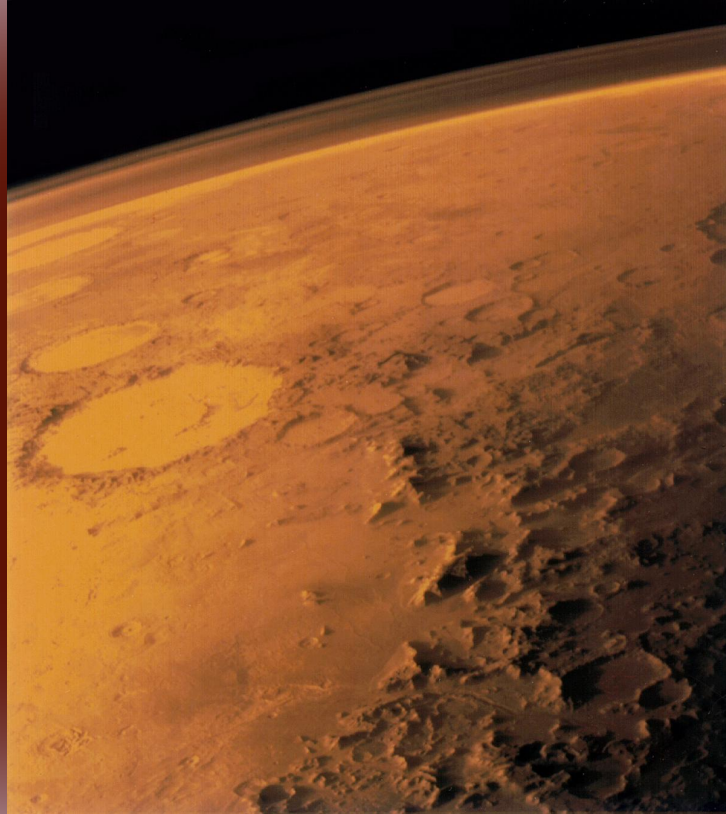


Projet conception : robot martien.



Contexte du projet.

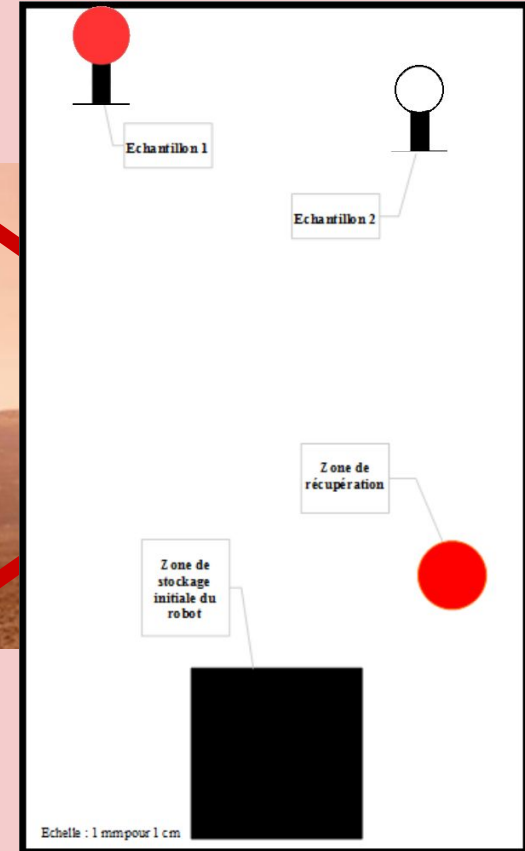
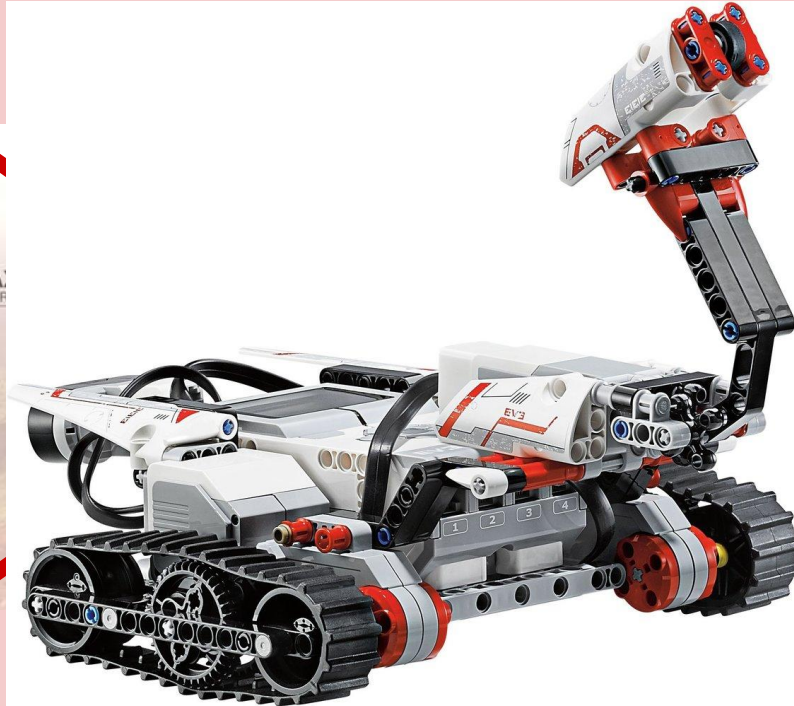
Contexte du projet.



