ROČNÍKOVÁ PRÁCE

Obor: Informatika

Vývoj chůze čtyřnohého mechanického robota

Autor: **Martin Znamenáček**
Konzultant: **Mgr. Čestmír Mniazga**
Škola: **Gymnázium ALTIS s.r.o., Dopplerova 351, 110 00 Praha**
Kraj: **Praha**
Ročník: **6. A (sexta), 2019/2020**

Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Praze dne 31. 5. 2020

Martin Znamenáček

Poděkování

Tímto bych velice rád poděkoval svému konzultantovi panu Mgr. Čestmíru Mniazgovi za cenné rady a připomínky v průběhu zpracovávání mé ročníkové práce, pomoc s nejasnostmi, se kterými jsem se setkal, motivaci a trpělivost.

Anotace

Ve své práci jsem se zabýval vývojem pavoučí chůze čtyřnohého robota. Jedná se o software vyvíjený v rámci programovacího jazyka Arduino a jeho vývojového prostředí.

Konečným výsledkem mé práce je čtyřnohý pavouk, který je schopen chůze vpřed, aniž by ztratil rovnováhu, či jakkoliv zavrávoral.

Jedná se tedy o projekt, jenž se zabývá inverzní kinematikou, jakožto médiem mezi souřadnicemi a mechanickým pohybem.

Klíčová slova

Arduino; C; C++; programování; robotika; inverzní kinematika; čtyřnohý pohyb; pavoučí chůze.

OBSAH

1 ÚVOD

Práce se převážně zabývá kódem, tedy objasněním řešení otázky čtyřnohé chůze mechanického robota. Od začátku až do konce tedy rozebírá, jak celý program funguje, popřípadě vysvětluje hlubší myšlenku za zvoleným řešením problému.

Mimo jiné obsahuje letmý popis programovacího prostředí a hlavně pak součástek, které byly k postavení pavouka.

Nechybí ani videa, která slouží jako perfektní nástroje pro pochopení aspektu zachování těžiště, jež je nezbytné pro chůzi samotnou.

2 CÍLE PRÁCE

- Postavit robota
- Rozchodit robota

- Naučit se pracovat s jednodeskovým počítačem Arduino
- Zlepšit si znalosti programovacího jazyka spadajícího do rodiny C
- Rozšířit si obzory ohledně robotiky a přenosu kódu na mechanickou práci

3 PROSTŘEDÍ

Veškerý kód pro řízení mikrokontroléru robota byl vyvíjen zapomocí softwaru Arduino IDE. Jedná se o jednoduché, ale flexibilní vývojové prostředí, jež je zdarma dostupné ke stažení na oficiálních stránkách https://www.arduino.cc/en/main/software.

Programovací jazyk Arduina vychází z rodiny C, konkrétně z její knihovny Wiring. Arduino IDE je potom založeno na intuitivním prostředí Processing.[1]

[1] Arduino: Co je to Arduino? [online]. [cit. 2020-05-31]. Dostupné z: https://arduino.cz/co-je-to-arduino/

