**BASIC APPLICATION FOR CONNECTING ROBODK TO EXTERNAL SENSORS AND PROGRAMMING BASED ON REAL SIGNALS**

**autori: CRISTOIU Cozmin ……**

**ABSTRACT**: Integrarea datelor senzorilor reali în simulările robotizate oferă un pas semnificativ către crearea de soluții de automatizare industrială mai precise și interactive. Această lucrare explorează metodologia de conectare a unei plăci Arduino cu software-ul RoboDK pentru a prelua date în timp real de la senzori și a le utiliza pentru a putea realiza si programa elementele din aplicație pe baza semnalelor reale. Scopul a fost de a realiza o aplicație de bază, de tip șablon, care poate fi utilizata în mod deosebit in scopuri educaționale de către studenți sau doctoranzi, sau chiar în etapele preliminare de automatizare industriala a unei aplicații robotizate. Prin două scripturi personalizate – unul pentru Arduino responsabil cu colectarea datelor de la senzori si transmiterea prin intermediul comunicației seriale către Robodk, si un script Python pentru recepționarea datelor si transformarea acestora in instrucțiuni Robodk astfel incat evenimente din cadrul simularii sa fie executate pe baza semnalelor receptionate. Modelul demo al aplicatiei din RoboDK si cele 2 scripturi vor fi impartasite la liber pe github (link in bibliografie).

1. **INTRODUCERE**

În automatizările industriale robotizate, utilizarea aplicațiilor software de simulare si programare au devenit indispensabile pentru programarea, testarea și optimizarea aplicațiilor înainte de execuția si implementarea lor in lumea reala. RoboDK, este unul dintre instrumentele proeminente din acest domeniu si permite inginerilor și cercetătorilor să simuleze procese din cadrul unor aplicații industriale robotizate într-un mediu simulat. Cu toate astea, acestor simulări le lipsește adesea integrarea dinamică cu date preluate de la senzorii din lumea reală (aceste semnale fiind de asemenea si ele simulate).

Acest studiu propune o abordare nouă pentru a elimina acest neajuns prin integrarea unor semnale reale preluate de la senzori externi in logica funcționarii unei aplicații robotizate. Platforma Arduino oferă o soluție simpla, versatila si ieftina pentru captarea datelor de la senzori externi, cum ar fi temperatura, distanta sau lumina etc., care pot fi transferate în RoboDK pentru interacțiune dinamică în cadrul unei simulări. Această abordare face posibila testarea simulării intr-un grad mai mare de similitudine cu realitatea, fiind posibila testarea întregii logici de funcționare pe baza semnalelor preluate de la senzorii reali. Lucrarea detaliază dezvoltarea unei interfețe simple între Arduino și RoboDK folosind două scripturi personalizate. Lucrarea este un „proof of concept” si un model gata de a fi preluat si dezvoltat/personalizat suplimentar și oferă o bază pentru îmbunătățiri viitoare în dezvoltarea, programarea si simularea aplicațiile industriale robotizate.

1. **MATERIALE SI METODE**

**2.1 Arhitectura Sistemului**

Următoarele trei componente au fost integrate pentru a atinge obiectivul de a executa acțiuni in cadrul simulării care sa fie bazate pe semnale reale:

1. Aplicatia software Robodk: aceasta este un mediu de programare si simulare destinata in mod deosebit roboților industriali si aplicațiilor industriale robotizate. Libraria Robodk include peste 1000 de modele de roboti industriali de la majoritatea brandurilor cunoscute, dar permite si configurarea unor modele noi de roboti. Acesti roboti pot fi programați pentru a executa anumite tascuri sau miscari dar la nivelul simulării virtuale pot fi programate sau lansate instrucțiuni si pentru alte echipamente care se pot afla in cadrul aplicației: sisteme perirobotice, conveioare, AGV-uri, obiecte de toate tipurile, camere video etc.
2. Placa Arduino dotata cu senzori si scriptul de colectare si transmitere a datelor.
3. Scriptul Python care a fost realizat cu scopul de a crea o punte intre placa Arduino si aplicatia RoboDK. Acest script interpretează datele recepționate de la placa Arduino prin comunicația seriala si apoi, prin intermediul API-ului Robodk sunt executate comenzi sau ajutați parametrii din cadrul simulării robotizate.

