UPSSITECH
ÉCOLE D'INGÉNIEURS

UNIVERSITÉ TOULOUSE III PAUL SABATIER
Université de Toulouse

Rapport de Projet

Projet : Moteur de fusion multimodale

Author Pinçon Pierre

Dorian SAURAT

Professeur Mr Truillet Phillipe

Date Novembre 2023

Année 2023-2024

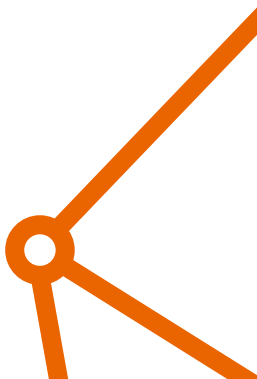


Table des figures

Introduction

Le domaine de l'interaction multimodale vise à améliorer l'expérience utilisateur en combinant diverses modalités telles que la reconnaissance vocale, la reconnaissance gestuelle et le pointage sur un écran. Mais aussi de manière plus large, d'autres outils comme par exemple la synthèse vocale Dans le cadre du bureau d'étude pour le projet 3A SRI 2023/2024, nous nous attelons à la conception et à l'implémentation d'un moteur de fusion multimodale, visant à interagir avec une palette de dessin dépourvue de boutons.

L'objectif principal de ce projet est d'aboutir à la réalisation d'un moteur capable de créer et de déplacer des formes sur la palette de dessin en utilisant trois modalités d'interaction distinctes : la reconnaissance de parole, la reconnaissance de formes, et le pointage avec la souris. Cette approche multimodale vise à offrir une expérience d'interaction riche et intuitive, rappelant le célèbre concept du MIT "put that there," datant de plusieurs décennies.

Au fil des pages de ce compte rendu, nous détaillerons la conception et l'architecture logicielle du moteur de fusion, explorant la mise en œuvre des grammaires de gestes et de parole, ainsi que la gestion des différentes formes sur la palette de dessin. Le choix des outils, notamment le langage de programmation, les outils de reconnaissance vocale et gestuelle, ainsi que la communication inter-outils via Ivy, sera également expliqué.

Nous aspirons à démontrer non seulement la fonctionnalité et l'efficacité de notre moteur de fusion multimodale, mais également à partager les enseignements tirés de ce projet, mettant en lumière les défis relevés et les solutions adoptées pour les surmonter. Ce compte rendu offre ainsi un aperçu détaillé du processus de conception et d'implémentation de notre moteur de fusion multimodale.

Table des matières

1 Cahier des charges

Avant tout, faisons un petit retour sur le cahier des charges et les attentes vis-à-vis de ce projet.

1.1 Objectifs fonctionnels

- Création de formes sur la palette de dessin :

  Le système doit permettre à l'utilisateur de créer des formes géométriques telles que rectangles, cercles, triangles et losanges sur la palette de dessin.

- Déplacement des formes :

  Les formes créées doivent être déplaçables à travers des actions spécifiques. L'utilisateur peut sélectionner une forme et la déplacer à l'endroit désiré sur la palette lorsqu'il indique au moyen du moteur de fusion qu'il souhaite réaliser un déplacement de forme.

- Actions multimodales pour dessiner et déplacer :

  Les interactions doivent être multimodales, combinant la reconnaissance vocale, gestuelle et manuelle pour créer et déplacer des formes. Par exemple, l'utilisateur peut dire "Dessiner" suivi d'un geste pour spécifier le type de forme.

1.2 Contraintes et spécificaitons

- Langage de programmation utilisé :

  Le système a at développé en utilisant le langage de programmation Python, assururant ainsi la flexibilité et la facilité de développement. Nous développerons ce choix ultérieurement.

- Communication inter-outils via Ivy :

  Les différents composants du système devront communiquer via Ivy, un bus logiciel permettant l'échange de messages entre les outils.

- Outils de reconnaissance vocale et gestuelle :

Les système utilisera des outils tels que speechX_1recognition et gesture des librairies python pour réaliser certaines tâches de la multimodalité.

- Interface graphique :

  Autant que le permet l'application, une interface graphique est utilisée pour rendre l'interaction plus agréable. Python nous permet d'utiliser tkinter à cette fin.

