Robotică

Titular curs si laborator:

Conf.dr.ing. Sanda Victorinne PATURCA

Competenţe - Obiective disciplina

- **Competenţe profesionale:**

Dezvolta cunoștințe despre diferite structuri de roboti, aplicabilitatile acestora, sarcinile de indeplinit, spațiul de lucru și interactiunea cu operatorul sau alti roboti

- Curs introduce studenților elementele de bază ale modelării, proiectării, planificării și controlului sistemelor robotizate.
- Curs acoperă principiile fundamentale ale manipulatorului și roboticii mobile. Subiectele care urmează să fie acoperite includ proiectarea controlului, actuatoare, senzori și software încorporat.
- Studentii vor afla despre o varietate de platforme robotizate, aplicațiile și utilizările lor. Vor învăța despre algoritmii de percepție senzorială și controlul robotului prin feedback senzorial.
- Acest curs va oferi experiență practică cu microcontrolere, servo drive-uri, control în timp real și software încorporat.
- Studentii vor aplica notiunile teoretice pentru a proiecta și programa sisteme robotizate in aplicatii de inginerie electrica.

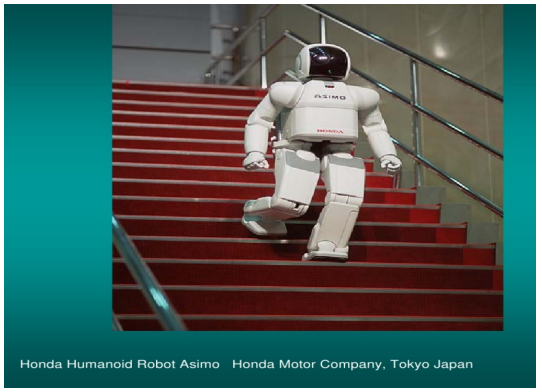
Notiuni de baza

- Ce sunt robotii? Unde sunt folositi?
- Cum functioneaza? Arhitecturi de principiu.
- Aplicatii azi; prognoze pentru viitor
- Principalele provocari pentru proiectarea/ construirea robotilor

Robotica – domeniu multidisciplinar, cu aplicabilitati multiple

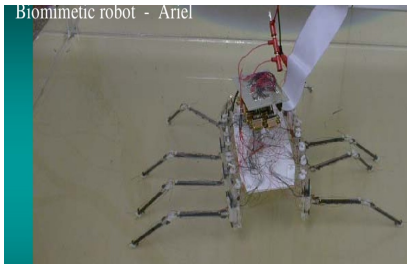
Roboţi industriali FANUC

Image credit: Biorobotics Laboratory



Credit: Biorobotics Laboratory, Carnegie Mellon University

Robot Nao



Directii de baza

- Intelegerea arhitecturilor a robotilor
- Intelegerea modului de interfatare cu exteriorul a robotilor prin diverse tipuri de senzori
- Cunoasterea structurilor de comenzda multiax distribuite folosite pentru robotii avansati
- Prezentarea algoritmilor pentru controlul deplasarii pe mai multe axe si controlul manipularii

Conţinut de baza - extras

- Percepție și senzoristica în mobilitate și manipulare robotică.
- Structura şi sarcinile sistemelor robotice.
- Sistemul de comandă/conducere pentru robot. Comanda distribuită: sistemul de comanda, funcţii şi structura tipică; Sisteme de operare în timp real; tipuri de procesoare folosite pentru sistemul de comandă
- Tehnici de urmărire (reglare) a mişcării roboţilor. Tehnici de reglare mişcarii si actionare, descentralizată sau centralizata.
- Modelare cinematică pentru structuri de robot braţ articulat. Algoritmi pentru controlul manipulării - generare şi urmărire traiectorii.
- Tehnici de generarea de trasee, planificare și navigație, evitarea obstacolelor pentru robotii mobili. Implementare de algoritmi pentru citirea si prelucrarea informatiei de la senzori, controlul deplasari.

