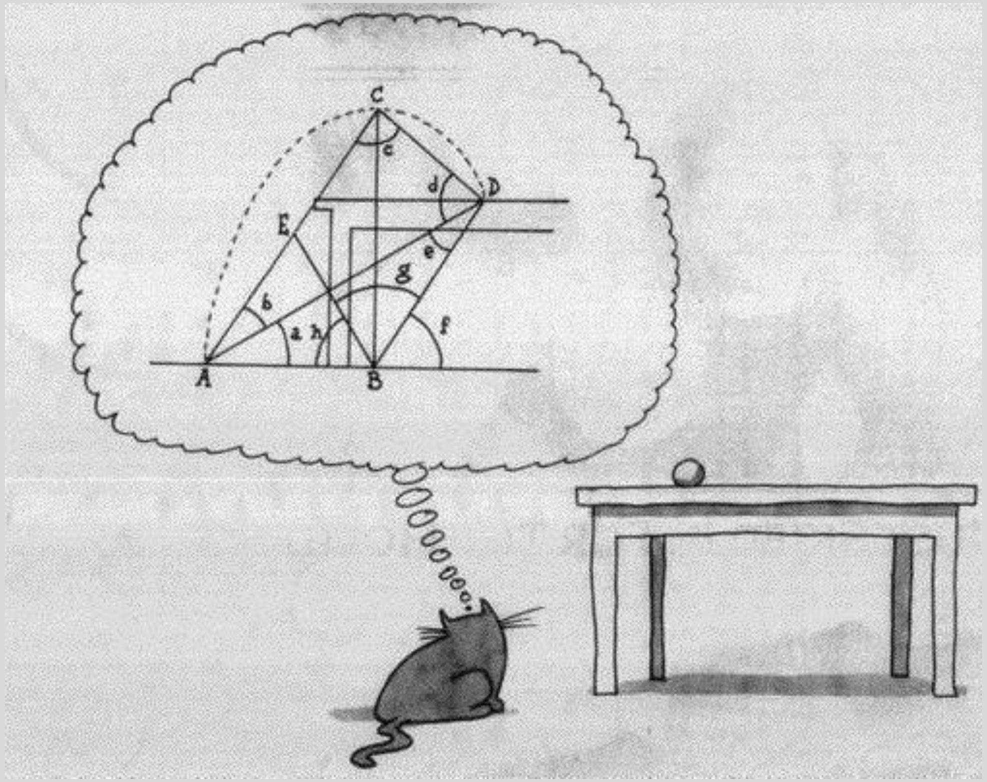
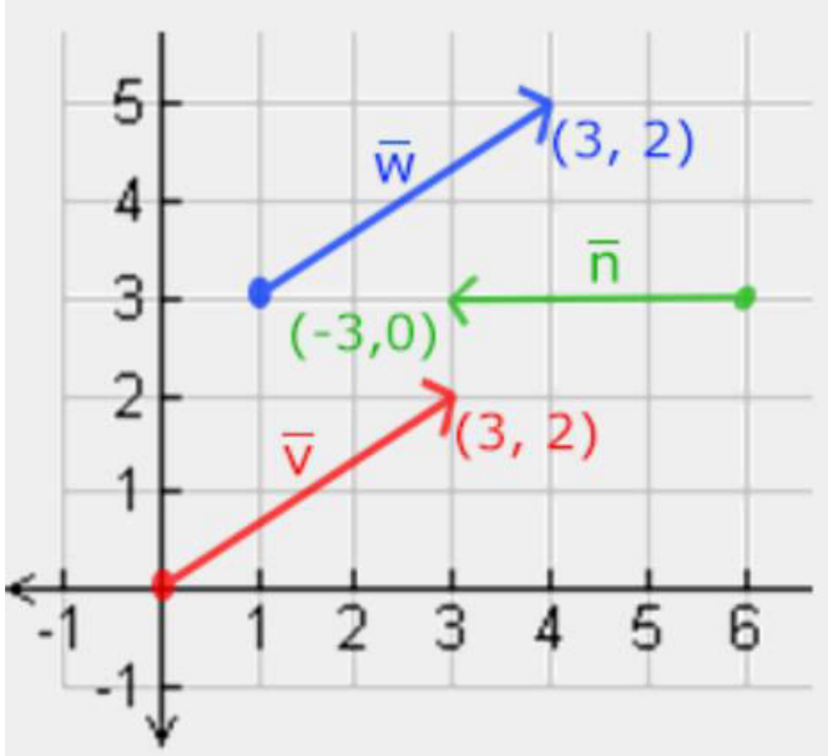


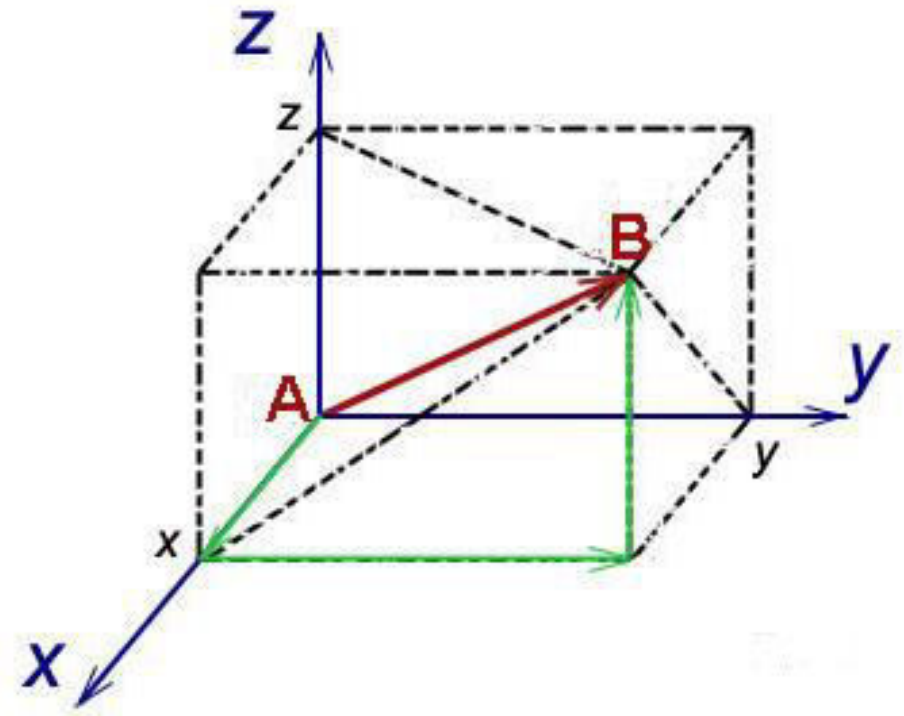
Лінійна алгебра та 3D-графіка. Трансформації

КУРС З 3D-ПРОГРАМУВАННЯ «ВІД ТРИКУТНИКА ДО СЦЕНИ».

