**PROJET INTEGRATEUR**

**Groupe 5**

****

|  |  |
| --- | --- |
|  |  |
|  |  |
|  | |  |
|  | |  |

**Voiture Autonome**

**X2027**

PATHFINDING

Qu’est-ce qu’un pathfinding ?

Pathfinding est un terme anglais qui peut se traduire en « **Trouver son chemin** ».

Notre véhicule devra donc être capable à partir d’un point de départ A de se déplacer jusqu’au point B en utilisant le chemin le plus court tout en suivant la route.

/!\ Notre véhicule ne peut que tourner à droite ou à gauche dans une intersection, il ne peut aller tout droit.

Découverte de la Mapp

Tout d’abord pour réaliser notre pathfinding il suffit de faire découvrir le chemin de notre Mapp ou de le faire suivre pour une entité donnée.

Voici la Mapp fourni pour ce projet ci :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **5** |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |
| **3** |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  |
| **1** |  |  |  |  |  |  |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** |

*Figure 1 : Mapp à la base*

Nous avons maintenant notre Mapp, il va falloir placer des repères sous formes de numéros correspondants à des coordonnées.

La voiture va pouvoir les emprunter dans les différentes intersections présente dans cette Mapp.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **5** | **4** |  | **1** |  | **13** | **14** |
| **5** |  | **3** |  | **2** |  |  |  |
| **4** |  |  |  |  |  |  |  |
| **3** | **7** |  |  |  |  | **12** | **15** |
| **2** |  | **10**  **6** |  |  |  | **11** |  |
| **1** | **8** | **9** |  |  |  |  | **16** |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** |  |

*Figure 2 : Mapp détaillé*

Maintenant que toutes les intersections sont énumérées de 0 (point de départ) à 16 nous allons pouvoir passer au comportement du véhicule.

Cependant quelques points sont à souligner :

* L’ordre des intersections importe peu, cet ordre tel qu’il est pour permettre de repérer
* Le point 0 n’existe plus juste après le départ véhicule, il est présent pour représenter le point de départ

# Comportement du véhicule

On considère que chaque intersection est un évènement. Le véhicule s’arrêtera donc à chaque intersection numérotée et prendra une décision programmée dans l’Arduino.

Nous allons donc réaliser un tableau qui va permettre de répertorier toutes les destinations possibles du véhicule.

* Points et destination(s) possible

|  |  |  |  |
| --- | --- | --- | --- |
| Points | Destination 1 | Destination 2 | Destination 3 |
| 1 | 2 | 4 |  |
| 2 | 3 |  |  |
| 3 | 4 | 6 |  |
| 4 | 5 | 1 |  |
| 5 | 6 |  |  |
| 6 | 3 | 7 |  |
| 7 | 8 |  |  |
| 8 | 9 |  |  |
| 9 | 10 | 16 |  |
| 10 | 11 |  |  |
| 11 | 12 |  |  |
| 12 | 13 | 15 |  |
| 13 | 14 | 1 |  |
| 14 | 13 | 15 |  |
| 15 | 16 | 12 |  |
| 16 | 9 | 15 |  |

Figure 3 : Répertoire de position des intersections

À la suite de cela il sera possible de faire un algorithme de déplacement du robot d’un point connu à un autre. Cependant, il est difficile de dire comment circuler entre deux points. Pour pouvoir remédier à ce problème, on créer un tableau qui va nous permettre de gérer facilement la notion de points et d’indices.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 00 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 1 |  | 1 |  | 1 |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  | 1 |  | 1 |  |  |  |  |  |  |  |  |  |  |
| 4 | 1 |  |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  | 1 |  |  |  | 1 |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  | 1 |
| 10 |  |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |  | 1 |  |  |  | 1 |
| 12 |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  | 1 |  |
| 13 | 1 |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  | 1 |  |
| 15 |  |  |  |  |  |  |  |  |  |  |  | 1 |  |  |  | 1 |
| 16 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  | 1 |  |

Figure 4 : Tableau de circulation dans la Mapp

Grâce aux indices de tableau, on peut voir facilement si un déplacement est possible ou non. Malheureusement à ce stade, on ne peut pas indiquer au robot quelle manœuvre il doit effectuer pour atteindre le point suivant. Il ne sait pas s’il doit tourner à gauche, à droite ou avancer tout droit.

* Points et leurs cordonnées dans le repère

Pour réaliser ceci, Nous allons définir un plan avec sa propre échelle et utiliser des vecteurs. Dans ce cadre, on va créer un nouveau tableau qui va contenir les coordonnées de tous les points.

Donc, voici le tableau qui contient les cordonnées : (L’origine de notre repère est 8)

Considérons 3cm comme unité de mesure on obtient :

|  |  |  |
| --- | --- | --- |
| Points | Abscisses | Ordonnées͢ |
| 1 | 9 | 8 |
| 2 | 9 | 6 |
| 3 | 3 | 6 |
| 4 | 3 | 8 |
| 5 | 0 | 8 |
| 6 | 0 | 6 |
| 7 | 0 | 3 |
| 8 | 0 | 0 |
| 9 | 3 | 0 |
| 10 | 3 | 2 |
| 11 | 12 | 2 |
| 12 | 12 | 3 |
| 13 | 12 | 8 |
| 14 | 14 | 8 |
| 15 | 14 | 3 |
| 16 | 14 | 0 |

Figure 5 : Vecteurs de déplacement

Une fois que nous avons réalisé ce tableau, suffit simplement de faire le calcul des vecteurs d’un point à un autre pour avoir la direction du robot.