Conţinut de baza - extras

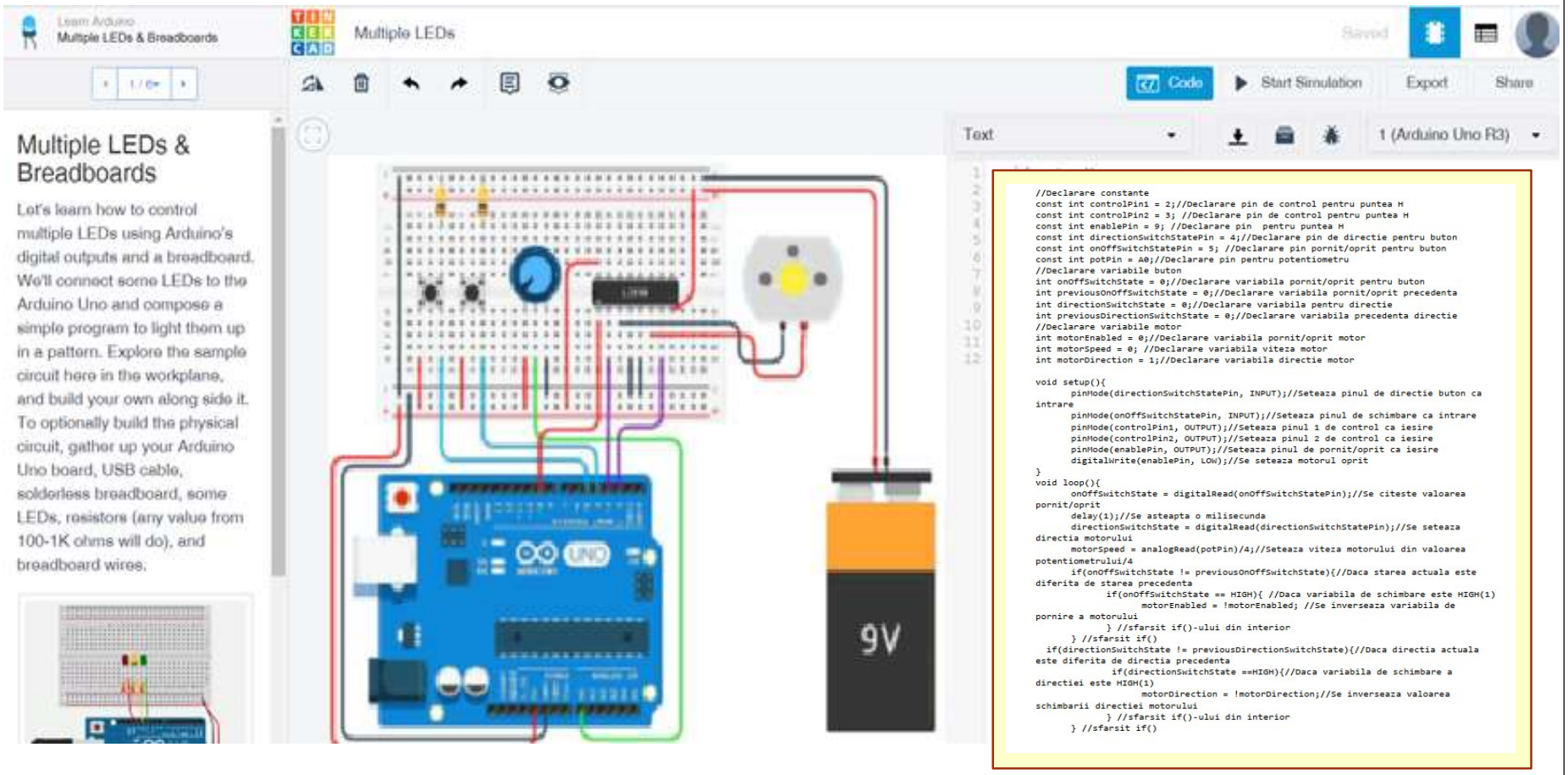
Seminar/laborator

- Familiarizare cu platforme didactice experimentale: structuri de roboti braţ articulat, structuri modulare reconfigurabile de roboţi, structuri de robot cu gripper.
- Aplicatii folosind instrumente de dezvoltare si programare, precum: LabVIEW-Robotics (National Instruments), MPLABX (Microhip), Matlab, si medii free: SimulIDE, TinkerCAD (Autodesk), MIT App Inventor.
- Configurare şi testare mecanică a platformelor experimentale reconfigurabile: alegerea axelor de mişcare, plasarea senzorilor şi a elementelor de execuţie.
- Testare cu senzori interni şi externi pentru sistemul robotic (exemple: senzori de mişcare, lumină, contact, ultrasunete, IR, s.a).
- Testare folosind brate robotice manipulatoare.
- Testare pe structuri de roboti mobili utilitari, autonomi, destinați dezvoltării de aplicații software orientate către explorare şi orientare.

Anexă: extras din conținutul cursului și al laboratorului disciplinei

Exemple

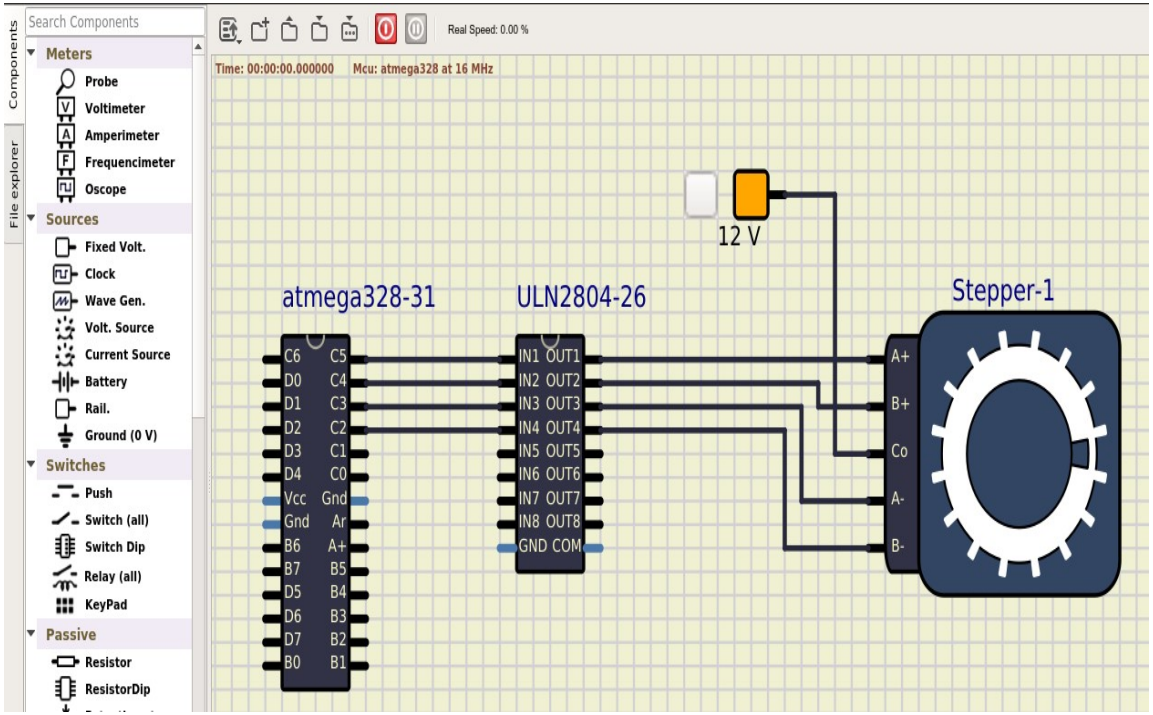
Mediul de simulare si programare **TinkerCAD**




```
//Declarare constante
const int controlPin1 = 2;//Declarare pin de control pentru puntea H
const int controlPin2 = 3; //Declarare pin de control pentru puntea H
const int enablePin = 9; //Declarare pin  pentru puntea H
const int directionSwitchStatePin = 4;//Declarare pin de directie pentru buton
const int onOffSwitchStatePin = 5; //Declarare pin pornit/oprit pentru buton
const int potPin = A0;//Declarare pin pentru potentiometru
//Declarare variabile buton
int onOffSwitchState = 0;//Declarare variabila pornit/oprit pentru buton
int previousOnOffSwitchState = 0;//Declarare variabila pornit/oprit precedenta
int directionSwitchState = 0;//Declarare variabila pentru directie
int previousDirectionSwitchState = 0;//Declarare variabila precedenta directie
//Declarare variabile motor
int motorEnabled = 0;//Declarare variabila pornit/oprit motor
int motorSpeed = 0; //Declarare variabila viteza motor
int motorDirection = 1;//Declarare variabila directie motor

void setup(){
      pinMode(directionSwitchStatePin, INPUT);//Seteaza pinul de directie buton ca
intrare
      pinMode(onOffSwitchStatePin, INPUT);//Seteaza pinul de schimbare ca intrare
      pinMode(controlPin1, OUTPUT);//Seteaza pinul 1 de control ca iesire
      pinMode(controlPin2, OUTPUT);//Seteaza pinul 2 de control ca iesire
      pinMode(enablePin, OUTPUT);//Seteaza pinul de pornit/oprit ca iesire
      digitalWrite(enablePin, LOW);//Se seteaza motorul oprit
}
void loop(){
      onOffSwitchState = digitalRead(onOffSwitchStatePin);//Se citeste valoarea
pornit/oprit
      delay(1);//Se asteapta o milisecunda
      directionSwitchState = digitalRead(directionSwitchStatePin);//Se seteaza
directia motorului
      motorSpeed = analogRead(potPin)/4;//Seteaza viteza motorului din valoarea
potentiometrului/4
      if(onOffSwitchState != previousOnOffSwitchState){//Daca starea actuala este
diferita de starea precedenta
            if(onOffSwitchState == HIGH){ //Daca variabila de schimbare este HIGH(1)
                  motorEnabled = !motorEnabled; //Se inverseaza variabila de
pornire a motorului
            } //sfarsit if()-ului din interior
      } //sfarsit if()
  if(directionSwitchState != previousDirectionSwitchState){//Daca directia actuala
este diferita de directia precedenta
            if(directionSwitchState ==HIGH){//Daca variabila de schimbare a
directiei este HIGH(1)
                  motorDirection = !motorDirection;//Se inverseaza valoarea
schimbarii directiei motorului
            } //sfarsit if()-ului din interior
      } //sfarsit if()
```

