

Guide du Projet de Simulation Numérique

ECAM 3e

2023



SOMMAIRE

Contenu

I.	Introduction.....	3
II.	L'algorithme de Balas-Hammer	4
III.	Cahier des charges.....	5
IV.	Organisation	8
V.	Évaluation.....	11
VI.	Guide du développeur	12
	Exemple de lecture de fichier txt	14
VII.	Manuel de secours	15
VIII.	Exemple de trombinoscope.....	16

I. Introduction

Le projet de simulation numérique allie des compétences en mécanique, robotique, recherche opérationnelle et informatique. Pour cette cuvée 2023, nous vous proposons de transformer le robot de Lego en AGV !



Le robot EV3 est un robot ludique développé par Lego. Comme tout robot, il est constitué de moteurs, de capteurs et d'un calculateur. Il est, comme tout jouet Lego, modulable, au gré de l'imagination de ses utilisateurs. La programmation se fait nativement avec un langage graphique, inspiré par LabView, et accessible (en théorie) dès l'âge de 10 ans. Toutefois, pour ce projet, vous devrez développer en Python sous Visual Studio Code, en utilisant le *firmware* Lejos (projet Open Source). La documentation nécessaire vous sera fournie pour y parvenir.

II. L'algorithme de Balas-Hammer

À la suite du cours de Recherche Opérationnelle, nous vous proposons de **développer en Python un programme qui planifiera les déplacements d'AGV !**

On considère comme exemple une usine avec 4 lignes de montage et 4 magasins de stockage (voir figure 1). On connaît les coûts de livraison des pièces du stock i à la ligne j , soit une matrice C . De plus on a, à un instant donné, les demandes en pièces de chaque ligne et les offres de chaque magasin. On suppose être dans le cas particulier où les demandes des 4 lignes égalent les stocks des 4 magasins.

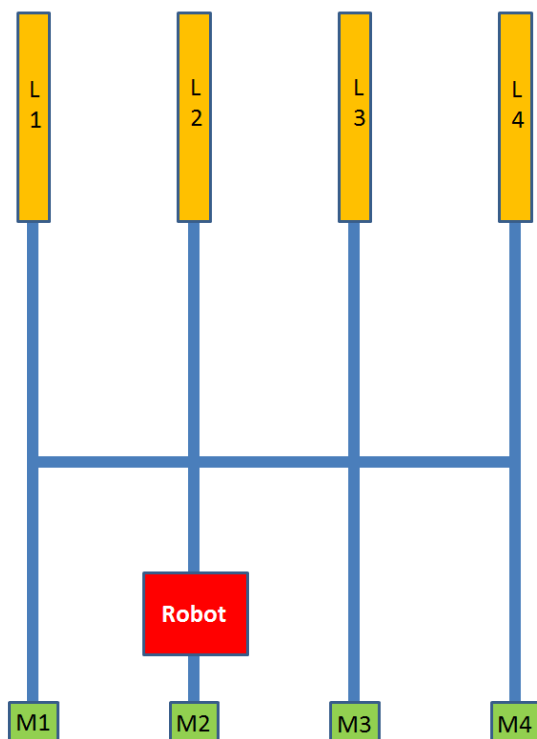


Figure 1 : plan de l'usine

Il s'agit de déterminer la matrice P , optimale au sens du coût total, telle que $P_{i,j}$ corresponde au nombre de pièces à livrer par le magasin i à la ligne j . Nous utiliserons pour ce faire l'algorithme de Balas-Hammer vu en cours de Recherche Opérationnelle.

III. Cahier des charges

On classe les exigences du client par niveau croissant. Vous pouvez travailler indépendamment sur les niveaux 0,1 d'une part et 2,3,4 d'autre part et donc vous partager le travail efficacement.

Niveau 0 :

En entrée : 1 fichier txt nommé *usine_i.txt* avec i un nombre. Les nombres sont séparés par un espace.

- La première ligne est un entier spécifiant le nombre de lignes de montage suivi d'un autre entier spécifiant le nombre de magasins de stockage. En effet le programme doit pouvoir tourner quel que soit le nombre de lignes et le nombre de magasins.
- Les lignes suivantes représentent la matrice des coûts dans l'ordre logique ...
- L'avant-dernière représente le vecteur des demandes
- La dernière ligne représente le vecteur des offres.

