

Structure de données II : Rapport de l'étape préliminaire du projet

Groupe 5 :
HUYLENBROECK Florent
DACHY Corentin

Année Académique 2018-2019
Bachelier en Sciences Informatiques

Faculté des Sciences, Université de Mons

Algorithm 1 belongsToScene

Entrées : BSP : Partition de recherche binaire
On assume que chaque noeud contient l'équation de la droite qu'il décrit et (facultatif) le segment qui lui est confondu, et chaque feuille contient un segment, décrit par une paire de points $S(S.x, S.y)$ et $S'(S'.x, S'.y)$.
 A : Point correspondant à une extrémité du segment.
Ce point a pour coordonnées $(A.x, A.y)$.
 B : Point correspondant à l'autre extrémité du segment.
Ce point a pour coordonnées $(B.x, B.y)$.

Sorties : Booléen, vrai si le segment appartient à la scene, faux sinon.

Effets : /

- 1: **procedure** BELONGSTOSCENE(BSP, A, B)
- 2: $d \leftarrow$ COEFICIENTANGULAIRE(A, B)
- 3: **retourner** RECHERCHER(BSP, A, B, d)
- 4: **end procedure**

Algorithm 2 rechercher

Entrées : BSP : Partition de recherche binaire
On assume que chaque noeud contient l'équation de la droite qu'il décrit et (facultatif) le segment qui lui est confondu, et chaque feuille contient un segment, décrit par une paire de points $S(S.x, S.y)$ et $S'(S'.x, S'.y)$.
 P : Point, première extrémité du segment recherché dans le BSP
 B : Point, deuxième extrémité du segment recherché dans le BSP
 d : Entier (ou valeur sentinelle $+\infty$), coefficient angulaire du segment recherché.
Sorties : Booléen, vrai si le segment PB appartient au BSP
Effets : /

```
1: procedure RECHERCHER( $BSP, P, B, d$ )
2:    $S[] \leftarrow$  nouvelle liste vide
3:   LOCALISER( $BSP, P, S[]$ )
4:   REDUIRE( $S[], d$ )
5:   if  $S[]$  vide then
6:     retourner False
7:   else
8:      $segment \leftarrow S[0]$ 
9:      $P' \leftarrow$  extrémité de  $segment$  qui n'est pas  $P$ 
10:    if  $P'$  pas sur un bord then
11:      if  $P' \neq B$  then
12:        retourner False
13:      else
14:        retourner True
15:      end if
16:    else
17:      retourner RECHERCHER( $BSP, P', B, d$ )
18:    end if
19:  end if
20: end procedure
```

Algorithm 3 rechercher

Entrées : $root$: Racine de la sous-partition de recherche binaire que l'on doit chercher

P : Point que l'on recherche.

$return[]$: Liste contenant les résultats de la recherche.

Sorties : /

Effets : La liste $return[]$ est mise à jour.

```
1: procedure LOCALISER( $root, P, return[]$ )
2:   if  $root$  est une feuille then
3:     if  $P \in root$  then
4:       ajouter  $root$  dans  $return[]$ 
5:     end if
6:   else
7:      $res \leftarrow$  résultat de la résolution de l'équation de la droite décrite par  $root$  avec  $P.x$  et  $P.y$ 
8:     if  $res \geq 0$  then
9:       LOCALISER( $root+, P, return[]$ )
10:    else if  $res \leq 0$  then
11:      LOCALISER( $root-, P, return[]$ )
12:    else
13:      if  $P \in root$  then
14:        ajouter  $root$  dans  $return[]$ 
15:      end if
16:      LOCALISER( $root+, P, return[]$ )
17:      LOCALISER( $root-, P, return[]$ )
18:    end if
19:  end if
20: end procedure
```

Algorithm 4 rechercher

Entrées : $S[]$: Ensemble de segments à réduire.

d : Entier (ou valeur sentinelle $+\infty$), coefficient angulaire du segment recherché.

Sorties : /

Effets : La liste $S[]$ ne contient plus qu'un ou zéro segments, celui qui à la direction d .

```
1: procedure REDUIRE( $S[], d$ )
2:   for all elements  $s$  de  $S[]$  do
3:      $sd \leftarrow$  COEFFICIENTANGULAIRE( $s.A, s.B$ )
4:     if  $sd \neq d$  then
5:       retirer  $s$  de  $S[]$ 
6:     end if
7:   end for
8: end procedure
```

Algorithm 5 rechercher

Entrées : A : Racine de la sous-partition de recherche binaire que l'on doit chercher
 B : Point que l'on recherche.

Sorties : Le coefficient angulaire de la droite passant par A et B .
 Une valeur sentinelle $+\infty$ sera retournée si la pente est verticale.

Effets : /

```
1: procedure COEFICIENTANGULAIRE( $A, B$ )
2:   if  $A.x - B.x == 0$  then
3:     retourner  $+\infty$ 
4:   else
5:     retourner  $(A.y - B.y)/(A.x - B.x)$ 
6:   end if
7: end procedure
```