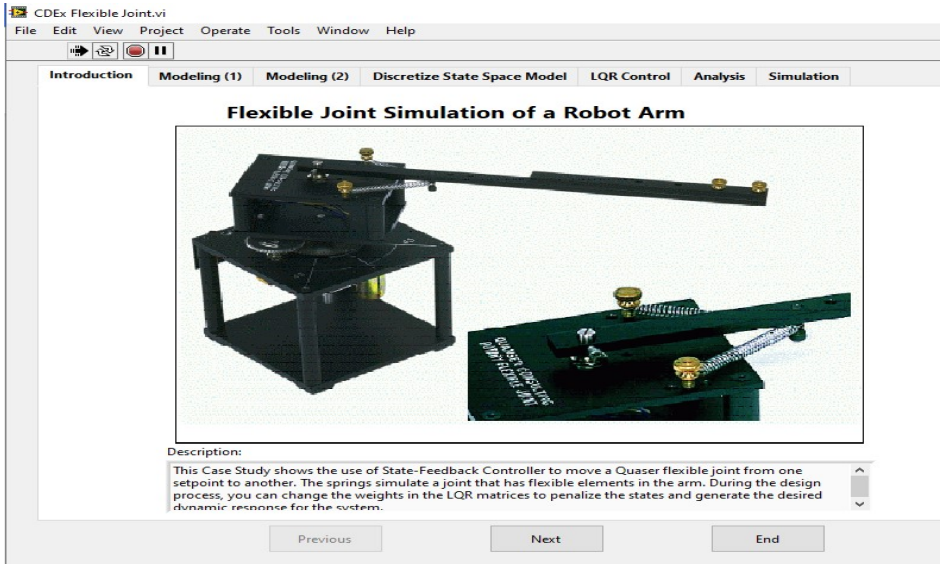
Exemple

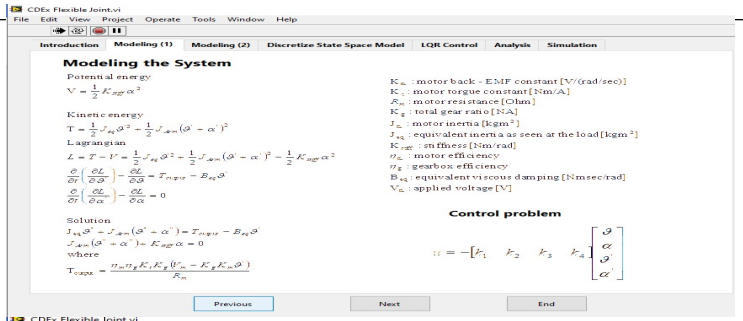
Mediul de simulare si programare **SimulIDE**

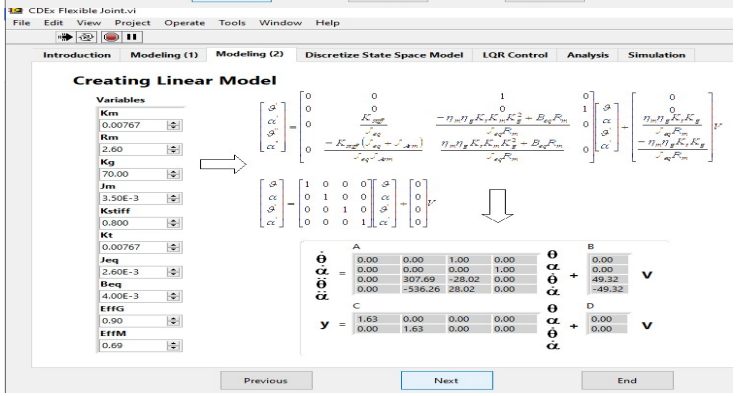


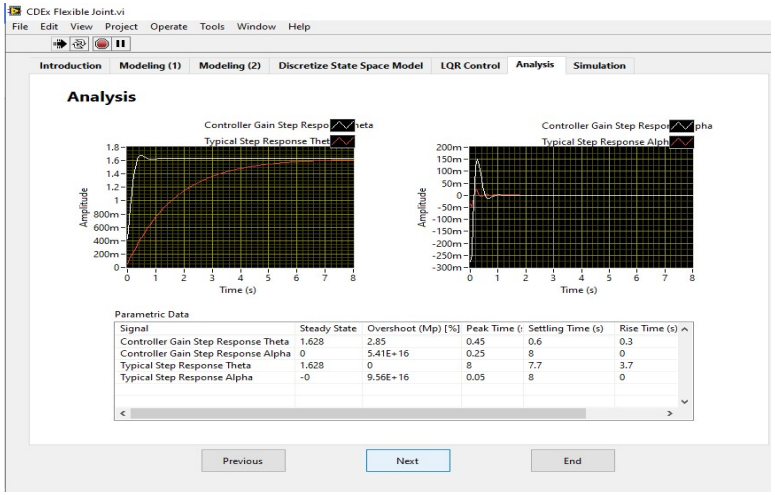

```
 * File:   main.c
 * Author: SP
 *
 * Created on April 27, 2021, 11:04 AM
 */


#include <avr/io.h>
#include <util/delay.h>

#define F_CPU 1000000UL

#define N_PASI              8
#define MASK_TURN_LOW_CMDS  0xC3
#define STEP_TIME   16*500  // inmultit cu 16 pentru ca sim

unsigned char tabelaComutatie[N_PASI] = {
                    0x20,   // A+       C5
                    0x30,   // A+ B+    C5, C4
                    0x10,   // B+       C4
                    0x18,   // B+ A-    C4, C3
                    0x08,   // A-       C3
                    0x0C,   // A- B-    C3, C2
                    0x04,   // B-       C2
                    0x24,   // B- A+    C2, C5
                  };
short k = 0;
//char inc = 1;

int main(void) {
    // setup
    // =====================================================
    // setam directia iesire pt pinii C2...C5
    DDRC |= 0x3C;


    // loop
    // =====================================================
```