Вектор



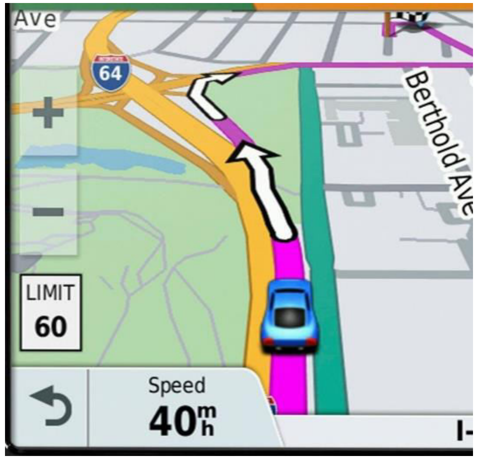
$$\bar{v} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



Вектор

```
float vertices[] = {
    // positions          // colors           // texture coords
     0.5f,  0.5f, 0.0f,   1.0f, 0.0f, 0.0f,   1.0f, 1.0f,   // top right
     0.5f, -0.5f, 0.0f,   0.0f, 1.0f, 0.0f,   1.0f, 0.0f,   // bottom right
    -0.5f, -0.5f, 0.0f,   0.0f, 0.0f, 1.0f,   0.0f, 0.0f,   // bottom left
    -0.5f,  0.5f, 0.0f,   1.0f, 1.0f, 0.0f,   0.0f, 1.0f    // top left
};
```

Вектор



Положення **Швидкість** **Напрямок**

Скаляр

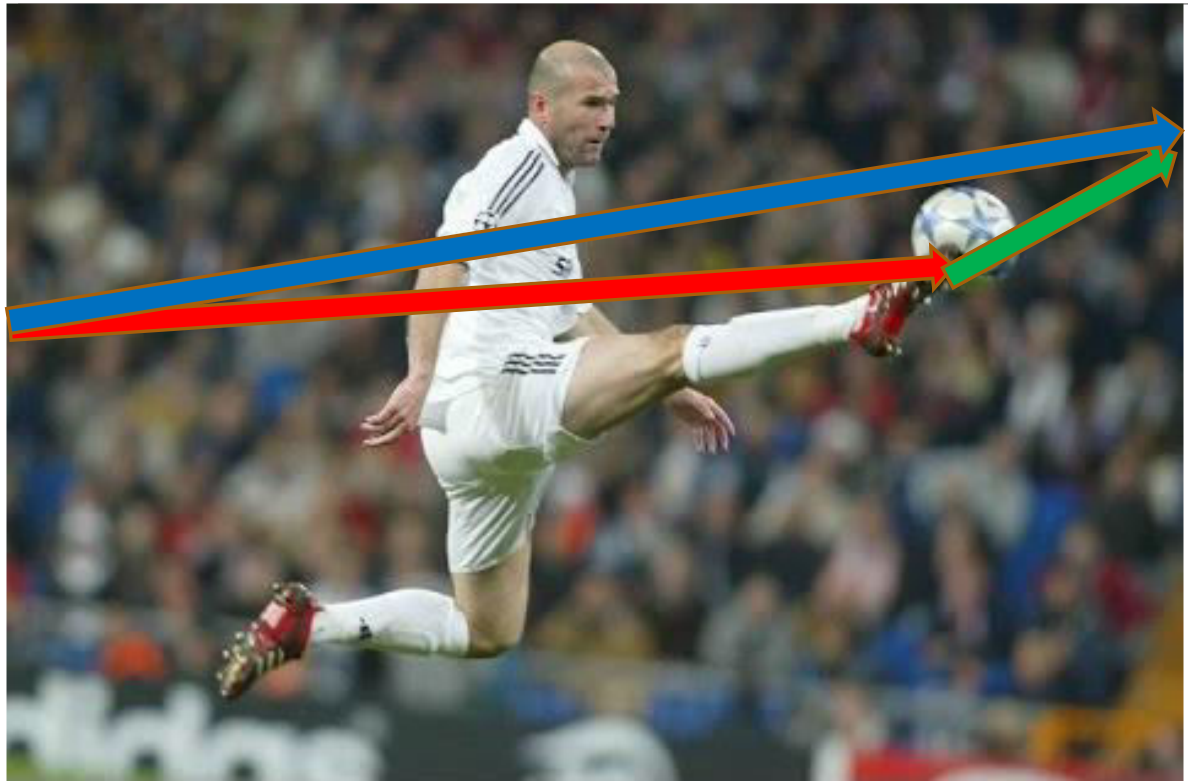
$$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} + x = \begin{pmatrix} 1 + x \\ 2 + x \\ 3 + x \end{pmatrix}$$

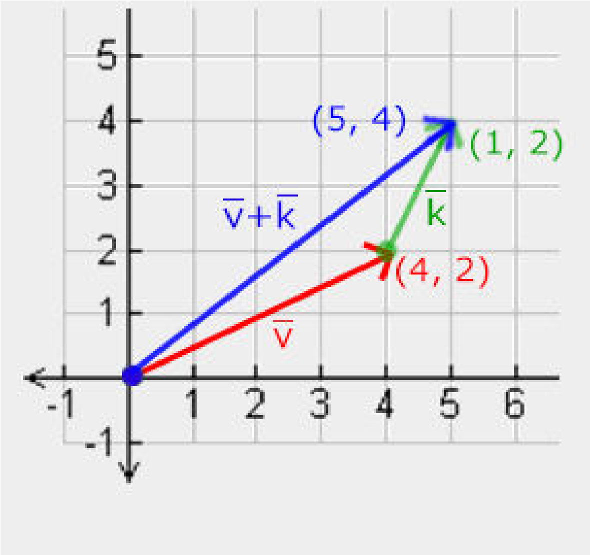
-, *, /

Швидкість руху супермена – (10, 20, 10), в кожному кадрі її потрібно зменшувати з урахуванням опору вітру на 0.9. Якою Буде нова швидкість в наступному кадрі?

0.9*(10, 20) = (0.9 * 10, 0.9 * 20, 0.9*10) = (9, 18, 9).

