**Controlul wireless al unui robot**

**-un hexapod cu radiocomanda-**

*Materiale folosite pentru proiect:*

-Arduino UNO

-Arduino Nano

-Driver de servomotoare (PCA9685)

-18x servomotoare (SG 90)

-regulator de tensiune Step-down  (D24V22F5)

-Joystick cu potentiometru

-Emitator si Receptor radio (MX-FS-03v si respectiv MX-05V)

-cutie pentru 4 baterii AA

-fire/conectori

-lemn/suruburi/saibe/piulite

***1. Etapa de design (a scheletului)***

Initial mi-am propus sa construiesc un hexapod cu 2 grade de libertate pe fiecare picior, pentru simplitate. Insa din entuziasm am decis sa merg pe 3 grade de libertate. Insa din cauza acestei decizii a aparut riscul ca servomotoarele (cele mai ieftine pe care le-am putut gasi) sa nu aiba suficienta putere. Cuplul servomotoarelor SG 90 este de 1.8kgf\*cm. Ceea ce inseamna ca la mai mult de 1.8 kg la un cm fata de ax nu mai exista garantia ca se vor misca. Pentru a ma asigura ca este posibil sa fac un asemenea hexapod cu materialele propuse, am cautat sa vad daca a mai fost facut vreun hexapod cu 3 grade de libertate folosind servomotoare de putere similara.

In urma cercetarii am descoperit ca intr-adevar cu un design clasic probabil servomotoarele nu ar functiona. Insa am gasit un design ce utilizeaza un sistem de scripeti pentru a reduce sarcina servomotoarelor in schimbul limitarii miscarilor. Neavand masuratorile persoanei care a facut acel design, am masurat ochiometric si am adaptat dimensiunile la servomotoarele si placa mea de programare.

***2. Etapa de fabricare (a scheletului)***

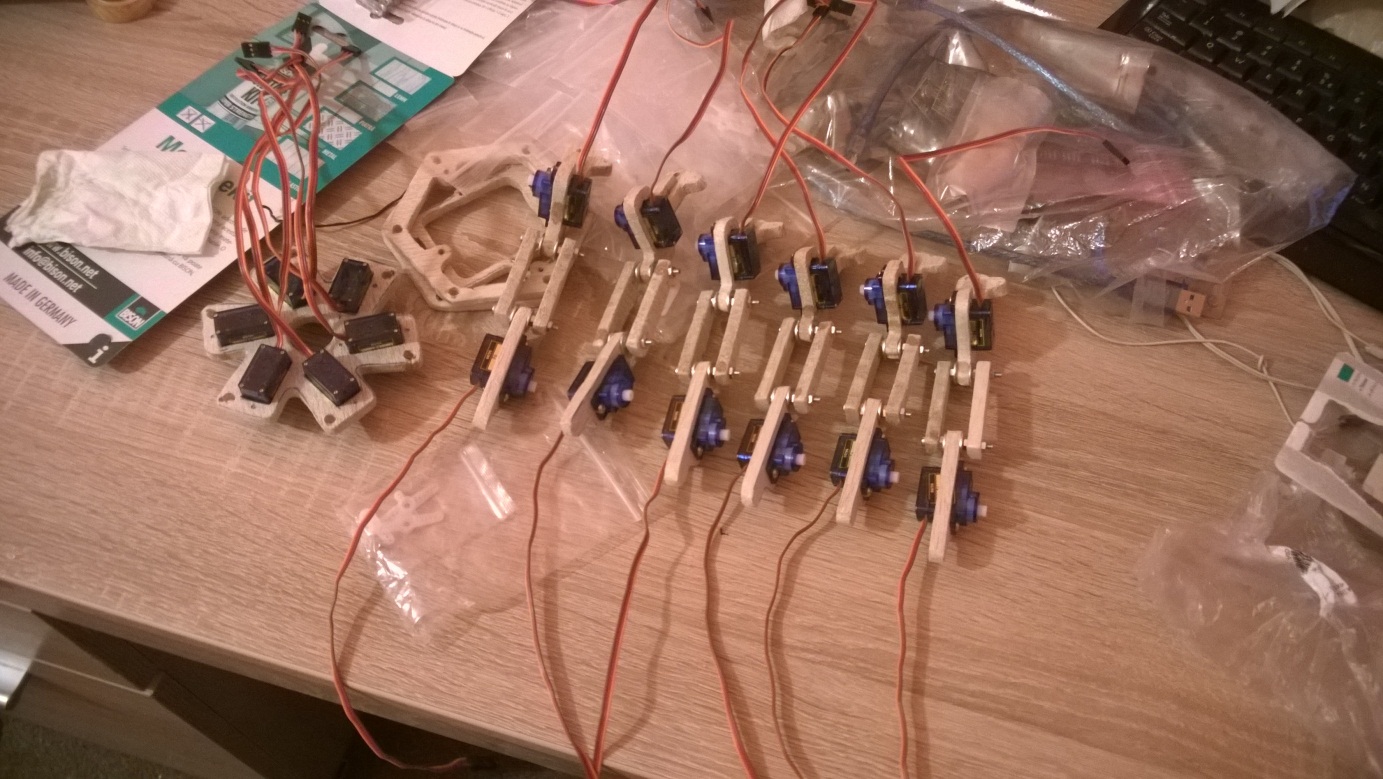
Utilizand o trusa de traforaj si o bormasina, am decupat piesele necesare scheletului dintr-un placaj de lemn cu densitate foarte mica si grosimea de 6mm. Desi lemn cu densitatea mai mare ar conferi o rezistenta mai ridicata, am preferat ceva mai usor, deoarece inca eram ingrijorat de greutatea finala.