- Utilisation manuelle de la palette de dessin :

  En plus des interactions vocales et gestuelles, l'utilisateur pourra interagir manuellement avec la palette de dessin en utilisant des dispositifs de pointage ou l'écran tactile, en fonction des fonctionnalités du système.

2 Détails de l'application

2.1 Architecture du projet

- Palette de Dessin :

  Interface utilisateur permettant la création, le déplacement et l'interaction avec les formes géométriques. La palette représente en quelque sorte la fon de la chaîne d'interaction. C'est elle qui permet à l'utilisateur de visualiser les différentes actions qu'il commande et leurs effets sur les formes crées ou déplacées.

- Moteur de Fusion Multimodale :

  Responsable de la fusion des informations provenant de la reconnaissance vocale, gestuelle et des interactions manuelles. Il contrôle également la logique de création et de déplacement des formes. Le moteur de fusion joue le rôle de centre névralgique de l'application. Sans le moteur, les différents agents ne sont que des interfaces monomodales. C'est lui qui permet la multimodalité.

- Reconnaissance Vocale : Utilise l'outil SpeechRecognition pour la conversion de la parole en commandes textuelles utilisables par Ivy et le reste de l'application.

- Reconnaissance Gestuelle :

  Intègre les outilssystème de reconnaissance de gestes interactif à l'aide de l'algorithme NDollar

- Communication entre les agents :

  Chaque script est un agent Ivy. Ainsi, il devient possible d'utiliser le bus de communication Ivy pour faire communiquer les programmes via des chaines de caractères.

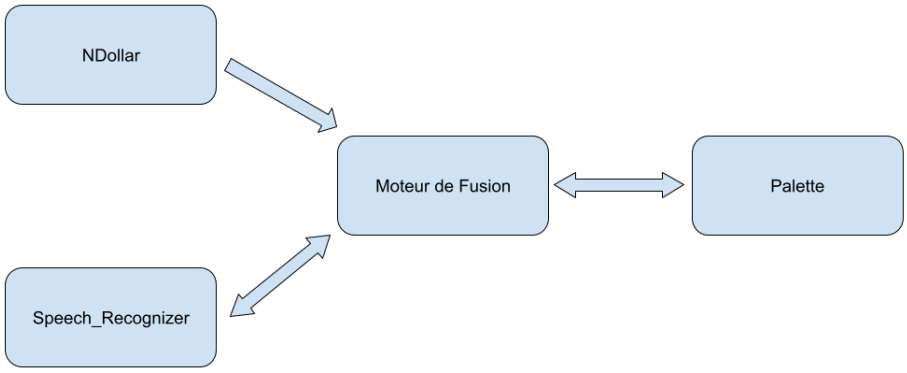


FIGURE 1 – Schema de l'architecture générale

Comme le montre le schéma ci-dessus, l'application ne communique pas de la même manière avec tous les agents. En effet, par exemple, pour des raisons pratiques, nous pouvons constater que le lien entre le moteur et la reconnaissance de formes est unidirectionnel. C'est intentionnel, car cela simplifie les interactions. Lorsque l'application de reconnaissance identifie une forme, elle publie un message sur le réseau pour informer le moteur de fusion de sa découverte. Ainsi, le moteur de fusion n'a pas besoin d'envoyer de requête à l'agent pour initier une reconnaissance de forme. La gestion en est simplifiée. Un inconvénient de ce mode de fonctionnement est peut-être que l'utilisateur perd un peu en confort d'utilisation et en fluidité.

2.2 Explication des composants et leurs interactions

De manière générale, l'utilisateur interagit de façon assez égale avec tous les agents de l'application.

Tout d'abord, nous avons au commencement de l'application la reconnaissance de parole. En effet, une interaction commence par une phrase de l'utilisateur, comme par exemple "Des-

siner un rectangle rouge ici" ou avec des éléments manquants tels que "Dessiner rouge" ou encore "Déplacer". Ces différents prompts sont ensuite traités, mis en forme et envoyés via Ivy avec le préfixe "message : " vers le moteur de fusion qui attend l'arrivée d'une chaîne commençant par ce préfixe.