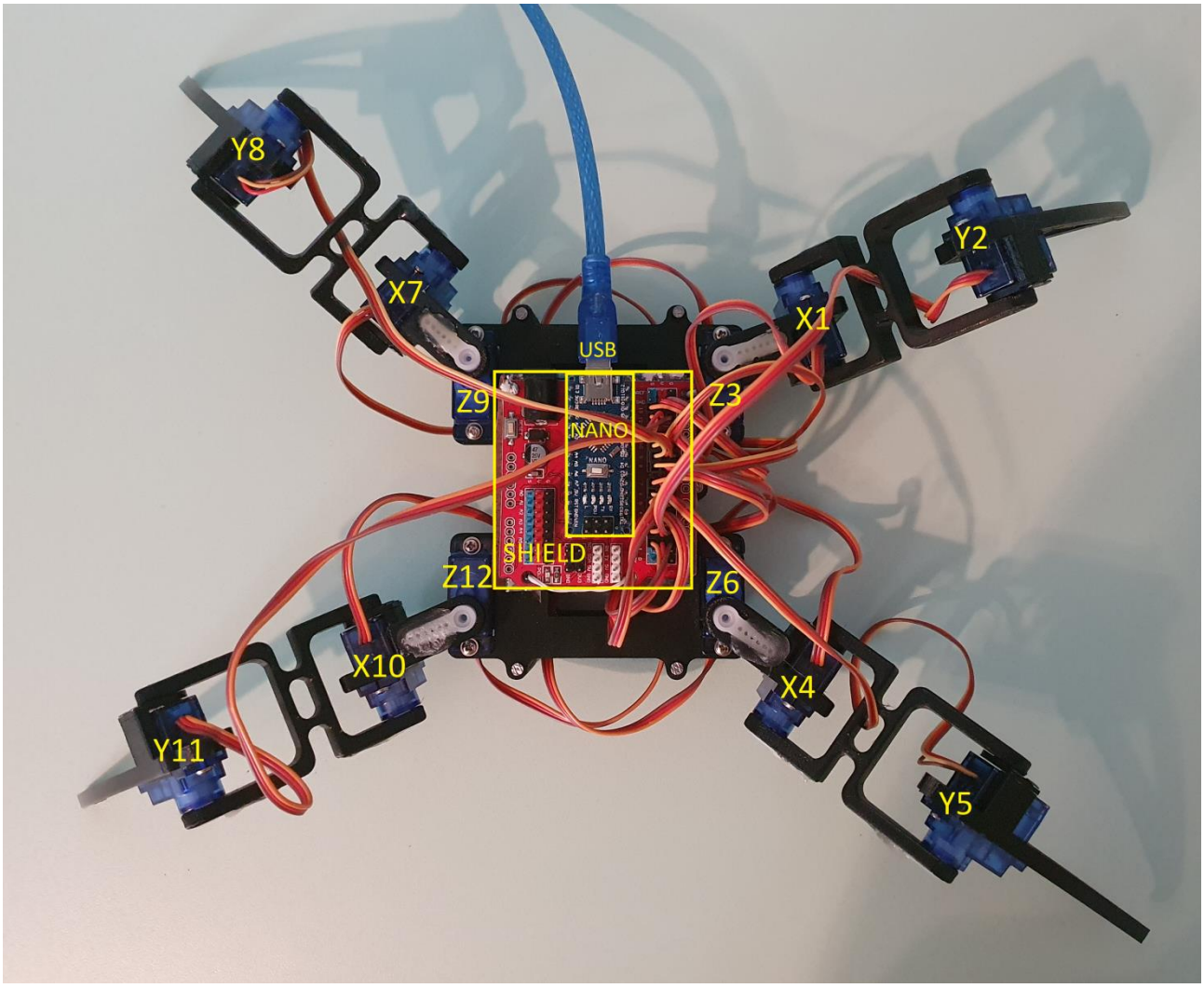
4 SOUČÁSTKY

Celkové náklady pro sestavení robota činily cirka 2 000 Kč. (Čip - 150 Kč, Shield - 150 Kč, 12 Servomotorů - 900 Kč, 3D části - 850 Kč).

4..1 Elektronické

Ovladačem, a tedy celým mozkem projektu je programovatelná deska, minipočítač Arduino NANO. Čip NANO je usazen na expanzním IO shieldu, tedy modulu, který umožňuje vytváření obvodů bez nutnosti pájení. Daná patice má piny vyvedené na jednotlivé konektory, již tak umožňují libovolné připojování a odpojování až 12 elektronických modulů.

Hybateli jsou páky modelářských 5V servomotorů, které umožňují pohyb ± 90 ° doleva a doprava. Jsou prozatímně napájeny mikro USB kabelem, připojeným k počítači. Je jich celkem 12, na každou ze 4 nohou připadají 3, a to pro pohyb na ose x, y a z.