Momentan, avand in vedere ca aplicatia are doar rol demonstrativ si ca lucrurile sa fie pe cat de clasre si simplu posibile, au fost conectate la intrarile digitale ale placii arduino doar 2 butoane pentru a genera semnale digitale si o fotodioda care sa furnizeze si un semnal analogic. Oricine doreste sa reutilizeze aceasta aplicatie poate adauga oricati senzori dispune (cat ii permite si modelul placii de arduino pe care o detine) si poate crea oricate rutine in Robodk in care se integreze noile semnale.

**2.2 Modelul aplicației din RoboDk**

In modelul virtual al aplicației au fost inserate 2 modele de roboti industriali de tip braț articulat. Pentru a putea avea un feedback vizual clar, in fata fiecărui robot a fost amplasata cate o lampa. Intre cei 2 roboti a fost amplasat un display virtual pe care sa poată fi afișate valori sau stări ale senzorilor si in format text. Fiecăruia dintre cei 2 roboti i-au fost alocate cate 4 puncte ținta si cate un program de parcurgere a unui traseu desemnat ce aceste patru puncte. Acest set-up general poate fi observat in figura 1.

A computer screen shot of a machine

Description automatically generated  
Fig. 1 Modelul virtual al aplicației

Au fost in program realizate sub-rutine care așteaptă valori specifice ale senzorilor si care declanșează executarea mișcărilor de către roboti in funcție de aceste semnale. Unele dintre semnale pot fi setate in prima faza manual cu ajutorul butoanelor din fereastra de dialog prezentata ce apare la lansarea scriptului Python. In figura 2 se poate observa cum din fereastra de dialog unul dintre cele 2 semnale a fost activat, schimbând-se atât culoarea lămpii indicatoare din fata robotului (lampa verde din fata robotului alb) dar si valoarea semnalului de pe displayul virtual.

A screenshot of a computer

Description automatically generated

Fig. 2 Activarea manuala a unui semnal

Cu ajutorul acestei ferestre de dialog minimaliste, putem schimba starea unuia dintre semnalele digitale (momentan sunt afișate doar 2 dar lista acestora poate fi extinsa oricât) in mod manual sau putem comuta pe modul automat. In modul automat, starea semnalelor se schimba automat in funcție de informațiile primite prin comunicația seriala de la placa Arduino. In figura figura 3, putem observa schimbarea lămpii din fata robotului portocaliu, a stării semnalului 2 afișat pe displayul virtual dar si execuția programului de mișcare al robotului portocaliu care a pornit automat la recepția semnalului prin comunicația seriala. De asemenea, pe display poate fi observata si valoarea transmisa de o fotodioda (utilizata pe post de senzor de lumina) conectata la Arduino.

A screenshot of a computer

Description automatically generated

Fig. 3 Schimbarea automata a stării unui semnalului Senzor 2 din RoboDK pe baza informațiilor recepționate prin comunicația seriala de la placa Arduino.

In cazul de fata, pe baza semnalelor digitale recepționate prin comunicația seriala de la unul dintre cele 2 butoane atașate plăcii Arduino, se executa un program prestabilit de mișcare al unuia dintre roboti pentru atingerea punctelor ținta dar unui semnal ii pot fi asociate si alte tipuri de instrucțiuni sau evenimente precum: schimbarea culorii unui obiect, afișarea sau ascunderea unui obiect, schimbarea valorilor unor variabile sau execuția unor scripturi create de noi. Fluxul de informații este prezentat in Figura 4.

