جامعة مولاي إسماعيل
UNIVERSITÉ MOULAY ISMAÏL

المدرسة الوطنية العليا للفنون و المهن
ÉCOLE NATIONALE SUPÉRIEURE D'ARTS ET MÉTIERS

Rapport du projet d'initiation IA

Autonomous tool-carrying robot

SWIFT

Réalisé par :
Bennis Sanae
Maghraoui Sana

Encadré par :
M. Masrour Tawfik

Remerciement

Avant tout, nous profitons de cette occasion pour exprimer nos sincères remerciements et notre profonde reconnaissance à l'Ecole Nationale Supérieure des Arts et Métiers, dont les responsables, le corps enseignant et le personnel administratif ont tout déployé pour nous donner la formation digne de cette prestigieuse institution.

Nous témoignons une ma profonde gratitude à notre encadrant et cher professeur Monsieur MASROUR Tawfik. pour ses conseils précieux et ses recommandations qu'il nous a prodigués durant toute la période du projet.

Nous voudrons également remercier nos familles pour leur soutien continuelle qui nous pousse toujours à se développer et fournir plus d'efforts.

Nous présentons nos vifs remerciements à toute personne qui a contribué de prés ou de loin à la réalisation de ce travail

Enfin, qu'il nous soit permis de remercier tout le corps professoral et administratif de l'école ainsi que toute personne ayant contribué à bien mener projet.

Introduction

Les projets d'initiation ont une grande importance dans la construction et l'instruction d'un élève ingénieur, il leur permet d'acquérir un ensemble de compétences, de se familiariser avec leur filière et d'ouvrir les yeux sur plusieurs pistes. Le projet initiation IA a pour but d'éclaircir la vision sur le domaine IA, de les ramener à apprendre et savoir comment et quoi apprendre, à découvrir ce domaine vaste et fascinant et à développer leurs propres méthodes de pensé, d'analyse, de critique et de création.

1 Chapitre 1 : introduction au projet

1.1 Mise en situation et but

De nos jours, la maintenance est au cœur de toute activité industrielle et constitue un enjeu majeur pour la productivité de l'entreprise. À cet égard, plusieurs technologies ont été développées au cours des dernières années afin de garantir la bonne gestion de la maintenance ainsi que faciliter considérablement le travail des équipes du terrain. L'un des problématiques de la maintenance consiste aux déplacements répétitifs de l'équipe pour ramener les outils nécessaires afin d'accomplir une tâche de réparation ce qui résulte une perte en terme du temps et d'effort. D'où vient l'idée de SWIFT qui a pour but de limiter ces déplacements.

SWIFT est un robot autonome porteur de charge visé pour les techniciens de maintenance. CET IGV est capable une fois qu'il reçoit l'ordre de trouver le chemin optimal pour arriver à l'atelier puis détecter l'outil convenable et l'apporter directement au technicien qui a fait la demande sans interrompre le processus de la maintenance. Tout cela à l'aide des outils de l'intelligence artificielle qui feront objet de notre étude. Cette solution est efficace pour minimiser le temps de panne donc l'arrêt de la ligne de production qui cause des surcouts lourds et des pertes pour l'entreprise.

1.2 Benchmarking

Notre projet s'intègre dans le cadre du véhicule guidé intelligent (IGV) qui est un robot mobile utilisé dans les centres de production et de distribution pour les tâches de manutention et de transport de matériaux. Ils sont équipés de capteurs, de systèmes de navigation et de logiciels de contrôle pour transporter des marchandises de manière autonome à l'intérieur d'une installation, en suivant des itinéraires prédéfinis ou en utilisant une cartographie en temps réel pour éviter les obstacles et atteindre leurs destinations.

1.2.1 Identification de point de comparaison :

- **Rapidité :** temps de réponse et de réalisation des missions
- **Efficacité opérationnelle :** taux d'utilisation des ressources, productivité…

- **Innovation** : taux de nouveaux produits lancés, investissements en R&D, nouvelles technologies adoptées.
- **Satisfaction client :** taux de réclamation, taux de fidélité, enquêtes de satisfaction
- **Coût :** coût total de possession, coûts de production, coûts de marketing

1.2.2 Identification des concurrents

Projet	**Solution**
Agilox	**Agilox est le premier et unique Véhicule à Guidage Intelligent, pour la manutention horizontale des marchandises.**

1.2.3 Conclusion et valeur ajoutée de la solution proposée

SWIFT, notre robot autonome a pour but de se déplacer au lieu du technicien en cas de panne afin de minimiser les déplacements de l'opérateur ainsi que le temps de panne donc l'arrêt de la ligne de production et économiser les surcoûts lourds et des pertes pour l'entreprise.