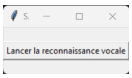


FIGURE 2 – IHM Speech Recognizer

À cet instant, le moteur a reçu la commande et commence à regrouper les informations dont il dispose ou non, sachant que pour effectuer une action sur la palette, cette dernière doit recevoir un mode de fonctionnement (dessiner, déplacer...), une forme, une couleur et une position. En effet, si le moteur ne dispose pas de la couleur, un message est envoyé à la reconnaissance de parole pour demander à l'utilisateur s'il souhaite la préciser, par exemple. Dans le cas contraire, le moteur génère une couleur aléatoire.

Ensuite vient la reconnaissance de forme. Si la forme n'est pas spécifiée, le moteur attend une réponse de la reconnaissance de geste. Lorsque l'utilisateur a dessiné sa forme et que le système l'a reconnue, elle est envoyée au moteur qui peut alors avancer. Reste la position.

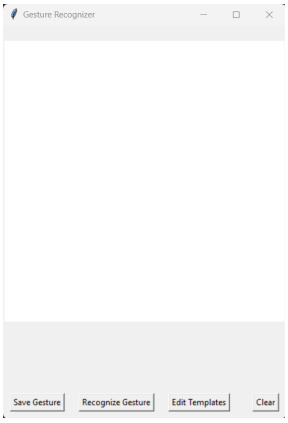


FIGURE 3 – IHM Gesture Recognizer

Dans cette étape, il reste deux options. Soit la position n'est pas renseignée dans le prompt initial, auquel cas le système la détermine lui-même de manière aléatoire, soit l'utilisateur a dit "ici". Dans ce cas, le système attend un clic sur la palette pour récupérer la position de la souris et la transmettre au moteur.

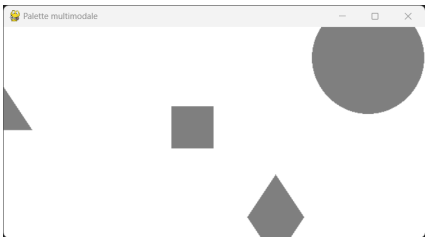


FIGURE 4 – IHM Palette

Le moteur dispose alors de toutes les informations et envoie une trame avec le préfixe "To draw :" contenant toutes les informations pour que la palette les récupère, traite l'information et dessine le nouvel objet. Ceci fait, la palette renvoie un "acquittement" via le message "drawn" vers le moteur de fusion pour que ce dernier puisse réinitialiser sa machine à états.

2.3 Conception des actions

En l'état catuel, l'application est capale de traiter correctement les types de prompts suivants : "*Dessiner* un *texte en italique couleur ici*"
"*Deplacer*"
"*forme* ou *"texte en italique"* (l'utilisateur donne une forme ou une couleur et le système par défaut se met à fonctionner en mode dessin)

Voici le différents chronogrammes associés :