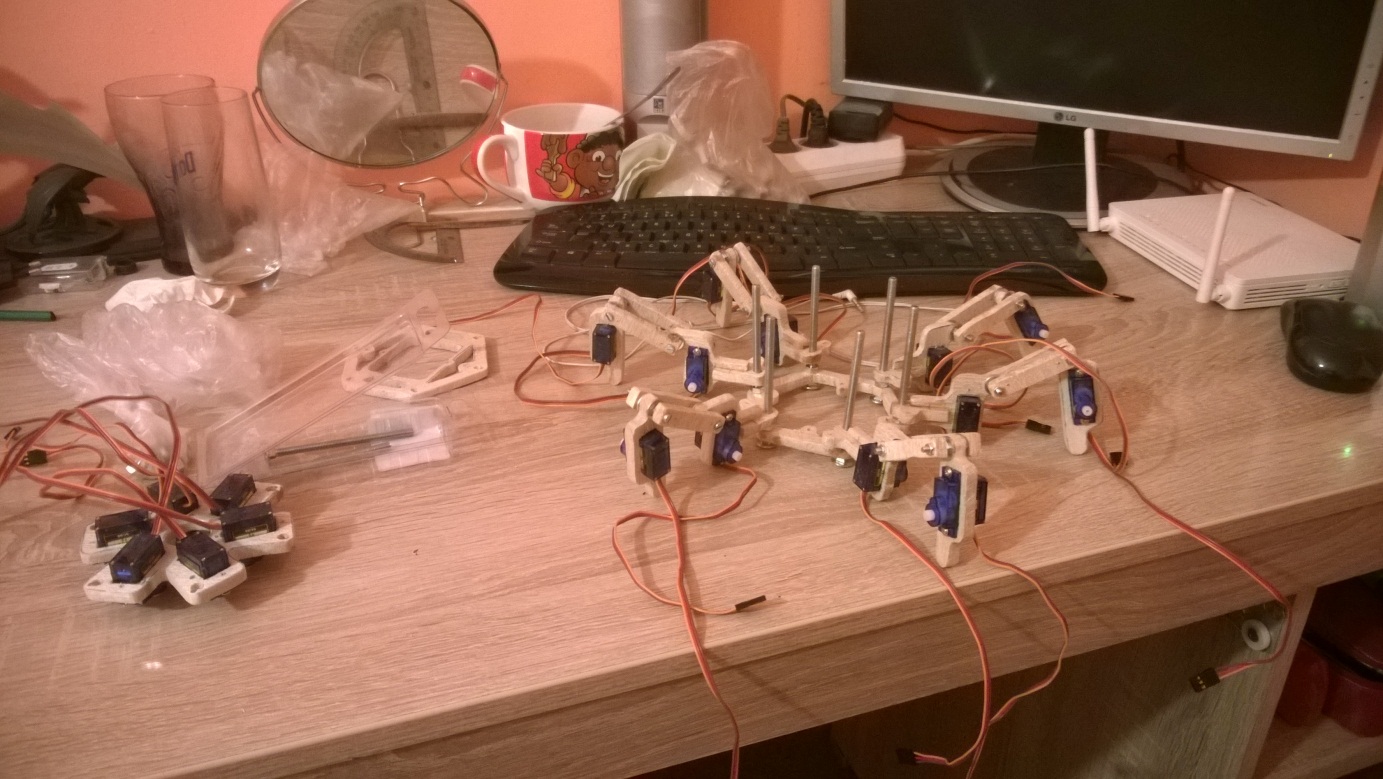


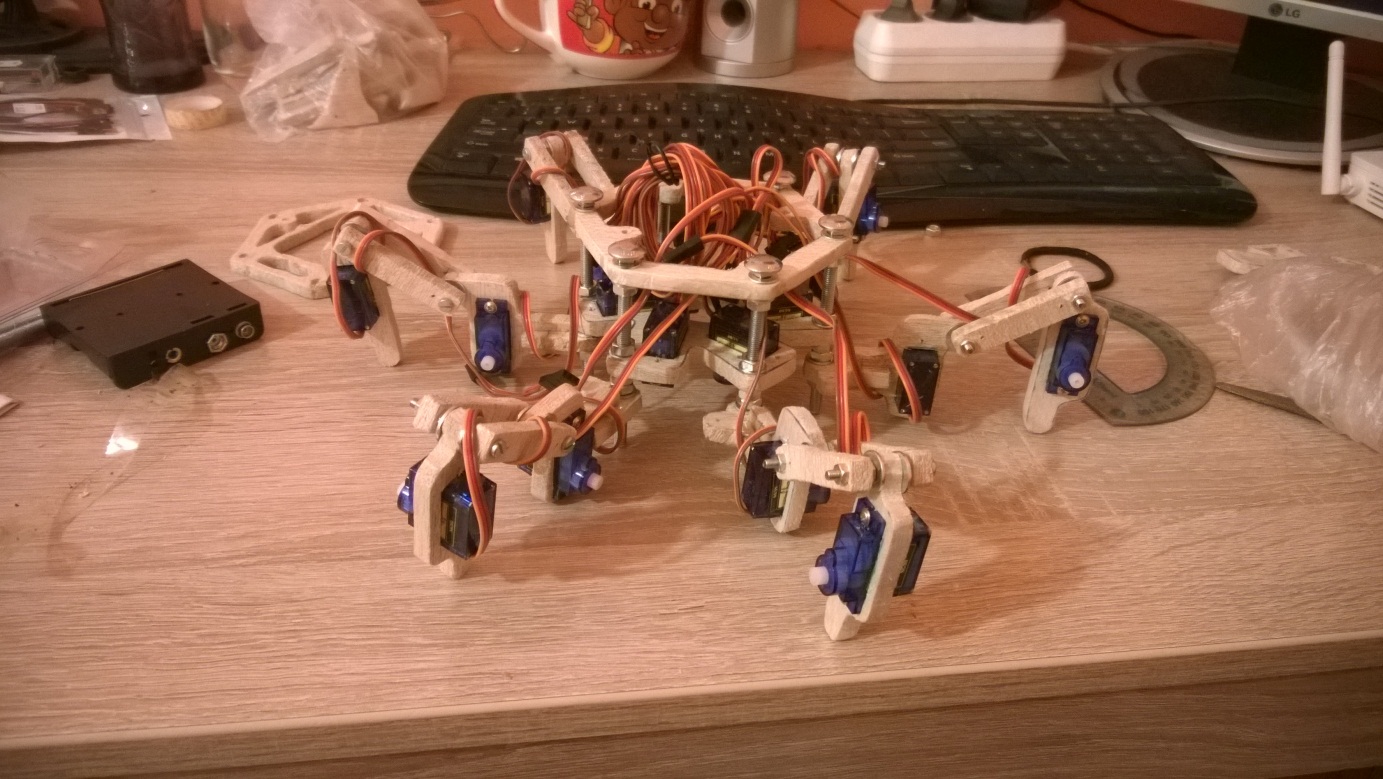
Din pacate lemnul utilizat avea o densitate atat de mica incat multe din piesele facute s-au rupt in timpul procesului.