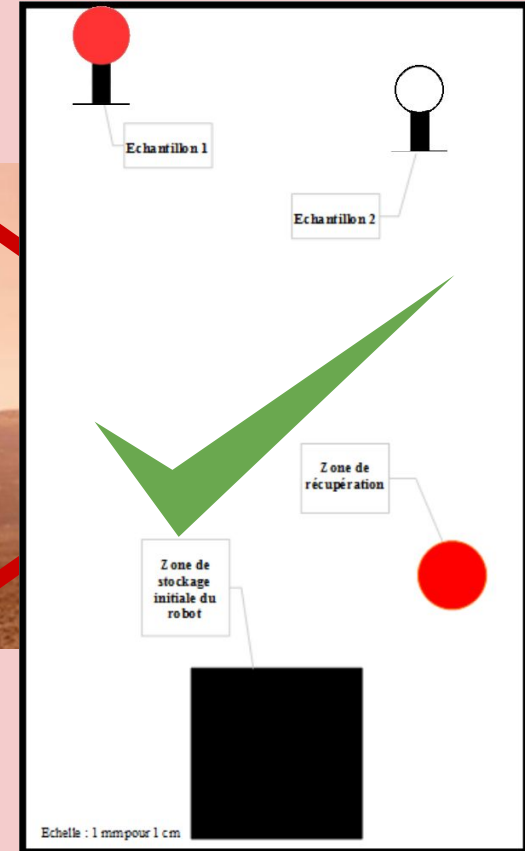
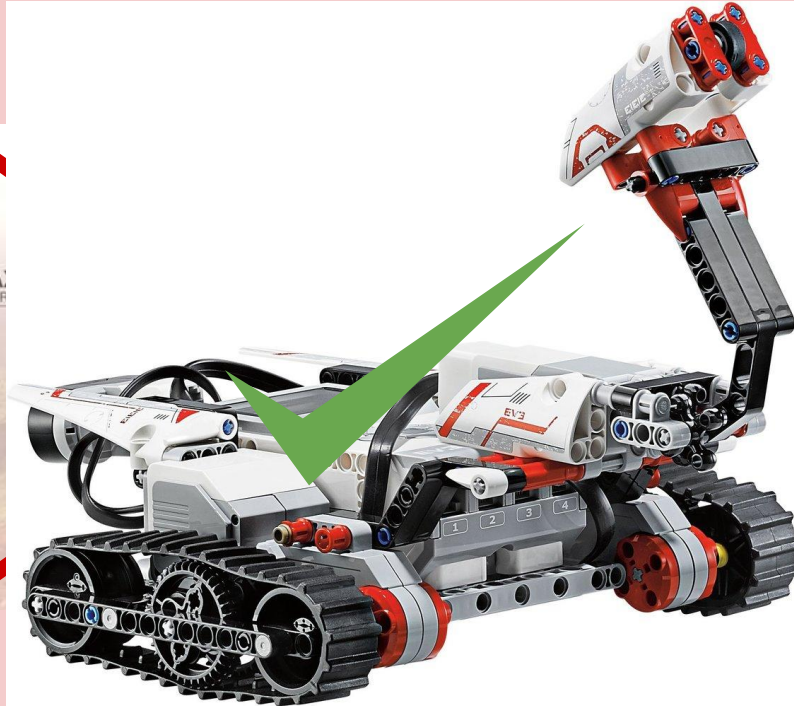
Contexte du projet.



Contexte du projet.



Contexte du projet.



