Tache6

Pascal Isak && Weber Loïc

Table Of Content

fonction distance_point_segment

geometrice.c

```
1  double distance_point_segment(Point P, Point A, Point B){
2      if (A.x == B.x && A.y == B.y){
3          return distance_point(A,P);
4      }
5      // On est dans le cas A =/= B :
6
7      // On commence par calculer lambda
8      double lambda = produit_scalaire(couple_point_to_vecteur(A,P),couple_point_to_vec
9      /produit_scalaire(couple_point_to_vecteur(A,B),couple_point_to_vecteur(A,B));
10
11     // Cas lambda < 0 :
12     if (lambda < 0){
13         return distance_point(A,P);
14     }
15
16     // Cas lambda > 1 :
17     if (lambda > 1){
18         return distance_point(B,P);
19     }
20
21     // Dernier cas 0 <= lambda <= 1 :
22     // On calcule le point Q :
23     Point Q = addition_point(A, produit_point(lambda, addition_point(B, negation_poin
24     //Point Q = nouveau_point(A.x + lambda * (B.x - A.x), A.y * lambda * (B.y - A.y))
25     return distance_point(Q,P);
26 }
```

test_simplification_contour

test_simplification_contour

```
1
```

```c
void jeu_de_test(){
    printf("Test simplification contour\n");
    printf("Test 1/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(0,0),
    nouveau_point(0,0), nouveau_point(0,0)) == 0); //P = A = B = (0,0)

    printf("Test 2/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(0.5,0),
    nouveau_point(0,0), nouveau_point(1,0)) == 0); // P = milieu(A,B)

    printf("Test 3/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(3.5,3.5),
    nouveau_point(5,2), nouveau_point(2,5)) == 0); // P = milieu(A,B)

    printf("Test 4/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(5,8),
    nouveau_point(5,8), nouveau_point(5,0)) == 0); // P = A

    printf("Test 5/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(5,8),
    nouveau_point(5,0), nouveau_point(5,8)) == 0); // P = B

    printf("Test 6/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(5,11),
    nouveau_point(5,0), nouveau_point(5,10)) == 1); // P ∈ droite(A,B)

    printf("Test 7/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(5,-1),
    nouveau_point(5,0), nouveau_point(5,10)) == 1); // P ∈ droite(A,B)

    printf("Test 8/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(7,8),
    nouveau_point(5,0), nouveau_point(5,10)) == 2); // 0 < λ < 1

    printf("Test 9/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(8,14),
    nouveau_point(5,5), nouveau_point(5,10)) == 5); // 1 < λ

    printf("Test 10/10 : ");
    afficher_resultat_test(distance_point_segment(nouveau_point(1,2),
    nouveau_point(5,5), nouveau_point(5,10)) == 5); // λ < 0
}
int main(int argc, char** argv){
    jeu_de_test();

    // Test en ligne de commande :
    if (argc != 7){
        printf("Usage: ./test_simplification_contour P.x P.y A.x A.y B.x B.y\n");
        return 1;
    }
    Point P = nouveau_point(atof(argv[1]),atof(argv[2]));
    Point A = nouveau_point(atof(argv[3]),atof(argv[4]));
    Point B = nouveau_point(atof(argv[5]),atof(argv[6]));

    double resultat = distance_point_segment(P, A, B);
    printf("distance_point_segment = %f \n",resultat);
```

```
58      return 0;
59  }
```