ETAT DES LIEUX DU PROJET RobotServiceJeunesse2023 (RSJ23) AU 5 MAI 2023 (jour de l'atelier avec les enfants)

1. Généralités - Rappel de l'objectif du projet RSJ23:

Faire assembler par les enfants un véhicule télécommandé et équipé d'une caméra permettant de mettre en oeuvre une application basée sur l'utilisation d'Intelligence Artificielle.Cet objectif a été précisé aux alentours de noel 22:

- voiture équipée d'une ESP CAM, de 2 moteurs à cc, et pilotée via une télécommande (TCD) Infrarouge (IR)
- la partie IA consistait à reconnaître différentes formes posées au sol, afin de découvrir une une séquence secrète de formes (sorte de jeu Master Mind)

Plusieurs aspects du projet n'ont pas été réussis. Malgré tout, grâce à la participation de l'équipe, nous avons réussi à proposer aux jeunes une séance attrayante et ludique.
On pourrait synthétiser les points défaillants ainsi:

- trop de changements dans le choix du microcontrôleur
- tests sur les éléments effectués trop tardivement et pas systématiquement, qui ne nous ont pas permis de détecter les éléments de mauvaise qualité
- difficultés de maîtriser l'apprentissage du RdN
- manque de documentation partagée

Les points positifs:

- la base du véhicule est validée et pourra être reprise telle quelle
- la reconnaissance des figures est en bonne voie
- la compréhension par l'équipe des concepts généraux du projet nous a permis de mener une séance avec les jeunes assez réussie

La conclusion générale est que l'on choisit de reconduire le projet dans son esprit (véhicule télé-piloté, reconnaissance optique de figures, support du jeu MasterMind) pour la saison 2023-2024, en corrigeant les défaillances constatées

En détail

- compléter et finaliser la nomenclature (Utilisation du logiciel Notion)
    - plans de câblage électrique
    - plans de montage mécanique
    - guide de montage

    - vérifier la composition des boîtes
- les tests
    - les tests en général n'ont pas été suffisants ni suffisamment détaillés
    - la motorisation est comprise (avec test unitaire)
    - la détection IR est comprise (avec test unitaire)
    - le couplage (Moteur + IR) n'est pas réalisé
- IA
    - le modèle d'entraînement n'est pas suffisamment solide malgré une phase de validation correcte, mais lorsqu'elles sont appliquées aux images réelles, les prédictions ne sont qu'approximatives (environ 30% de réussite seulement)
- problèmes sur les accus (de récupération) qui demandent à être reconditionnés

2. Etat du projet au 5 mai 23

2.1. Le matériel

- une définition matérielle est acquise mais non écrite
- le matériel correspondant a été approvisionné et réparti dans 8 boîtes kit
- la partie de l'assemblage nous incombant (soudures, ….) est réalisée pour les 6 kits destinés aux enfants
- 2 autres véhicules ont été entièrement assemblés pour nos besoins de développement (tutoriel, checklist d'assemblage, essai de validation,...)
- un proto "pieuvre" (composants et câblages sur table) à des fins de validations électriques
- des équipements supplémentaires ont été approvisionnés en cours d'année au cours de évolutions du projet (mini tourelle pan tilt pour pointer la caméra, manette joystick, microcontrôleur et shield,….). A recenser.

2.2. L'assemblage mécanique:

Une procédure d'assemblage mécanique a été définie (mais non écrite) et semble donner satisfaction avec 2 points à noter:

- la définition du support de l'ESP CAM risque d'être à revoir en fonction des essais de la caméra.
- est ce que le scotch double face utilisé pour assembler l'ESP CAM, l'ESP moteurs, la diode IR à la structure ne perturbe pas le bon fonctionnement de ces éléments? (isolement trop faible par exemple)
- même remarque pour fixer la caméra à son support.

2.3. Le fonctionnel:

2.3.1. La prise d'image , la transmission des images par wifi, l'affichage des images sur téléphone portable, l'affichage et le Traitement des Images (TI) par le PC:

On a vérifié:

- que l'ESP CAM créait des images
- que le TI de reconnaissance de formes fonctionne suffisamment rapidement pour notre besoin
- mais que ce TI donne un taux de bonne détection beaucoup trop faible pour notre besoin. Pour dérouler une partie de Mastermind il faut que le taux de fausse détection soit très faible.