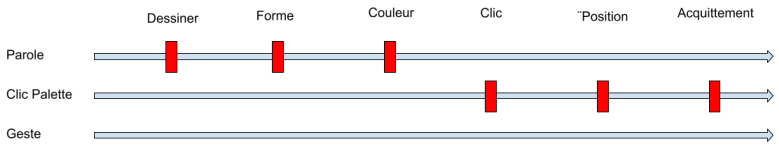


FIGURE 5 – Chronogramme 1

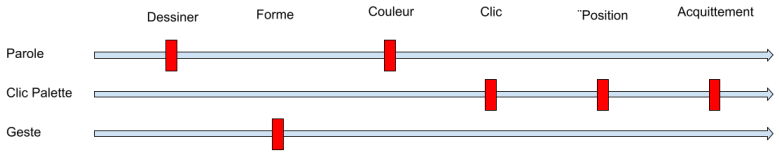


FIGURE 6 – Chronogramme 2

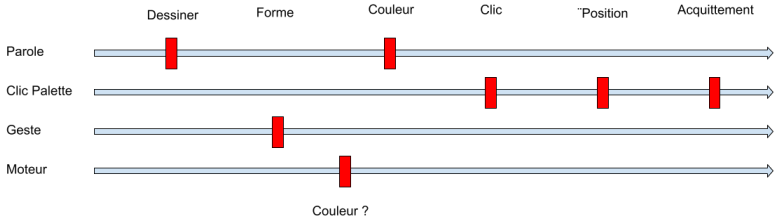


FIGURE 7 – Chronogramme 3

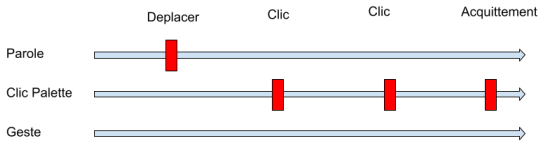


FIGURE 8 – Chronogramme 4

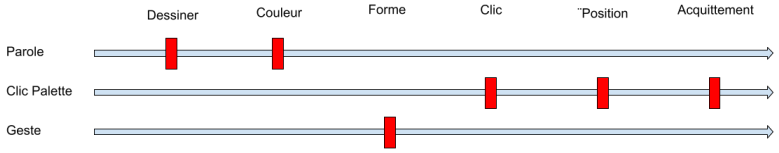


FIGURE 9 – Chronogramme 5

2.4 Detail des modes dessin

2.4.1 Création de Formes

L'utilisateur peut dire "Dessiner" suivi d'un geste pour spécifier le type de forme (rectangle, cercle, triangle, losange) ou donc le spécifier sur la palette de reconnaissance de formes. Des options de couleur peuvent être ajoutées vocalement après la création de la forme.

2.4.2 Déplacer des Formes

Pour déplacer une forme, l'utilisateur doit dire "déplacer" suivi du type de forme. Théoriquement, nous aurions voulu pouvoir ajouter des options de formes et de couleur pour compléter cette commande, mais nous en sommes resté au clic souris par manque de temps.

2.4.3 Autres options

2.4.4 Supprimer des formes

Pour supprimer un item, l'utilisateur doit spécifier "Supprimer" dans l'outil de gestion de la parole puis cliquer sur la forme à supprimer.

2.4.5 Modifier des formes

La modification est sommaire. Elle ne permet pour le moment que de changer la couleur d'une forme. Cependant, nous avons rencontré un souci que nous n'avons pas eu le temps de régler. En effet en local sur la palette la modification marche parfaitement, seulement. En mode multi-modal cette modification pose problème et fait crasher l'appli. C'est très certainement la forme du message trasmettant la couleur qui pose problème, mais nous n'avons pas eu le temps de le modifier.

3 Outils et Environnement

3.1 Langage de programmation et environnement de développement :

Le projet est développé en utilisant le langage de programmation Python. Python a été choisi en raison de sa simplicité, de sa flexibilité et de la disponibilité de nombreuses bibliothèques utiles pour le traitement du langage naturel, la reconnaissance vocale, la manipulation d'interfaces graphiques, et d'autres tâches pertinentes pour le projet. L'environnement de développement utilisé principalement est Visual Studio Code. Le code a été développé à la fois sous windows et Ubuntu.

3.2 Reconnaissance Vocale (librairie SpeechRecognition) :

Le module de reconnaissance vocale est implémenté à l'aide de la bibliothèque SpeechRecognition. Cette bibliothèque offre une interface simple pour interagir avec différents moteurs de reconnaissance vocale, y compris Google Web Speech API utilisé dans cet exemple. La reconnaissance vocale permet à l'utilisateur de donner des instructions verbales pour créer, déplacer ou manipuler des formes sur la palette de dessin. Cette methode nous a permi d'éviter d'avoir à faire nos propres grammaires etc mais est également bien plus performant que ce que nous pourrions produire avec les quelques minutes nécessaires à son implémentation. Elle compte de nombreaux avantages :

- Facilité d'utilisation :

  SpeechRecognition fournit une API simple et facile à utiliser pour intégrer la reconnaissance vocale dans des applications Python. Cela permet aux développeurs d'ajouter rapidement des fonctionnalités de reconnaissance vocale sans avoir à comprendre les détails complexes des moteurs sous-jacents.

- Prise en charge de plusieurs moteurs :

  La bibliothèque prend en charge plusieurs moteurs de reconnaissance vocale, offrant ainsi une flexibilité pour choisir celui qui convient le mieux aux besoins du projet. Cela inclut

des moteurs basés sur le cloud et des moteurs hors ligne.

