

Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Járó robot készítése

Babics Mátyás

Konzulens

Nagy Ákos

Budapest, 2018

# Bevezetés

A feladat egy mozgásra képes robot elkészítése volt, az ehhez tartozó modell és a vezérlést végző hardware és software tervezése és elkészítése. A robot egy quadruped lett, vagyis négy lába van. Ezen helyezkedik el az akkumulátor és a vezérlő áramkör. Az irányító egység fő feladata a lábak motorjaihoz vezérlő jelet biztosítani a kívánt mozgás eléréséhez, azaz kiszámolni az egyes lábak által elérendő pozíciót, ezalapján meghatározni az egyes motorok állását, majd az ezt eredményező jelet kiadni. Ezen felül a project előre haladtával az illesztett eszközöket, mint távolságérzékelőt és wifi modult fog kezelni.

# Modell

A robot modellje 3D nyomtatással készült, amihez a tervek Autodesk Inventorral készültek. A modell fő részei a teste és 4 lába, amik további 3 részre bonthatók. Az egyes részek egy-egy dimenzióban el tudnak forogni egymáshoz képest. A lábaknak 3 szabadságfokuk van a tetszőleges pozíció eléréséhez.

Készített alkotóelemek (láb részei felső sorban, test részei és motor alsó sorban):

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Láb vége | Motortartó | Lábszár |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Test fő része | Test fő részek között támasz | Motor |

Az alkatrészeken látható nagyobb lyukak 2mm, a kisebbek 1mm átmérőjűek. A részeket 2mm-es csavarok tartják össze a nagyobb lyukakba csavarva. A kisebb lyukak a használt motorok fejéhez lettek mérve. A fejet megfogva a motor forgó részéhez rögzítik magukat mozgást eredményezve. Az elemek vastagsága 3mm.

A láb vége elem egy motortartóhoz illeszkedik, meghosszabbítva a láb végső szakaszát 80mm-re. Vége lekerekített, így orientációja nem sokat befolyásolja a talajjal érintkezés pontos helyzetét a testhez képest. A robottól független tökéletlenségek miatt, mint például egyenetlen talaj, praktikusan nem lehetséges kiszámolni a láb pontos helyzetét.

A motortartó feladata a motor hozzácsatolása a modell többi részéhez. A motor fülein 2mm-es lyukak vannak, amikkel rögzíthető az álló rész. A tartó ezeken fogja a motort.

Lábszárból 2-t egymáshoz csavarva H alakot kapva adódik a láb középső része. A tengelyek között 30mm távolságot tart. A két fül között 34mm van, ide illeszkedik a motortartó és a motor.

A test fő részéből 2 van egymás fölött, közöttük 4 támasz helyezkedik el. A test 96mm hosszó és 86mm széles. A támaszok 34mm hosszúak, ugyanakkora helyet adva a motortartók-motor együttesnek, mint a lábszár. A lábak a testen látható négy elálló fülön helyezkednek el.

|  |  |
| --- | --- |
|  |  |
| Test | Láb |
|  | C:\Users\Matyi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\20180519_164855.jpg |
| Robot modellje | Robot egy lábbal |

A modell dizájnjához egy létező quadruped adta az inspirációt. A méretek meghatározásának fő szempontja a motorok és vezérlést biztosító nyák mérete volt. A végtagok a testhez képest hosszúak lettek, hogy messze elérjenek. A láb középső része (30mm) lényegesen rövidebb a végső részénél (80mm), így a robot teste a talaj fölött maradhat úgy is, hogy a térde a test fölött van. Ez tisztán kinézet kérdése, valójában nagyobb térfogat lenne lefedhető egyforma tengelytávolságokkal, viszont járásnál semmi megkötést nem jelent.

# Hardware

A hardware egy nyomtatott áramkörre készült el. A vezérlést egy STM32L476RET6 típusú mikrokontroller végzi. Ehhez órajelet egy 8MHz-es kristály oszcillátor biztosítja. A mikrokontrolleren 12 timer kimenet PWM módban vezéreli az SG-90 típusú motorokat. A tápellátást két sorosan kapcsolt 3.7V-os Li-ion akkumulátor adja. A feszültség meglétét egy kék LED jelzi. A feszültség egy harmadoló osztón csatlakozik a mikrokontroller egy analóg bemenetére. A 7.4V-ból egy PTH08T231WAD típusú konverter 5V-ot állít elő a motoroknak. 5V-ból egy TS1117BCW-3.3 típusú LDO 3.3V-ot állít elő a mikrokontrollernek.