Apporter une nouvelle et unique solution qui vise la maintenance

Choix du chemin optimal

Facilité de communication avec le robot

1.3 Importance de la solution

Le diagramme de Gantt ci-dessous issu d'une étude des arrêts lors d'un stage à l'EMO (Les Eaux Minérales d'Oulmès) illustre l'importance de notre projet :

A : l'opérateur arrête la machine et appui sur le bouton d'alerte

B : les mécaniciens apportent leurs matériels et provient à la salle de production

C : la recherche de la machine en panne et discussions avec l'opérateur

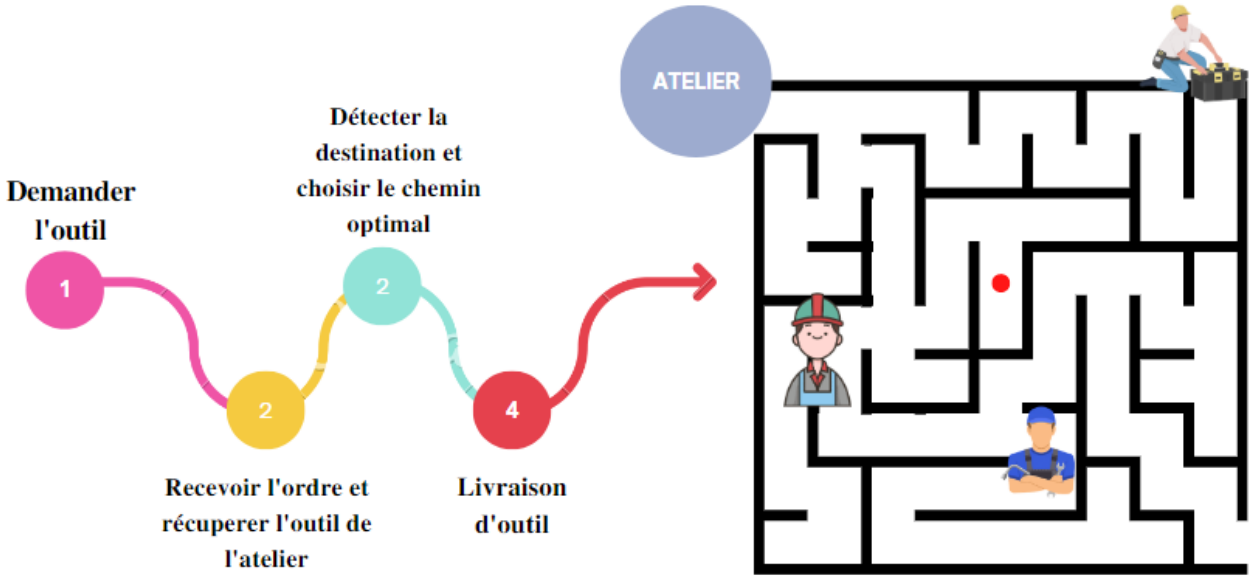
D : la réparation de la machine

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A															
B															
C															
D															

