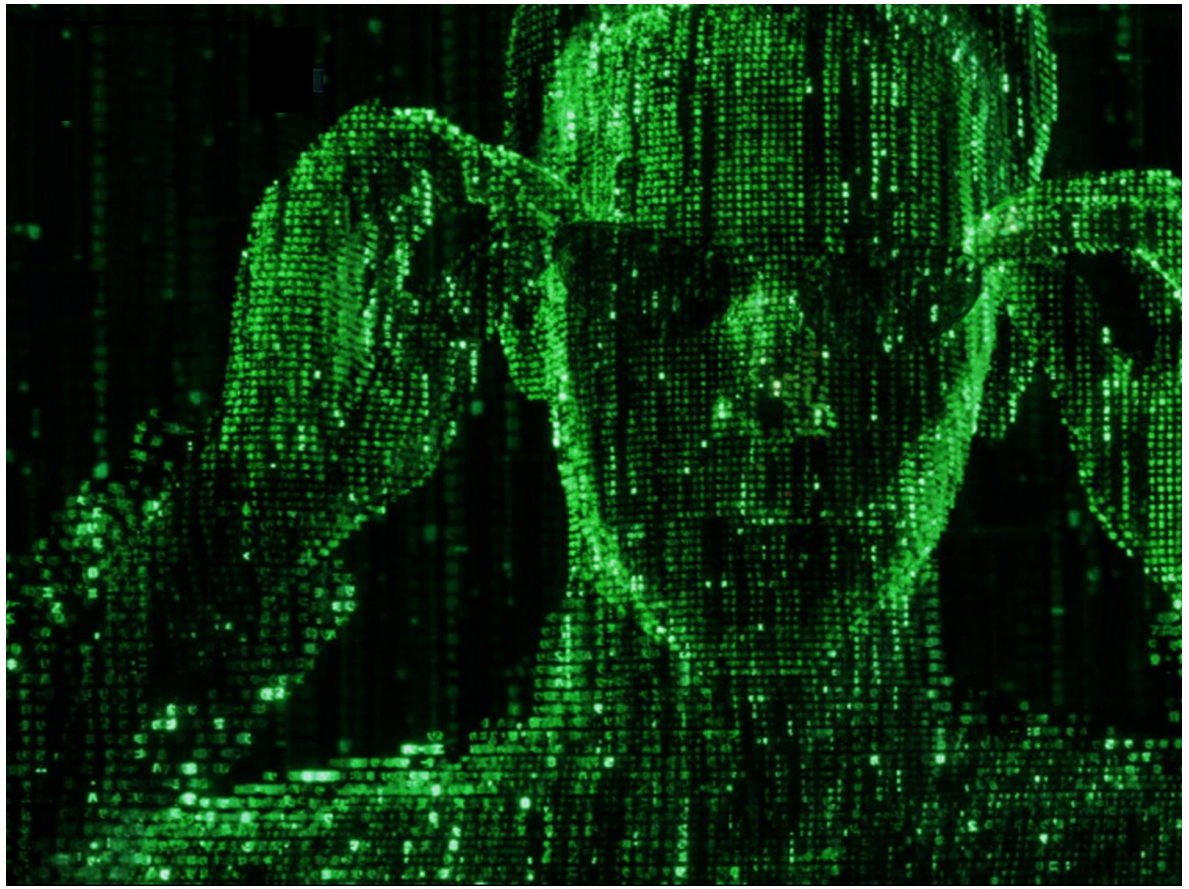
Swizzle!!!

Swizzle

HLSL/CG Syntax:

```
a.xyz = b.xxy;
a.rgb = b.aaa;
a.rgba = b.xyzw;
```

Vectors and Matrices



Matrices

The Matrix is a 1999 American-Australian science fiction action film written and directed by The Wachowskis, starring Keanu Reeves, Laurence Fishburne, Carrie-Anne Moss, Hugo Weaving, and Joe Pantoliano. It depicts a dystopian future in which reality as perceived by most humans is actually a simulated reality called "the Matrix", created by sentient machines to subdue the human population, while their bodies' heat and electrical activity are used as an energy source. Computer programmer "Neo" learns this truth and is drawn into a rebellion against the machines, which involves other people who have been freed from the "dream world".