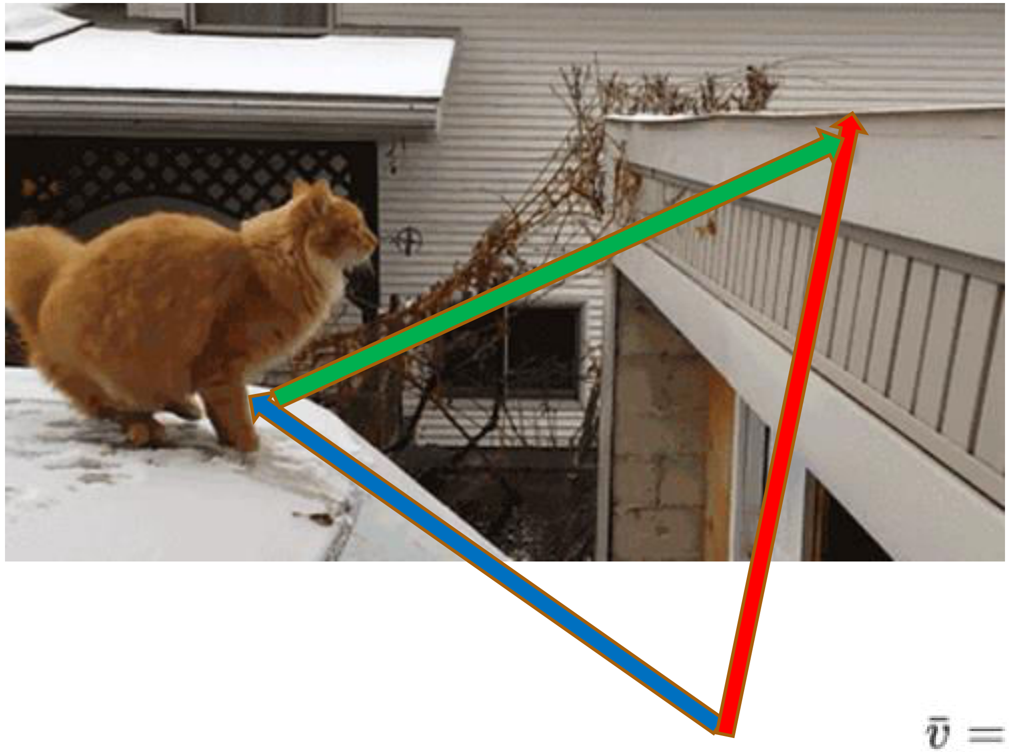
Додавання векторів

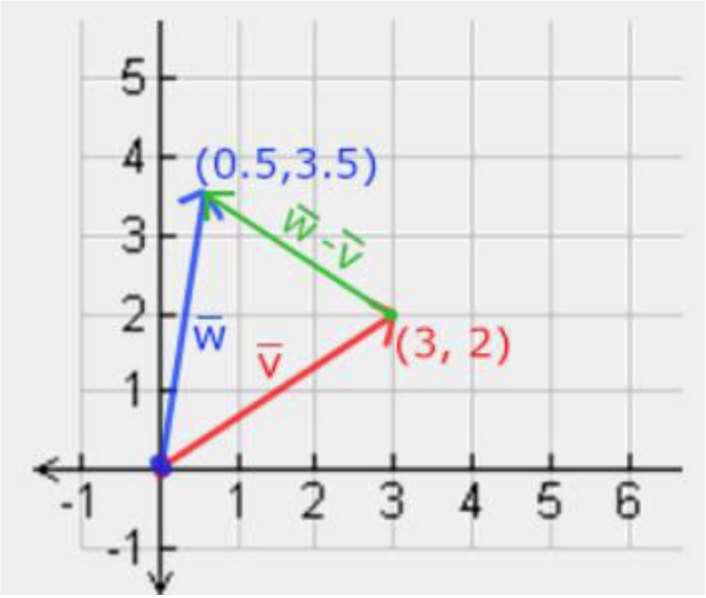




$$\bar{v} = \begin{pmatrix}1\\2\\3\end{pmatrix}, \bar{k} = \begin{pmatrix}4\\5\\6\end{pmatrix} \rightarrow \bar{v} + \bar{k} = \begin{pmatrix}1+4\\2+5\\3+6\end{pmatrix} = \begin{pmatrix}5\\7\\9\end{pmatrix}$$

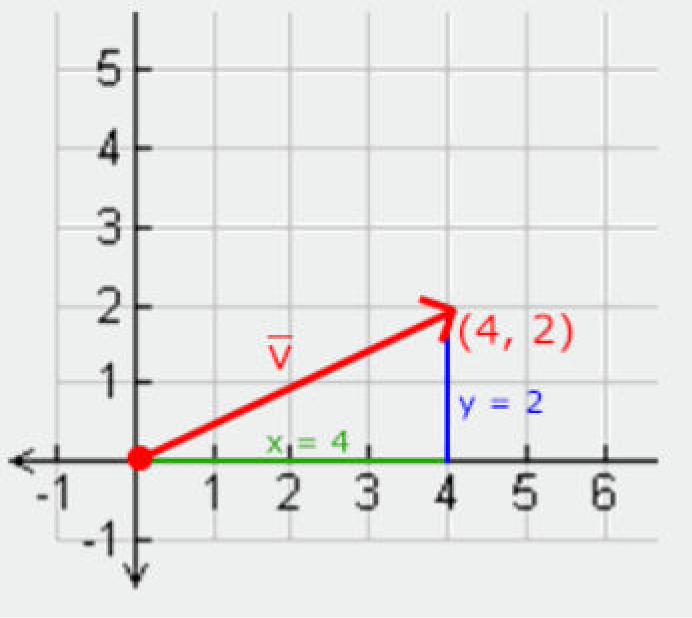
Віднімання векторів





$$\bar{v} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \bar{k} = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} \rightarrow \bar{v} + -\bar{k} = \begin{pmatrix} 1 + (-4) \\ 2 + (-5) \\ 3 + (-6) \end{pmatrix} = \begin{pmatrix} -3 \\ -3 \\ -3 \end{pmatrix}$$

Довжина (модуль) вектора

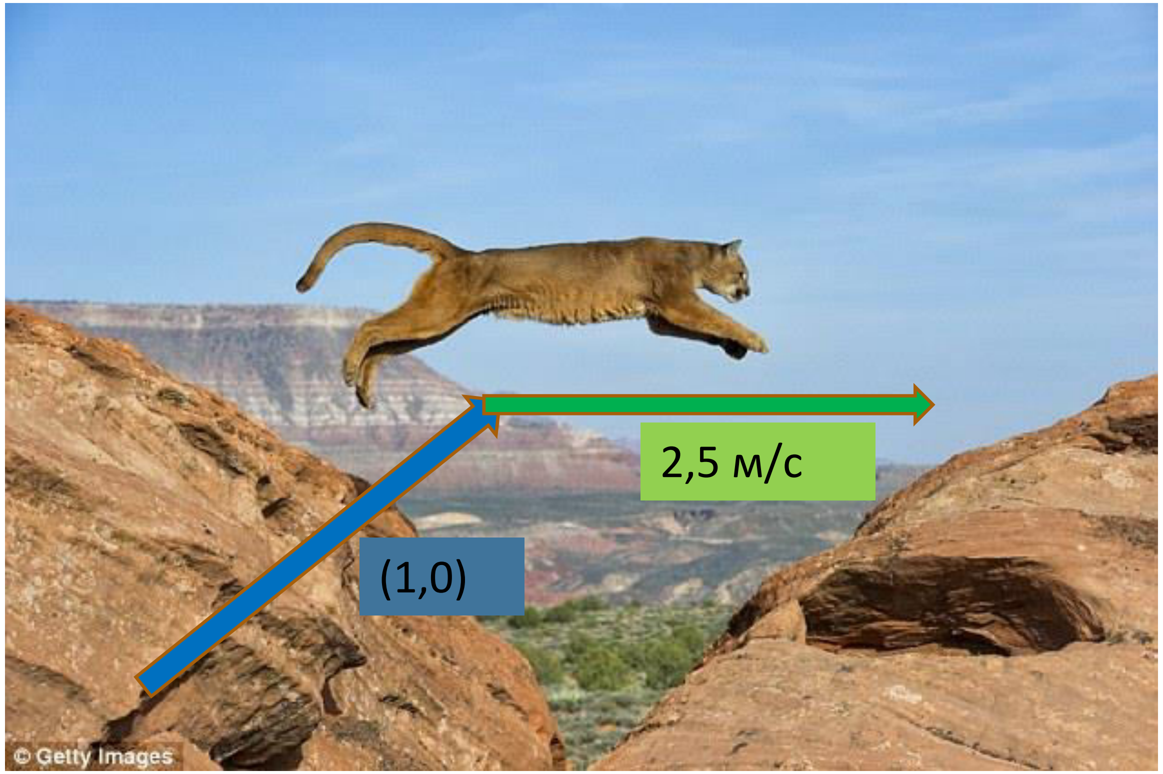


Швидкість удару в більярді (1.5, 1.2). На яку відстань відкотиться куля після удару?