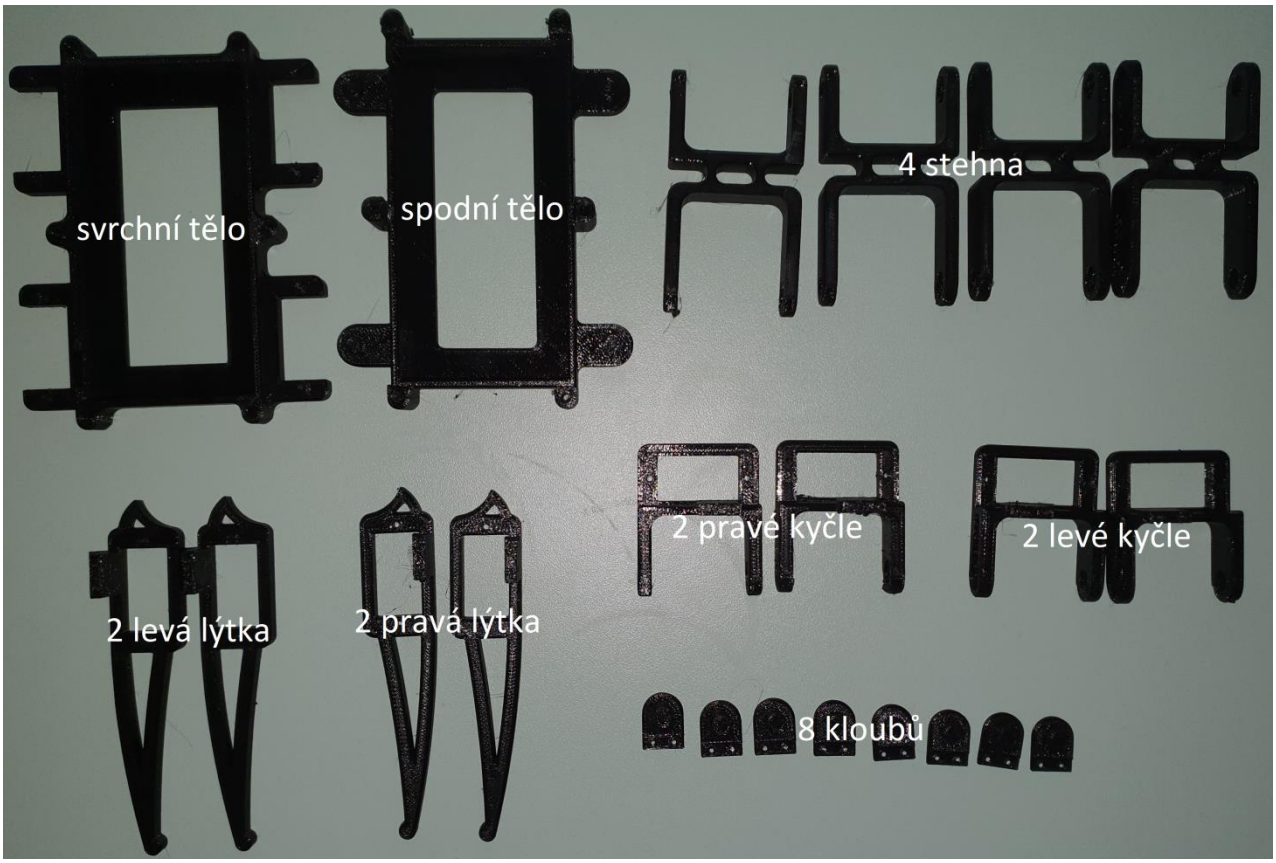


Obrázek 1: Přehled elektronických součástek, včetně popisu os, na kterých servomotory operují