Une perte de 11 minutes avant de commencer la réparation de la panne

1.4 Fonctionnement

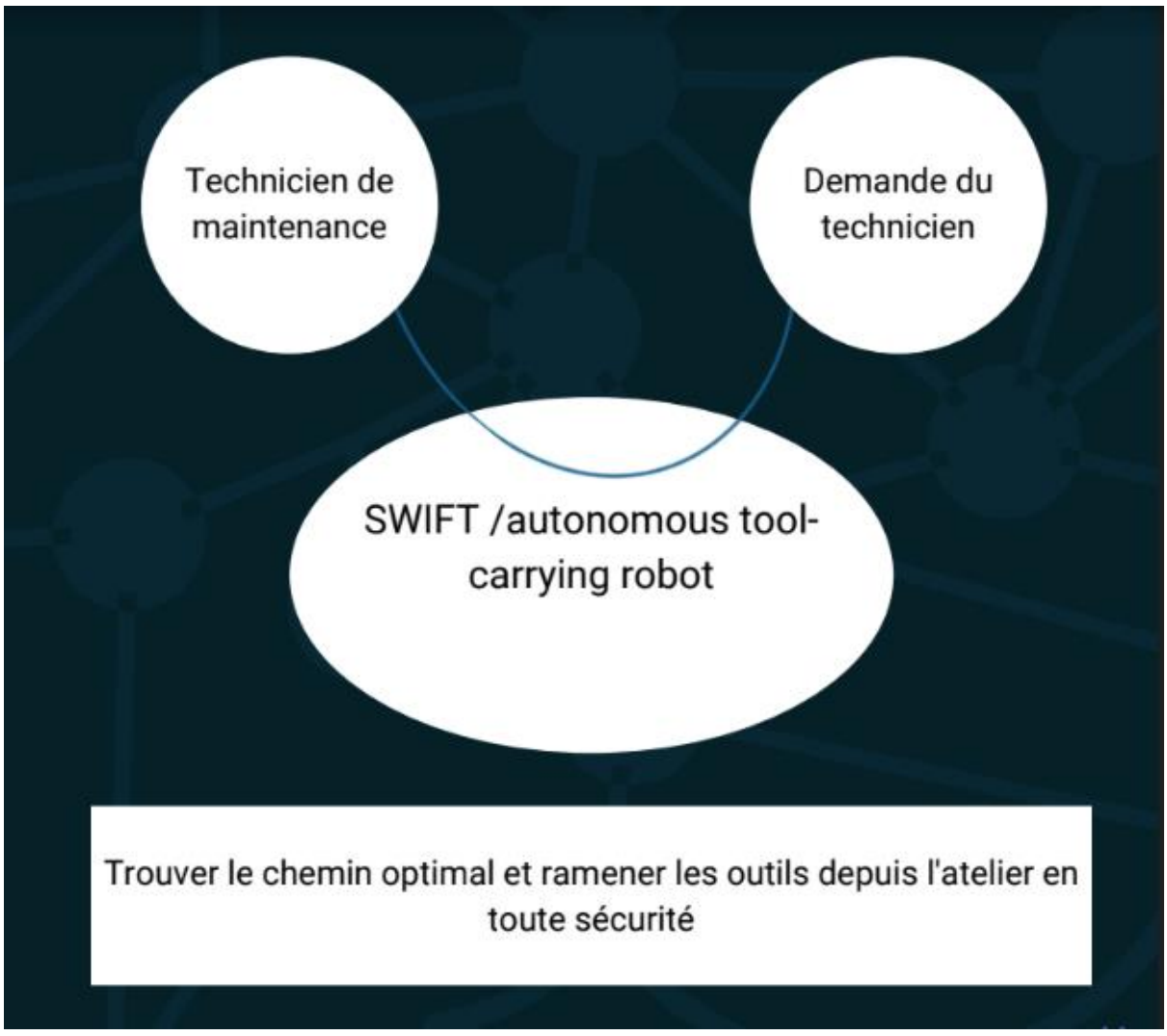
Une fois l'opérateur fait le diagnostic de la panne il n'a qu'à demander les outils qu'il a besoin à travers un web server connecté au robot. C'est ainsi que ce dernier doit choisir d'une façon autonome le chemin optimal vers l'atelier des outils en évitant tout obstacle. Une fois arrivé à sa destination SWIFT doit détecter à travers une caméra l'outil choisi le prendre et retourner rapidement vers son opérateur. Tout cela est affiché dans le web server ou l'opérateur peut suivre en temps réel l'avancement du robot.