Ezeken kívül van a nyákon:

* 3 nyomógomb, 1 resethez, 2 a mikrokontrollerre bemenetnek.
* 3 sárga LED tetszőleges jelzéshez, és 1 piros akkumulátor töltöttsége jelzéséhez
* 5V, 3.3V, GND kivezetés
* UART kommunikáció külső eszközzel
* Programozó interfész
* ESP alapú wifi modul
* Sharp sensor távolságérzékelő
* 8 GPIO

## Mikrokontroller

A választás egy STM32L476RET6 típusú mikrokontrollerre esett. A választás szempontjai:

* Megfelelő számú kimenet, ebből 12 PWM
* Elegendő számítási teljesítmény, FPU
* Tokozás
* Fejlesztői környezet
* Ismeretség
* Ár

A választott mikrokontroller mindnek eleget tesz. A 12 PWM kimenetet TIM1, TIM2 és TIM3 időzítők biztosítják, mindegyik 4-4 kimeneten. USART3-n kommunikál a wifi modullal, UART4-en a kívülről illesztett egységgel, mint például számítógéppel. Az akkumulátor töltöttségét és a távolságérzékelő analóg kimenetét az ADC1 1-es és 2-es csatornáin figyeli. A PB0-PB7 lábak ki vannak vezetve tetszőleges felhasználásra. Az egyéb illesztett ki- és bemenetek elhelyezés szerint kedvező GPIO lábakra csatlakoznak.

## Motor

A motorok SG-90 típusú mikroszervók. Táplálásuk 5V-ról történik. Vezérlésükhöz 50Hz-es PWM jelet kell bemenetükre adni adatlap szerint 1-2ms-es kitöltési tényezővel, ami hatására 0-180°-ba fordulnak. A lábak pozícióba állításához ez kifejezetten előnyös. A motor továbbá kis méretű, könnyű és szolgáltat elegendő nyomatékot.

## Táp

Az energiaellátást két sorba kapcsolt 18350 Li-ion akkumulátor szolgálja, kapcsolóval megszakítható. Ennek feszültsége 7.4-8.2V.

Ebből egy PTH08T231WAD DCDC konverter 5V-ot állít elő. A konverter bemeneti feszültsége 4.5V-tól 14V-ig terjedhet, kimeneti feszültsége egy ellenállással állítható. 160Ω-os ellenállás használatával a pontos kimenet 5.02V. Mivel a 7.4V-ot ez az átalakító, egy ellenállással sorosan kapcsolt LED és 3-mal leosztva a mikrokontroller egy analóg, 5V toleráns lába kapja meg, a tápellátás értéke felmehet 14V-ig, viszont 9.9V fölött az analóg bemeneten információ veszik el. Nagyobb feszültség hatására egyéb mellékhatás a tápellátást jelző LED erősebb fénye. A konverter maximális kimeneti árama 6A, hatásfoka 1A-től 85% fölött van, és akár 95%-ot is elérhet.

A mikrokontrollerhez szükséges 3.3V-ot egy TS1117BCW-3.3 típusú LDO állítja elő 5V-ból. Ugyan itt rosszabb a hatásfok, a konverter kis mérete és a rajta átfolyó kis teljesítmény miatt esett rá a választás.

## Nyák

A motorok csatlakozója 3-mas csoportokban a nyák négy sarkánál vannak. A táp a nyák hátulján (képen jobb oldal) érkezik, és itt konvertálja át 5V-ra és 3.3V-ra. Itt helyezkedik el a kapcsoló, a bekapcsolást jelző kék LED és az akkumulátor merülését jelző piros LED. A 3 nyomógomb és 3 sárga LED a nyák elején (képen bal oldal) vannak. A mikrokontroller a nyák közepére került, hogy a legegyszerűbben ki lehessen vezetni a lábakat az adott eszközökhöz. E fölött és alatt van a kristály oszcillátor és a kivezetések programozáshoz és illeszthető eszközökhöz.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Nyák felső rétege | Nyák alsó rétege | Forrasztott nyák |

# Software

A mikrokontrolleren futó program C-ben készült a System Workbench for STM32 eclipse alapú IDE használatával. A perifériák kezeléséhez tartalmazza az LL (low layer) könyvtárat, amibe típusdefiníciókat, függvényeket és makrókat implementáltak megkönnyítve a programozói munkát. A projectet a CubeMX software generálta. Ezzel létrejött a perifériák inicializálásának alapja.

## Időzítők

A motorok vezérléséhez 12 PWM jelre volt szükség. Ezeket a TIM1, TIM2 és TIM3 szolgálta 4-4 csatornán. Egy időzítő egy meghatározott frekvencián tud működni, amihez egy prescaler érték és egy autoreload érték tartozik, és minden időzítő ugyan arról az órajelforrásról jár, ami a 80MHz-es órajel. Ez a kristály oszcillátor jele PLL-lel 10-szeresére gyorsítva. A prescaler azt határozza meg, hogy mennyi órajel után lépjen egyet a számláló, az autoreload pedig azt, hogy mekkora értéknél lépjen vissza 0-ra. Ezzel az időzítő frekvenciája ahol az oszcillátor frekvenciája, és sorra a prescaler és autoreload. PWM kimenethez csatornánként megadható egy összehasonlító érték, amit ha az időzítő elér, alacsony szintre váltja a kimenetet. A következő periódus kezdetén visszavált magasra. Ezáltal csatornánként változtatható a kitöltési tényező. Mivel a motorok mind egyforma frekvencián működnek és csak a pulzus hossza szükséges eltérjen, ez semmi problémát nem jelent. Mindegyik időzítő ugyanúgy lett beállítva. A prescaler értéke 49 lett, az autoreload értéke 31999. Az elért frekvencia 50Hz, és a PWM jel kitöltési tényezőjének a felbontása , ami ~0.1°-os pontosságot jelent.