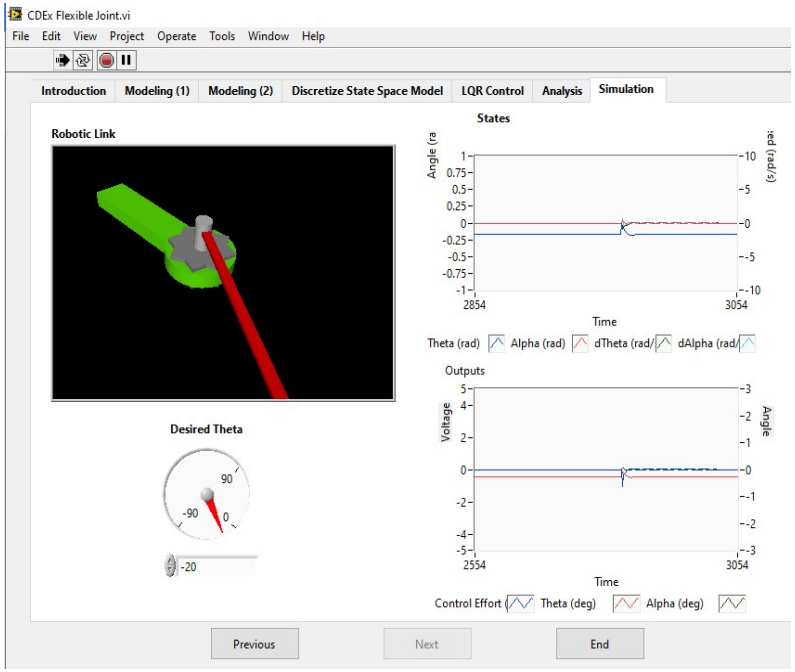
Exemple in simulator



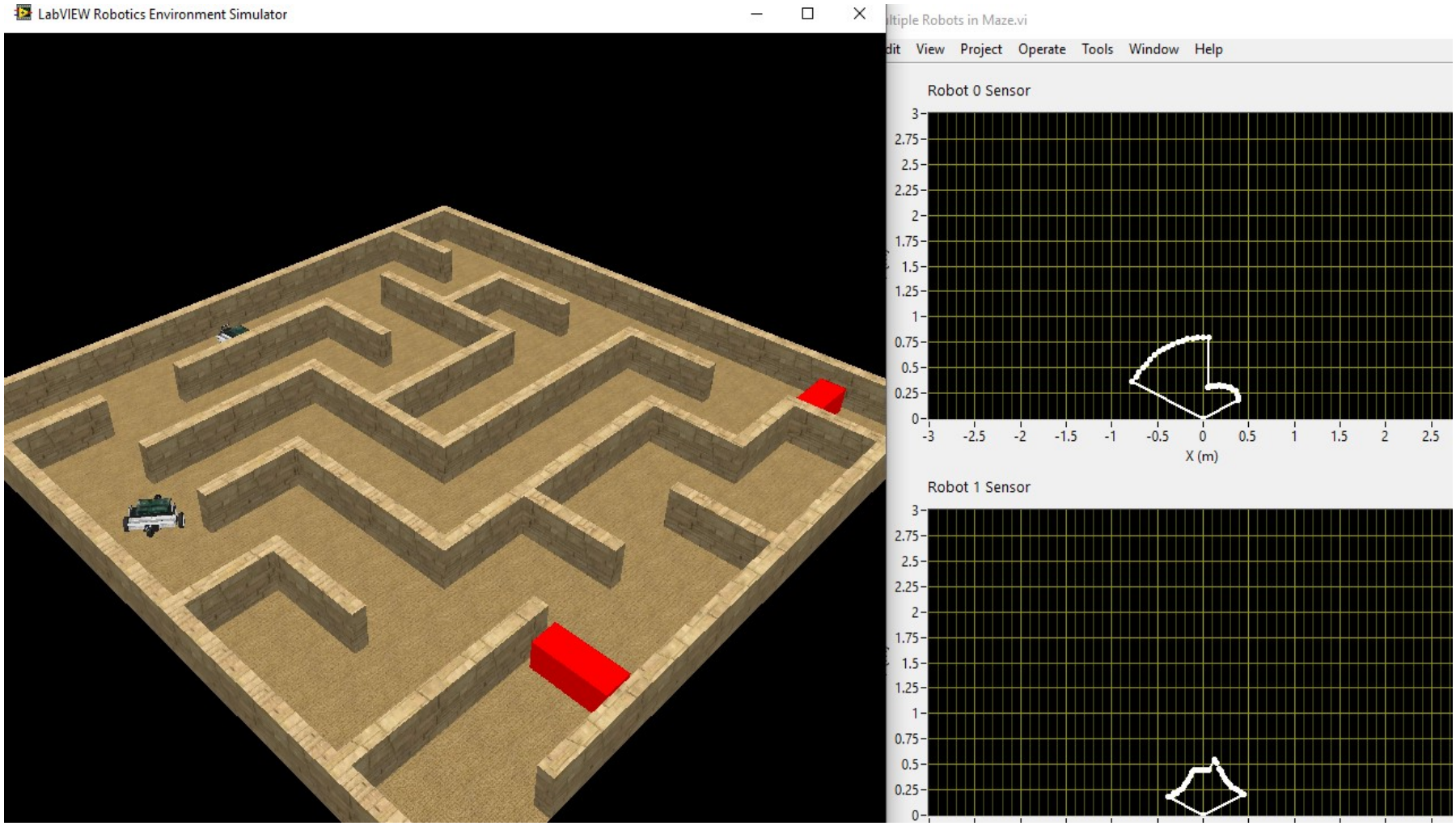




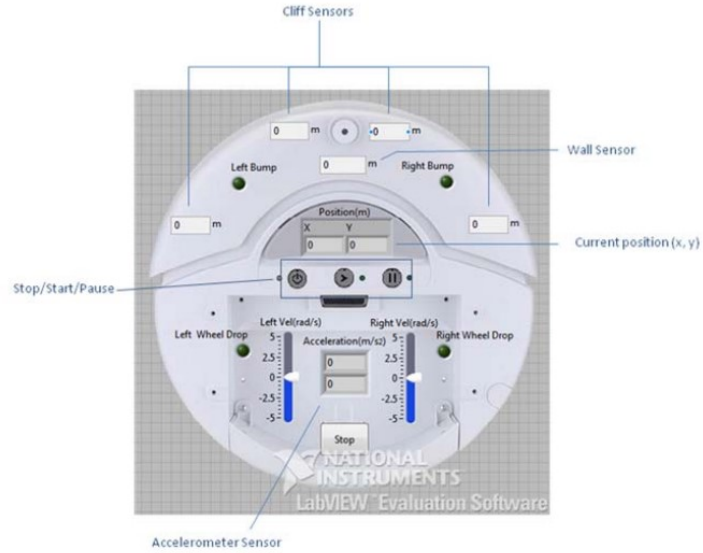


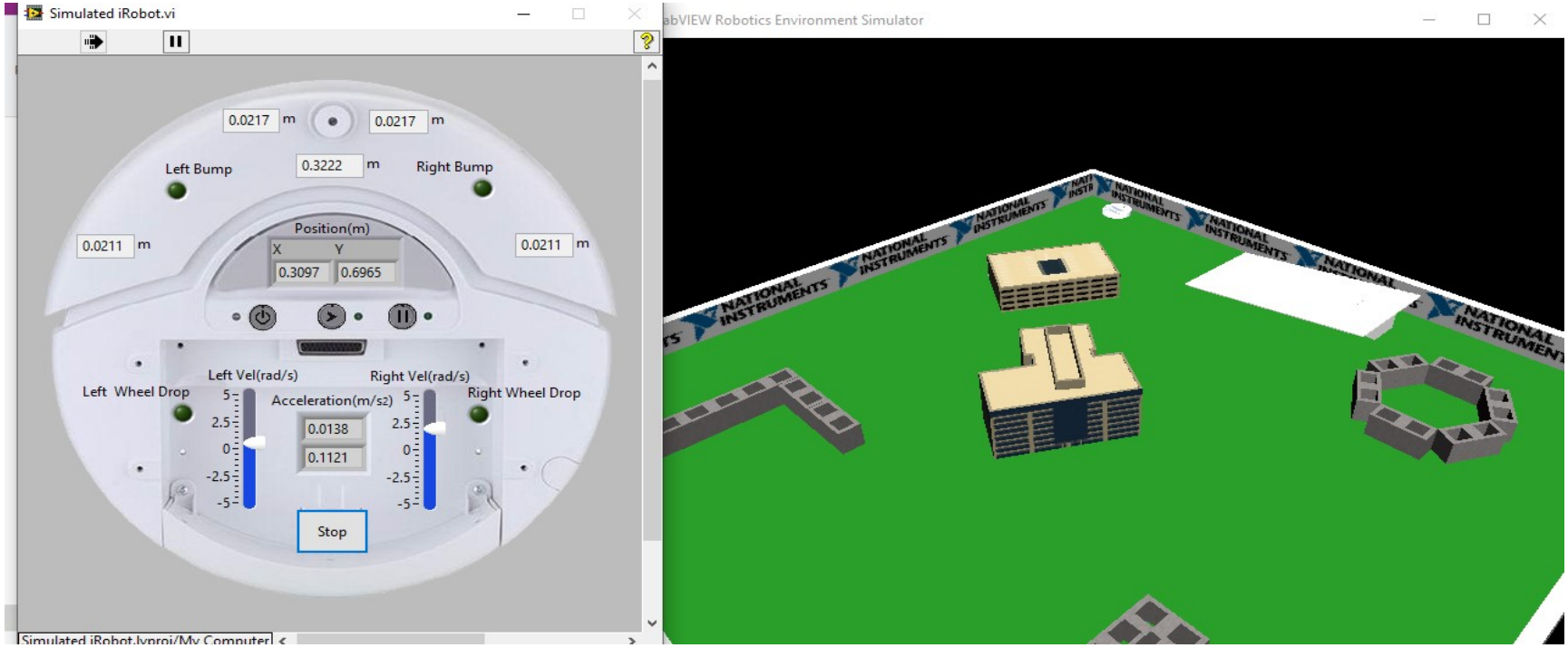


Exemple in simulator



Exemple in simulator





Structuri disponibile in laborator

- roboti reconfigurabili, roboti mobili, roboţi educationali K-TEAM, structuri de roboţi configurabili de tip NXT, EV3, braţe robotice KSR10 Velleman.