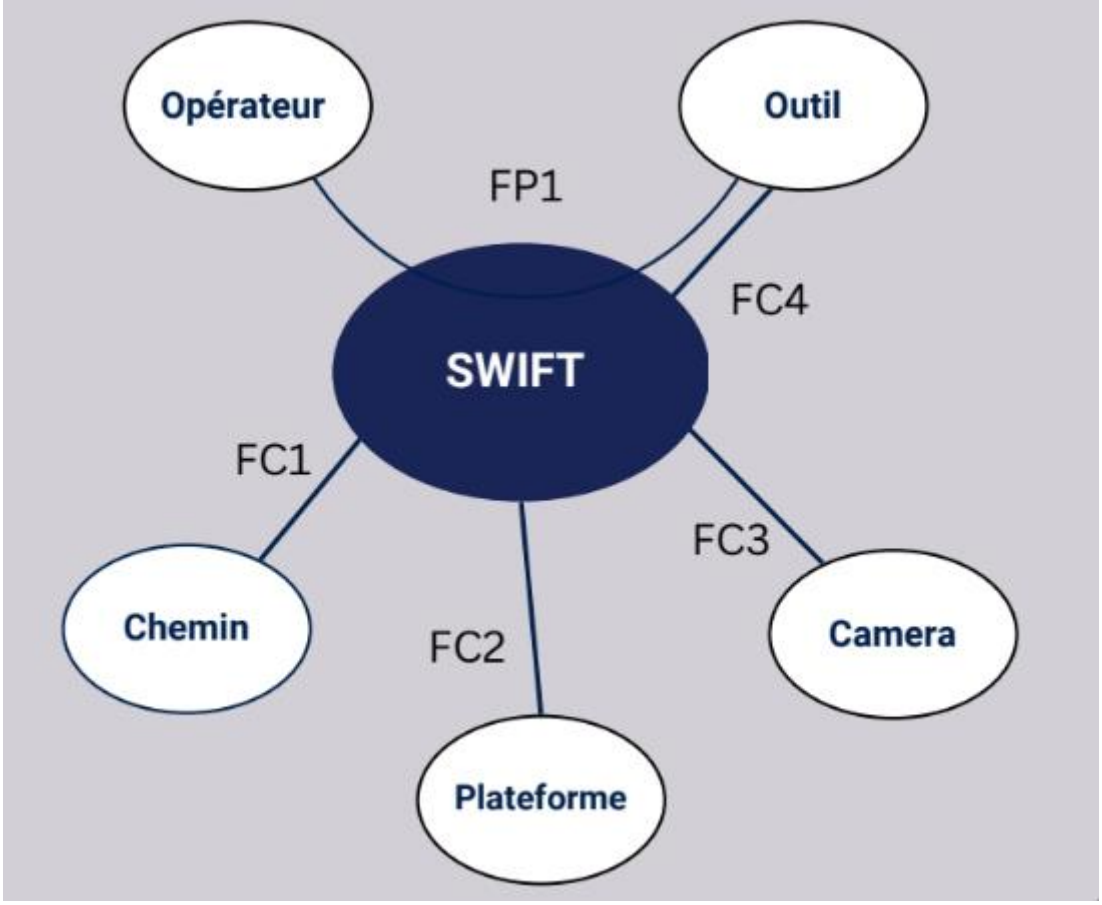
1.5 Analyse fonctionnelle

L'analyse fonctionnelle est une approche structurante au service de l'équipe en charge de la création ou de l'amélioration d'un produit. L'objectif consiste à identifier le besoin client, pour déterminer les fonctions du produit, avant de rechercher les solutions techniques et technologiques à mettre en œuvre.

1.5.1 Béte à corne :



1.5.2 Diagramme Pieuvre



FP1	Ramener l'outil à la demande de l'opérateur
FC1	Optimiser le chemin suivi et éviter les obstacles
FC2	Permettre à l'utilisateur de controler le robot et suivre ses mouvements
FC3	Contribuer avec la camera pour détecter l'emplacement de l'outil
FC4	Porter l'outil et garantir son arrivée

2 Chapitre 2 : Réalisation du projet

2.1 Étapes de la réalisation

On a tracé un plan de 6 étapes pour la réalisation du projet, la première consiste à **la préparation de data** qui va être utilisé pour **le développement du système de détection des outils**, la troisième étape se focalise sur le **développement du système de pathfinding,** puis on arrive au **développement du système de communication avec le robot** et vers la fin nous allons faire **une simulation du projet.**