Par exemple, le fichier ci-dessous représente une usine avec 6 lignes de montage et 4 magasins.

```
6 4
5 4 11 5 11 8
7 2 18 11 4 5
10 7 11 17 8 12
11 7 15 15 11 16
100 210 200 250 200 120
120 300 410 250
```

La matrice des coûts est :

$$C = \begin{pmatrix} 5 & 4 & 11 & 5 & 11 & 8 \\ 7 & 2 & 18 & 11 & 4 & 5 \\ 10 & 7 & 11 & 17 & 8 & 12 \\ 11 & 7 & 15 & 15 & 11 & 16 \end{pmatrix}$$

Le vecteur des demandes est :

$$D = (100 \quad 210 \quad 200 \quad 250 \quad 200 \quad 120)$$

Et le vecteur des offres (ou stocks) est :

$$S = (120 \quad 300 \quad 410 \quad 250)$$

Vous devez écrire un programme qui lira le fichier et récupérera les données qu'il placera dans les bons attributs de votre classe, que vous afficherez ensuite.

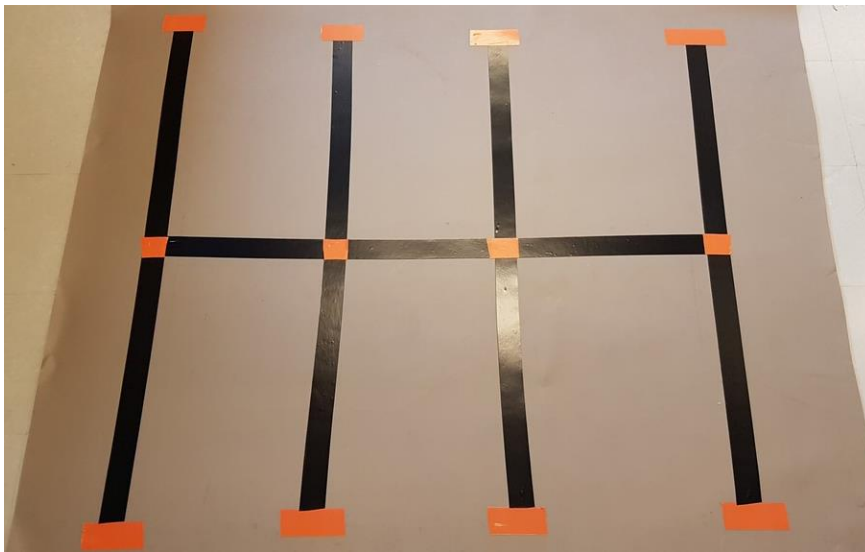
Vous trouverez en annexe un exemple de lecture d'un fichier *txt*. Les intervenants seront également à votre disposition pour vous aider.

Niveau 1 :

Implémentez l'algorithme de Balas-Hammer et déduisez-en la matrice P qui indique le nombre de pièces à livrer pour chaque couple (stock, ligne). Il est fortement recommandé de découper votre programme en fonctions à définir avant de coder et à valider par l'encadrant ! Il est également conseillé de bien tester chaque fonction avant de passer à la suivante. Bien entendu votre programme doit fonctionner pour un nombre quelconque de lignes de montage et de stocks.

Niveaux > 1 :

Pour la deuxième partie concernant le robot, on représentera au sol un plan de l'usine correspondant à l'exemple précédent (figure 1).



Les chemins autorisés des AGV seront représentés par un adhésif noir. Les carrefours, les magasins de stockage et les lignes seront représentés par un adhésif orange. Pour charger et décharger les pièces (matérialisées par une pièce lego), vous pourrez utiliser vos mains pour simplifier.

Niveau 2 :

Vous pouvez orienter votre robot comme vous le souhaitez au départ.

Vous devez construire et programmer un robot convoyeur qui simule le transport des pièces des magasins de stockage jusqu'aux lignes de montage, selon le résultat fourni par votre programme. Votre robot devra bien entendu suivre les « routes » noires.

Vous disposez dans votre kit¹, entre autres, de 2 gros moteurs, pouvant actionner des roues, d'un système de "roue folle", d'un capteur de couleurs devant être très proche du sol (3 cm max) et d'un capteur gyroscopique vous permettant de contrôler l'angle de vos virages. Les caractéristiques techniques de ces différents éléments se trouvent facilement sur Internet.