(https://en.wikipedia.org/wiki/The_Matrix)

Matrices

In mathematics, a **matrix** (plural **matrices**) is a rectangular *array*[1] of numbers, symbols, or expressions, arranged in *rows* and *columns*.[2][3] For example, the dimensions of matrix (**1**) are *2* × *3* (read "two by three"), because there are two rows and three columns.

$$\begin{bmatrix} 1 & 9 & -13 \\ 20 & 5 & -6 \end{bmatrix}$$

(https://en.wikipedia.org/wiki/Matrix_(mathematics))

Matrices

- Matrices come in all kinds of sizes
  - 3x3 Matrix
  - 4x4 Matrix
  - 4x3 Matrix
  - 3x4 Matrix
  - 2x2 Matrix

Matrices

HLSL/CG Syntax:

```
float3x3 mat = {1,2,3,1,4,2,3,1,2};
```

Matrices

What can we do with Matrices?

What can we do with Matrices?

- Transform Positions
  - Move
  - Rotate
  - Scale

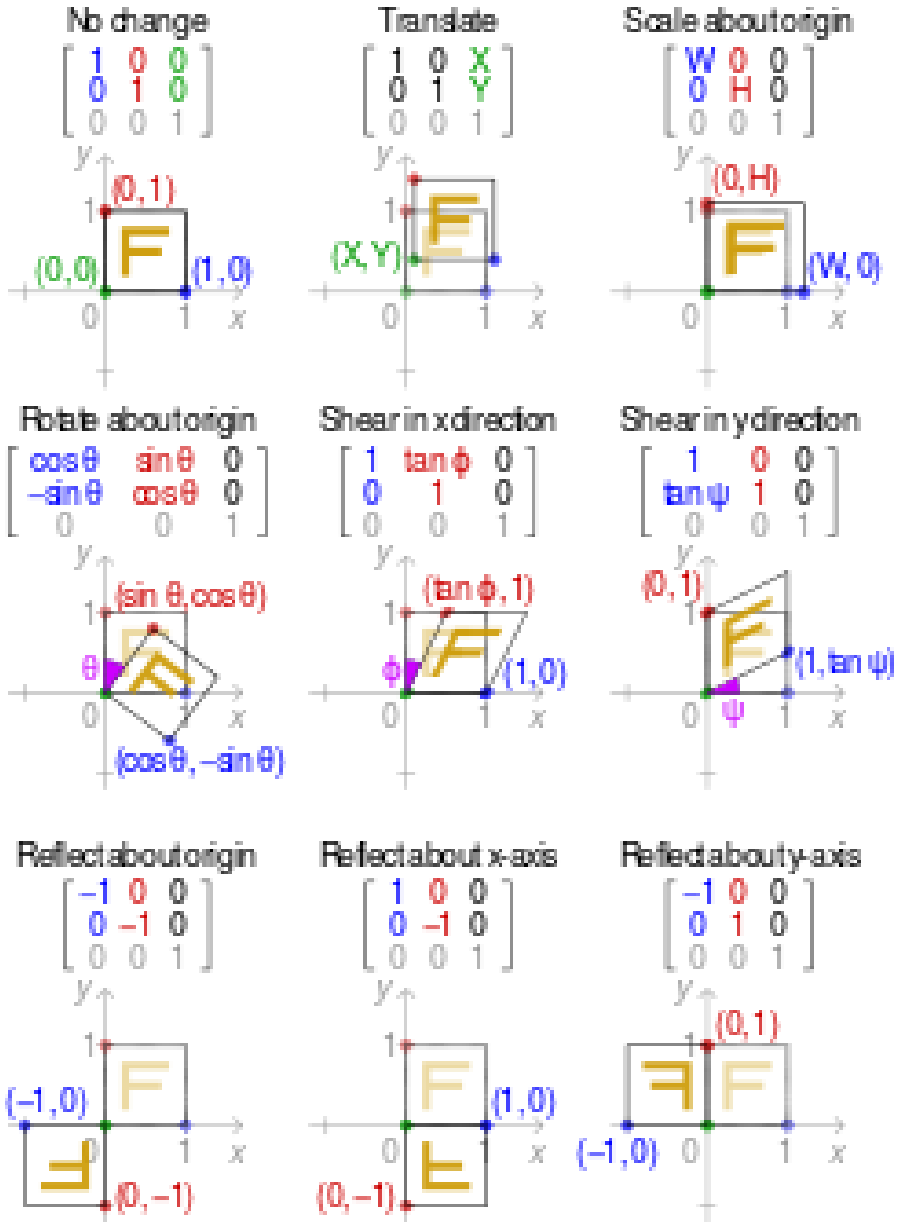
- Perspectively Distort Positions

What can we do with Matrices?

