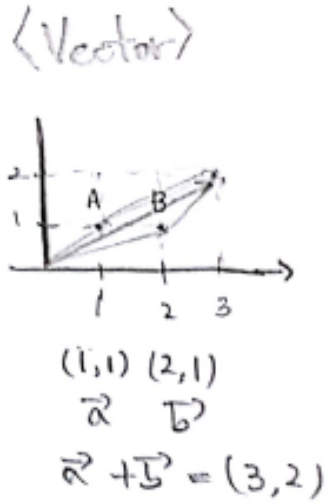
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

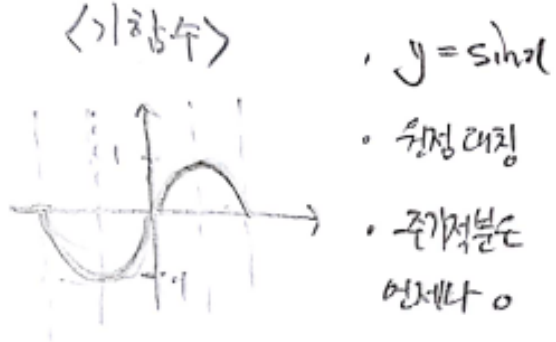
강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

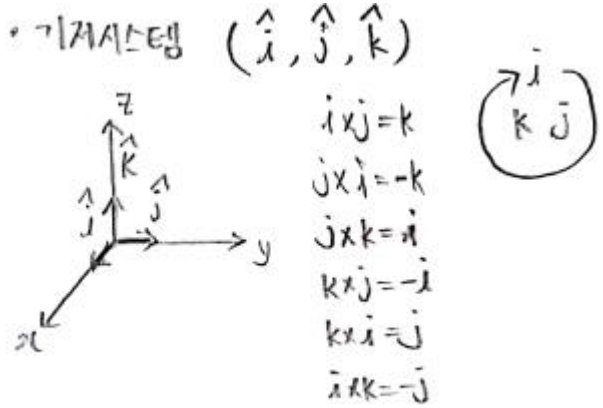
목차

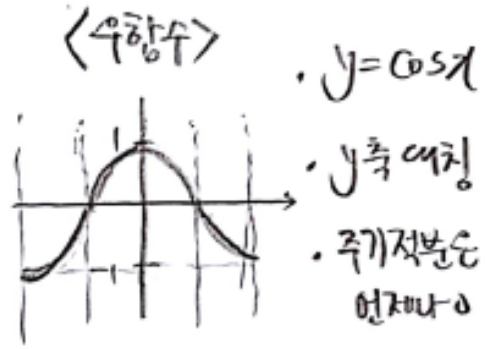
6. 수학

6. 수학 – vector 성질, 우함수와 기함수 특성









6. 수학 – vector 내적과 외적

〈내적 (Inner Product)〉

$\vec{a} \cdot \vec{b}$ $\langle \vec{a}, \vec{b} \rangle$

$\vec{a} = (a_x, a_y, a_z)$

$\vec{b} = (b_x, b_y, b_z)$

$\vec{a} \cdot \vec{b} = a_x b_x + a_y b_y + a_z b_z$

$= \|\vec{a}\| \|\vec{b}\| \cos\theta$

$= \dfrac{\langle \vec{a}, \vec{b} \rangle}{\|\vec{b}\|^2} \vec{b}$

- 성질 $\vec{a} \cdot \vec{b} = 0 \Rightarrow$ 직교
- 각도를 몰라도 두 벡터만 알면 값을 구할 수 있다.

〈외적 (Outer Product)〉

$\vec{a} \times \vec{b} \Rightarrow \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix}$

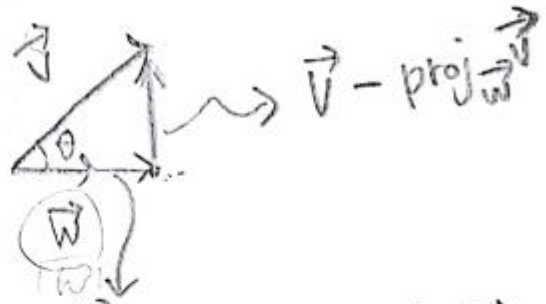
$= (a_y b_z - a_z b_y)\hat{i}$

$-(a_x b_z - a_z b_x)\hat{j}$

$(a_x b_y - a_y b_x)\hat{k}$

6. 수학 – Orthogonal Projection & Gram-Schmidt Orthonormalization

〈Orthogonal Projection〉



① $\|\vec{v}\|\cos\theta = \text{proj}_{\vec{w}}\vec{v}$

② $\vec{v}\cdot\vec{w} = \|\vec{w}\|\|\vec{v}\|\cos\theta$

③ $\dfrac{\vec{v}\cdot\vec{w}}{\|\vec{w}\|} = \|\vec{v}\|\cos\theta$ (스칼라)

④ 성분 $\Rightarrow \dfrac{\vec{v}\cdot\vec{w}}{\|\vec{w}\|}\cdot\dfrac{\vec{w}}{\|\vec{w}\|} = \dfrac{\langle\vec{v},\vec{w}\rangle}{\|\vec{w}\|^2}\cdot\vec{w}$ (단위벡터)

