

# Projet de Bureau d'étude

YOUSRE OUALI & HICHEME BEN GAIED

HE2B: ISIB

54667@etu.he2b.be

## Résumé

*L'objectif de notre projet est de programmer le robot KUKA afin qu'il puisse taper sur un clavier. On a décidé de faire un compte Twitter au robot et lui faire Twitter des phrases. Ces phrases auront pour sujet Pokemon. Il nous dira quelles sont les évolutions des Pokemon.*

**Mots-clefs :** Robot, KUKA, Clavier, Twitter, Pokemon, Souris

## I. INTRODUCTION

Notre projet consiste à programmer un robot KUKA de telle sorte qu'il puisse écrire sur un clavier seul.

### I. KUKA

Dans le cadre de mon projet d'étude, on a développé un système permettant à un bras mécanique KUKA de taper des choses sur un clavier. L'automatisation des tâches est devenue essentielle dans de nombreux domaines, et l'utilisation de bras mécaniques offre de vastes possibilités pour améliorer l'efficacité et la précision des processus.

Kuka est avant tout une entreprise de robotique. C'est une société mondiale d'automatisation. Le siège de la société se trouve en Allemagne.

## II. DESCRIPTION DES DIFFÉRENTES PARTIES DU PROJET

Lors de ce projet, nous sommes passés par plusieurs étapes. Voici une description brève des étapes ;

- Maintenir une connexion avec le robot.
- Configurer ses entrées pour qu'il puisse recevoir des coordonnées.

- Programmer en KRL afin qu'il puisse utiliser ses entrées et procéder à des mouvements.
- Bouger le robot avec les coordonnées envoyées.
- Prendre les coordonnées des touches du clavier.
- Bouger le robot pour qu'il puisse taper sur le clavier juste en lui donnant la phrase à taper.
- Initialiser les étapes à suivre pour que le robot sache où appuyer pour réaliser un post.

## III. MATÉRIEL ET MÉTHODE

Les outils utilisés sont les suivants ;

- Un robot KUKA
- Un ordinateur
- Un clavier mécanique
- Un support pour le clavier

En ce qui concerne le langage utilisé lors de ce projet. On a dû utiliser deux langages de programmation, **Python** et le **KRL**.

Python a été utilisé pour instancier le serveur. Ce serveur permet de maintenir la connexion, recevoir/envoyer des coordonnées, vérifier des coordonnées et séquencer la phrase que le robot doit taper. Les informations que le serveur envoie au robot est sous format **XML**.

Le KRL permet de se connecter au serveur et d'utiliser les informations que le serveur lui envoie. Pour que la connexion se fasse, le programme a besoin d'informations tels que l'adresse IP et le port du serveur. Ces informations se trouvent dans un fichier XML. Ce fichier possède aussi d'autres informations tels que les données en entrée et en sortie du robot.

Et en ce qui concerne le support de clavier, ce support est obligatoire en vue de respecter la zone de travail du robot. En effet, le robot possède une zone de travail. Lorsque le robot est hors de cette zone, le robot est inutilisable. Cette zone permet d'avoir une sécurité. Elle permet en outre d'éviter que le robot soit en contact avec le plan de travail par exemple.

## I. Connexion

En ce qui concerne la connexion. Le robot est connecté en **VLAN**. Donc, il n'est pas directement lié au réseau. Pour pouvoir communiquer avec celui-ci, on est obligé de se connecter au VLAN.

Lorsque la communication entre le serveur et le robot est assurée, on peut passer à l'étape suivante qui consiste à créer un fichier XML. Ce fichier XML contiendra toutes les informations afin d'assurer la connexion. Comme par exemple, l'adresse IP du serveur, le port que le serveur utilise et le protocole utilisé pour communiquer.

Et enfin, lorsque le fichier XML est bien configuré, on peut passer au programme KRL. On a créé un objet **RSI** de type **ST\_ETHERNET**. Cet objet spécifique au langage **KRL** permettra d'initialiser la connexion. Lorsque la connexion a été établie, le robot commence par envoyer le premier fichier XML au serveur. Ce premier fichier contiendra des informations comme les coordonnées, mais aussi un **IPOC**. Cette valeur est composée de chiffres. Elle devra être renvoyée au robot selon un temps imparti. Si cette valeur n'est pas renvoyée dans les plus brefs délais, alors la connexion s'interrompt.