Dupa decuparea pieselor, au fost necesare cateva retusuri pentru a atasa servomotoarele.



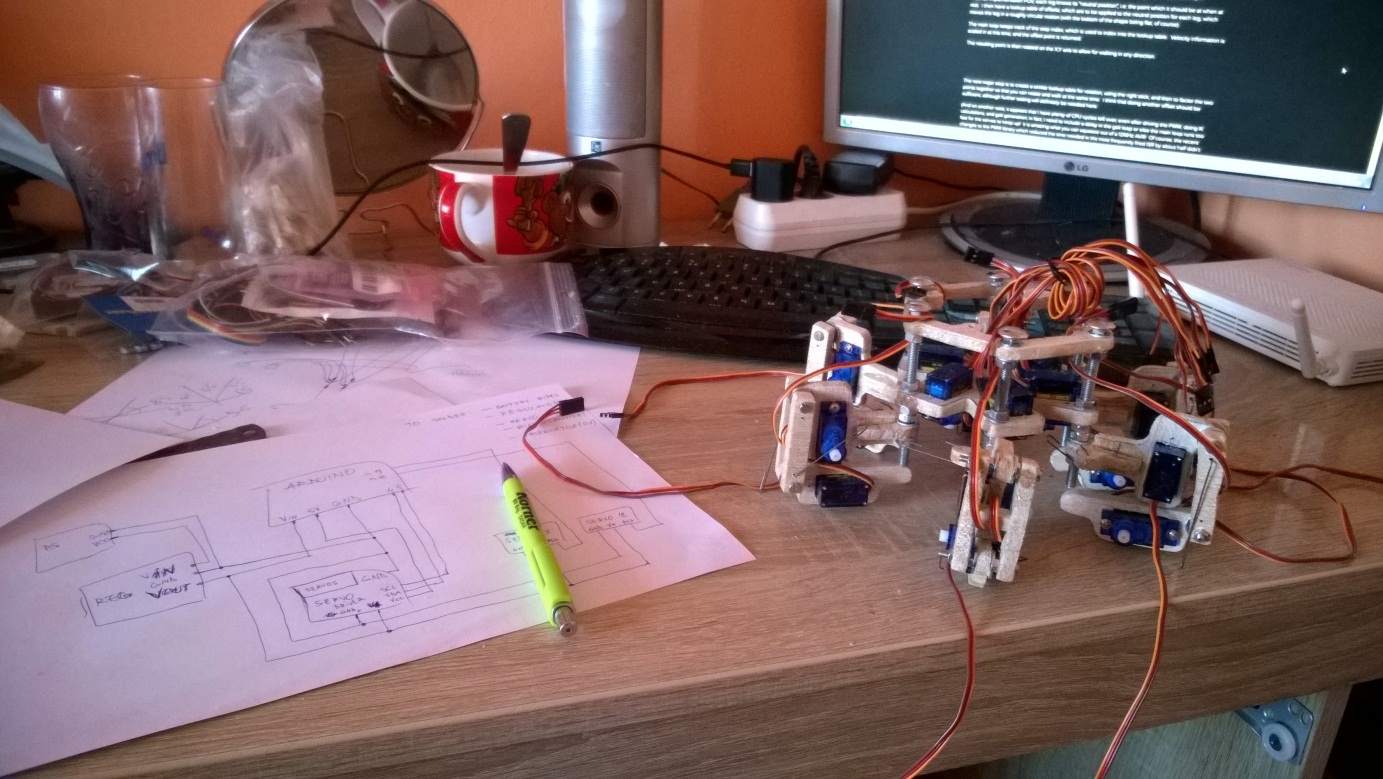
Dupa care a urmat asamblarea componentelor.





