# Tache 6 Partie 1

Code source de la fonction pour le calcul de la distance point segment :

double distance\_point\_segment(Point P1, Segment S1)

{

if ((S1.A.x == S1.B.x) && (S1.A.y == S1.B.y))

{

return distance(P1, S1.A);

}

else

{

double λ;

Vecteur AP;

Vecteur AB;

AP = creer\_vecteur(S1.A, P1);

AB = creer\_vecteur(S1.A, S1.B);

λ = produit\_scalaire(AP, AB)/produit\_scalaire(AB, AB);

if (λ < 0)

{

return distance(P1, S1.A);

}

if (λ > 1)

{

return distance(P1, S1.B);

}

else

{

Point Q;

Point q;

q = set\_point(S1.B.x-S1.A.x, S1.B.y-S1.A.y);

q = produit(q, λ);

Q = add\_point(S1.A, q);

return distance(Q, P1);

}

}

}

Avec profil de Segment le suivant :

typedef struct Segment\_

{

Point A, B;

} Segment;

Code source du programme test :

#include <stdint.h>

#include <string.h>

#include<stdlib.h>

#include <stdio.h>

#include "geom2d.h"

int main(int argc, char \*\*argv)

{

//Test no 8 (Tache 6 partie 1)

printf("Test distance point-segments Tache 6\n");

Segment S;

//Les coordonnes pour les tests marques ci dessous

S.A = set\_point(0.0, 0.0);

S.B = set\_point(0.0, 2.0);

Point P1 = set\_point(1.00, 1.0);

double distance\_seg = distance\_point\_segment(P1, S);

printf("%f distance point-segment\n\n", distance\_seg);

}

Jeu de test :

Test no 1:

A: (0,0)

B: (3,0)

P: (-1, 0)

Résultat: 1

Test no 2:

A: (1,1)

B: (1,4)

P: (2,2)

Résultat: 1

Test no 3:

A: (0,4)

B: (0,0)

P: (0,5)

Résultat : 1

Test no 4 :

A : (0,0)

B : (0,2)

P : (1,1)

Résultat : 1