Démarche suivie.

Démarche suivie.



inspiration d'un modèle physique : le track3r

manuel
track3r

Démarche suivie.

- ☐ inspiration d'un modèle physique : le track3r
- ☐ modification de la structure *on the fly*

manuel
track3r

observer des
défauts

adapter le design

Démarche suivie.

- ☐ inspiration d'un modèle physique : le track3r
- ☐ modification de la structure *on the fly*
- ☐ familiarisation avec la bibliothèque leJOS

API leJOS →
<https://lejos.sourceforge.io/ev3/docs/>

manuel
track3r

observer des
défauts

adapter le design

Démarche suivie.

- ☐ inspiration d'un modèle physique : le track3r
- ☐ modification de la structure *on the fly*
- ☐ familiarisation avec la bibliothèque leJOS
- ☐ développement en parallèle d'algorithmes indépendants

API leJOS →
<https://lejos.sourceforge.io/ev3/docs/>

manuel
track3r

Rover.*explore*

Rover.*harvest*

observer des
défauts

adapter le design

Démarche suivie.

- ☐ inspiration d'un modèle physique : le track3r
- ☐ modification de la structure *on the fly*
- ☐ familiarisation avec la bibliothèque leJOS
- ☐ développement en parallèle d'algorithmes indépendants
- ☐ tests unitaires des capteurs et des moteurs
- ☐ tests unitaires des algorithmes
- ☐ tests grandeur nature

API leJOS →
<https://lejos.sourceforge.io/ev3/docs/>

manuel
track3r

Rover.*explore*

Rover.*harvest*

observer des
défauts

adapter le design

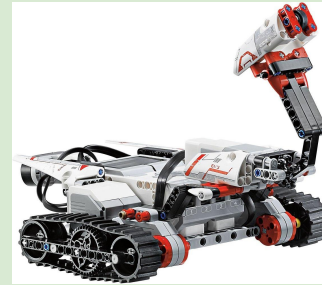
tests unitaires
et complets

Des résultats.

Des résultats.



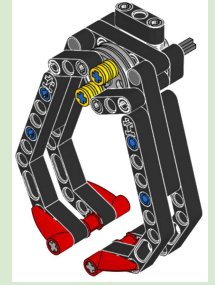
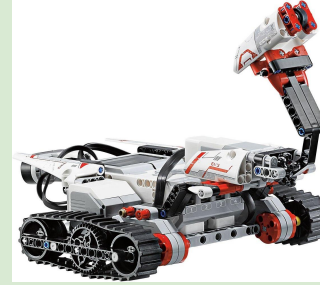
inspiration d'un modèle physique : le track3r



Des résultats.

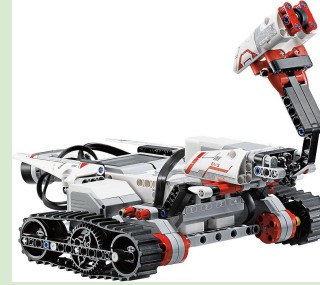
☒ inspiration d'un modèle physique : le track3r

☒ modification de la structure *on the fly*

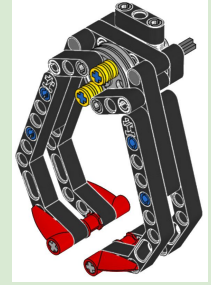


Des résultats.

- ✓ inspiration d'un modèle physique : le track3r
- ✓ modification de la structure *on the fly*
- ✓ familiarisation avec la bibliothèque leJOS

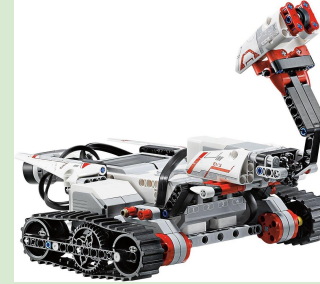


x2

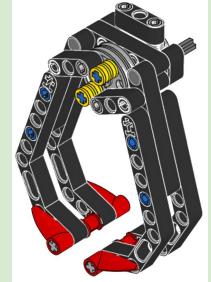


Des résultats.