2.2 Système de détection des outils

2.3 Système de pathfinding

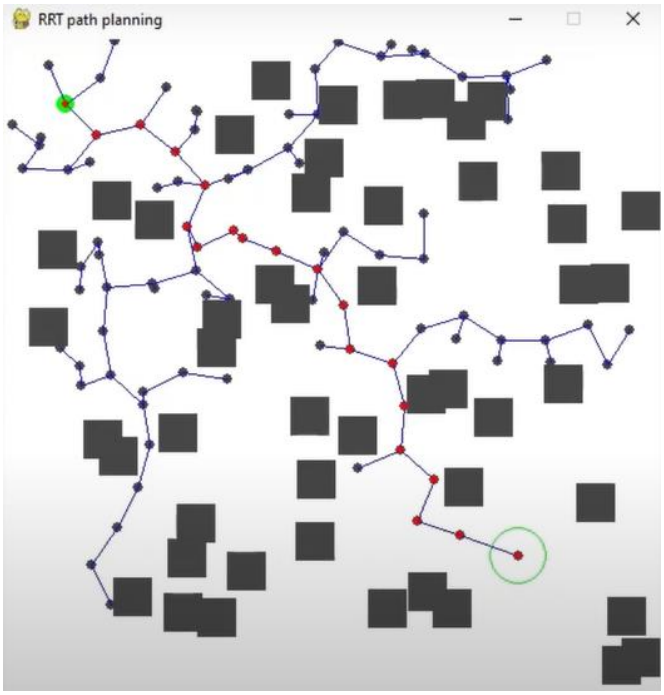
A l'aide des algorithmes d'optimisation notre robot peut choisir le chemin optimal on suppose qu'un ordre es déjà reçu donc notre robot peut savoir le point de départ et le point d'arrivé.

Pour illustrer l'efficacité de ces algorithmes nous avons fait une simulation via Pygame et là vous pouvez voir le résultat obtenu

- **RRT Algorithm**

RRT.py > main

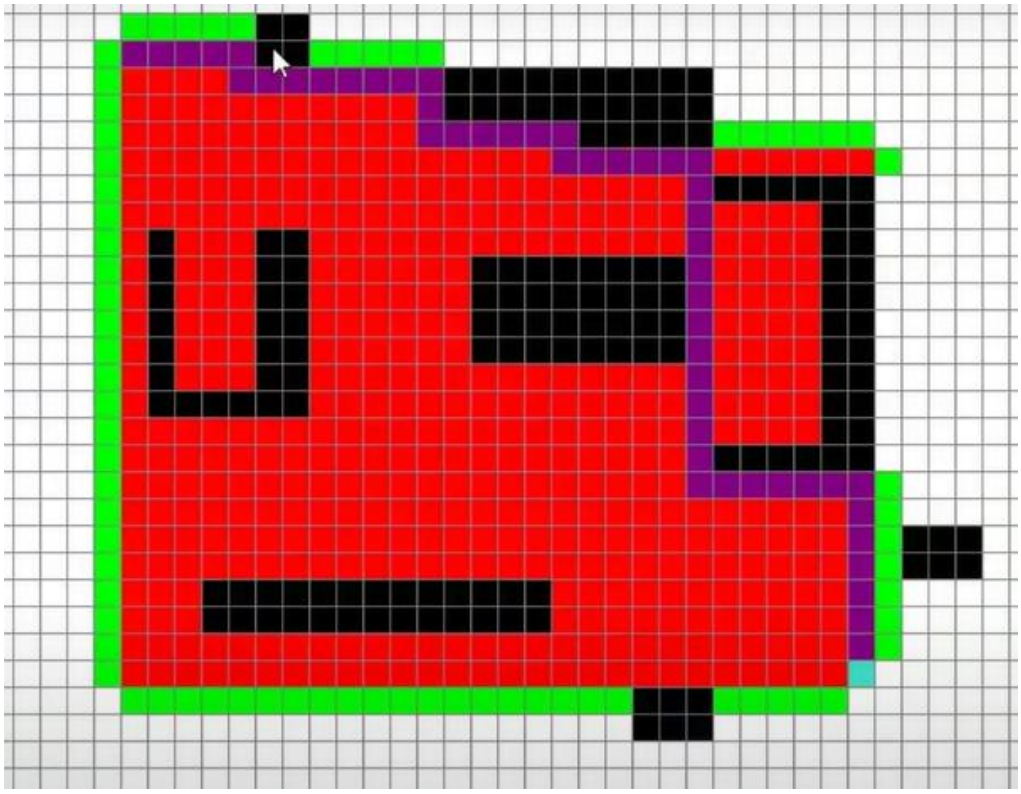
```
    t1=time.time()
    while (not graph.path_to_goal()):
        time.sleep(0.01)
        elapsed=time.time()-t1
        t1=time.time()
        #raise exception if timeout
        if elapsed > 10:
            print('timeout re-initiating the calculations')
            raise

        if iteration % 10 == 0:
            X, Y, Parent = graph.bias(goal)
            pygame.draw.circle(map.map, map.grey, (X[-1], Y[-1]), map.nodeRad*2, 0)
            pygame.draw.line(map.map, map.Blue, (X[-1], Y[-1]), (X[Parent[-1]], Y[Parent[-1]]),
                             map.edgeThickness)

        else:
            X, Y, Parent = graph.expand()
            pygame.draw.circle(map.map, map.grey, (X[-1], Y[-1]), map.nodeRad*2, 0)
            pygame.draw.line(map.map, map.Blue, (X[-1], Y[-1]), (X[Parent[-1]], Y[Parent[-1]]),
                             map.edgeThickness)

        if iteration % 5 == 0:
            pygame.display.update()
        iteration += 1
    map.drawPath(graph.getPathCoords())
    pygame.display.update()
    pygame.event.clear()
    pygame.event.wait(0)
```