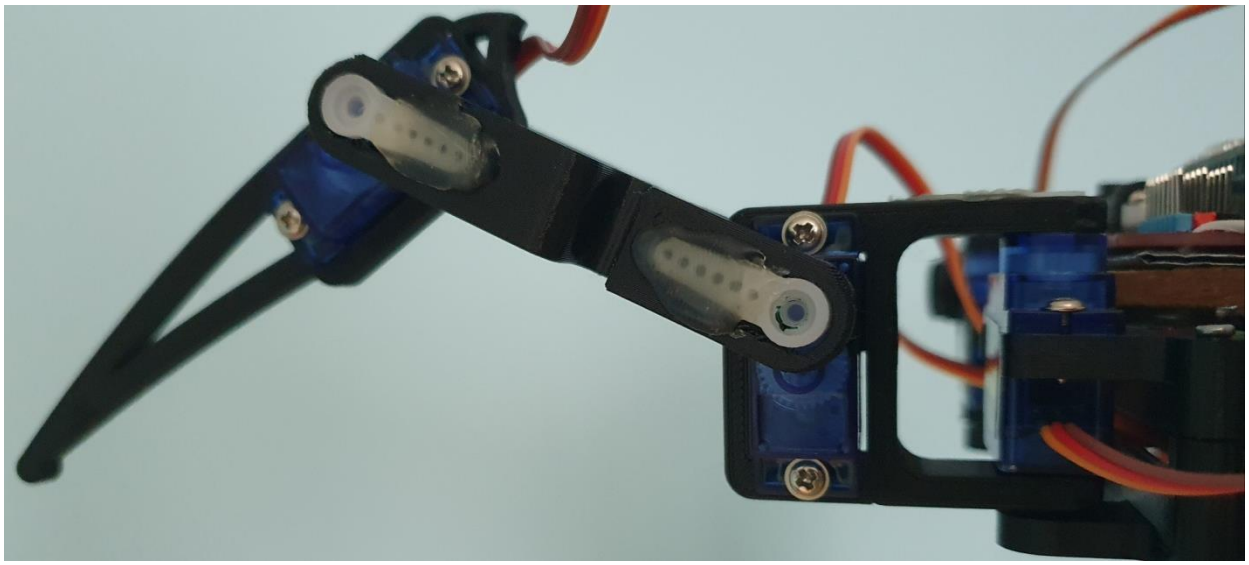
4..2 3D

Kromě elektronických součástek bylo ještě zapotřebí opatřit samotné robotí tělo. K tomu posloužily 3D tiskárny Sharplayers, které mi za 850 Kč umožnily vymodelování těla, končetin a spojů.

Celý tisk čítal 22 součástek: 2 tělní části, svrchní a spodní, 4 stehna, 2 pravé a 2 levé kyčle, 2 levá a 2 pravá lýtka a 8 kloubů pro fixaci lýtek a kyčlí na stehna.



Obrázek 2: Veškeré 3D součástky



Obrázek 3: Jednostranný pohled na nohu č. 3 sestávající se z 5 součástek připevněných k tělu

5 KÓD

5..1 Knihovny

Aby celý program mohl fungovat, bylo zapotřebí dvou knihoven:

<Servo.h> přidává funkci *attach(pin)*, pro připojení servomotorů k pinům desky, a *write(x)*, pro stanovení úhlu *x*, který má osa servomotoru protínat, přičemž povoluje hodnoty od nehybných *90* do *0*, či *180*, tedy pohyb vlevo, respektive vpravo, a to o úhel *x - 90* [°]. *Servo* pak definuje uspořádané pole hodnot, tedy 3 osy *[x,y,z]* pro 4 nohy *[0,1,2,3]* s příslušným názvem, kterým pak lze odkazovat na jednotlivé servomotory ve zmíněných funkcích, třeba: *servomotor[2][1].funkce*.

<FlexiTimer2.h> zavádí časovač, jenž pro každý interval *x* [ms], iniciovaný funkcí *start()*, provede blok *void*, který se nastaví pomocí *set(x,void)*. Každých *x* milisekund tedy vykoná část kódu, která je potřeba k přesnému ovládání servomotorů.

```
1  #include <Servo.h>
2  Servo servomotor[4][3];
3  #include <FlexiTimer2.h>
```

5..2 Konstanty

Pro pohyb servomotorů musí být přesně definované souřadnice, kam se osy *x*, *y* a *z* mají pohybovat. K tomu slouží jak startovní (*xyz_start*), tak pohybové (*xyz_krok*) pozice. Jakožto brzda potom slouží konstanta *nehybat*, pro jejíž hodnotu je servomotor nehybný, to je pozice při 90 °.

Při pohybu je taktéž potřeba pro dané osy *xyz* a nohy *0123* uchovávat současné a budoucí pozice v datových řadách, a to pro aproximaci délky ještě nedokonaného pohybu.

```
4  const float x_start = 60;
5  const float y_start = 0, y_krok = 40;
6  const float z_start = -30, z_krok = -50;
7  const byte nehybat = 255;
8  volatile float soucasna_pozice[4][3], budouci_pozice[4][3];
```

Robotí rychlost je pak závislá na *prodlevách*, které předchází přehřátí, a na rychlostech jednotlivých pohybů (*nohy*, *těla*, *vstávaní*/*sedání*), jež se přiřazují proměnné *rychlost_pohybu*.

Dané rychlosti jsou nakonec přepočítány pro každou část dané nohy zvlášť, a to na krátkodobou rychlost, kterou se servomotor při pohybu přibližuje k cílové pozici (*budouci_rychlost_pohybu*).

Modifikator_rychlosti pak slouží k upravení tempa robota při zachování definovaných poměrů.

```
9  const int prodleva = 250;
10 const float modifikator_rychlosti = 1.1;
11 const float rychlost_pohybu_nohy = 10;
12 const float rychlost_pohybu_tela = 5;
13 const float rychlost_pohybu_vstavani = 2;
14 float rychlost_pohybu, budouci_rychlost_pohybu[4][3];
```

5..3 Inicializace

V rámci prvotního *void* bloku se všech 12 motorů připojí k pinům a přednastaví se jejich pozice, aby nedošlo k automatickému načtení 90 °, jež způsobí zaseknutí nohou do těla.