- ✓ inspiration d'un modèle physique : le track3r
- ✓ modification de la structure *on the fly*
- ✓ familiarisation avec la bibliothèque leJOS
- ✓ développement en parallèle d'algorithmes indépendants



x2



```
1 tant que (waypoint_index < path.length-1) faire
2   rotation vers le point de passage path[waypoint_index+1] non bloquante
3   tant que (rotation en cours) faire
4     distance <- mesure ultrasonique
5     // calcule des coordonnées de l'objet avec la position du rover et la distance à l'objet
6     objet <- object from distance
7     si (objet dans la zone) et (objet pas encore vu) et (objet pas dans la zone de récup) alors
8       sortir de explore pour appeler harvest
9     fin si
10  fin tant que
11
12  translation vers le point de passage path[waypoint_index+1] non bloquante
13  tant que (translation en cours) faire
14    distance <- mesure ultrasonique
15    objet <- object from distance
16    si (objet dans la zone) et (objet pas encore vu) et (objet pas dans la zone de récup) alors
17      sortir de explore pour appeler harvest
18    fin si
19  fin tant que
20  waypoint_index <- waypoint_index + 1
21 fin tant que
```

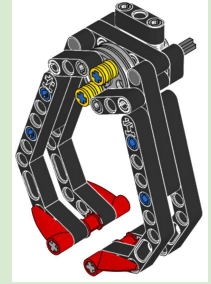
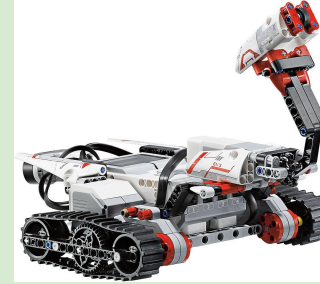
explore

```
22 // fin de
23 fin si
24 fin tant que
25
26 // on est en face de l'échantillon
27 noter la position de l'échantillon
28 // pour ne plus le récolter si le rover le rencontre à nouveau
29 ouvrir la pince
30 avancer de (distance - distance entre le centre de rotation et la pince)
31 // pour aligner la pince et l'échantillon
32 fermer la pince
33 rotation pour rapprocher la zone de récupération
34 translation pour approcher la zone de récupération
35 ouvrir la pince // pour lâcher l'échantillon
36 reculer
37 fermer la pince
```

harvest

Des résultats.

- ✓ inspiration d'un modèle physique : le track3r
- ✓ modification de la structure *on the fly*
- ✓ familiarisation avec la bibliothèque leJOS
- ✓ développement en parallèle d'algorithmes indépendants
- ✗ tests unitaires des capteurs et des moteurs



```
1 tant que (waypoint_index < path.length-1) faire
2   rotation vers le point de passage path[waypoint_index+1] non bloquante
3   tant que (rotation en cours) faire
4     distance <- mesure ultrasonique
5     // calcule des coordonnées de l'objet avec la position du rover et la distance à l'objet
6     objet <- object from distance
7     si (objet dans la zone) et (objet pas encore vu) et (objet pas dans la zone de récup) alors
8       sortir de explore pour appeler harvest
9     fin si
10  fin tant que
11
12  translation vers le point de passage path[waypoint_index+1] non bloquante
13  tant que (translation en cours) faire
14    distance <- mesure ultrasonique
15    objet <- object from distance
16    si (objet dans la zone) et (objet pas encore vu) et (objet pas dans la zone de récup) alors
17      sortir de explore pour appeler harvest
18    fin si
19  fin tant que
20  waypoint_index <- waypoint_index + 1
21 fin tant que
```

explore

```
22 fin si
23 fin tant que
24
25 // on est en face de l'échantillon
26 noter la position de l'échantillon
27 // pour ne plus le récolter si le rover le rencontre à nouveau
28 ouvrir la pince
29 avancer de (distance - distance entre le centre de rotation et la pince)
30 // pour aligner la pince et l'échantillon
31 fermer la pince
32 rotation pour aligner la pince et la zone de récupération
33 translation pour approcher la zone de récupération
34 ouvrir la pince // pour lâcher l'échantillon
35 reculer
36 fermer la pince
```

harvest

Constructeurs

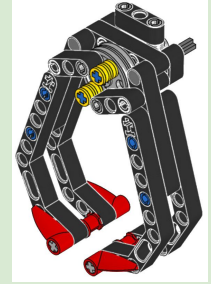
Navigator

travel

rotate

Des résultats.