Pentru a putea misca fiecare segment al piciorului, a fost gandit un sistem de scripeti din bare metalice de grosimea sarmei.

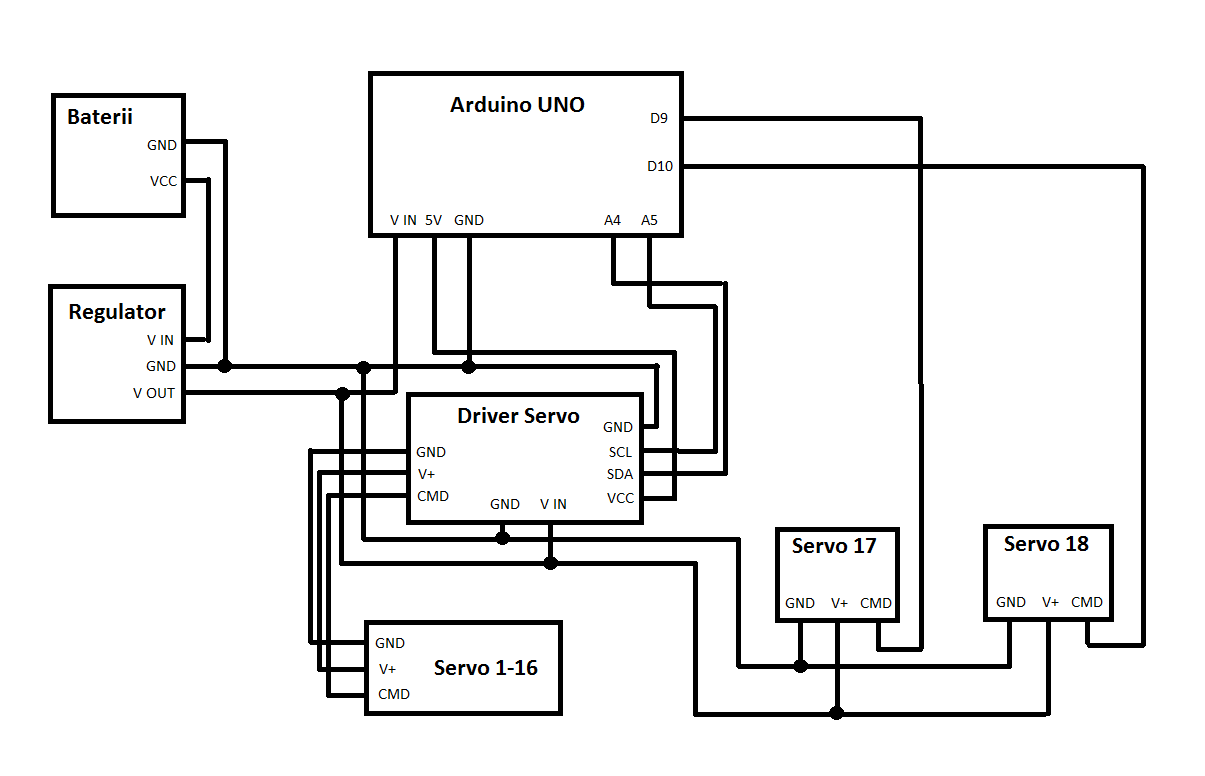
Probabil un sfert din efortul construirii scheletului s-a dus in indreptarea, decuparea si indoirea unei sarme la dimensiunile necesare barelor responsabile cu formarea scripetilor.