## II. Configuration du fichier XML

Comme expliqué précédemment, le fichier XML permet de configurer la connexion du robot au serveur. Toutefois, ce fichier contient d'autres informations. Voyons voir de plus près l'architecture de ce fichier.

La balise racine de ce fichier se nomme "**ROOT**". Dans la balise racine se trouvent 3 autres balises. Il y a une balise qui se nomme "**CONFIG**", une autre "**SEND**" et la dernière s'appelle "**RECEIVE**".

La balise **CONFIG** va permettre de configurer les paramètres de communication. On va y retrouver 6 autres balises ;

- "**IP\_NUMBER**" contenant l'adresse IP du serveur.
- "**PORT**" permettant d'avoir le port utilisé par le serveur.
- "**PROTOCOL**" qui définit le protocole utilisé pour la communication. On a deux types de protocoles utilisables, on a le droit d'utiliser le protocole **TCP** ou **UDP**.
- "**SENDTYPE**" est l'identifiant du système externe. Cette valeur va être l'identifiant pour chaque paquet reçu au robot.
- "**PROTOCOLLENGTH**" est une balise permettant d'activer la transmission de la longueur de byte du protocole. Elle n'a que deux valeurs possibles ("**ON**" ou "**OFF**").
- Et enfin, la balise "**ONLYSEND**" permet de définir la direction des échanges de données. Elle peut être mise à "**TRUE**", ceci permet au robot d'envoyer des données mais pas d'en recevoir. Tandis que, lorsqu'il est mis à "**FALSE**", alors le système peut envoyer et recevoir des données.

Concernant la balise **PROTOCOLLENGTH**, il a été initialisé à **OFF** et la balise **ONLYSEND** a été initialisé à **FALSE**.

Ensuite, la balise **SEND** permet de définir ce que le robot peut envoyer. Cette balise contient une sous-balise "**ELEMENTS**" qui en contient

aussi une autres sous-balise "**ELEMENT**". La balise **ELEMENT** a des attributs tels que ;

- **TAG** est le nom du tag qui sera généré par le robot.
- **TYPE** est le type de donnée.
- **INDX** est le numéro de sortie de l'objet.
- Et enfin, **UNIT** est l'unité utilisé pour la donnée.

Pour ce projet, j'ai juste besoin des coordonnées du robot. Le système possède déjà un objet préconfiguré qui se nomme "**DEF\_R1st**". C'est dans l'attribut **TAG** où on précise le nom de cette objet. Le type de cette objet est un **DOUBLE**, le numéro de sortie est **INTERNAL** et l'unité est 0.

Remarque, il existe plein d'autres mots-clés comme **DEF\_RSol** (envoie les commandes de positions cartésienne), ou le **DEF\_MACur** (envoie les courants moteur du robot de l'axe A1 à A6).

Et enfin, la balise **RECEIVE** est la balise qui définit la sortie du robot. Cette balise a la même architecture que la balise **SEND**. Toutefois, la balise **ELEMENT** contient un attribut en plus, c'est le **HOLDON**. Ce nouveau attribut va définir le comportement de l'objet reçu au robot lorsque celle-ci à une erreur. Elle a deux valeurs possibles ;

- **0** permet de réinitialisé la valeurs reçu.
- **1** permet de maintenir l'ancienne valeur valide alors que la nouvelle valeur a une

erreur.

Dans ce projet, tous les éléments ont un **HOLDON** à 1 afin d'assurer une localisation toujours valide.

Passons maintenant dans le langage KRL,

### III. Mouvement

HICHEME

### IV. Twitter

HICHEME

### IV. PROBLÈME RENCONTRÉ

HICHEME

### V. RÉSULTAT FINAL

YOUSRI

### VI. AMÉLIORATIONS POSSIBLES

YOUSRI

### RÉFÉRENCES

[Pozyx] 2022

UWB is here to stay

<https://www.pozyx.io/technology/uwb-technology#UWB-is-here-to-stay>