- ☒ inspiration d'un modèle physique : le track3r
- ☒ modification de la structure *on the fly*
- ☒ familiarisation avec la bibliothèque leJOS
- ☒ développement en parallèle d'algorithmes indépendants
- ☐ tests unitaires des capteurs et des moteurs
- ☒ tests unitaires des algorithmes



```
1 tant que (waypoint_index < path.length-1) faire
2   rotation vers le point de passage path[waypoint_index+1] non bloquante
3   tant que (rotation en cours) faire
4     distance <- mesure ultrasonique
5     // calcule des coordonnées de l'objet avec la position du rover et la distance à l'objet
6     objet <- object from distance
7     si (objet dans la zone) et (objet pas encore vu) et (objet pas dans la zone de récup) alors
8       sortir de explore pour appeler harvest
9     fin si
10  fin tant que
11
12  translation vers le point de passage path[waypoint_index+1] non bloquante
13  tant que (translation en cours) faire
14    distance <- mesure ultrasonique
15    objet <- object from distance
16    si (objet dans la zone) et (objet pas encore vu) et (objet pas dans la zone de récup) alors
17      sortir de explore pour appeler harvest
18    fin si
19  fin tant que
20  waypoint_index <- waypoint_index + 1
21 fin tant que
```

explore

```
22 fin si
23 fin tant que
24
25 // on est en face de l'échantillon
26 noter la position de l'échantillon
27 // pour ne plus le récolter si le rover le rencontre à nouveau
28 ouvrir la pince
29 avancer de (distance - distance entre le centre de rotation et la pince)
30 // pour aligner la pince et l'échantillon
31 fermer la pince
32 rotation pour aligner la pince et la zone de récupération
33 translation pour approcher la zone de récupération
34 ouvrir la pince // pour lâcher l'échantillon
35 reculer
36 fermer la pince
```

harvest

Constructeurs

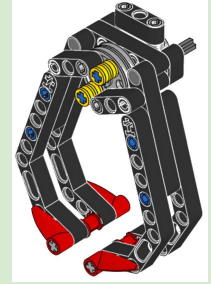
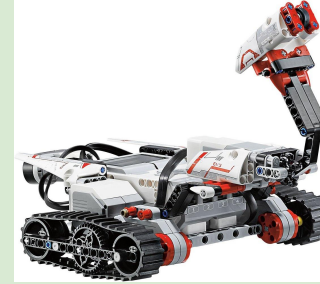
Navigator

travel

rotate

Des résultats.

- ☒ inspiration d'un modèle physique : le track3r
- ☒ modification de la structure *on the fly*
- ☒ familiarisation avec la bibliothèque leJOS
- ☒ développement en parallèle d'algorithmes indépendants
- ☐ tests unitaires des capteurs et des moteurs
- ☒ tests unitaires des algorithmes
- ☒ tests grandeur nature



```
1 tant que (waypoint_index < path.length-1) faire
2   rotation vers le point de passage path[waypoint_index+1] non bloquante
3   tant que (rotation en cours) faire
4     distance <- mesure ultrasonique
5     // calcule des coordonnées de l'objet avec la position du rover et la distance à l'objet
6     objet <- object from distance
7     si (objet dans la zone) et (objet pas encore vu) et (objet pas dans la zone de récup) alors
8       sortir de explore pour appeler harvest
9     fin si
10  fin tant que
11
12  translation vers le point de passage path[waypoint_index+1] non bloquante
13  tant que (translation en cours) faire
14    distance <- mesure ultrasonique
15    objet <- object from distance
16    si (objet dans la zone) et (objet pas encore vu) et (objet pas dans la zone de récup) alors
17      sortir de explore pour appeler harvest
18    fin si
19  fin tant que
20  waypoint_index <- waypoint_index + 1
21 fin tant que
```

explore

```
22 fin si
23 fin tant que
24
25 // on est en face de l'échantillon
26 noter la position de l'échantillon
27 // pour ne plus le récolter si le rover le rencontre à nouveau
28 ouvrir la pince
29 avancer de (distance - distance entre le centre de rotation et la pince)
30 // pour aligner la pince et l'échantillon
31 fermer la pince
32 rotation pour aligner la zone de récupération
33 translation pour approcher la zone de récupération
34 ouvrir la pince // pour lâcher l'échantillon
35 reculer
36 fermer la pince
```

harvest

Constructeurs

Navigator

travel

rotate

Conclusion et enseignements.

construire

Conclusion et enseignements.

anticiper

prévoir

construire

Conclusion et enseignements.

tests
anticiper
prévoir
construire

Conclusion et enseignements.

r é p a r t i t i o n
anticiper
proposer tests
prévoir
construire
parallélisme

Conclusion et enseignements.

r é p a r t i t i o n
anticiper
proposer tests **temps** prévoir
construire parallélisme

FIN