***3. Etapa de proiectare a circuitului.***

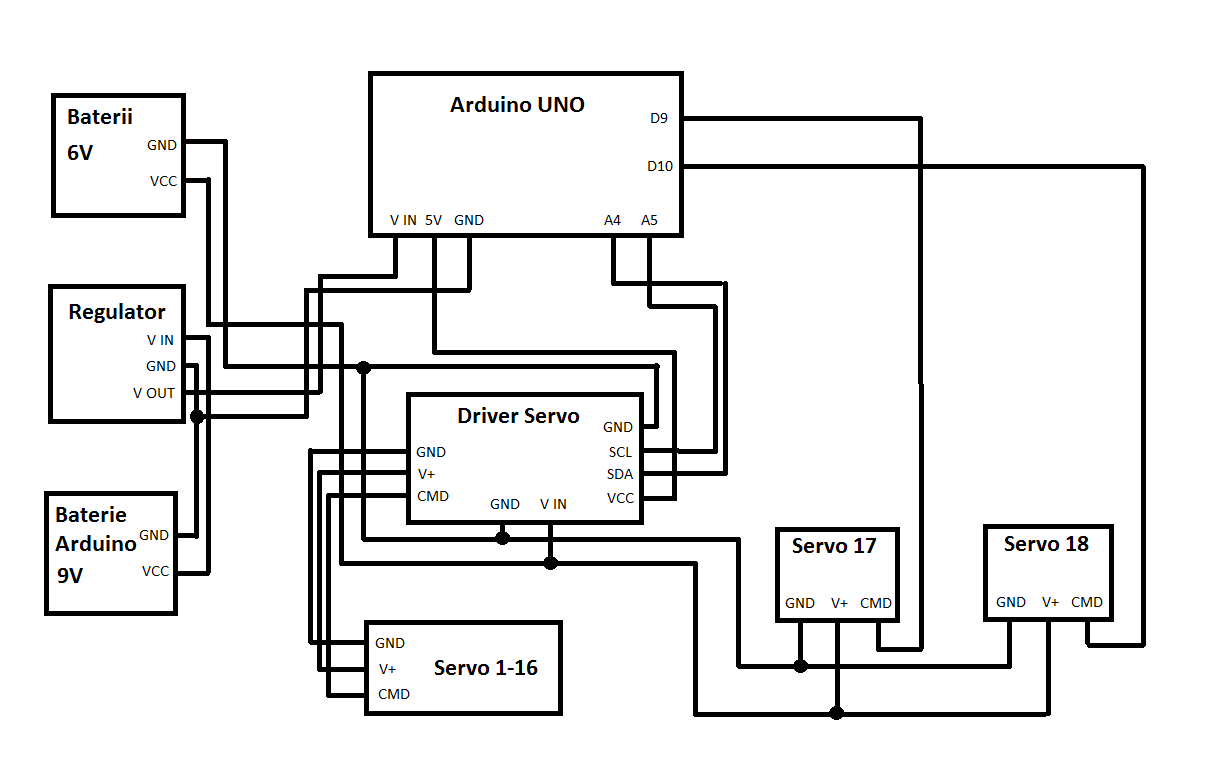
Inca de la inceput am decis ca robotul va fi comandat de Arduino UNO, in timp ce Arduino Nano va fi utilizat pentru telecomanda. Insa datorita numarului mare de servomotoare ce trebuiesc comandate simultan, am avut nevoie de un generator de semnale PWM (Arduino UNO are doar 6, nici pe aproape de cate sunt necesare pentru acest proiect), astfel am decis sa cumpar un driver de servomotoare ce poate genera pana la 16 semnale PWM.

Deoarece driverul de servomotoare pana la 16, 2 servomotoare vor fi comandate direct de arduino. In design-ul initial doream sa alimentez intreg circuitul de la cele patru baterii AA pe care le voi situa dedesubt, insa servomotoarele trag foarte mult curent, iar tensiunea din intreg circuitul scade drastic, restartand arduino-ul permanent. Ca o masura de precautie, designul initial avea si un regulator de tensiune in el (am ars deja un Arduino din greseala) doar ca sa fie.