Žádný z motorů se ale ve skutečnosti nepohne, poněvadž ihned po iniciaci pohybu dojde k jeho ukončení, a to nastavením přednastavené pozice za budoucí, tedy končenou.

Zároveň se přiřadí a zahájí zmíněný časovač, který po každém uběhnutí 20ms intervalu vyvolá funkci *ovladac_servomotoru*, jež slouží k přesné kontrole pohybu.

```
15 void setup(){
16   int pin_servomotoru[4][3] = {{2, 3, 4},{5, 6, 7},{8, 9, 10},{11, 12, 13}};
17   for (int i = 0; i < 4; i++){
18     for (int j = 0; j < 3; j++){
19       servomotor[i][j].attach(pin_servomotoru[i][j]);
20       nastavit_pozici(i, x_start, y_start, z_start, 0);
21       soucasna_pozice[i][j] = budouci_pozice[i][j];}}
22   FlexiTimer2::set(20, ovladac_servomotoru);
23   FlexiTimer2::start();}
```

5..4 Itinerář

Daná sekce *void* slouží jako plán fungování celého robota. Nejprve tedy vstane, po prodlevě započne pohyb vpřed, a to po dobu 25 kroků a po konečné pauze si sedne.

```
24 void loop(){
25   vstat(1); delay(4 * prodleva);
26   pohyb_vpred(25); delay(3 * prodleva);
27   vstat(0); delay(6 * prodleva);}
```

5..5 Vstávání a sedání

Aby mohl samotný pohyb započít, robot potřebuje výchozí stabilní pozici, ze které může vykročit vpřed. Jakmile veškerý pohyb dokončí, je třeba motorům odlehčit a celého robota zase posadit.

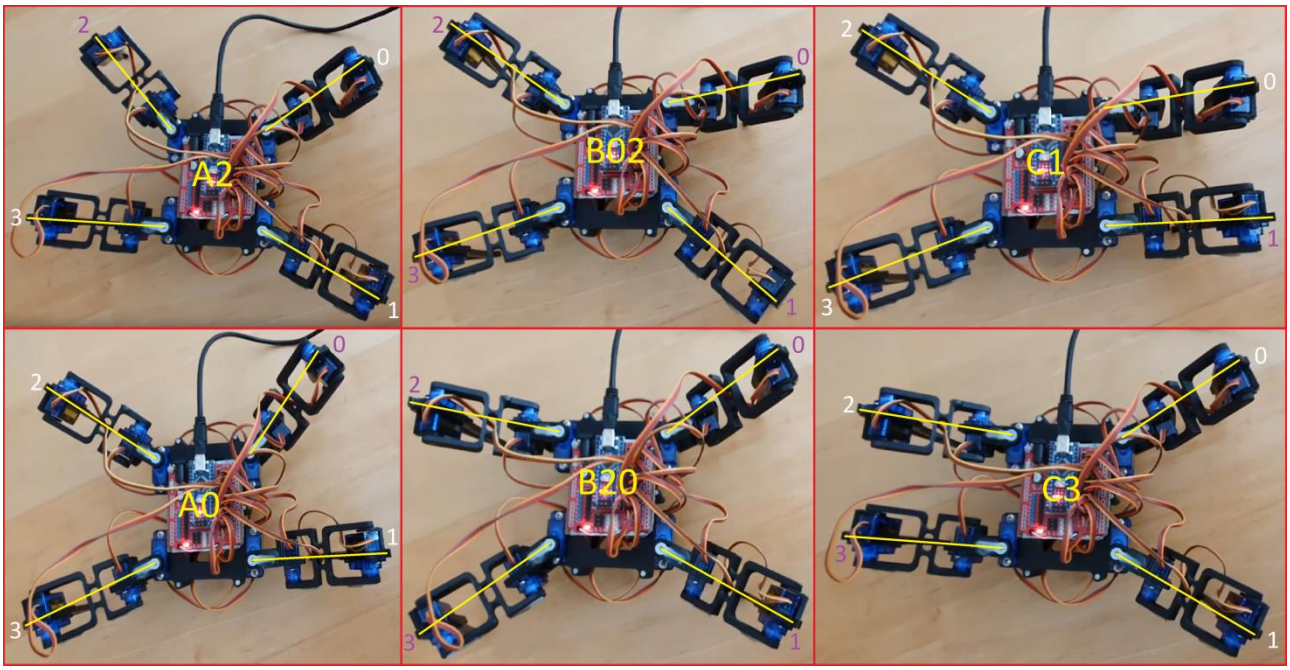
Daný blok kódu tedy na základě pravdivostní hodnoty *orientace* (*true* ... vztyk, *false* ... sed) ponechá osám *x* a *y* mrtvou 90° pozici a skrze *z*, pozvedne tělo, jej spustí k zemi, přičemž dané nohy provedou pohyb paralelně a nakonec na sebe počkají (*noha / 3* totiž bude pravdivá pouze v případě 4., tedy posledního opakování).

```
28 void vstat(int orientace){
29   orientace = (orientace == 1) ? (z_krok) : (z_start);
30   rychlost_pohybu = rychlost_pohybu_vstavani;
31   for (int noha = 0; noha < 4; noha++)
32     nastavit_pozici(noha, nehybat, nehybat, orientace, noha / 3);}
```