Mais on n'a pas vérifié:

- le bon positionnement de notre caméra par rapport au sol en fonction de la précision de pilotage du véhicule. La définition du support de la caméra pourrait être remis en cause

2.3.2. La gestion et l'affichage du jeu Master Mind:

Rien n'a été testé sur ce sujet.
Est ce que quelque chose a été effectué sur ce sujet (*voir le paragraphe sur les développements RdN*)
Aucune définition de besoin n'a été validée par le groupe.

2.3.3. Les câblages:

On a défini et vérifié les câblages:

- entre les 3 pattes de la diode IR et l'ESP moteurs
- entre l'ESP moteurs et le pont en H
- entre le Boîtier Accu (BA), le domino l'ESP moteurs et l'ESP CAM

2.3.4. La vitesse et le sens de rotation des moteurs

On a vérifié:

- la vitesse de rotation max des moteurs à plusieurs tours par seconde pour le code 255, dans chaque sens de rotation.
- la vitesse de rotation min (environ 1 à 2 tr/s) pour un code compris entre 60 et 100, dans chaque sens de rotation
- il faut noter que ces mesures ont été faites avec les moteurs fixés sur un support, sans aucune charge sur les roues.Ils seront à reprendre sur un véhicule posé au sol

2.3.5. Estimation de la vitesse linéaire et de la vitesse de rotation du véhicule pour pouvoir assurer un positionnement correct de la caméra par rapport aux formes posées au sol:

- la taille des formes posées au sol est de l'ordre de 3 cm de diamètre
- la taille de la zone observée par la caméra située à 7 cm de hauteur, est environ de 6 cm de diamètre (pour un champ de la caméra de 45°)
- on a estimé que pour pouvoir positionner précisément la caméra au dessus des formes il faudrait limiter la vitesse avant/arrière à 10 cm/s soit environ 0.5tr/s et limiter la vitesse de rotation du véhicule à 10°/s soit une survitesse d'une roue de 0.1tr/s par rapport à l'autre roue.

2.3.6. Le pilotage du véhicule par télécommande IR

Elle n'est pas fonctionnelle:

- si la 1ère commande de la TCD est bien prise en compte, les suivantes sont ignorées

2.4. Développements réalisés autour du Réseau de Neurones Artificiels (RdN)

2.4.1. Une première phase a consisté à entraîner un Réseau de Neurones Artificiels (RdN) destiné à reconnaître des pastilles de couleurs

Mais en fait ici, on n'a pas vraiment besoin d'un RdN et une simple reconnaissance des couleurs (*librairie OpenCV*) donne la réponse, associée à l'utilisation d'un détecteur de couleurs sous la forme d'une diode triple.

Ceci permet une implémentation satisfaisante du jeu MasterMind (*développée en Python*) démontrant les bases de l'algorithme du jeu, (*sans pour autant aborder l'IHM pour l'interaction du joueur, et la réaction du robot*)

Mais ceci ne remplit pas un des objectifs du projet: montrer des éléments de la technologie IA !!

2.4.2. Donc on oriente alors le projet sur une deuxième phase

On remplace la détection de pastilles de couleurs par une détection de figures géométriques associée à un RdN entraîné.

- L'entraînement du RdN sera assuré par un modèle **Keras**
- Les figures de bases sont dessinées géométriquement par la librairie **OpenCV**
- 8 figures sont sélectionnées (*parmi 12 ou plus*) afin de limiter la possibilité de confusion entre elles et donc de diminuer la complexité logicielle.