S=sqrt(2.25+1.44) = 1.92 м

$$||\bar{v}|| = \sqrt{x^2 + y^2}$$

Нормалізований вектор



$$\hat{n} = \frac{\bar{v}}{||\bar{v}||}$$

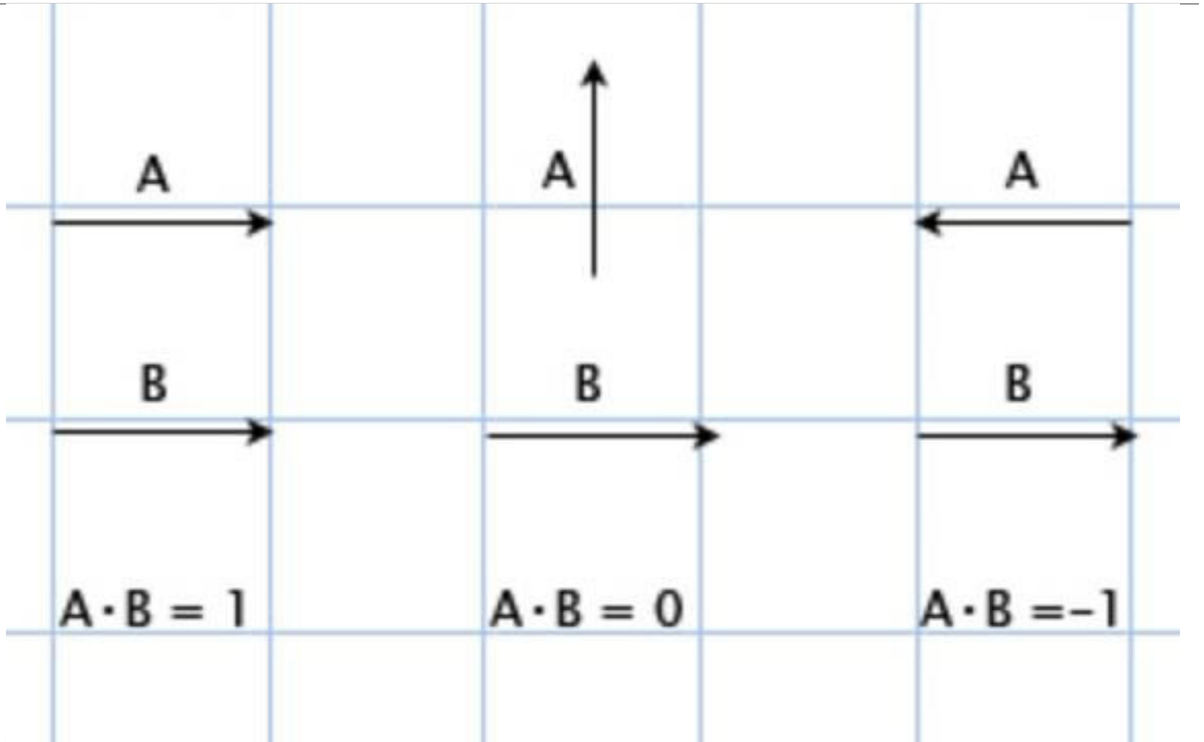
Пума стрибає з місця в напрямку (1, 0)

зі швидкістю 2,5 м/с.

На яку відстань вона вистрибне?

Скалярний добуток

$$\bar{v} \cdot \bar{k} = ||\bar{v}|| \cdot ||\bar{k}|| \cdot \cos\theta$$



$$\begin{pmatrix} 0.6 \\ -0.8 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = (0.6 * 0) + (-0.8 * 1) + (0 * 0) = -0.8$$

Скалярний добуток

$$\bar{v} \cdot \bar{k} = 1 \cdot 1 \cdot \cos\theta = \cos\theta$$

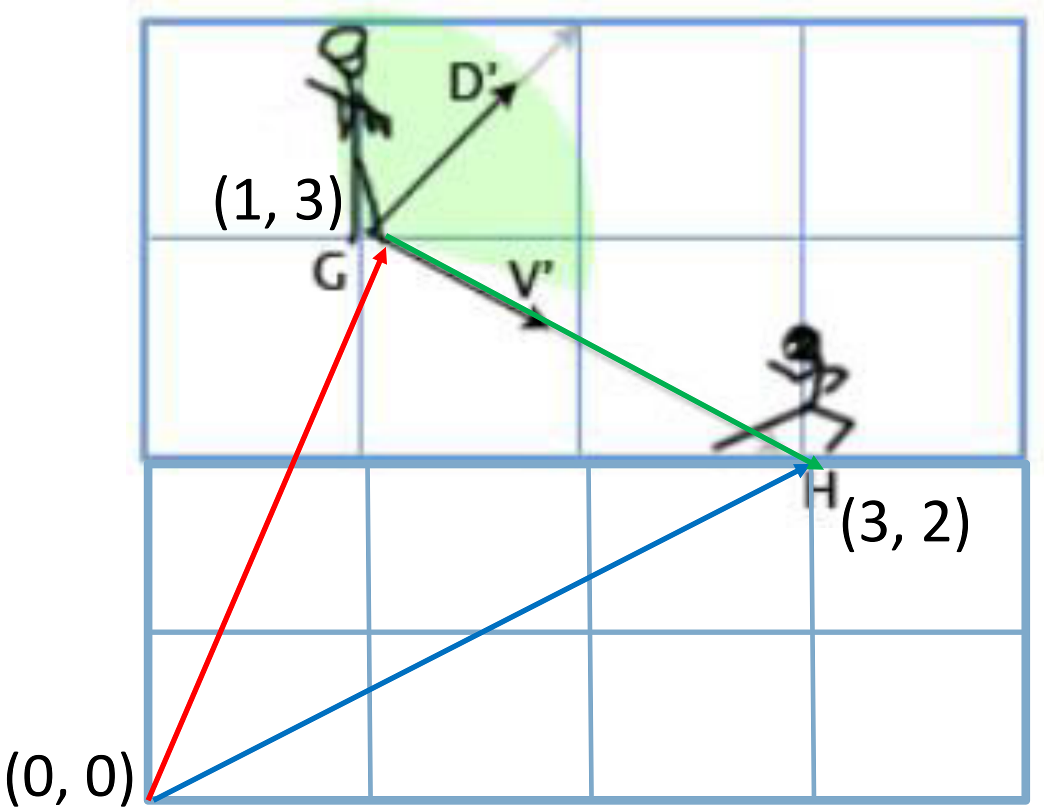
$$\theta = acos\ (v \bullet k)$$

Охоронець розташований у G(1, 3) дивиться у напрямку D(1,1), з кутом огляду 120 ◦.

Головний герой знаходиться на позиції H(3, 2).

Чи потрапить герой у поле зору охоронця?