5..6 Pohyb vpřed

Podstatou celého programu je pohyb vpřed, který jsem rozdělil do třech logických bloků. Jedná se o soběstačné celky, při kterých jsou nohy robota umístěny tak, že robot neztratí rovnováhu.

Toho je docíleno v pohybech *A* a *B* tím, že vždy tři z nohou přímo vystoupí z rohů pod 45° úhlem, díky čemuž vždy těžiště zůstane uvnitř trojúhelníku, který nohy utváří. To umožňuje volný pohyb 4. nohy, která stojí mimo tento trojúhelník, aniž by tím bylo těžiště jakkoliv ovlivněno. V pohybu *C* svírají dva páry 45° a 90° úhly, čímž formují rovnoramenný lichoběžník, u kterého se těžiště sice mírně vychýlí ze středu, a to k páru nohou, jenž vytyčují 90° úhly, avšak zůstane v rámci těla, a tak robot neztratí balanc.



Obrázek 4: Schéma 2 konečných pozic kroků vpřed A(2), B(0,2), C(1), respektive A(0) B(2,0) C(3)

Celý proces je ohraničen počtem kroků, neboť je jinak indefinitní. Jeden krok značí sekvenci pohybů *A*, *B*, *C*, po jejímž ukončení dochází ke kroku druhému, jež je navazující sekvencí identických pohybů, avšak zrcadlových motorů. Po skončení převrácených pohybů se robot opět vrátí do prvotní pozice a započne tak další pár kroků.

```
33 void pohyb_vpred(unsigned int krok){
34   while (krok-- > 0){
35     if (soucasna_pozice[2][1] == y_start){
36       pohyb_vpredA(2);
37       pohyb_vpredB(0,2);
38       pohyb_vpredC(1);}
39     else{
40       pohyb_vpredA(0);
41       pohyb_vpredB(2,0);
42       pohyb_vpredC(3);}}}
```

Pohyb *A* je zodpovědný za napřažení přední nohy co nejdále od robota ve směru chůze. Nejprve se nadzdvihne, posléze pootočí na ose *y* a nakonec dopadne zpět na zemskou z souřadnici.

```
43 void pohyb_vpredA(byte noha){
44   rychlost_pohybu = rychlost_pohybu_nohy;
45   nastavit_pozici(noha, x_start, y_start, z_start, 1);
46   nastavit_pozici(noha, x_start, y_start + 2 * y_krok, z_start, 1);
47   nastavit_pozici(noha, x_start, y_start + 2 * y_krok, z_krok, 1);}
```

Pohyb *B* je nejdůležitější, neboť jako jediný rozhýbe celé tělo. Zprvu oddálí pár nohou, jež se nachází na straně opačné vůči té z pohybu *A*, čímž posune tělo protichůdným směrem, tedy vpřed. Následovně přisune zbylé 2 nohy k tělu pro udržení stability.