- **A* Algorithm :**

De même pour l'algorithme A*, là nous avons créé un environnement virtuel et il va donner le chemin optimal.

```
def algorithm(draw, grid, start, end):
    count = 0
    open_set = PriorityQueue()
    open_set.put((0, count, start))
    came_from = {}
    g_score = {spot: float("inf") for row in grid for spot in row}
    g_score[start] = 0
    f_score = {spot: float("inf") for row in grid for spot in row}
    f_score[start] = h(start.get_pos(), end.get_pos())

    open_set_hash = {start}

    while not open_set.empty():
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()

        current = open_set.get()[2]
        open_set_hash.remove(current)

        if current == end:
            reconstruct_path(came_from, end, draw)
            end.make_end()
            return True

        for neighbor in current.neighbors:
            temp_g_score = g_score[current] + 1

            if temp_g_score < g_score[neighbor]:
                came_from[neighbor] = current
```



2.4 Système de communication

2.5 Simulation

A l’aide de webots, un simulateur open-source de robotique, nous arrivons faire la simulation du robot.
On a commencé par la création de l’environnement avec l’ensemble des objets nécessaires, puis la création du robot qui se compose d’un corps de base dont on a lié 4 roues et nous avons ajouté 2 capteurs de distance, droit et gauche, pour permettre au robot de détecter la distance entre le robot et l’obstacle, de plus, nous avons intégré une camera pour la détection des objets, finalement on a entré le code de control.

my_controller.cpp

```cpp
#include <webots/DistanceSensor.hpp>
#include <webots/Motor.hpp>
#include <webots/Robot.hpp>
#include <webots/Camera.hpp>
#define TIME_STEP 67

using namespace webots;

int main(int argc, char **argv) {
  Robot *robot = new Robot();
  DistanceSensor *ds[2];
  char dsNames[2][10] = {"ds_right", "ds_left"};
  for (int i = 0; i < 2; i++) {
    ds[i] = robot->getDistanceSensor(dsNames[i]);
    ds[i]->enable(TIME_STEP);
  }

  Camera *cm;
  cm=robot -> getCamera("CAM");
  cm -> enable(TIME_STEP);
  cm -> recognitionEnable(TIME_STEP);

  Motor *wheels[4];
  char wheels_names[4][8] = {"wheel1", "wheel2", "wheel
  for (int i = 0; i < 4; i++) {
    wheels[i] = robot->getMotor(wheels_names[i]);
    wheels[i]->setPosition(INFINITY);
    wheels[i]->setVelocity(0.0);
```