- Compatibilité avec les services cloud :

  SpeechRecognition offre une prise en charge intégrée de plusieurs services de reconnaissance vocale basés sur le cloud, tels que Google Web Speech API, Sphinx, Wit.ai, etc. Cela permet aux développeurs de tirer parti des puissants moteurs de reconnaissance vocale fournis par ces services.

- Documentation détaillée :

  SpeechRecognition est bien documenté, avec des exemples pratiques qui aident les développeurs à comprendre rapidement comment intégrer la reconnaissance vocale dans leurs projets.

- Gratuit et open source :

  SpeechRecognition est un projet open source, ce qui signifie qu'il est gratuit à utiliser et que les développeurs peuvent examiner, modifier et contribuer au code source.caractères.

- Compatibilité multi-plateforme :

  La bibliothèque fonctionne sur plusieurs plateformes, y compris Windows, macOS et Linux, ce qui la rend polyvalente pour différents types de projets.

3.3 Utilisation du visionneur pour le développement et la simulation des interactions :

Pour faciliter le développement et la simulation des interactions, un visionneur nous a été fourni pour pouvoir visionner les inéractions Ivy. Ce visionneur permet de visualiser les gestes et les commandes vocales détectés par les différents modules du système. Il offre une interface graphique pour suivre et déboguer les interactions entre les différents composants. Cela facilite également le processus de développement en nous permettant de tester et d'ajuster les fonctionnalités de nos scripts en envoyant manuellement certains message par exemple.

4 Validation et tests

4.1 Exemple 1

Dans cette section, nous allons explorer les différents cas d'utilisation de cette application.

Commençons tout d'abord par un cas nominal, qui est assez courant et peu multimodal.

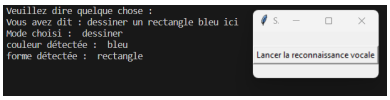


FIGURE 10 – Initialisation de l'intéraction

Dans ce cas, l'utilisateur a prononcé la phrase "dessiner un rectangle bleu ici" de fais, les mots clé "dessiner", "bleu", "rectangle", "ici" vont être transmis au moteur de fusion via une trame de la forme suivante : "message : dessiner rectangle bleu ici".
Dans le cas présent, le moteur de fusion n'a pas beaucoup d'informations à fusionner, une fois la trame reçue le moteur de fusion n'a qu'à attendre la position donnée par un clic de souris sur la palette comme le spécifie le mot "ici".

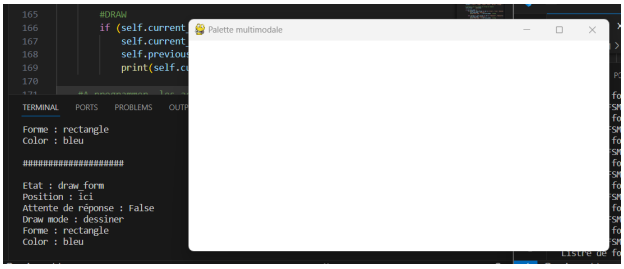


FIGURE 11 – Attente du clic

Comme nous pouvons le voir la machine a etats est dans l'état draw, mais le dessin n'est pas encore apparu sur la palette. Le message attend d'être complété de la position pour être envoyé. Mais syr l'image suivant, la palette a été cliquée. Une methode récupère le clic et la position de la souris au moment du clic pour envoyer le bessage sur le bus Ivy. A cet instant, le moteur de fusion qui attend un message avec la balise "Position :" intercepte ce

message, complète sa variable self.pos. Cette variable contient maintenant la position ce qui permet l'envoi d'un meaasge contenant les informations avec la balise "To draw :" que scrute la palette. Ce message est récupéré par la palette qui dessine le rectangle puis envoie le message "drawn" pour réinitialiser le moteur de fusion.

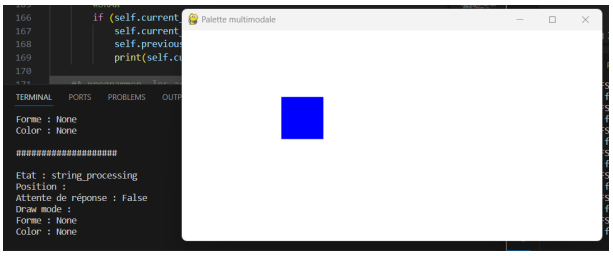


FIGURE 12 – Attente du clic

Maintenant que nous y sommes, nous pouvons tester la fonction de supression. L'utilisateur dit "supprimer" comme le montre l'image suivant :

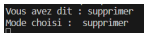


FIGURE 13 – Prompt suppression

Le moteur de fusion se met donc dans l'etat draw attendant le clic comme précédement.