A computer with a computer screen

Description automatically generated with medium confidence

Fig. 4 Fluxul de informații

* 1. **Structura elementelor din Robodk pentru simulare**

Pentru a realiza o simulare in Robodk, trebuie creata o stație. Prin stație, se înțelege un ansamblu de elemente care pot cuprinde: modele virtuale 3D, puncte si traiectorii, sisteme de coordonate, mecanisme virtuale, fișiere care conțin instrucțiuni sau programe de lucru pentru roboti si echipamente si scripturi. Aceste elemente trebuie organizate într-o structura logica si sunt afișate sau accesibile din arborele de specificații afișat in partea din stânga unei stații. Pentru simularea noastră, am realizat o stație care conține: a) 2 roboti (fiecare dintre ei cu câteva puncte ținta asociate), b) 7 fișiere care conțin instrucțiuni de mișcare pentru roboti sau instrucțiuni de manipulare a semnalelor c) modelele virtuale pentru 4 lămpi si un display d) un script python. Aceste elemente pot fi observate in arborele de specificații din figura 5.

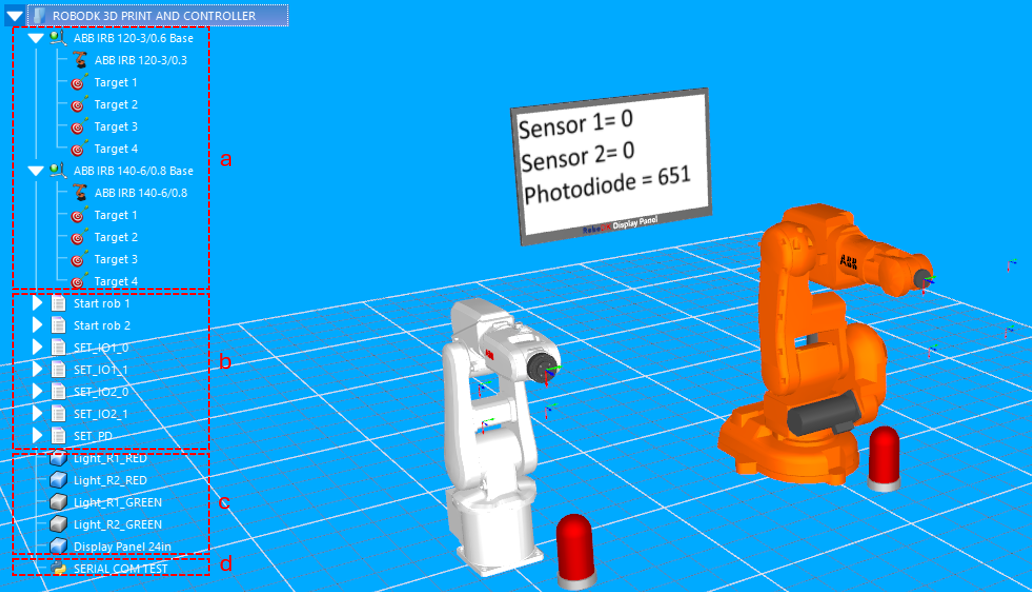


Fig. 5 Structura elementelor din Robodk

Pentru ca un robot sa poata executa un program trebuie ca acesta sa aiba in primul rand asociate un set de puncte tinta. Utilizand aceste puncte tinta, se pot crea programe cu instructiuni de deplasare in aceste puncte. In figura 6 pot fi observate instructiunile de miscare pentru robotul alb (rob 1) de tip MoveJ (move joint) astfel incat acesta sa se deplaseze in bucla intre cele 4 puncte asociate.

A computer screen with a blue background

Description automatically generated

Fig. 6 Programul de mișcare al robotului 1

Astfel de programe pot fi executate ca atare, pot fi folosite pe post de rutine si apelate in cadrul unor programe mai mari, sau pot fi lansate din cadrul unor scripturi. In cazul de fata se poate observa cea de-a doua instrucțiune WAIT IO\_1=1 care așteaptă ca semnalul digital 1 sa devina pozitiv. Execuția programului este suspendata pana când aceasta condiție este îndeplinita. Cu alte cuvinte, mișcarea robotului va începe când semnalul devine 1. In mod normal, schimbarea stării acestor semnale se poate face creând programe care sa cuprindă instrucțiuni de tip „SET\_IO”, așa cum se observa in figura 7, dar in aceasta maniera acest semnal este doar unul simulat. Se schimba doar o variabila/un parametru asociat stației fără ca valoarea acestuia sa provină de la un senzor sau echipament real.