- diverse tipuri de senzori: ultrasonici, tactili, PIR, de temperatura, de sunet, fotoelectrici, s.a.; si module WIFI si Bluetooth;
- module programabile si kit-uri de dezvoltare Microchip şi TI, kit-uri ATmega2560, pentru dezvoltare de structuri robotice.

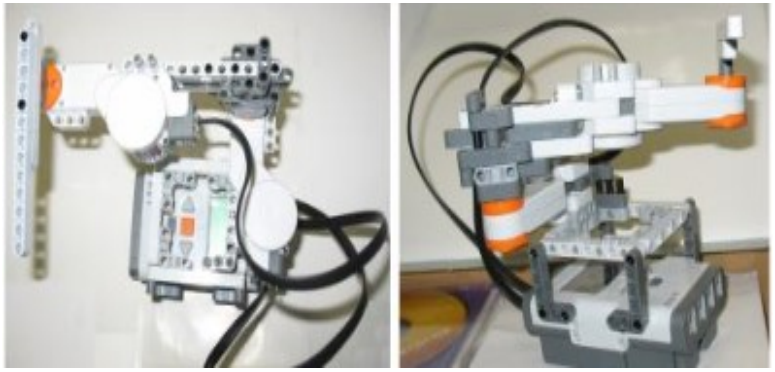
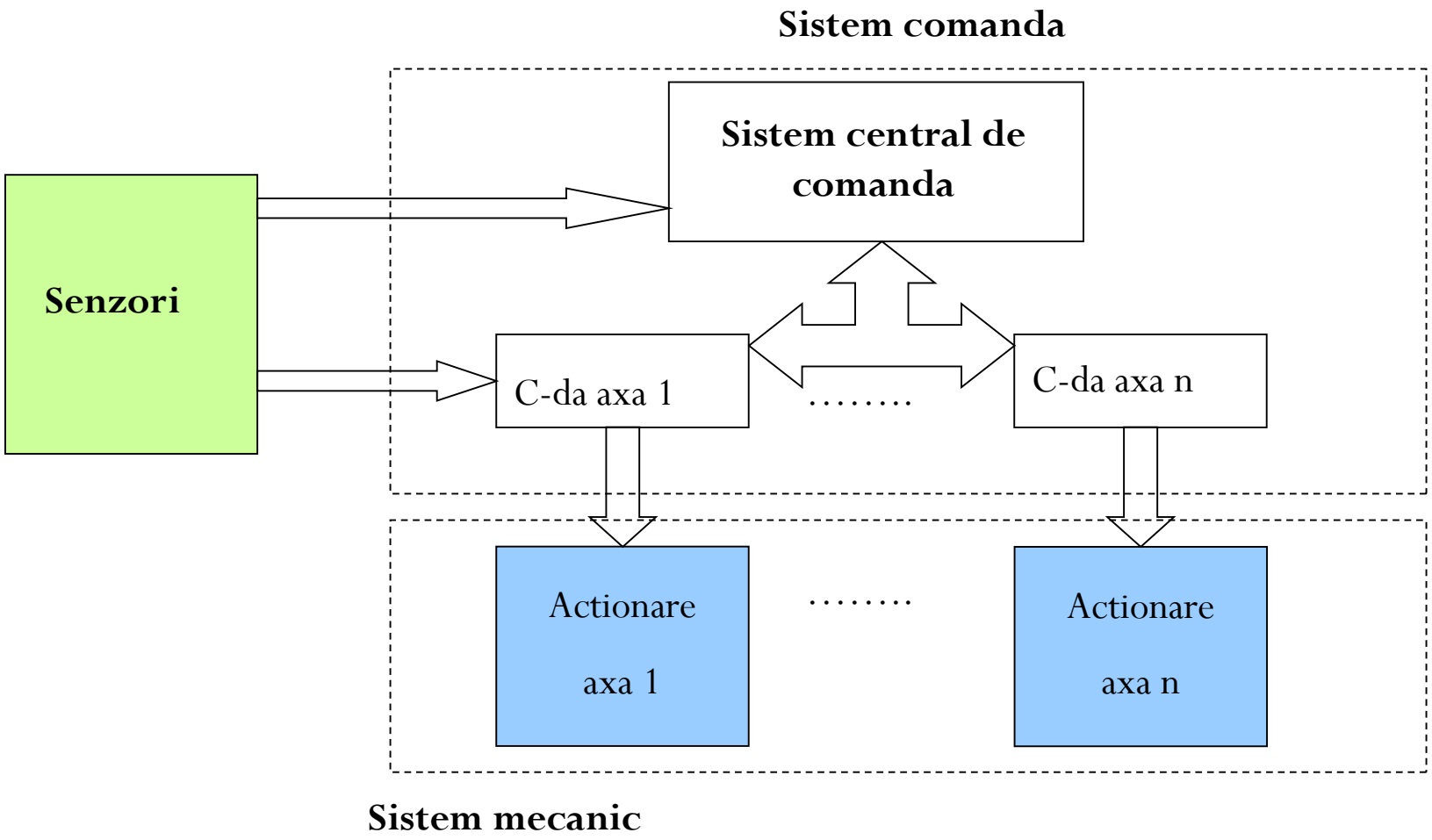