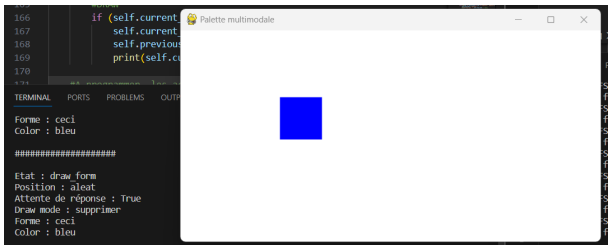


FIGURE 14 – Attente du clic pour supprimer

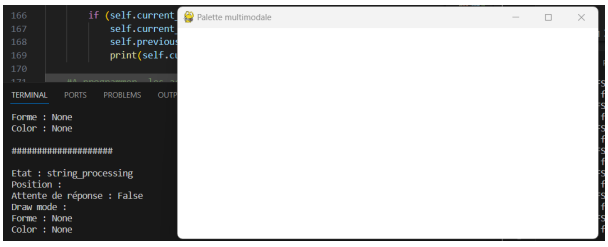


FIGURE 15 – Suppression après le clic

Une fois cliqué, le rectangle est bien supprimé.

4.2 Exemple 2

Dans ce second exemple, nous allons un peu plus mettre en avant la multimodalite. Nous allons partir d'un prompt dans lequel il manque des informations telles que la forme, la couleur ou la position.

Dans un premier temps, l'utilisateur annonce "Deplacer" uniquement, rien de plus.

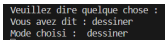


FIGURE 16 – Prompt dessiner

A cet instant, le moteur ne dispose que du mode dessin, "dessiner". Il cherche donc a se procurer la forme comme le montre le résultat de l'affichage de l'état.

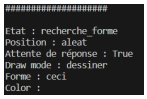


FIGURE 17 – Mae dans l'état recherche de forme

De fait, l'utilisateur décide de renseigner la forme dans le script de reconnaissnace de gestes et fait un cercle

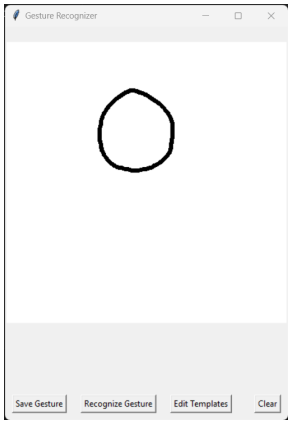


FIGURE 18 – Reconnaisance de geste (dessin)

Une fois qu'il clique sur recognize, le système reconnait un cercle :

figs/Chapitre4/reconnaissenace.png

FIGURE 19 – Reconnaisance de geste (confirmation)

Lorsque l'utilsateur clique sur OK, cela envoie un message de la forme "forme : cercle" qui est récupéré par le moteur de fusion qui s'en saisit. Le moteur cherche maintenant à se procurer la couleur et fait une requette "color needed" à la reconnaissance de parole qui donne le chois à l'utilisateur de donner la couleur ou non. Ici nous avons choisi de donner la couleur rouge :

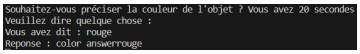


FIGURE 20 – Correction de la couleur

Le système est alors maintenant en mesure de desiner le cercle rouge :

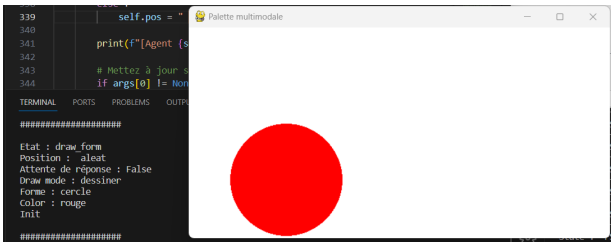


FIGURE 21 – Reconnaisance de geste (confirmation)