A blue grid with a wire and a blue background

Description automatically generated with medium confidence

Fig. 7 Schimbarea stării unui semnal in simulare

Pentru a fi ușor de urmărit starea sau valorile unor semnale, in stație au fost adăugate si modelele virtuale ale unui display peste care au fost suprapuse linii de text si 4 lămpi, suprapuse 2 cate 2. Textul afișat pe display poate fi schimbat programatic dar pentru efect vizual, s-a folosit un workaround prin care jonglam programatic cu optiunile show/hide ale obiectelor. Doua lampi sunt de culoare rosie si doua lampi sunt de culoare verde. Tot programatic, in functie de starea semnalelor 1 sau 2 se apeleaza instructiuni de ascundere/vizibilitate a lampilor rosii sau verzi.

Pentru simulare, asemnalele de care vorbesc trebuie sa fie definite ca parametrii ai stației. Acești parametrii sunt de fapt doar simple variabile care au valori constante. Aceste valori pot fi setate chiar in momentul definirii parametrilor sau pot fi schimbate utilizând instrucțiuni de tip SET\_IO ca in cazul precedent. Cei 3 parametri demonstrativi utilizați in aceasta aplicație sunt prezentați in figura 8.

A computer screen shot of a machine

Description automatically generated

Fig. 8 Parametri stației

Faptul ca valorile acestor parametrii sunt constante si nu pot fi modificate interactiv, împiedica realizarea unei simulări realiste, bazata pe semnale provenite de la echipamente reale in urma interacționării lor normale. Aici intervine necesitatea dezvoltării scriptului in Python care face legătura dintre senzorii adevărați si instrucțiunile de schimbare a valorilor unor semnale din simulare.

* 1. **Scriptul Python**

Scriptul din Python are nevoie de urmatoarele librarii:

1. TKINTER – pentru elementele de interfata grafica / ferestre de dialog;
2. Serial – pentru preluarea datelor prin comunicatie seriala de la Arduino;
3. Threading – used to start a thread responsible for continuously readingdata from Arduino;
4. Robolink – used to control and interact with Robodk

The logic diagram of the script is presented in figure 9.

A diagram of a program

Description automatically generated

Fig. 8 Diagrama logica a scriptului Python

Comunicatia seriala se realizeaza pe unul dintre porturile la care este conectata placa Arduino (in cazul nostru portul 4) cu un baudrate comun ajustabil de 9600 bits per second. Pentru a putea utilza informatiile setate in fereastra grafica de dialog si pentru a procesa datele au fost create urmatoarele functii:

1. o functie pentru schimbarea valorilor asociate semnalelor din fereastra de dialog si pentru schimbarea culorii campurilor care afiseaza daca unul dintre semnale este activ sau nu.
2. o functie pentru citirea datelor provenite pe comunicatia seriala de la arduino, date care sunt de fapt siruri de caractere de forma „I01\_0”, „I02\_0”, „I01\_1”, „I02\_1” sau „PD\_value” care sunt asociate de fapt butoanelor atasate intrarilor digitale de la placa arduino sau fotodioda atasata uneia dintre intrarile analogice. Aceasta functie proceseaza informatiile si executa aproi din Robodk anumite programe specifice sau actualizeaza constant valoarea unor parametrii.
3. o functie de schimare a modului de functionare: manual / automat. In modul manual semnalele pot fi alterata apasand pe butoanele ferestrei de dialog. In modul automat, semnalele sunt alterate in functie de informatiile primite pe comunicatia seriala.

Codul este unul relativ simplu, care pe lângă librăriile incluse cuprinde doar 112 linii de cod. Totul a fost creat, doar ca un schelet de baza peste care oricine poate construi oricâte elemente suplimentare necesita. Scopul a fost de îndeplinire a unui singur obiectiv si anume de a putea interacționa cu comenzi sau evenimente din cadrul unei stații Robodk pe baza unor semnale provenite de la echipamente din exterior. Aceasta capabilitate este una de general si poate fi utilizata pentru a configura orice fel de simulari si aplicatii robotizate in Robodk.