Fig. Robot educational NXT

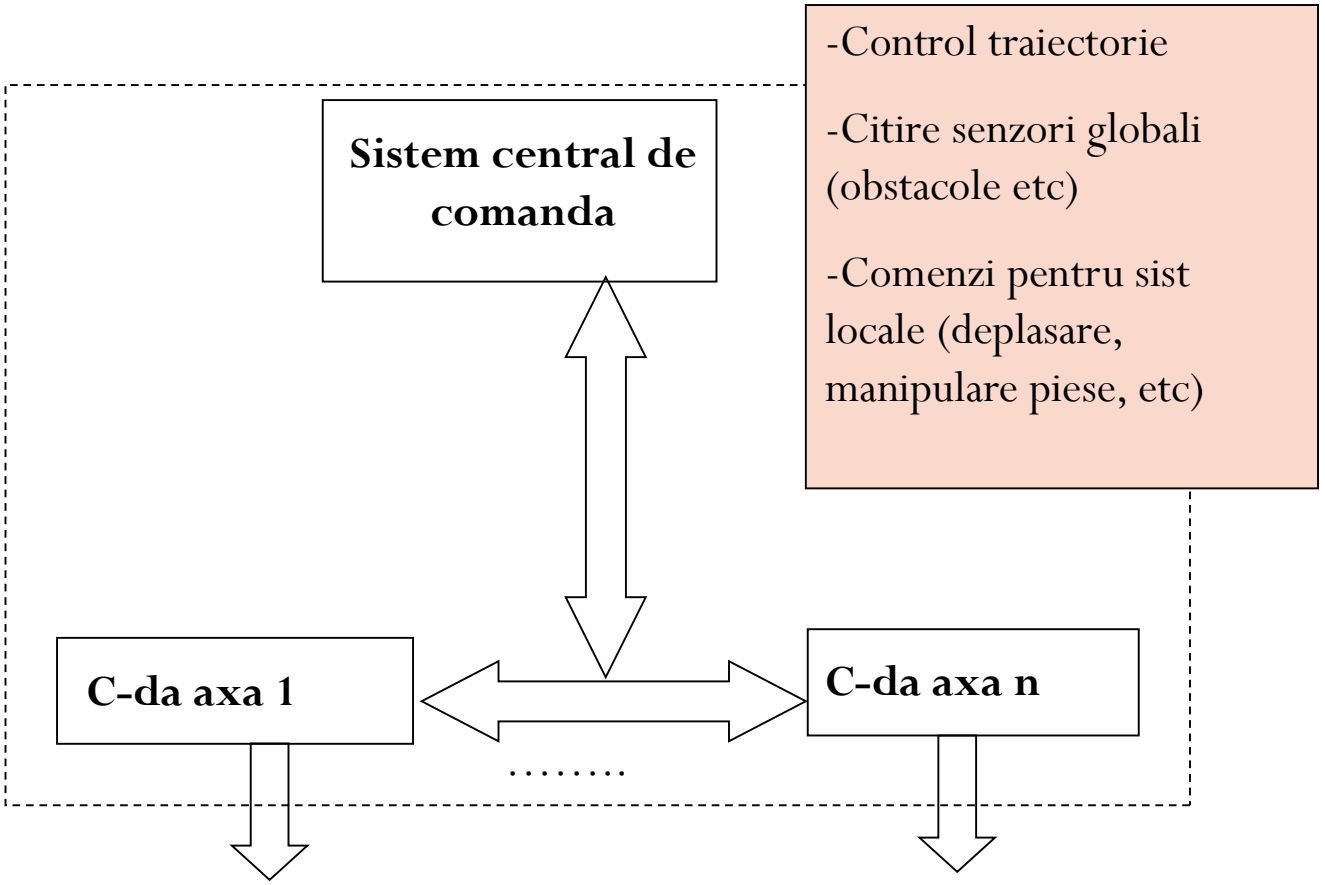
Fig. Robot educational K-TEAM

Fig. Brat robotic cu 6 GDL

Arhitectura de principiu



Impartirea functionala a comenzii



Kit robot educational

Componentele kit-ului robot educational NXT, alcătuiesc o structură tipică de robot autonom, compusă din unitate centrală îmbarcată pe platformă (on-board), sistem senzorial, elemente de execuţie şi elemente mecanice.

Robotul dispune de unitatea de procesare încorporată :

- unitatea de procesare on-board are acces direct la hardware-ul robotului, în timp ce unităţile off-board trebuie să apeleze la un anumit protocol care să asigure interfaţarea între componente şi unitate;
- lipsa întârzierilor asociate comunicaţiei între o unitate de procesare externă robotului şi componentele hardware ale acestuia;
- autonomia fizică faţă de PC-ul gazdă.