De facto se celý robot jako krakatice pohne směrem k vyčuhující noze z pohybu *A*, a to odražením protějších nohou od země.

```
48 void pohyb_vpredB(byte noha0, byte noha1){
49   rychlost_pohybu = rychlost_pohybu_tela;
50   nastavit_pozici(noha0, x_start, y_start, z_krok, 0);
51   nastavit_pozici(noha0 + 1, x_start, y_start + 2 * y_krok, z_krok, 0);
52   nastavit_pozici(noha1, x_start, y_start + y_krok, z_krok, 0);
53   nastavit_pozici(noha1 + 1, x_start, y_start + y_krok, z_krok, 1);}
```

Pohyb *C* uvede křížovou nohu vůči té z *A* do pozice, ze které lze opětovně navázat pohybem *A*. Stejně jako v *A* nohu nadzdvihne, pootočí na *y* ve směru pohybu a nechá dopadnout. Oproti *A* ale z hlediska hodnot *y* dochází ke zcela opačnému jevu, neboť se noha navrací z pohybu *B*, a proto se musí na ose y na rozdíl od *A* odečítat.

```
54 void pohyb_vpredC(byte noha){
55   rychlost_pohybu = rychlost_pohybu_nohy;
56   nastavit_pozici(noha, x_start, y_start + 2 * y_krok, z_start, 1);
57   nastavit_pozici(noha, x_start, y_start, z_start, 1);
58   nastavit_pozici(noha, x_start, y_start, z_krok, 1);}
```

5..7 Převaděč a ovladač motorů

Nejprve je třeba zjistit, zda je potřeba daným servomotorem hýbat. Hýbání je totiž zapotřebí, pouze pokud je vzdálenost pozice budoucí od té současné větší nežli dálka, kterou je daný motor v intervalu schopen překonat (*budouci_rychlost_pohybu*). V opačném případě je pohyb ukončen přenastavením současné pozice na budoucí.

Pro servomotory, které se mají účastnit pohybu, je pak třeba definovat úhly alfa, beta a gama, které určí vychýlení osičky, která modul, jež je na ni připevněn, rozpohybuje. Servomotory totiž podporují pouze hodnoty od 0 do 180 °, a tak je třeba převodu souřadnic z kartézské soustavy na polární a z ní pak konečně na obloukovou míru. K tomu slouží vzorce pro inverzní kinematiku, které vychází z kosinových vět. Nakonec je ještě třeba pro úhel vyřešit otázku orientace - zda servomotor leží na straně pravé, či levé:

$$x \geq 0 \ldots j = 1 \ \wedge \ x < 0 \ldots j = -1$$
$$v = \left(j \cdot \sqrt{x^2 + y^2}\right) - 27{,}5$$

$$i \in \{0, 3\} \ldots k = \ -1 \ \wedge \ i \in \{1, 2\} \ldots k = 1$$
$$\alpha = 90 + k \cdot \left(\frac{180}{\pi} \cdot \left(atan2(z, v) + acos\left(\frac{a^2 - b^2 + v^2 + z^2}{2a\sqrt{v^2 + z^2}}\right)\right)\right)$$
$$\beta = 90 + k \cdot 90 - k \cdot \left(\frac{180}{\pi} \cdot \left(acos\left(\frac{a^2 + b^2 - v^2 - z^2}{2ab}\right)\right)\right)$$
$$v \geq -27{,}5 \ldots l = 1 \ \wedge v < -27{,}5 \ldots l = -1$$
$$\gamma = 90 - k \cdot \left(\frac{180}{\pi} \cdot \left(atan2(l \cdot y, l \cdot x)\right)\right)$$

K vzorcům samotným jsou nutné rozměry a i b robota, tedy frontální a boční části základny, kromě toho pak Ludolfovo číslo.

Po převedení ze souřadnic na orientovaný úhel ve stupních se pro všechny tři osy dané nohy servomotory pohnou.

```
59 void ovladac_servomotoru(void){
60   static float alfa, beta, gama;
61   for (int i = 0; i < 4; i++){
62     for (int j = 0; j < 3; j++){
63       if (abs(soucasna_pozice[i][j] - budouci_pozice[i][j]) >=
         abs(budouci_rychlost_pohybu[i][j]))
64         soucasna_pozice[i][j] += budouci_rychlost_pohybu[i][j];
65       else
66         soucasna_pozice[i][j] = budouci_pozice[i][j];}
67     volatile float x = soucasna_pozice[i][0], y = soucasna_pozice[i][1], z =
       soucasna_pozice[i][2];
68     const float delka_a = 56.5, delka_b = 79.1, pi = 3.1415926;
69     int k = ((i == 0) ||(i == 3)) ? (-1) : (1);
70     float v = ((x >= 0 ? 1 : -1) * (sqrt(sq(x) + sq(y)))) - 27.5;
71     alfa = 90 + k * ((atan2(z, v) + acos((sq(delka_a) - sq(delka_b) + sq(v) +
       sq(z)) / 2 / delka_a / sqrt(sq(v) + sq(z)))) / pi * 180);
72     beta = 90 + k * 90 - k * ((acos((sq(delka_a) + sq(delka_b) - sq(v) - sq(z))
       / 2 / delka_a / delka_b)) / pi * 180);
73     gama = 90 - k * (((v >= -27.5) ? atan2(y, x) : atan2(-y, -x)) / pi * 180);
74     const float uhel[3] = {alfa, beta, gama};
75     for (int j = 0; j < 3; j++){
76       servomotor[i][j].write(uhel[j]);}}}
```