Скалярний добуток



1) Знаходимо різницю векторів:

V = H — G = (3, 2) — (1, 3) = (3-1, 2-3) = (2, -1)

2) Нормалізуємо вектори V і D:

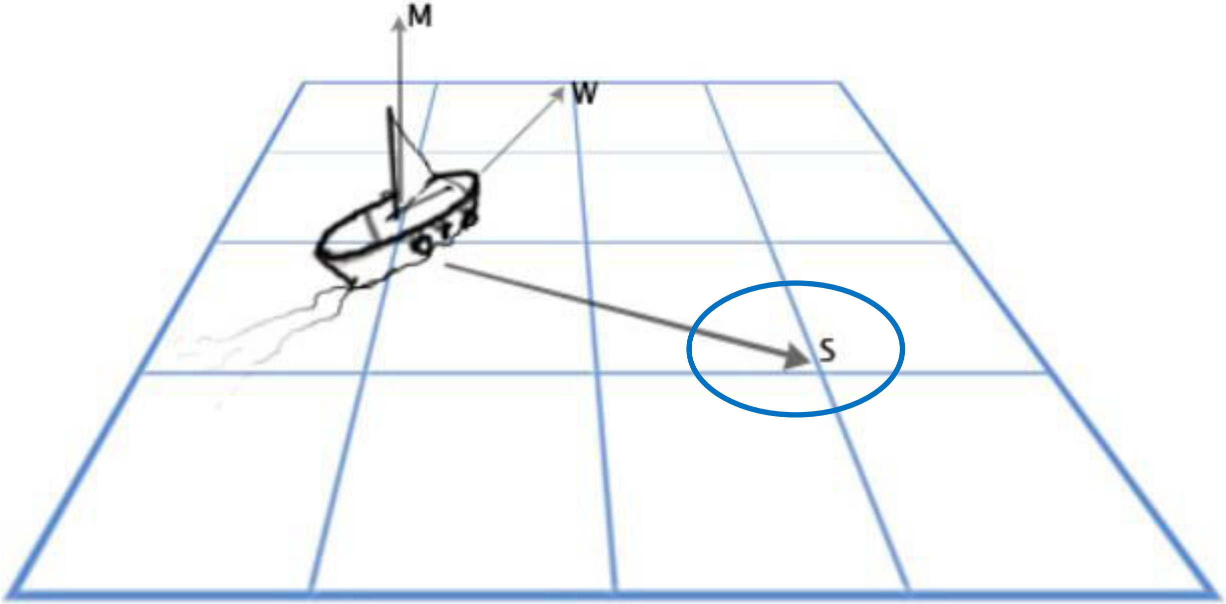
D' = D / |D| = (1, 1) / sqrt(1^2 + 1^2) = (1, 1) / sqrt(2) = (0.71, 0.71)
V' = V / |V| = (2, -1) / sqrt(2^2 + $(-1)^2$) = (2,-1) / sqrt(5) = (0.89, -0.45)

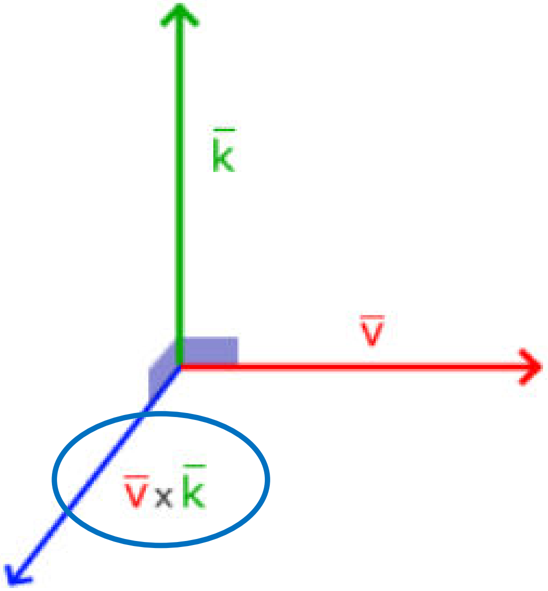
3) Знаходимо кут між ними:

Θ = acos(D'V') = acos(0.71*0.89 + 0.71*(-0.45)) = acos(0.31) = 72 градуси. >(120/2)

НЕ БАЧИТЬ

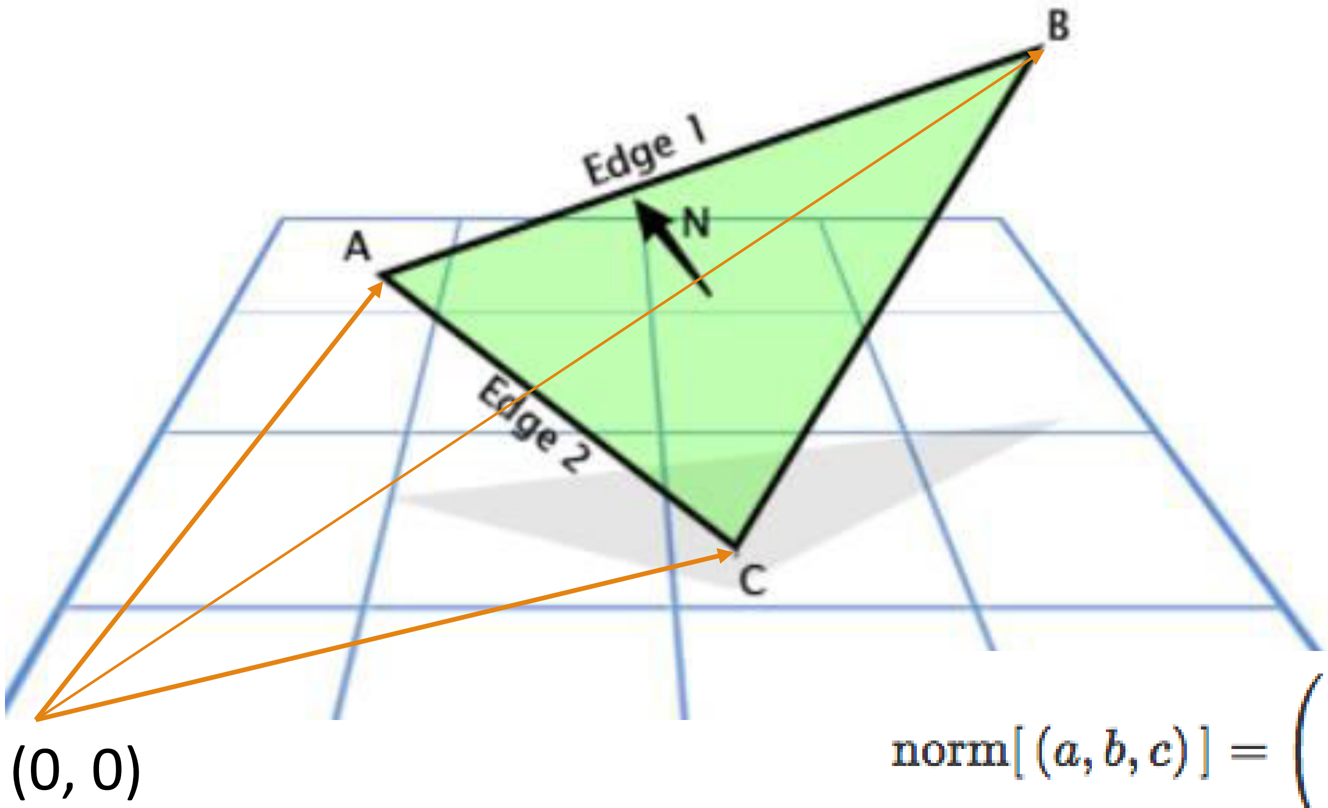
Векторний добуток





$$\begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} \times \begin{pmatrix} B_x \\ B_y \\ B_z \end{pmatrix} = \begin{pmatrix} A_y \cdot B_z - A_z \cdot B_y \\ A_z \cdot B_x - A_x \cdot B_z \\ A_x \cdot B_y - A_y \cdot B_x \end{pmatrix}$$