¹ Il y a 2 familles de kits EV3 : une version grand public et une version éducation. Nous avons complété ces 2 versions de manière à les rapprocher au mieux et rendre ce projet équitable.

Vous avez toute latitude pour la construction de votre modèle. De préférence, votre robot doit pouvoir transporter des pièces. En cas de difficultés, l'intervenant se tient à votre disposition pour vous guider dans vos choix.

Niveau 3 :

Des obstacles peuvent surgir sur la route. Le robot doit être capable de les détecter, de s'arrêter, d'émettre un son en forme d'avertissement et de repartir une fois l'obstacle disparu. Vous disposez dans votre kit de capteurs permettant de détecter des obstacles.

IV. Organisation

Le déroulement d'une séance

Au début de chaque séance, chaque élève doit :

- émarger sur la feuille de son équipe épinglée sur le panneau en liège près du Petit Amphi,
- noter les instructions de la séance figurant sur ce même panneau (horaire de passage avec les différents référents s'il y a lieu).
- Pour chaque équipe, le premier élève arrivé doit indiquer sur la feuille de présence, dans quelle(s) salle(s) se trouve l'équipe.

Les robots seront stockés à l'entrée du département informatique, dans des casiers cadenassés. Le numéro du casier et le code d'accès vous seront communiqués en début de projet. Un élève par équipe sera responsable du kit robotique. Il devra venir chercher le robot en début de séance et l'indiquer sur la feuille *ad hoc*, et ramener le robot en fin de séance. **Ne pas oublier le câble USB !**

Le contexte du projet

Vous êtes salariés d'une société de conseil en informatique. Un de vos clients (l'ECAM Rennes) a émis un appel d'offre, concrétisé par un cahier des charges, auprès de différents prestataires. Chaque équipe projet appartient à une société différente et elle conçoit une proposition en réponse à cet appel d'offre. En plus des exigences du cahier des charges, vous devez tenir compte d'autres contraintes du client :

- Date de remise de votre proposition
- Date de fin du projet

... et de votre propre société :

- Le respect des méthodes de travail de votre société.
- Le nombre de collaborateurs composant l'équipe projet.

Le commercial de votre société a réussi à avoir une idée du budget dont dispose le client. Celui-ci est surtout sensible au respect du délai et du cahier des charges. Votre société est très attachée à la réussite de ce projet dont découle l'obtention d'autres projets.

Chaque proposition est validée par la direction de chaque société. Elle peut être amenée à vous demander de la modifier avant la remise au client.

Quelques éléments peuvent évoluer en cours de projet.

Un cahier des charges vous est remis dans ce guide. À partir de celui-ci, vous devez remettre une proposition financière à Christophe Boisramé, en début de séance 3

Planification

Séance	Interventions de l'école	Modalités	Livrables attendus par l'école	Encadrants (*)
Présentation du sujet+ Gestion de Projet		Toute la promo		Tous
Séance 1	Notions sur Python Objet.	Avec les responsables développement.	Trombinoscope de l'équipe avec répartition des rôles.	NG + JML
Séance 2	Intervention GeP en début de séance avec les chefs de projet.	1h avec CB avec l'ensemble des chefs de projet en début de séance à distance ou présentiel (téléphone + partage d'écran).	Identification des tâches à réaliser. Affectation de ces tâches à des membres de l'équipe. Imputation commencée. Diagramme de classes.	CB + NG + JML
Séance 3	Revue de la proposition financière Point développement.	30 mns par équipe au complet	Proposition financière envoyée avant la séance. Suivi de projet à jour.	CB + NG + JML
Séance 4	Séance non encadrée			
Séance 5	Revue de projet. Point développement.	30 mn par équipe pour le GeP	Suivi de projet à jour. Niveau 1 terminé.	CB + NG + JML
Séance 6	Point développement.		Niveau 2 terminé.	NG + JML
Séance 7	Séance non encadrée			
Séance 8		1 h avec l'ensemble des chefs de projet en début de séance.	Niveau 3 terminé	NG + JML + CB
Validation	Recette		Tous les livrables de la GeP à jour. Formulaire audit de GeP complété.	Tous