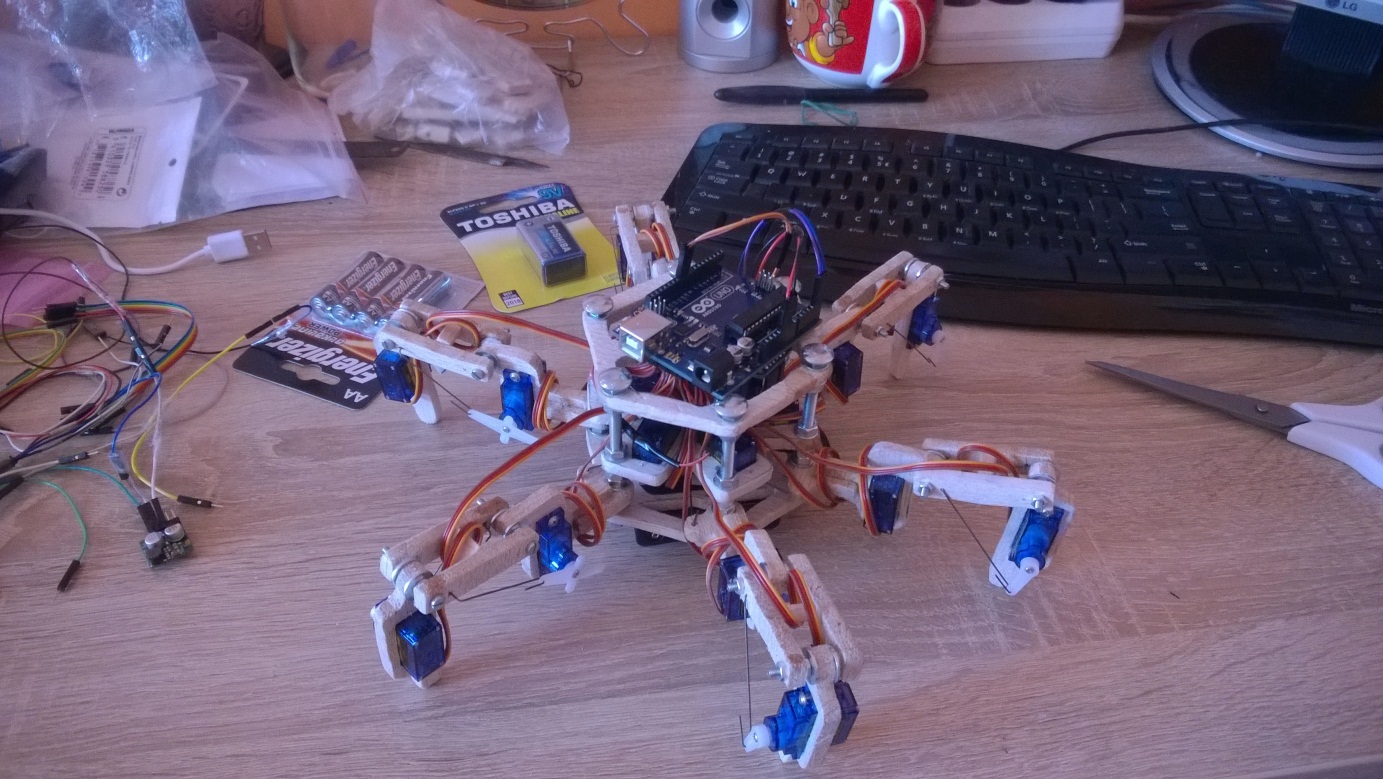


Am crezut ca problema ar putea fi rezolvata utilizand un condensator suficient de mare, insa s-a dovedit imposibil.

Singurele solutii ramase erau scaderea semnalelor PWM sau alimentarea separata a arduino-ului. Am considerat mai viabila solutia cea din urma, iar astfel circuitul a devenit:



***4. Etapa de conectare a componentelor***



***5. Etapa de programare***

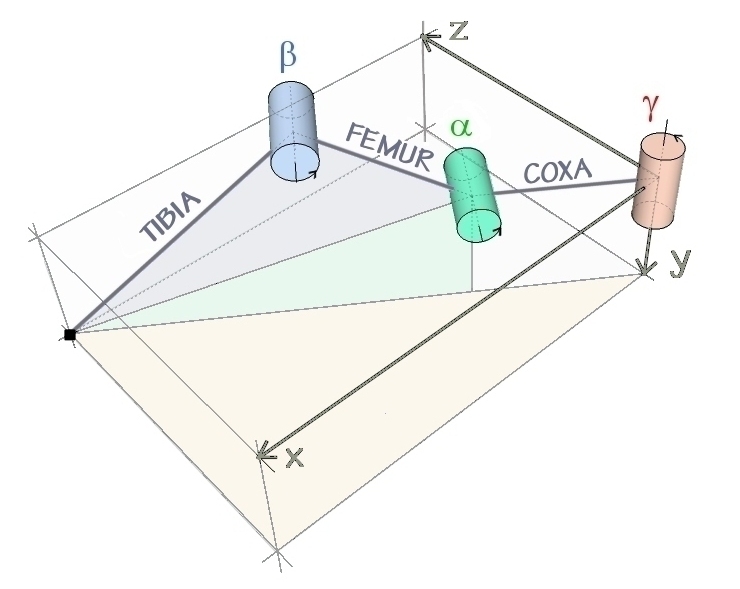
Pentru ca fiecare servomotor este diferit (chiar si daca sunt acelasi model), este necesar ca mai intai sa le calibram. Am facut acest lucru din software, pentru ca e mai simplu decat sa desfac fiecare servomotor si sa umblu la el. Acest lucru se face dand o comanda de diferite pulsatii pana se gaseste pulsatia minima si maxima pentru fiecare servomotor. In cazul de fata, pulsatia minima va duce servomotorul la 0 grade, iar cea maxima la 180. Acest proces a fost mai epuizant decat speram.

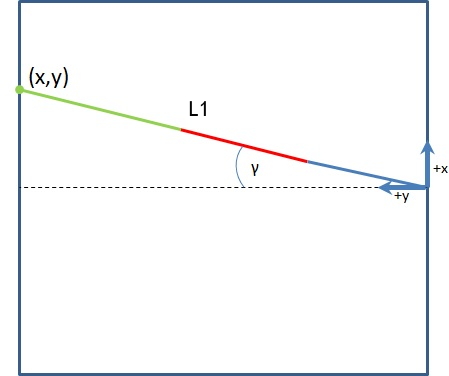
Pentru a comanda servomotoarele, am utilizat biblioteca oferita de Adafruit pentru driverul lor Adafruit\_PWMServoDriver.h si Servo.h (ulterior schimbata in ServoTimer2.h din cauza unor dificultati tehnice).

Desi initial credeam ca pentru a deplasa robotul doar voi da pozitii predeterminate fiecarui picior si le voi repeta intr-un ciclu, am descoperit ca o metoda populara de a pune in miscare hexapozii este folosind cinematica inversa. Practic aceasta metoda presupune calcularea unghiurilor cand se stiu dreptele.

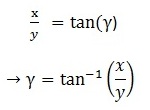
**Algoritmul de aflare al unghiurilor:**

In total trebuie aflate 3 unghuiri corespunzatoare fiecarui servomotor: alpha, beta si gamma.