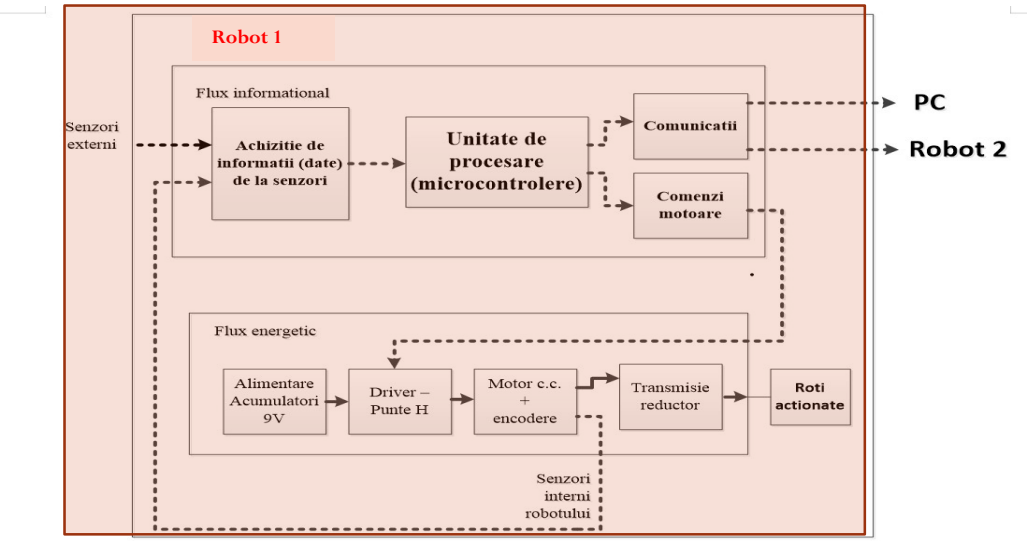
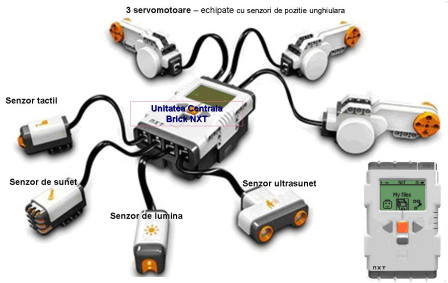


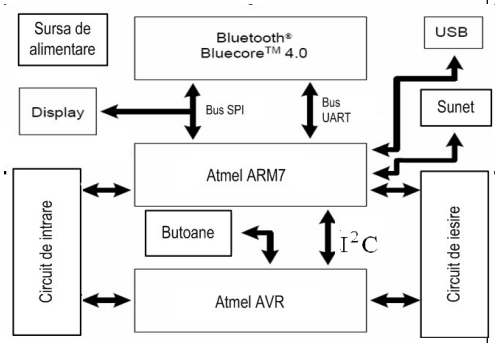
Fig. 1.1.2 Fluxuri informaţionale şi de energie, care caracterizează sistemul NXT

Componente ale robot NXT



Unitatea centrală a sistemului NXT include :

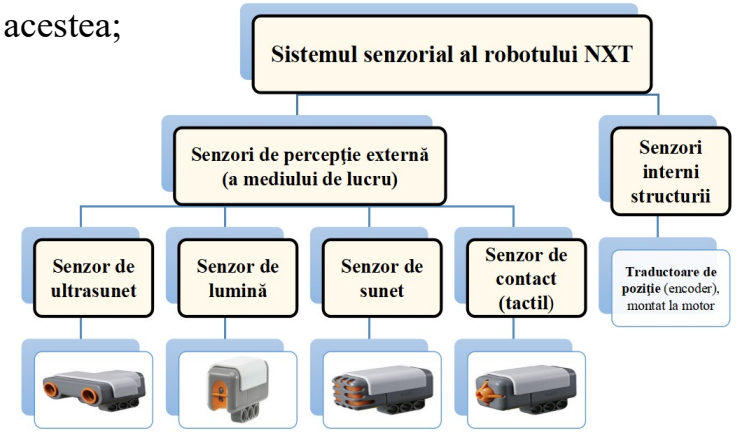
- -Microcontrolere
  - microcontroler principal, care execută programele utilizatorului: Atmel ARM7, AT91SAM7S256, pe 32 biţi (arhitectură RISC),
  - microcontroler auxiliar destinat controlului PWM al servomotoarelor: Atmel AVR, ATmega48, pe 8 biţi (arhitectură RISC);
- -Interfaţe de comunicaţie NXT cu exteriorul:
  - USB 2.0 full speed (12 Mbiţi/s) - utilizată pentru download de programe de pe PC-ul gazdă
  - Bluetooth CSR BlueCoreTM 4 v2.0 + sistem EDR, Controler - memorie RAM internă de 47 Kbyte, memorie FLASH externă 8 Mbit, 26MHz;



- -Porturi pentru conectarea motoarelor şi a senzorilor: porturi de intrare pentru senzorii analogici şi digitali (senzorul de ultrasunet al kit-ului NXT este digital - 8 biţi, iar ceilalţi sunt analogici) şi porturi de ieşire/intrare pentru alimentarea motoarelor NXT şi citirea datelor de la encoderele montate la acestea;

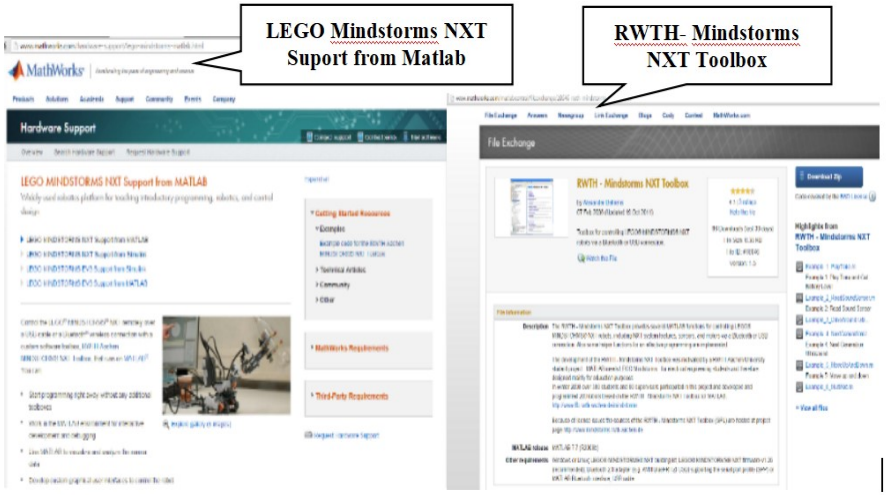
Porturi NXT pentru conectarea motoarelor şi a senzorilor

- **porturi de intrare pentru senzorii analogici şi digitali** - senzorul de ultrasunet al kit-ului NXT este digital - 8 biţi, iar ceilalţi sunt analogici.
- **porturi de ieşire/intrare** pentru alimentarea motoarelor şi citirea datelor de la encoderele montate la acestea.



Medii de dezvoltare şi programare specifice

- **National Instruments** cu **toolkit-ul MINDSTORMS NXT**, pachet gratuit, ce lucrează cu firmware-ul standard LEGO, şi care rulează ca modul adiţional podusul de bază LabVIEW.
- **Mathworks,** cu **toolbox-uri MINDSTORMS NXT**, care se pot adăuga la produsul de bază MATLAB-Simulink.