(*) NG = Noussaïba GASMI, CB = Christophe BOISRAMÉ, JML = Jean-Marc LAFERTÉ

La gestion de projet

Christophe Boisramé interviendra :

- Lors de la présentation du projet avec toute la promotion)
- En début de séance 2 à distance ou présentiel auprès des CdP
- En séance 3 pour une rencontre individuelle d'environ 30 mn avec chaque équipe complète (présentation de la proposition financière)
- En séance 5 pour une rencontre individuelle avec chaque équipe complète (suivi du projet)
- En début de séance 8 à distance ou présentiel auprès des CdP
- Lors de l'audit final

Il consultera régulièrement le suivi de projet déposé sur l'espace partagé (drive).

Chaque équipe disposera d'un suivi de projet à jour régulièrement ainsi qu'à la fin du projet (toutes les charges remplies), lors de l'audit.

Pour la séance de rencontre avec les équipes, les élèves le rejoignent dans une salle prédéfinie, à la fréquence d'un groupe toutes les 35mn. Excepté le 1^{er} groupe, qui se présente à l'heure prévue, chaque groupe est prévenu par le groupe précédent qu'il doit se présenter à la réunion de gestion de projet. Cela évite les attentes en cas de retard.

Les créneaux horaires sont mentionnés dans la présentation initiale.

Toutes les ressources (documents, tutos) se trouvent sur le moodle.

Le groupe doit être en mesure de projeter tous les documents de gestion de projet lors de la séance (PC portable, clef USB, accès à l'espace partagé).

Les livrables

La proposition financière

L'équipe fournira une proposition financière au début de la séance 3.

Elle sera disponible AVANT le début de la séance 3, la date du fichier faisant foi. Les versions ultérieures devront avoir une version différente.

Voir les détails dans la présentation de la Gep.

Les fichiers de suivi projet

Ce fichier regroupe les données d'avancement du projet.

Ce fichier qui évolue tout au long du projet, est mémorisé, avec un nom différent, avant chaque séance de Gestion de Projet.

Il doit donc y avoir un fichier par séance en fin de projet (donc 8 fichiers)

Voir les détails dans la présentation de la Gep.

V. Évaluation

La validation est réalisée en 3 temps décrits ci-dessous et donnant lieu à 3 notes. L'ordre de passage sera affiché en début de séance sur le panneau habituel.

I. Évaluation de la Gestion de projet – Auditeur : Christophe BOISRAMÉ – Durée : 25 mns - élèves : le chef de projet + les équipiers désirant assister à cette recette.

Lors des séances 3 et 5 une évaluation de votre suivi de projet sera réalisée.

La recette consiste en un audit de la GeP par l'intervenant en GeP, avec le chef de projet et les équipiers désirant y assister.

Les points abordés lors de cette recette sont précisés dans la présentation de la GeP (slide « la soutenance pour la GeP »).

Cet audit nécessite la communication au préalable de tous les éléments nécessaires. Il s'appuie sur le formulaire « audit » se trouvant dans le fichier de suivi de projet. Ce formulaire devra donc être rempli avant l'audit.

II. Recette algorithme sur PC – Noussaïba GASMI – Durée : 25 mns – élèves : développeurs

Les niveaux 0 et 1 seront évalués sur ordinateur et un audit du processus de développement du programme sera réalisé. Votre programme doit fonctionner pour un nombre quelconque de lignes de montage et de stocks.

III. Recette Robot - Jean-Marc LAFERTÉ – Durée : 25 mns – élèves : reste de l'équipe

Les niveaux ≥ 2 seront testés durant cette partie. L'usine sera symbolisée au sol comme expliqué dans ce document.

La recette aura lieu dans la salle Amorgos (Département Informatique).