Векторний добуток



A (Ax, Ay, Az), B(Bx, By, Bz), C(Cx, Cy, Cz)

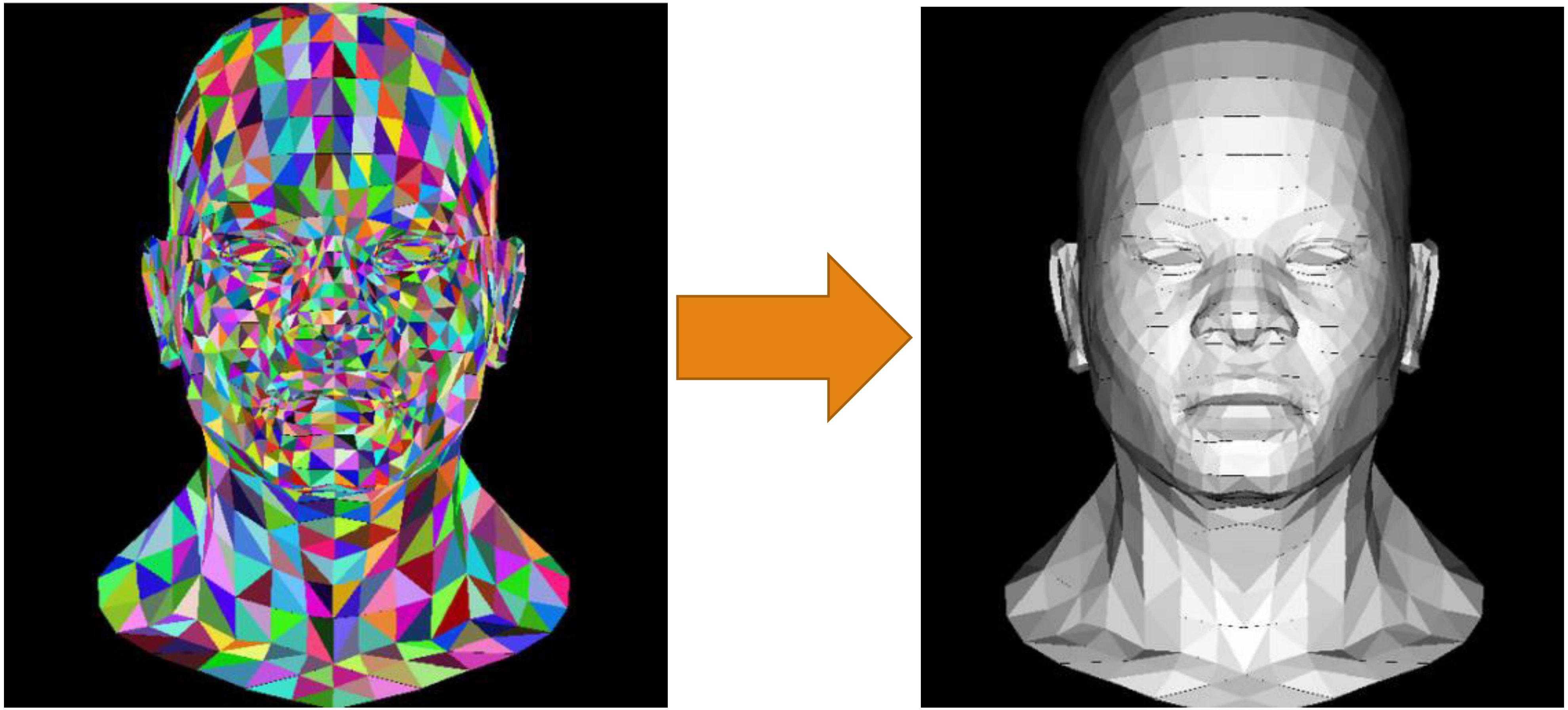
Edge1= B-A

Edge2=C-A

N=Edge1×Edge2

$$\mathrm{norm}[\,(a,b,c)\,] = \left(\frac{a}{\sqrt{a^2+b^2+c^2}}, \frac{b}{\sqrt{a^2+b^2+c^2}}, \frac{c}{\sqrt{a^2+b^2+c^2}}\right)$$

Векторний добуток і нормалі



Освітлення і векторний добуток

Вектор джерела світла S (**0, 0, 1**).

Визначаємо одиничну нормаль Ni для кожного трикутника.

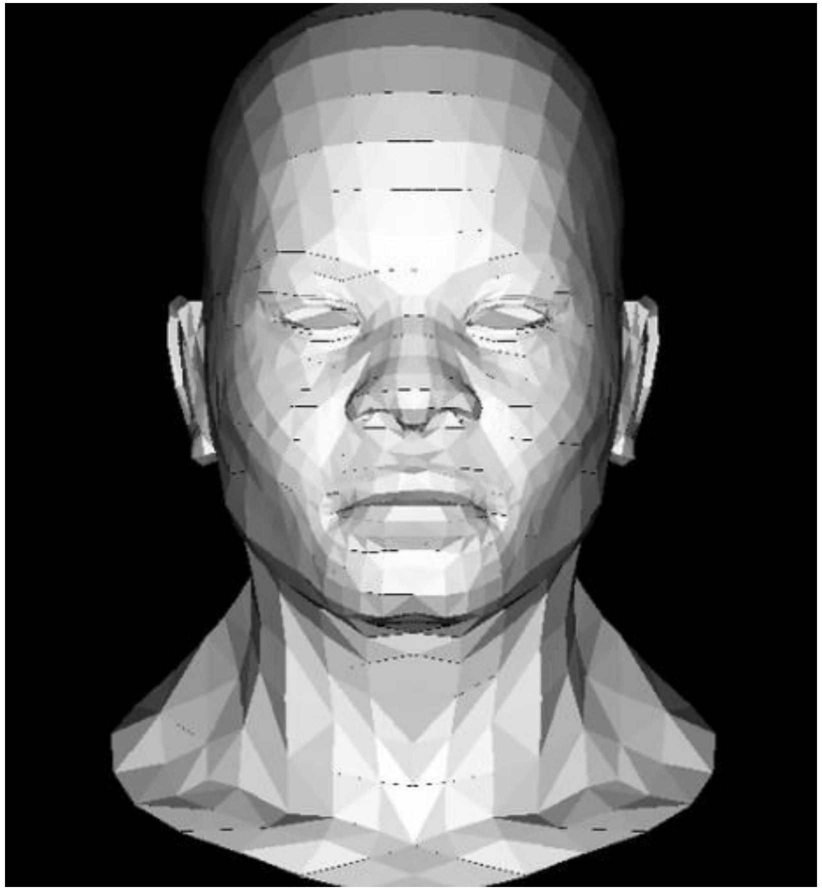
Що дасть скалярний добуток S·N*i ?* -

Коефіцієнт оствітлення трикутника:

— 0, зовсім не освітлений – чорний.

— 0.5, напівосвітлений - сірий 50%.

— 1, повністю освітлений, білий.