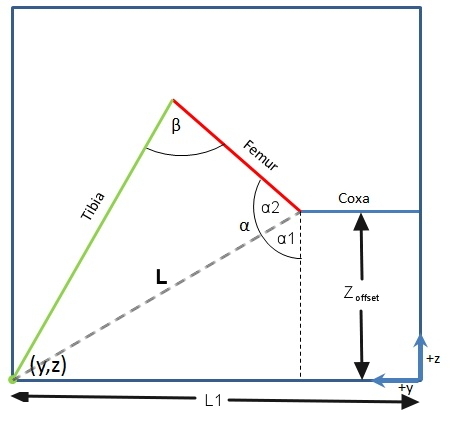




Pentru a afla gamma, este simplu, doar aplicam formula

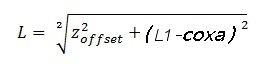


Unde P(x,y) este punctul destinatie.



Pentru a afla alpha si beta este putin mai dificil. In primul rand impartim alpha in 2 unghiuri alpha 1 si alpha 2.

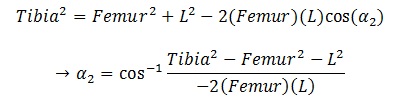
Pentru calcularea lui alpha 1 avem



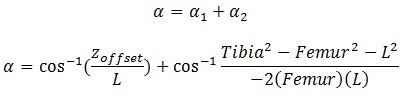
Din care rezulta ca alpha 1 este

C:\Users\Shpitzy\Desktop\61.jpg

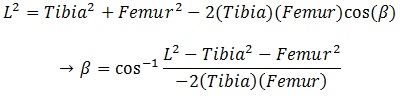
Similar, dar pentru un triunghi fara unghi drept calculam si alpha 2



Cunoscand alpha 1 si alpha 2, trebuie doar sa la adun.