5..8 Vymezovač pozice

Daný *void* blok je nezbytný pro nastavování pozic, kam se mají servomotory hýbat.

Má-li se na dané ose servomotor hýbat, změří se délka, kterou má na ose překonat. Z *délek x, y, z* se stanoví celková *délka*, která je nakonec použita k nastavení *budoucí rychlosti pohybu*, tedy délky pohnutí, kterou má v časovém úseku motor stihnout vykonat. Zároveň předefinuje nastavené souřadnice jako budoucí pozice, ke kterým se budou motory již spočítanou rychlostí přibližovat.

Důležitou součástí je funkce, která počká na dokončení veškerého pohybu. Jedná se o jednoduchý opakovač, který poběží do té doby, než všechny motory dosáhnou kýžených pozic, což je velice důležité pro synchronní pohyb celého robota, neboť by bez této části byly některé nohy napřed a jiné pozadu, čímž by došlo k nežádoucímu vychýlení těžiště ven.

```
77 void nastavit_pozici(int noha, float x, float y, float z, boolean
   pockat_na_dokonceni_pohybu){
78   volatile float xyz[3] = {x, y, z};
79   float delka_x, delka_y, delka_z;
80   float delkaxyz[3] = {delka_x, delka_y, delka_z};
81   for (int i = 0; i < 3; i++){
82     if (xyz[i] != nehybat) delkaxyz[i] = xyz[i] - soucasna_pozice[noha][i];
83     float delka = sqrt(sq(delkaxyz[0]) + sq(delkaxyz[1]) + sq(delkaxyz[2]));
84     budouci_rychlost_pohybu[noha][i] = delkaxyz[i] / delka * rychlost_pohybu *
        modifikator_rychlosti;
85     if (xyz[i] != nehybat) budouci_pozice[noha][i] = xyz[i];}
86   if (pockat_na_dokonceni_pohybu)
87     for (int i = 0; i < 4; i++)
88       for (int j = 0; j < 3; j++)
89         while (1)
90           if (soucasna_pozice[i][j] == budouci_pozice[i][j])
91             break;}
```

6 ZÁVĚR

V rámci práce se mi úspěšně podařilo rozchodit robota vpřed. Zároveň jsem dospěl k vytvoření ovladače, jenž lze v budoucnosti použít pro další robotí chůzi, a to třeba vzad nebo do strany.

Robot je zároveň připraven k dalším více soběstačným vylepšením, jakými mohou být instalace externího zdroje napájení, zavedení čidla pro monitoring prostředí, spárování pro bezdrátové ovládání skrze Bluetooth či hlas, nebo zabudování čidla, schopného vyhodnotit bezpečnost cesty či navrhnout její trasu.

7 PŘÍLOHA 1: VIDEA

Namísto exhibice naživo byla robotí chůze zdokumentována a nahrána na platformu YouTube.

7..1 Hyperrychlá chůze

Trojnásobná rychlost všech pohybů a 0ms pauza mezi 70 kroky A, B, C.

https://youtu.be/2IphBBQ6WMI

7..2 Pomalá chůze

Poloviční rychlost všech pohybů a 500ms pauza mezi 10 kroky A, B, C.

https://youtu.be/WiGCmEijPZY

7..3 Hyperpomalá chůze

Poloviční rychlost všech pohybů a 1000ms pauza mezi 15 kroky A, B, C.

https://youtu.be/j8jVyETSt_w

8 PŘÍLOHA 2: KÓD

Kolekce všech souborů.

https://drp.mk/183vsNyPj7

8..1 .INO

Online verze .ino souboru, tedy veškerého kódu, jenž je zodpovědný za pohyb robota.

https://martinznamenacek.dropmark.com/817395/23253553

8..2 .TXT

Online verze .ino souboru ve formátu .txt, jenž lze otevřít na jakémkoliv zařízení.

https://martinznamenacek.dropmark.com/817395/23253570