Матриці

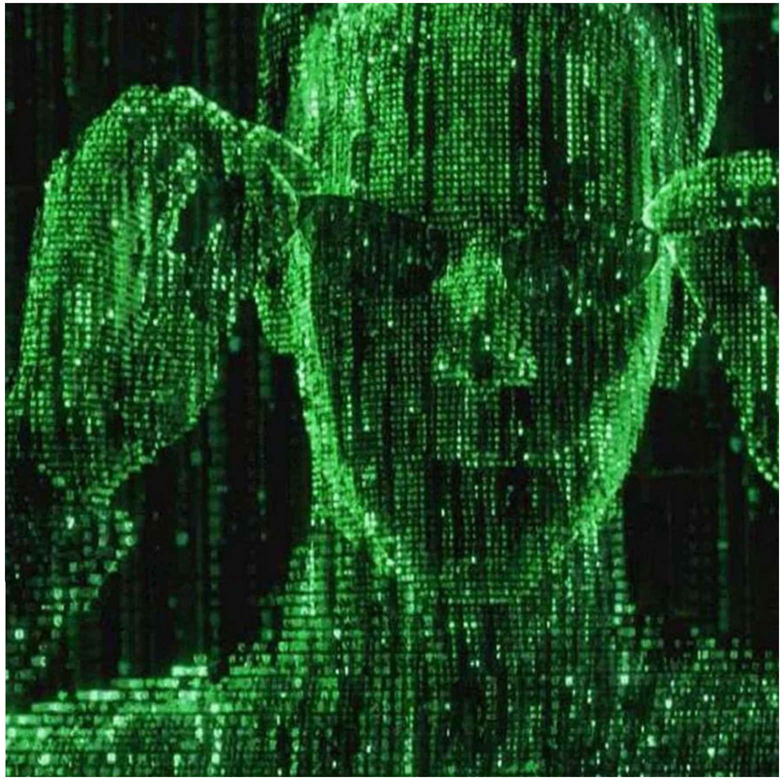
$$a = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

Добуток на скаляр:

$$2 \cdot \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 2 \cdot 1 & 2 \cdot 2 \\ 2 \cdot 3 & 2 \cdot 4 \end{bmatrix} = \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix}$$

Додавання/віднімання однорозмірних матриць:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1+5 & 2+6 \\ 3+7 & 4+8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$



Множення матриць

1. Можна множити матриці, в яких к-ть стовпців першої збігається з к-тю рядків другої.

2. A * B != B * A.

3. Результат: матриця (n,m), де n – к-ть рядків лівої матриці, а m – к-ть стовпців правої.

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1 \cdot 5 + 2 \cdot 7 & 1 \cdot 6 + 2 \cdot 8 \\ 3 \cdot 5 + 4 \cdot 7 & 3 \cdot 6 + 4 \cdot 8 \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

Нічого не нагадує?

$$\begin{pmatrix} 0.6 \\ -0.8 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = (0.6 * 0) + (-0.8 * 1) + (0 * 0) = -0.8$$

Скалярний добуток векторів

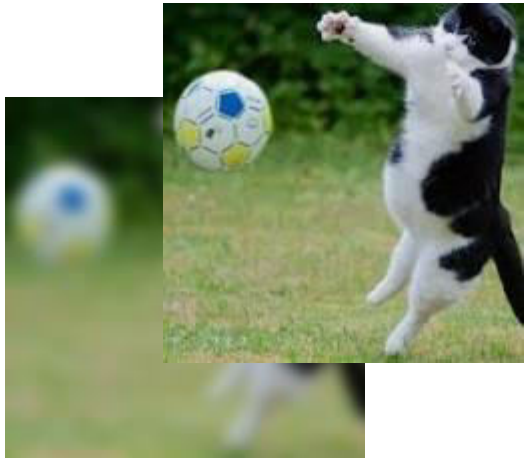
Матриця x Вектор

- Матриця - 4x4
- Вектор – матриця 4x1
- Результат: оновлений вектор 4x1

$$\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} ax + by + cz + dw \\ ex + fy + gz + hw \\ ix + jy + kz + lw \\ mx + ny + oz + pw \end{bmatrix}$$

Трансформації фігур

Вихідна фігура



Зсув
(Translation)

Обертання
(Rotation)

Масштабування
(Scale)

GLM: OpenGL Mathematics

```
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
```

В програмі на C++:

```
glm::mat4 myMatrix;
glm::vec4 myVector;
// fill myMatrix and myVector somehow
glm::vec4 transformedVector = myMatrix * myVector;
```

В шейдері на GLSL:

```
mat4 myMatrix;
vec4 myVector;
// fill myMatrix and myVector somehow
vec4 transformedVector = myMatrix * myVector;
```

Матриця трансформації: одинична

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} 1*x+0*y+0*z+0*w \\ 0*x+1*y+0*z+0*w \\ 0*x+0*y+1*z+0*w \\ 0*x+0*y+0*z+1*w \end{bmatrix} = \begin{bmatrix} x+0+0+0 \\ 0+y+0+0 \\ 0+0+z+0 \\ 0+0+0+w \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

Щоб створити одиничну матрицю:

```
glm::mat4 myIdentityMatrix = glm::mat4(1.0f);
```

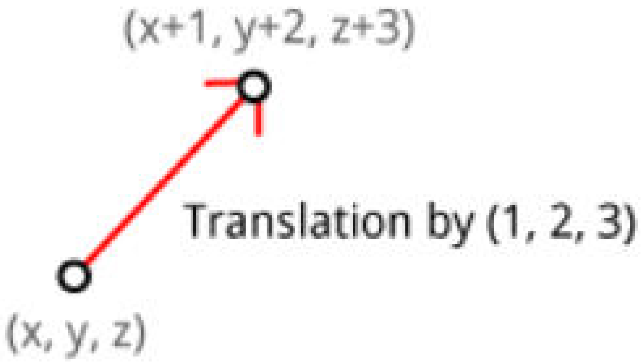
w – це гомогенна координата, використовується для 3D-візуалізації.

Коли **w=1** , тоді вектор (x,y,z,1) – положення в просторі

Коли **w=0** , тоді вектор (x,y,z,0) – напрямок.

Зсув (Translation)

$$\begin{bmatrix} 1 & 0 & 0 & X \\ 0 & 1 & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x + X \cdot 1 \\ y + Y \cdot 1 \\ z + Z \cdot 1 \\ 1 \end{pmatrix}$$



glm:: mat4 IdMatrix; // за замовчуванням одинична матриця

IdMatrix=glm:: translate(IdMatrix, glm::vec4(0.5f, -0.5f, 1.0f, 1.0f))

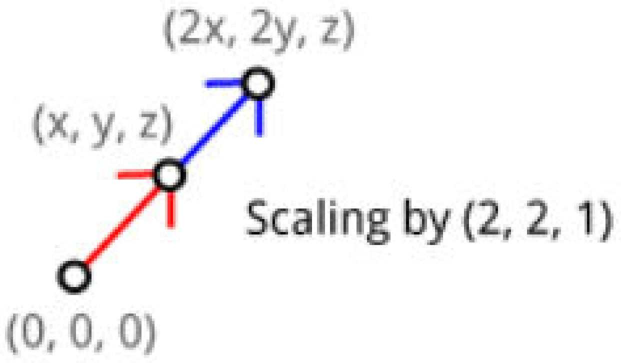
$X \quad Y \quad Z \quad w$

glm:: vec4 MyVector (0.5f, 0.5f, 0.0f, 1.0f);

glm:: vec4 TransVector=IdMatrix*MyVector;

Масштабування (Scale)

$$\begin{bmatrix} SX & 0 & 0 & 0 \\ 0 & SY & 0 & 0 \\ 0 & 0 & SZ & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} SX \cdot x \\ SY \cdot y \\ SZ \cdot z \\ 1 \end{pmatrix}$$



glm:: mat4 IdMatrix;