Les fichiers txt représentant les usines seront chargés via un PC. **Vous devrez venir avec votre robot démarré pour ne pas perdre de temps et avec le câble USB !**

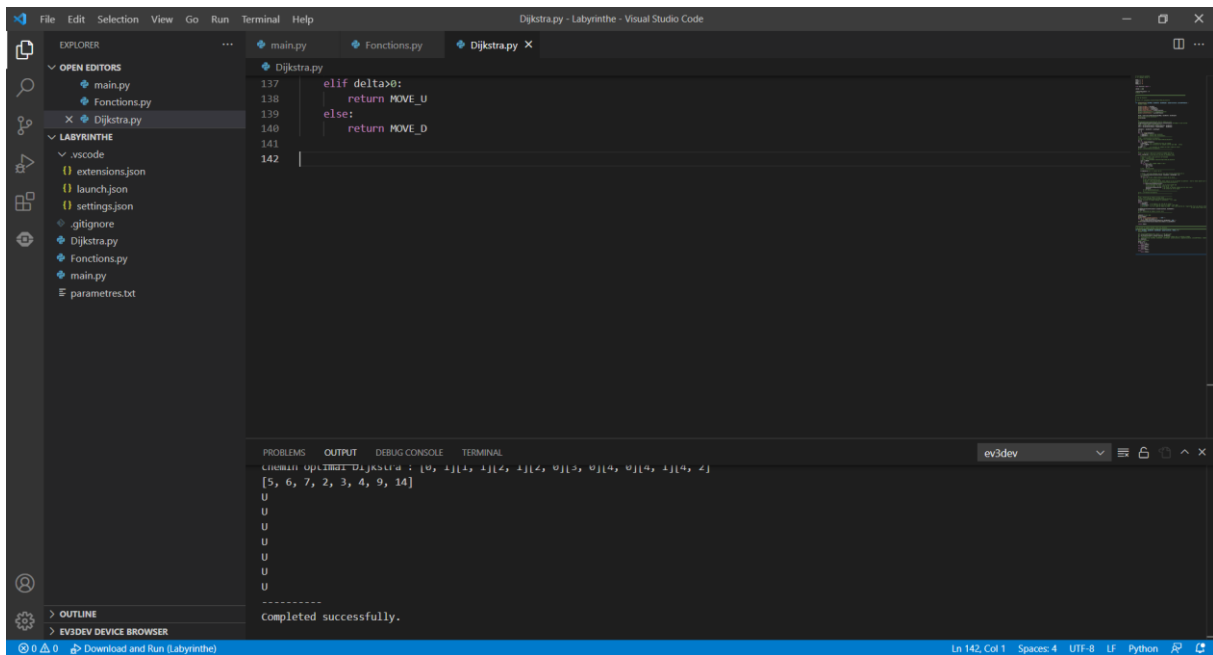
On appréciera la navigabilité du robot, sa rapidité, sa fiabilité et à un degré moindre ... son design !

VI. Guide du développeur

Installation de Visual Studio Code (VS Code) :

<https://education.lego.com/en-us/product-resources/mindstorms-ev3/teacher-resources/python-for-ev3>

Vue de l'interface de VSCode :



À gauche, on a l'explorateur de projets avec les différents fichiers du projet ouvert (si cette vue n'est pas activée par défaut, aller dans le menu View/Explorer). À droite en haut, on trouve l'éditeur avec le code en cours. Enfin en bas à droite, on a la console où peuvent s'afficher les résultats (print).

Création d'un projet sous VS Code² :



Pour créer un nouveau projet sous VS Code, cliquer sur l'icône de la barre de boutons à gauche.

Pour ajouter un fichier, positionnez la souris dans le cadre de gauche au niveau du nom du projet puis cliquer sur l'icône "New File" (avec un +). Pensez à changer le nom du fichier avec la bonne extension !

Pour exécuter un programme, reliez votre robot au PC avec le câble USB puis aller dans le menu Run.

² NB : si vous utilisez un PC de l'école, il faut se rendre dans extension > puis rechercher EV3 et cliquer sur installer.

Vous pouvez exécuter en mode debug, ce qui vous permet notamment de voir affichés sur votre ordinateur tous les print, mais il faut que le robot soit connecté au PC, ce qui n'est pas très pratique ! Si c'est impossible ou si vous êtes sûr de votre code, vous pouvez exécuter en mode normal. La première fois il vous faudra sans doute demander à se connecter au robot (voir message). Attention le robot va bouger rapidement mais vous pouvez l'arrêter en cliquant sur le bouton en haut à gauche de la brique. L'exécution provoque le chargement du programme sur la brique. Vous pouvez ensuite exécuter votre programme à partir de la brique en allant dans le menu "File Browser", puis sélectionner le bon projet puis le module main contenant le programme principal (nous vous conseillons de l'appeler main).