Le moteur est maintenant de nouveau dans l'état initial. Il est possible d'essayer de déplacer le cercle. Tout d'abord, l'utilisateur dit "deplacer" :

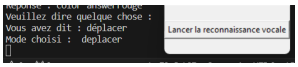


FIGURE 22 – Speech deplacer

Maintenant il reste à clique une première fois sur la forme :

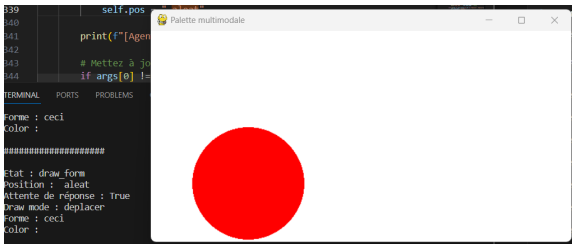


FIGURE 23 – clic 1 deplacement

Pour ensuite cliquer sur la destination :

FIGURE 24 – clic 2 deplacement destination

Ainsi se termine ce second test. Avec ces deux maninipulaitons, il est possible de se rendre conte du fonctionnement de cette application.

5 Conclusion

philo câbles tirés pour l'appli

In conclusion, it can be affirmed that the prospects for Plastic Omnium and the Industry 4.0 team regarding these projects are quite promising.

The robotics projects, focusing on the assembly of plastic clips () and the cleaning of vehicle parts (), exhibited successful strides in terms of trajectory planning, hardware integration, and real-time execution. The rigorous testing and validation processes underscored the feasibility and practicality of implementing cobotic systems within the manufacturing environment. The adaptation of trajectories and programming techniques to optimize task completion time, ensure accuracy, and maintain operational safety was a crucial aspect of these projects. These robotic endeavors demonstrated that carefully orchestrated collaboration between humans and robots can yield tangible efficiency gains while adhering to safety standards.

The artificial vision project also presented remarkable developments in the context of QR code detection and decoding. The iterative refinement of algorithms and techniques showcased the ability to accurately extract information from QR codes even in challenging conditions, which are common in industrial settings. The integration of OpenCV libraries and the exploration of advanced image processing techniques highlighted the potential for automated data extraction from visual inputs. Furthermore, even if this application is not used for its initial purpose, it can be applied to numerous cases of logistics and material management.

Through extensive testing and meticulous analysis, the projects revealed the importance of robustness, repeatability, and adaptability within industrial processes. Addressing real-world challenges such as varying floor conditions, dynamic environments, and quality control necessitated a holistic approach that incorporated technical expertise, collaborative efforts, and thorough validation procedures.

Moreover, the intersection of hardware and software elements underscored the significance of integrating robotics and vision technologies seamlessly into existing workflows. The establishment of communication protocols, safety mechanisms, and coordinated operations emer-

ged as pivotal factors that facilitate the seamless coexistence of human operators and automation systems.