- Multiply them with other Matrices
- Multiply them with Vectors

How does it work?

- Transformation Matrices
- Multiply a Position with the Matrix



Matrices

HLSL/CG Syntax:

```
mul(matA, vecB);
```

How does it work?

- Multiply two matrices to combine their effects

Matrices

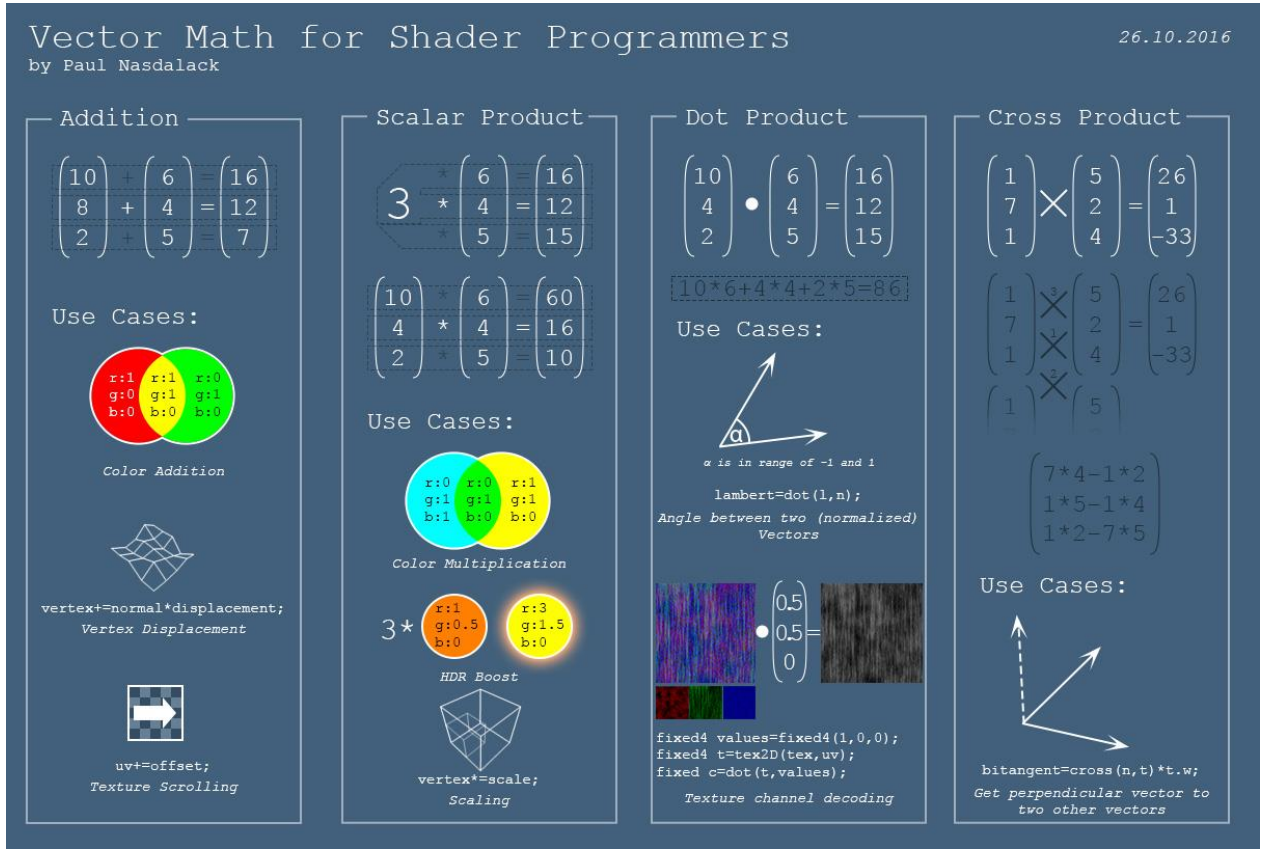
HLSL/CG Syntax:

```
matC = mul(matA, matB);
vecA = mul(matB, vecA);
vecA = mul(matA, vecA);
//multiplying with matC is the same as
//first multiplying MatB and then MatA
vecB = mul(matC, vecB);
```

Popular Matrices (in Shaders)

- Model/World Matrix
  - Moves the model to its position
- View/Camera Matrix
  - Moves the „World“ in front of the camera
- Projection Matrix
  - Distorts the world perspectively
- MVP Matrix
  - all the above multiplied together

Cheat Sheet!



http://bit.ly/2dQzwvD