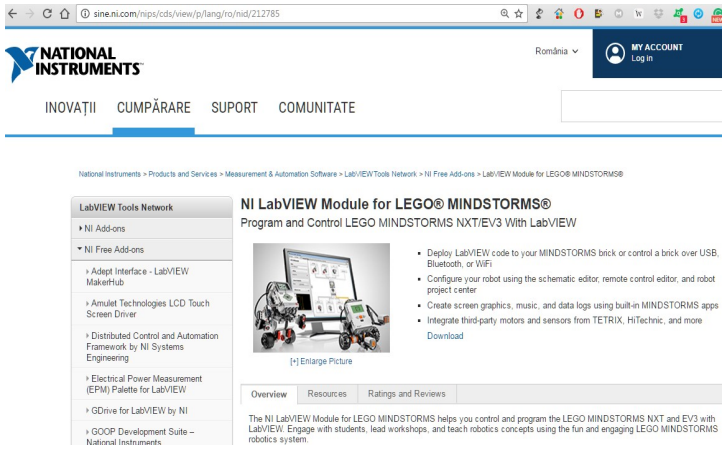


Fig.2.2.b Pagini Mathworks pentru descărcare de programe demonstrative

Pentru construirea aplicaţiilor de control la distanţă este disponibil mediul **MIT App Inventor 2**, care permite dezvoltarea rapidă şi facilă de aplicații Android, direct sub un browser Web.
App Inventor este un limbaj de programare vizuală open-source, inițiat de Google și preluat ca produs funcţional din 2012 de Massachusetts Institute of Technology (MIT).

Exemple

Navigarea către o ţintă fixă şi ocolirea obstacolelor întâlnite

Problema: Se cere să se dezvolte o arhitectură de control reactiv pentru comanda în timp real a mişcării (navigaţiei) unui robot mobil având sistemul de locomoţie de tip tribot, astfel încât robotul să fie capabil să realizeze task-uri de navigaţie autonom, în medii de lucru necunoscute, cu configuraţii de obstacole staţionare.

Rezolvare problema

Exemple

Deplasarea unui robot mobil între locaţii cerute din spaţiul de lucru

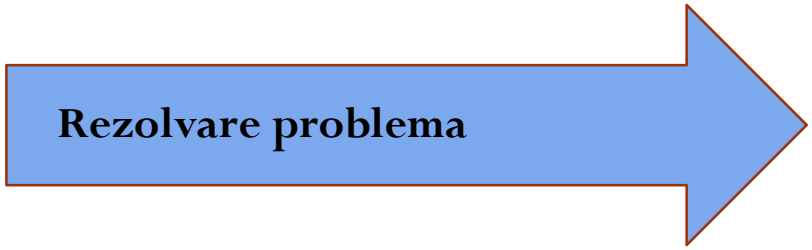
Problema: Fiind dat un robot mobil autonom care poate evolua într-un mediu de lucru fara obstacole, se cere programarea acestuia pentru a se deplasa între puncte de coordonate date, folosind ca mărime de feedback informaţia primită de la senzorii interni (traductoarele de pozitie - encodere). Punctele între care se deplasează sunt date prin coordonate carteziene (x,y). Tipurile de mişcare de implementat sunt: schimbarea orientării prin pivotare şi deplasarea în linie dreaptă către un punct ţintă.

Rezolvare problema

Exemple

Realizarea hărţii unui teren cu obstacole şi generarea traiectoriei optime printre acestea

- O categorie importantă de aplicaţii în robotică necesită rezolvarea problemei de găsire a drumului optim între două puncte. Problema constă în găsirea traseului de la un punct de start la un punct final, pe un teren în care sunt prezente obstacole. Traseul trebuie determinat pe o hartă bidimensională.



Exemple

Problema: Fiind dat robotul cu 2 gdm, care se mişcă în plan, având două articulaţii realizate prin cuple cinematice de rotaţie (aşa numitul **robot plan bimobil**), se cere să se determine modelul geometric direct al acestuia.

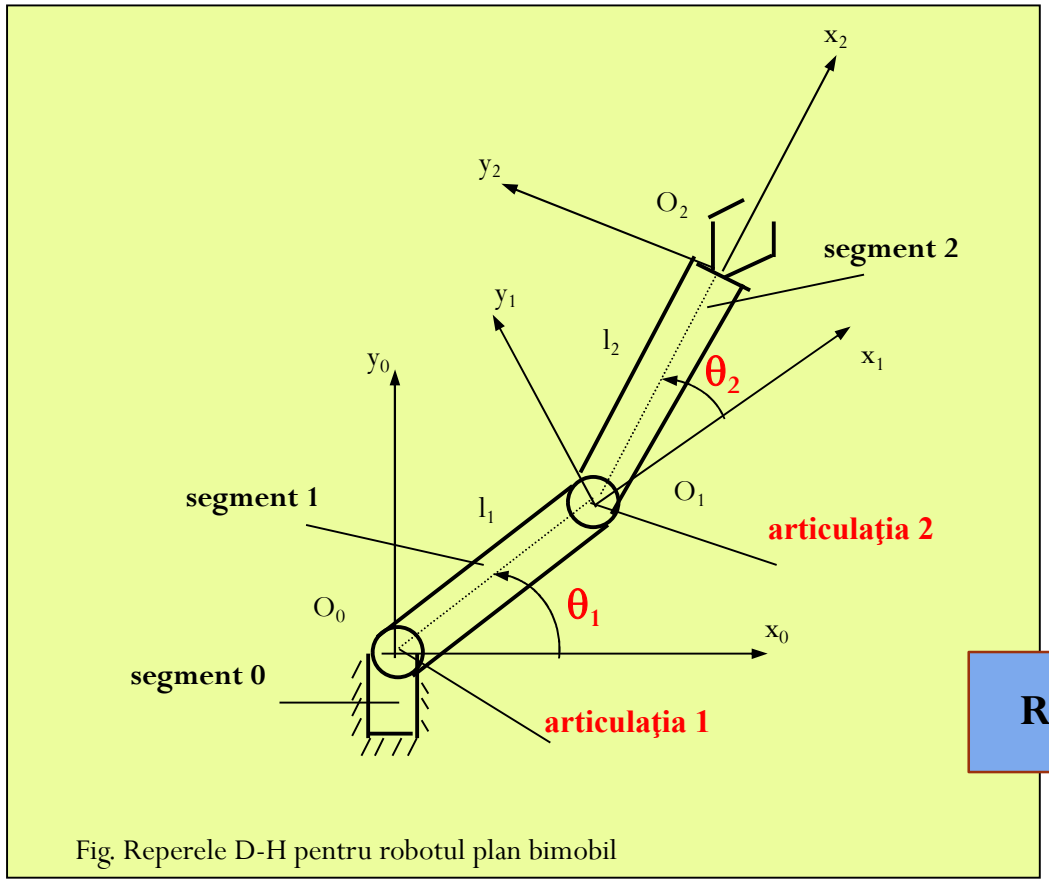


Fig. Reperele D-H pentru robotul plan bimobil

Rezolvare problema

Robotică
(extras din Fișa Disciplinei)

Punctaj activitate de laborator/seminar:

Media notelor pe parcurs la laborator si seminar: 60 puncte

Temă de casă: 20 puncte

Colocviu final: 20 puncte

Vă mulțumesc pentru atenție!