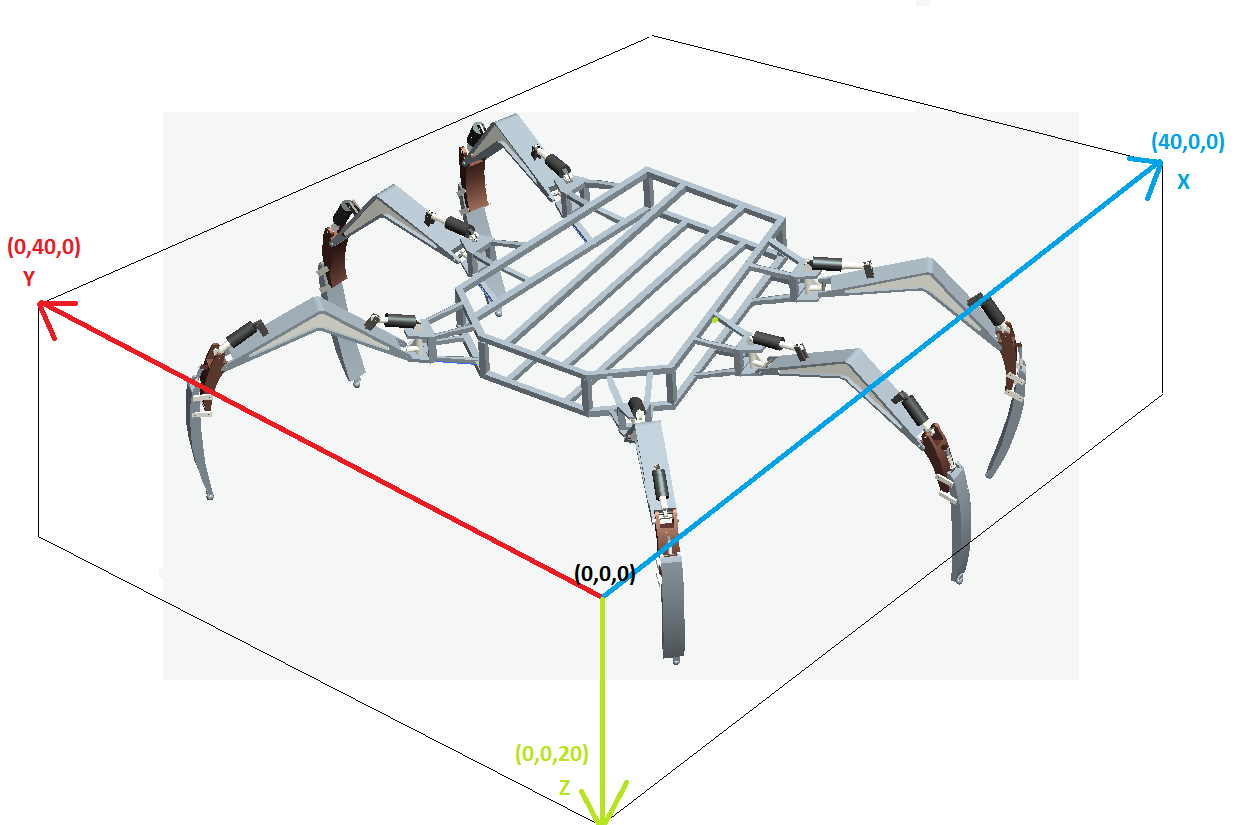


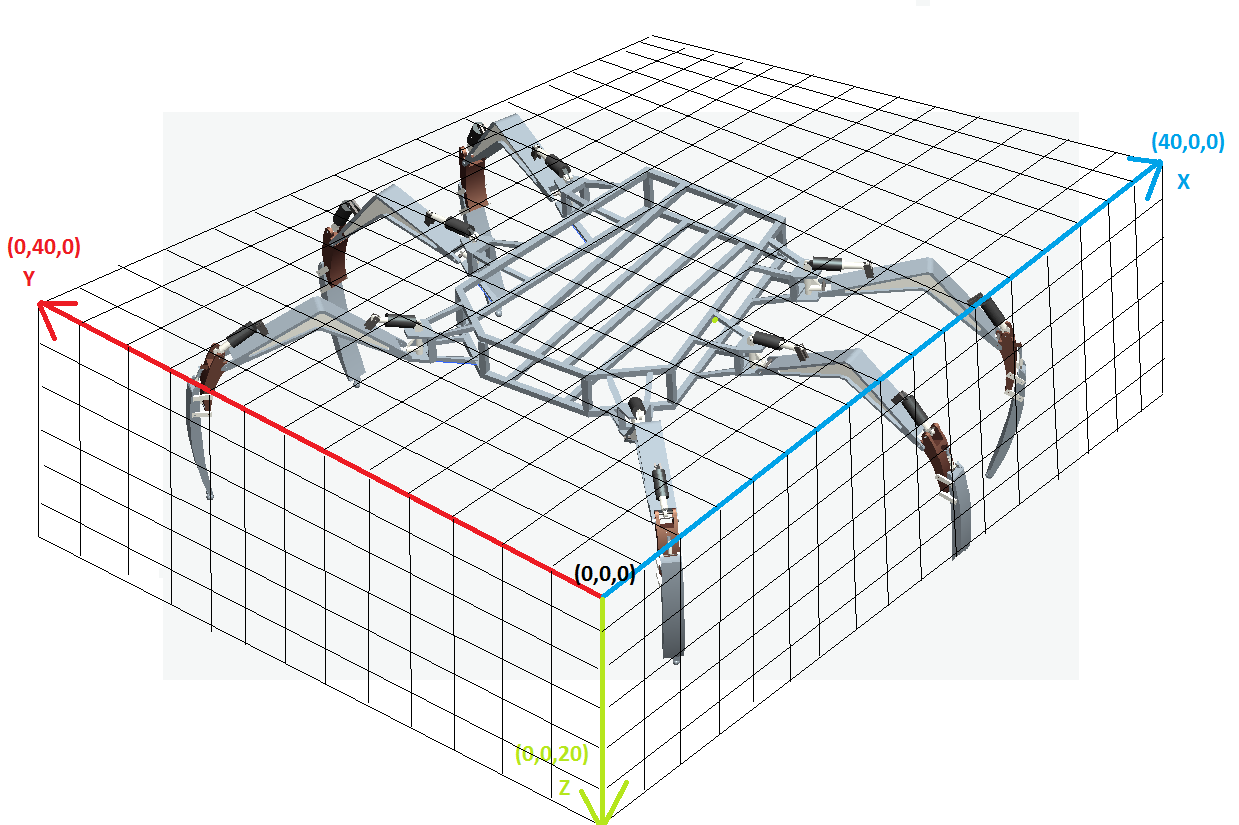
Beta se calculeaza aproape identic cu alpha 2.



Acum stim ce comanda sa dam unui picior daca stim care este punctul in care vrem sa ajunga.

Cum se pune aceasta teorie in practica? Nu stiu cum se face de obicei, insa eu am decis sa imi imaginez ca hexapodul meu este in interiorul unui paralelupiped dreptunghic intr-un sistem de coordonate 3 dimensional.



Unde fiecare unitate reprezinta un centimetru in lumea reala.

Singurele valori pe care a trebuit sa le introduc manual au fost punctele initiale ale picioarelor. Acestea reprezentand pozitia de repaus a robotului. Pentru a le misca din pozitia initiala, nu mai sunt necesare nici coordonate, nici unghiuri, doar trebuie sa specific directia si distanta, iar algoritmul va calcula automat ce valori trebuie sa dea servomotoarelor pentru a misca picioarele in acel punct.

Utilizand acest algoritm, am creat functia de miscare care imi comanda toate cele 6 picioare (18 servomotoare) avand ca input doar un unghi, reprezentand directia dorita deplasarii si distanta, cat de mari sa fie pasii (exista o limita; robotul nu poate face pasi mai mari de 3 cm).

***6. Etapa de control***

Pentru a construi telecomanda am utilizat un Arduino Nano, un joystick cu potentiometru ce imi va permite sa ofer mai mult de 4 directii de deplasare si un emitator radio.

Comunicarea intre cele 2 placi se va face prin intermediul setulului de emitator si recepror radio cu frecventa de 315MHz, a caror programare se va face cu ajutorul bibliotecii VirtualWire.h. Aceasta biblioteca face utilizarea celor 2 module foarte simpla. Astfel, din telecomanda, trimit la hexapod un unghi reprezentand directia in care vreau sa se deplaseze in raport cu sistemul sau propriu de coordonate.

vw\_send(message, length) este functia de transmitere a mesajelor.

vw\_get\_message(buf, &buflen)) este functia pe care o folosesc pentru a primi mesaje.

Mesajul pe care il trimit este un int pe care il convertesc in char.