Éléments de langage

Attention pyBricks est basé sur MicroPython, un langage adapté aux systèmes embarqués. La syntaxe est bien du Python mais tous les modules que vous trouvez habituellement ne sont pas utilisables ici. Ainsi de numpy. Nous travaillerons donc avec des listes plutôt que des tableaux.

De nombreux import vous seront nécessaires. En voici une liste non exhaustive :

```
#!/usr/bin/env pybricks-micropython
from pybricks.hubs import EV3Brick
from pybricks.ev3devices import (Motor, TouchSensor, ColorSensor,
                                InfraredSensor, UltrasonicSensor, GyroSensor)
from pybricks.parameters import Port, Stop, Direction, Button, Color
from pybricks.tools import wait, StopWatch, DataLog
from pybricks.robotics import DriveBase
from pybricks.media.ev3dev import SoundFile, ImageFile
```

Initialisation du robot avec 2 moteurs :

```
ev3 = EV3Brick()
# Initialize the motors.
left_motor = Motor(Port.B) # vérifier vos branchements !
right_motor = Motor(Port.C)
# Initialize the drive base.
robot = DriveBase(left_motor, right_motor, wheel_diameter = 43,
axle_track=138) # toutes les distances sont en mm
robot.settings(220, 100) # speed and acceleration
ev3.speaker.beep() # émet un son pour s'assurer qu'il fonctionne
```

Mouvements basiques (programmation de type "objet") :

```
robot.turn(45) # angle en degrés (positif ou négatif)
                # NB : raisonne dans le sens horaire !
robot.straight(100) # distance en mm (si négatif, recule)
robot.stop()
```

Capteurs (exemple avec le capteur à ultrasons) :

```
obstacle_sensor = UltrasonicSensor(Port.S4)
while obstacle_sensor.distance() > 300:
    wait(10)
```

Afficher du texte sur l'écran LCD (print ne fonctionne pas sur le robot) :

```
ev3.screen.draw_text(50, 50, "hello") # positionné en (50,50)
```

Vous trouverez une documentation détaillée sur <https://pybricks.com> ou plus synthétique sur http://numerique.ostralo.net/robotLego_python (oubliez les fautes d'orthographe !).

Exemple de lecture de fichier txt

En prenant l'exemple du fichier txt donné dans ce document ...

```
file = open("test1.txt", "r")
lignes = file.readlines()
for ligne in lignes:
    print(ligne)
    liste = ligne.split(" ")
    print(liste)
    for x in liste:
        n = int(x)
        print(n)
file.close()
```

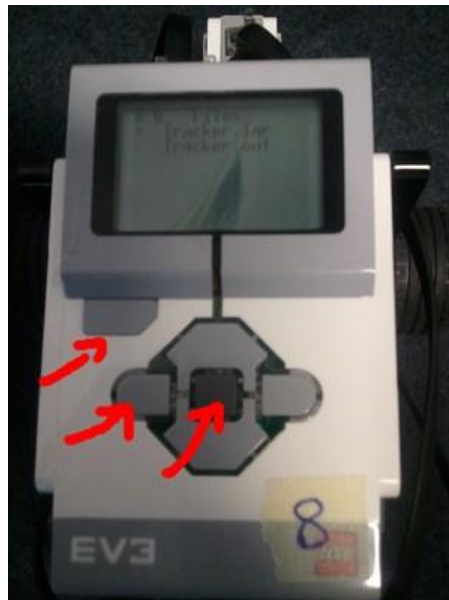
VII. Manuel de secours

Si l'ordinateur ne trouve pas le robot, débranchez puis rebranchez le câble USB.

Rappel : vous pouvez arrêter brutalement le programme en cliquant sur le bouton en haut à gauche.



Si le robot ne "répond" plus :


- reboot robot en appuyant simultanément sur les boutons suivants : haut gauche, central et gauche plusieurs secondes ...
- En dernier recours : enlever la batterie et la remettre.






VIII. Exemple de trombinoscope


Equipe Projet A1



Samuel Toukio
Responsable Robot


Adrien Itsweire
Assistant Développement


Maxime Veirier
Scrum Master



Gwenole Le Calonnec
Responsable Développement


Frank Filoda
Responsable Etudes