Les éléments logiciels suivants sont alors développés:

1. création des figures 8 génériques (**OpenCV**)
2. création des données d'entraînement **par multiplication et modification aléatoire des figures génériques** (*méthode d'* "*augmentation*") jusqu'à produire 50 000 (*ou plus*) versions modifiées des figures
3. création d'un jeu de données comparables aux données d'entraînement pour servir de validation du modèle
4. entraînement du modèle sur les données d'entraînement
5. validation du modèle utilisant les données de validation (*ce ne sont pas encore des images réelles*)
6. production de graphiques de la mesure de la qualité

⇒ **Cette première phase donne satisfaction !! (> *99% de prédictions correctes*) mais il faut passer aux images réelles !!**

2.4.3. Troisième phase de développement :

On va ici utiliser des vraies données (*images prises par la caméra du ESP32-CAM*)

- on produit à nouveau les 8 images graphiques sur papier (*images de base*)
- on produit alors environ une centaine d'images (*images fixes*) en faisant varier l'angle de prise de vue et la distance de la caméra au papier
- mais aussi des photos ne montrant que le fond (= *absence de figure*)

ensuite on reprend la même séquence logicielle que précédemment:

1. **augmentation** des images
2. modification du modèle par ajout de la classe supplémentaire correspondant à la détection du fond (=> *9 classes*)
3. entraînement du modèle sur les données d'entraînement
4. validation du modèle par les données construites
5. production de graphiques de mesure de la qualité

⇒ Cette phase donne à nouveau satisfaction !! Il reste à tester les prédictions sur des images réelles de la caméra

2.4.4. On effectue alors les prédictions sur des images réelles prises par la caméra:

- développement d'une application logicielle Python "réaliste" où des photos sont prises en boucle avec la caméra de l'ESP32-CAM
- chaque photo est alors confrontée au modèle entraîné (*en temps réel*)

⇒ ce test lui ne donne qu'une efficacité de "prédiction" assez mauvaise (environ 20% ou 30%)
Mais par contre on montre que la performance est suffisante pour analyser les images en temps réel (donc pendant le déplacement du véhicule)

Une analyse basée sur la documentation Keras, à ce stade du modèle RdN montre que l'on pourra appliquer une méthode d'amélioration consistant :

- à développer un modèle étendu et plus sophistiqué (*complexe*)
- à entraîner ce modèle à partir des paramètres de l'ancien modèle (*la complexité de ce modèle étendu ne permettrait pas d'obtenir une convergence de l'entraînement à partir de paramètres aléatoires*)

2.5. Préparation d'une application finale

Ce développement a pour but de produire un "package" contenant tous les éléments logiciels nécessaires pour offrir

- le modèle pré-entraîné du RdN
- le logiciel du jeu MasterMind
- l'IHM pour recevoir les images de la caméra et les réactions du robot (comportement du jeu)

Cette étape a été réalisée sous forme d'un package **PyPi** (*assez comparable à la logique "Play Store" d'Androïd, ou "Apple Store"*)

Ceci permet de créer un package téléchargeable à partir d'une adresse Web, et contenant tout ce qui est nécessaire à installer cette application sur un PC (*librairies, le modèle pré-entraîné, l'application IHM*), puis faire fonctionner le jeu MasterMind.

Le package a été créé, installé, évidemment dans sa version incomplète (sans l'IHM du jeu) (https://pypi.org/project/AnumbyRobotSJ/)

2.6. Dépôt Github

Tous les développements logiciels, que ce soit le code Arduino ou les développements Python pour le RdN et l'application MasterMind sont gérés dans le dépôt Github dédié au projet version 2023

https://github.com/anumby-source/RobotServiceJeunesse2023

3. Propositions d'améliorations

- adoption des deux logiciels collaboratifs
    - **Notion** pour la gestion de projet, et l'organisation des tâches
    - **Github** pour les gestion des versions des documents

- mise en place d’un contrôle-qualité sur les composants électroniques
    - ajouter une antenne efficace pour la transmission directe des images et des commandes
- mise en place d’un programme de tests:
    - tests unitaires sur chaque composant avec mesure des performances et des caractéristiques (*codes arduino*)
    - tests système (*codes combinés motorisation + IR*)
- tests fonctionnels (*mesure de performance, et fiabilité*)
- poursuite des développements sur l’IA
    - mise en place des options de Ré-apprentissage
- séances de mise à niveau (*codage, pratique des logiciels collaboratifs*) pour l’équipe

Commenté [1]: Je ne comprends pas ce que tu veux dire par "antenne efficace". Actuellement on utilise notre fameux ESP32-CAM pour la vidéo et ESP8266 pour les commandes.