* 1. **Conditii de functionare**

In primul rând trebuie ca stația din RoboDK sa fie configurata complet, incluzând programele care ar trebui sa fie apelate la nevoie. Scriptul Python trebuie adaugat in stația Robodk ca element component al proiectului. Plăcută Arduino (la care sunt conectati senzorii) trebuie sa fie conectat la portul de comunicație definit in cod iar viteza de transmitere a datelor trebuie sa fie aceeași (atat la trimitere de la Arduino către calculator, cat si la recepționare in codul Python).

Denumirile variabilelor care stochează valorile senzorilor in Arduino trebuie sa fie identice cu formatul care trebuie sa fie interpretat in scriptul Python. De exemplu, in aceasta aplicație semnalele de la arduino sunt notate cu denumiri precum: IO1\_value, IO2\_value, PD\_value. Unde IO1, IO2 sunt denumirile (input/output) senzorilor sau butoanelor atasate unora dintre pinii digitali ai plăcii arduino, si PD este denumirea fotodiodei atașata unui pin analogic. După simbolul „ \_” este valoarea efectiva a senzorilor. Pentru semnalele digitale 0 sau 1 sau pentru semnalele analogice o valoare de tip întreg. Este important protocolul de denumire al acestor semnale pentru ca apoi in partea de receptie (din scriptul Python) se caută exact acest format de scriere. Informațiile recepționate (sub forma unor stringuri de caractere) sunt împărțite in funcție de locația simbolului „\_” si se extrag denumirea senzorului si valoarea acestuia. Aceste date sunt apoi trimise mai departe către Robodk pentru a executa anumite programe sau instrucțiuni.

In Robodk trebuie configurate apoi programe care sa includă instrucțiuni de tip „WAIT\_IO”, care așteaptă recepția semnalelor specifice astfel încât apoi sa lanseze execuția unor instrucțiuni. Suplimentar, monitorizarea comunicației seriale din IDE-ul Arduino, trebuie neapărat sa fie oprita altfel la rularea scriptului Python vom recepționa o eroare care ne spune ca nu poate fi pornita comunicația cu placa Arduino pentru ca este deja ocupata.

1. **Rezultate**

In aceasta configuratie importanta este durata de timp care trece de la transmiterea semnalelor de la Arduino pana la executia instructiunilor din Robodk. Aceasta durata este formata din doua componente: timpul care reflecta viteza de comunicatie dintre Arduino si computer si timpul necear pentru ca actiunile corespunzatoare sa fie executate in Robodk de la trimiterea datelor de la Arduino. Aceasta durata totala de timp, de la transmiterea datelor de la arduino pana la procesarea si executia instructiunilor specifice in Robodk poarta denumirea de „end-to-end latency”. Aceast parametru reflecta intocmai capacitatea sistemului (Arduino + Python + RoboDK) sa finalizeze anumite actiuni bazate pe sensors inputs. Aceasta durata de timp poate fi influentata de factori precum: viteza setata pentru transmisia datelor, performantele procesorului sau eficienta algoritmului sau codului scris. Avand in vedere modelul de placuta Arduino Uno, a fost setata viteza maxima la care poate acest model de placa sa trimita date si anume un baudrate de 115200. Computerul pe care este rulat scriptul Python si aplicatia RoboDK este un laptop uzual cu un procesor Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 1.99 GHz.

Pentru a analiza promptitudinea sistemului, acesta a fost lasat sa functioneze aproximativ 60 de minute, inregistranduse timestamp-ul si end-to-end latency la fiecare schimbare de semnale de-a lungul acestei ore. Fisierul numit latency\_log.txt poate fi gasit in repo-uri alaturi de celalte fisiere. Acesta contine in total 4126 de inregistrari. Facand media acestor durate de timp s-a obtinut o valoare medie de 23,97 ms. In figura urmatoare poate fi observat graficul evolutiei latentei in perioada in care s-au facut inregistrari.

1. **Concluzii**

**AKNOWLEDGMENT   
This research was funded by National University of Science and Technology “Politehnica” Bucharest, grant “GNAC ARUT 2023” contract no. 116/4/12/2023**

1. **Bibliografie**