〈Gram-Schmidt orthonormalization〉

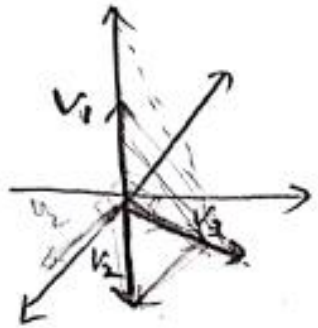
$u_1 = v_1$

$u_2 = v_2 - \text{proj}_{\vec{u}_1}\vec{v}_2$

$u_3 = v_3 - \text{proj}_{\vec{u}_1}\vec{v}_3 - \text{proj}_{\vec{u}_2}\vec{v}_3$

$\vdots$

$u_k = v_k - \sum_{j=1}^{k-1} \text{proj}_{\vec{u}_j}\vec{v}_k$



6. 수학 – vector 프로그래밍 1

< vector3d.c >

```c
#include "vector_3d.h"
#include <stdio.h>

int main(void)
{
    vec3 A = {3, 2, 1};
    vec3 B = {1, 1, 1};
    vec3 X = {1, 0, 0};
    vec3 Y = {0, 1, 0};
    vec3 v[3] = {{0, 4, 0}, {2, 2, 1}, {1, 1, 1}};
    vec3 w[3] = {};
    vec3 R = {0, 0, 0,
              vec3_add, vec3_sub, vec3_scale,
              vec3_dot, vec3_cross, print_vec3,
              gramschmidt_normalization};

    R.add(A, B, &R);
    R.print(R);

    R.sub(A, B, &R);
    R.print(R);

    R.scale(3, R, &R);
    R.print(R);

    printf("A dot B = %f\n", R.dot(A, B));

    R.cross(X, Y, &R);
    R.print(R);

    R.gramschmidt(v, w, R);

    return 0;
}
```

< vector3d.h >

```c
#ifndef __VECTOR_3D_H__
#define __VECTOR_3D_H__

#include <stdio.h>
#include <math.h>

typedef struct vector3d vec3;

struct vector3d
{
    float x;
    float y;
    float z;

    void (* add)(vec3, vec3, vec3 *);
    void (* sub)(vec3, vec3, vec3 *);
    void (* scale)(float, vec3, vec3 *);
    float (* dot)(vec3, vec3);
    void (* cross)(vec3, vec3, vec3 *);
    void (* print)(vec3);

    void (* gramschmidt)(vec3 *, vec3 *, vec3);
};

void vec3_add(vec3 a, vec3 b, vec3 *r)
{
    r->x = a.x + b.x;
    r->y = a.y + b.y;
    r->z = a.z + b.z;
}

void vec3_sub(vec3 a, vec3 b, vec3 *r)
{
    r->x = a.x - b.x;
    r->y = a.y - b.y;
    r->z = a.z - b.z;
}

void vec3_scale(float factor, vec3 a, vec3 *r)
{
    r->x = a.x * factor;
    r->y = a.y * factor;
    r->z = a.z * factor;
}

float vec3_dot(vec3 a, vec3 b)
{
    return a.x * b.x + a.y * b.y + a.z * b.z;
}

void vec3_cross(vec3 a, vec3 b, vec3 *r)
{
    r->x = a.y * b.z - a.z * b.y;
    r->y = a.z * b.x - a.x * b.z;
    r->z = a.x * b.y - a.y * b.x;
}

void print_vec3(vec3 r)
{
    printf("x = %f, y = %f, z = %f\n", r.x, r.y, r.z);
}

float magnitude(vec3 v)
{
    return sqrt(v.x * v.x + v.y * v.y + v.z * v.z);
}
```

6. 수학 – vector 프로그래밍 2

< vector3d.h >

```
68 void gramschmidt_normalization(vec3 *arr, vec3 *res, vec3 r)
69 {
70     vec3 proj[3] = {};
71     float dot[3] = {0}, mag[2] = {0};
72
73     mag[0] = magnitude(arr[0]);  // arr[0] 크기
74     r.scale(1.0 / mag[0], arr[0], &res[0]);  // 단위벡터 W0
75     printf("W0 :");
76     r.print(res[0]);
77
78     mag[0] = magnitude(res[0]);    //  W0의 크기
79     dot[0] = r.dot(arr[1], res[0]);  // <W0, V1> 내적
80     r.scale(dot[0] * (1.0 / pow(mag[0], 2.0)), res[0], &proj[0]);
81     r.sub(arr[1], proj[0], &res[1]);
82     printf("W1 :");
83     r.print(res[1]);
84
85     mag[1] = magnitude(res[1]);
86     dot[1] = r.dot(arr[2], res[0]);  // <W0, V2> 내적
87     dot[2] = r.dot(arr[2], res[1]); // <W1, V2> 내적
88     r.scale(dot[1] * (1.0 / pow(mag[0], 2.0)), res[0], &proj[1]);
89     r.scale(dot[2] * (1.0 / pow(mag[1], 2.0)), res[1], &proj[2]);
90     r.sub(arr[2], proj[1], &res[2]);
91     r.sub(res[2], proj[2], &res[2]);
92     printf("W2 :");
93     r.print(res[2]);
94
95
96 }
97
98 #endif
```

< result >

```
hyunwoopark@hyunwoopark-P65-P67SG:~/Homework/sanghoonlee/lec/math$ ./a.out
x = 4.000000, y = 3.000000, z = 2.000000
x = 2.000000, y = 1.000000, z = 0.000000
x = 6.000000, y = 3.000000, z = 0.000000
A dot B = 6.000000
x = 0.000000, y = 0.000000, z = 1.000000
W0 :x = 0.000000, y = 1.000000, z = 0.000000
W1 :x = 2.000000, y = 0.000000, z = 1.000000
W2 :x = -0.200000, y = 0.000000, z = 0.400000
```