IdMatrix=glm:: scale(IdMatrix, glm::vec4(0.5f, 0.5f, 0.5f, 1.0f))

SX SY SZ w

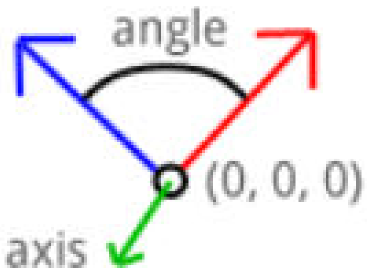
glm:: vec4 MyVector (1.0f, 1.0f, 1.0f, 1.0f);

glm:: vec4 TransVector=IdMatrix*MyVector;

Обертання (Rotate)

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ \cos\theta \cdot y - \sin\theta \cdot z \\ \sin\theta \cdot y + \cos\theta \cdot z \\ 1 \end{pmatrix}$$

Навколо X



$$\begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta \cdot x + \sin\theta \cdot z \\ y \\ -\sin\theta \cdot x + \cos\theta \cdot z \\ 1 \end{pmatrix}$$

Навколо Y

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta \cdot x - \sin\theta \cdot y \\ \sin\theta \cdot x + \cos\theta \cdot y \\ z \\ 1 \end{pmatrix}$$

Навколо Z

Обертання

Поворот навколо вісі Z на 90°:

glm::radians(90.0)

```
glm:: mat4 IdMatrix;
IdMatrix=glm:: rotate(IdMatrix, 90.0, glm::vec3(0.0f, 0.0f, 1.0f));

glm:: vec4 MyVector (0.5f, 0.5f, 0.0f, 1.0f);
glm:: vec4 TransVector=IdMatrix*MyVector;
```

Матриця трансформації в шейдері

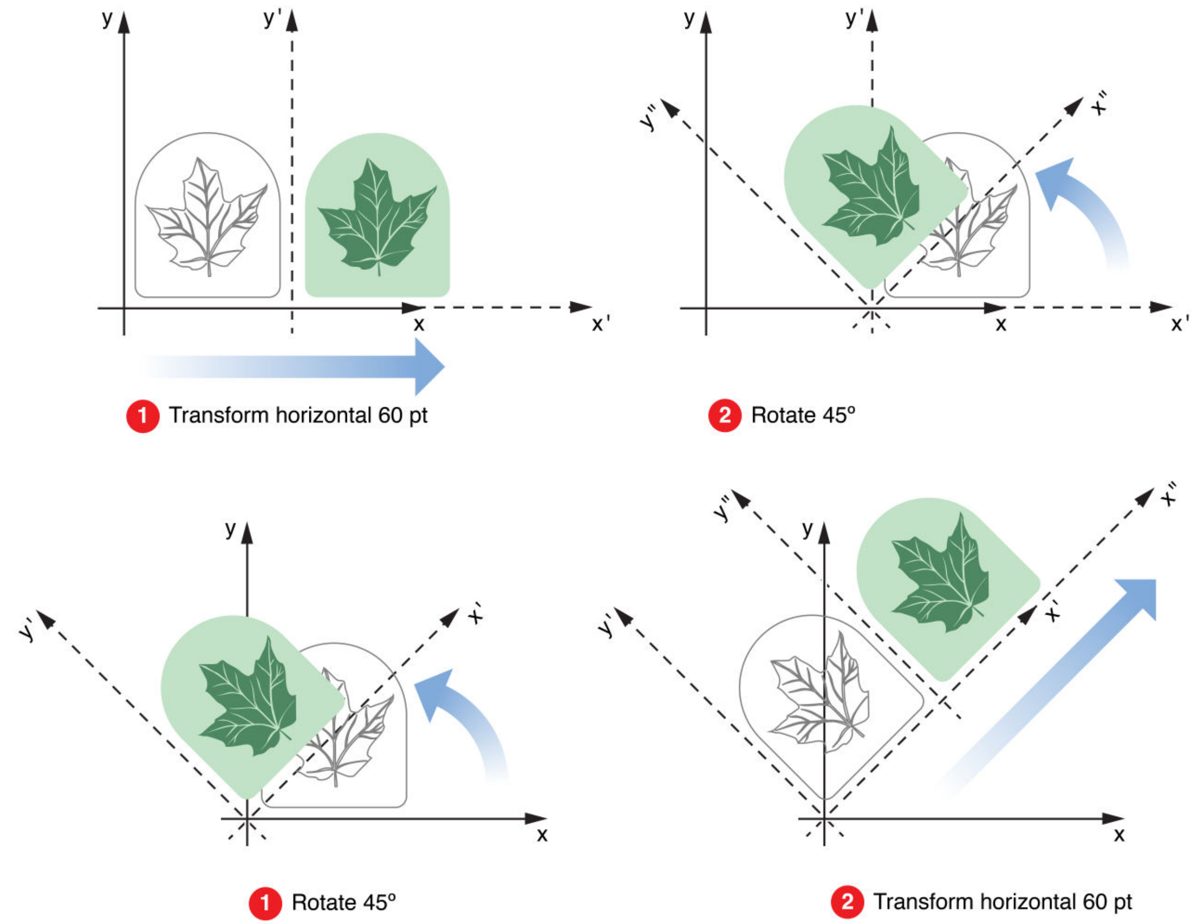
```
#version 330 core
layout (location = 0) in vec3 position;
layout (location = 1) in vec3 color;
layout (location = 2) in vec2 texCoord;

out vec3 ourColor;
out vec2 TexCoord;

uniform mat4 transform;

void main()
{
    gl_Position = transform * vec4(position, 1.0f);
    ourColor = color;
    TexCoord = vec2(texCoord.x, 1.0 - texCoord.y);
```

Поєднання трансформацій

$$M_{\text{translate}} \cdot M_{\text{scale}} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 1 \\ 0 & 2 & 0 & 2 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 0 & 0 & 1 \\ 0 & 2 & 0 & 2 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} 2x + 1 \\ 2y + 2 \\ 2z + 3 \\ 1 \end{pmatrix}$$

Порядок трансформацій



Поєднання трансформацій

```
TransformedVector = TranslationMatrix * RotationMatrix * ScaleMatrix * OriginalVector;
```

$3 \leftarrow 2 \leftarrow 1$

```
glm:: mat4 IdMatrix;
IdMatrix =glm:: translate(IdMatrix, glm::vec4(0.5f, -0.5f, 1.0f, 1.0f));
IdMatrix=glm:: rotate(IdMatrix, 90.0, glm::vec3(0.0f, 0.0f, 1.0f));
IdMatrix=glm:: scale(IdMatrix, glm::vec4(0.5f, 0.5f, 0.5f, 1.0f))

glm:: vec4 MyVector (0.5f, 0.5f, 0.0f, 1.0f);
glm:: vec4 TransVector=IdMatrix*MyVector;
```

Корисні посилання:

1. Уроки OpenGL Game Institute: http://gameinstitute.ru/uroki-opengl/

2. Математика для розробників ігор: https://agulev.com/matematika-dlya-razrabotchikov-igr-chast-2-podborka-statej/

3. Матрицы: http://www.opengl-tutorial.org/beginners-tutorials/tutorial-3-matrices/


Вот і всьо
Амінь