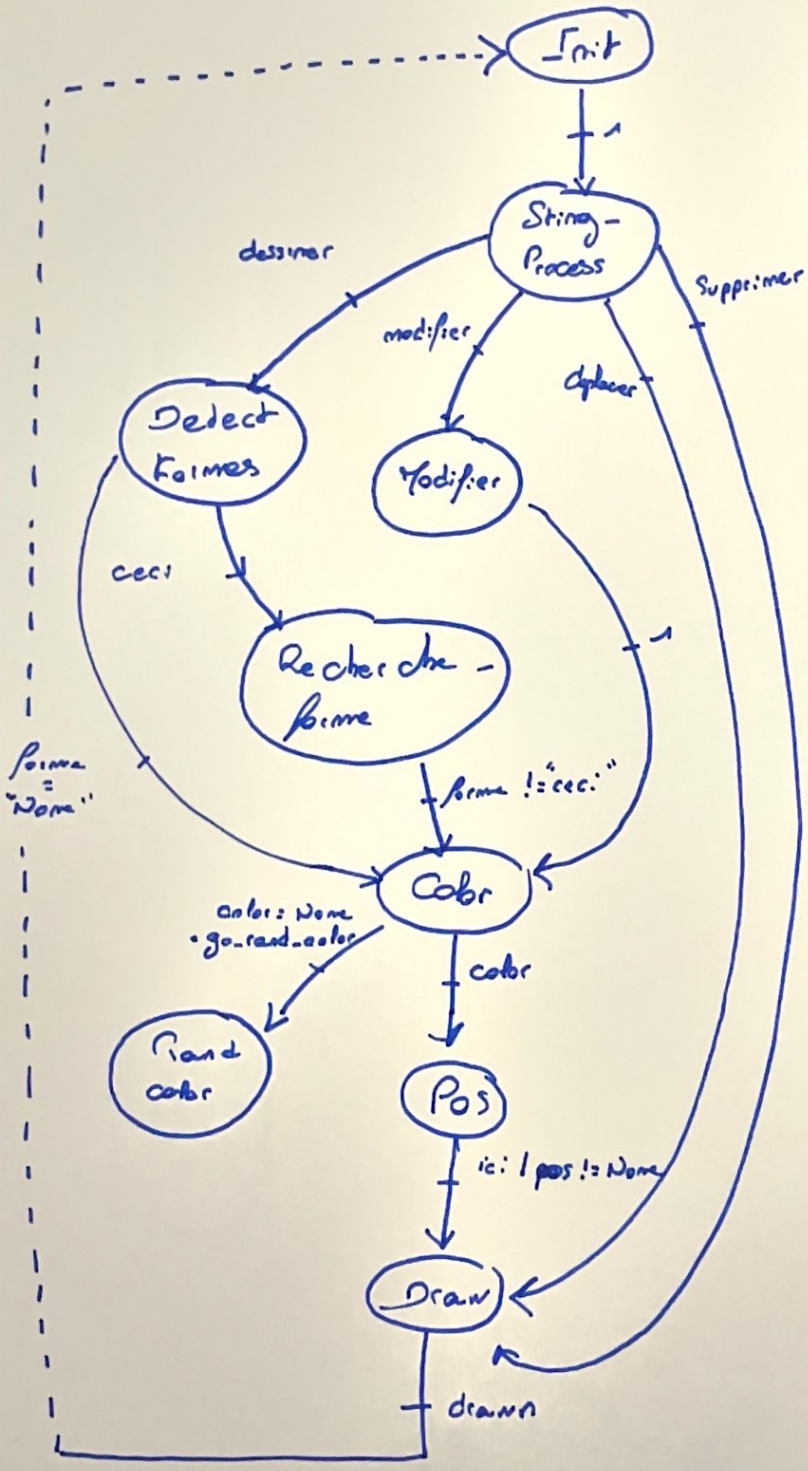
In essence, the convergence of the robotics and artificial vision projects, supported by thorough testing and insightful analysis, has resulted in significant contributions to the field of industrial automation for the Arevalo Factory. These initiatives have highlighted the potential for heightened productivity, efficiency, and quality control through the strategic implementation of cutting-edge technologies. The garnered technical insights not only solidify the progress made but also lay the groundwork for continuous advancements and innovations in the realm of industrial automation. The outcomes of these projects offer promising prospects for Plastic Omnium and its Industry 4.0 team, setting the stage for a future full of new cobotic projects based on the architecture and the knowledge acquired during this project.[**leister2015mq**] [test]

Appendix

A MAE du moteur de fusion

MAE

Moteur de Fusion



Etats:
- INIT
- STRING_PROCESS
- DETECT_FORMES
- RECHERCHE_FORMES
- MODIFIER
- COLOR
- RAND_COLOR
- POS
- DRAW

Evènements:

EXTERNES

- dessiner : Prompt contient "dessiner"
- modifier : Prompt contient "modifier"
- déplacer : Prompt contient "déplacer"
- Supprimer : Prompt contient "Supprimer"
- ceci : la forme n'est pas renseignée dans le prompt donc speech_recognizer envoie "ceci"
- color : la variable color contient la couleur transmise dans le prompt ou générée aléatoirement
- ici : le prompt a transmis "ici" pour être stocker dans la position
- drawn : Lorsque la forme est dessinée, modifiée, supprimée ou toute autre action, la palette renvoie un "acquittement" "drawn".

Internes

- go_rand_color : Si Color est vide, le programme spécifie qu'il faut générer une couleur.

Note: La position aléatoire est gérer dans la palette avec le mot clé "allant"