A TIM4-es időzítő másodpercenként 50-szer interuptot eredményez, ami által hívott függvény növel egy globális változót mutatva a program futásának idejét, és igazra állít egy *update* nevű változót. A fő ciklus ellenőrzi, hogy az *update* igaz-e, és ha igen, hamisra állítja, és futtatja a ciklikusan futtatandó műveleteket. Mivel ezek a műveletek sokáig tarthatnak, interupton kívül futnak. A program itt fogja kiszámolni a lábak pozícióját a járáshoz, és abból az egyes motorok állását. Mivel 50Hz-es PWM jel vezéreli a motorokat, legfeljebb 50-szer lehet változtatni a helyzetüket, ezért a TIM4 frekvenciája.

## Szimuláció

A robotnak jelenleg egy lába van, és azt tudja mozgatni és a rajta lévő LEDeket villogtatni, viszont a hardware és modell tervezése előtt-közben készült egy számítógépes szimuláció, ami más paraméterű ugyan, de azon kívül a robottal megegyező. A láb pozíciók számításában az offsetek átírásával az algoritmus használható a valódi roboton.

Ez a szimuláció C++-ban készült, a megjelenítéshez DirectX 11-et használ. A robotot modellje téglatestokból áll, minden külön mozgó elemhez egy téglatest. A szimuláció minden megjelenítés előtt kiszámolja a lábak következő helyzetét a megadott útvonal alapján, és ebből meghatározza az egyes csuklók állását, majd az ezek alapján eredményezett transzlációs és rotációs mátrixokkal a megfelelő helyre rajzolja fel a testrészeket. A szimulációban nincs fizika, az alakzatok mindig annyival előrébb kerülnek, amennyivel a földet érő lábaik hátrább, ezzel lépés illúzióját keltve. A modellek minden 6 másodpercben visszakerülnek a kezdő pozíciójukba, hogy ne menjenek ki a képből. A programban lehetőség van mozgásra a számítógépes játékokból ismert WASD + egér használatával.

|  |
| --- |
|  |
| Screenshot a szimulációról. Kúszó mozgás bal oldalt, sétáló jobb oldalt. |

## Mozgás leírása

A robothoz két mozgás került leírásra: kúszó mozgás és sétáló mozgás.

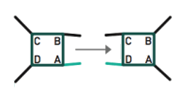
### Kúszó mozgás

Kúszó mozgás közben a robotnak 3 lába mindig a földön van, súlypontja ezek által alkotott háromszögön belülre esik. Ennek a mozgásnak előnye, hogy stabil, bármikor megállhat mozgás közben, nem dől el, hátránya, hogy lassú. A mozgás hat lépésből áll.

|  |
| --- |
| https://i0.wp.com/makezine.com/wp-content/uploads/2017/05/QuadFigureB.png?resize=620%2C190 |
| Kúszó mozgás lépései  *(forrás:* [*https://makezine.com/2016/11/22/robot-quadruped-arduino-program/*](https://makezine.com/2016/11/22/robot-quadruped-arduino-program/)*)* |

A kezdeti állapotot a hatodik ábra mutatja a fenti képen. Először jobb hátsó lábát előre viszi 2 egységgel, majd jobb első lábát szintén előre viszi 2 egységgel. Ebből az állapotból előre viszi a testét, vagy a robot inerciarendszere szerint hátra tolja mind a négy lábát 1 egységgel. Ugyanezt megcsinálja a bal lábakkal, majd kezdi elölről. Az egység hossza szabadon választható, határai, hogy az első ábrán látható helyzetben a két jobboldali láb ne ütközzen, és hogy a második ábrán a jobb első láb el tudja érni a helyzetet.

### Sétáló mozgás

Sétáló mozgás közben a robotnak 2 átellenes lába a földön van, a másik kettő lép. Súlypontja a két földön lévő lábát összekötő egyenes közelében van, így a lépés idejére nem dől sokat. Előnye a kúszó mozgáshoz képest gyors haladás, hátránya, hogy nem stabil, ha lassan végzi a lépést, rádől az egyik lépő lábára.

Ennek a mozgásnak lényegesen egyszerűbb a leírása. Első lépésben jobb első és bal hátsó lábát megemelve előre, másik két lábát a földön hátra viszi egy egységgel, tolva magán egy egységet, majd ugyanezt felcserélve a lábakat. Az egység hosszát a kúszó mozgásnál említett korlátozások befolyásolják.

## Lábak inverz kinematikája

A használt koordinátarendszer szerint *x* irány vízszintesen jobbra, *z* irány a kép belseje felé, *y* irány felfele megy a szimulációban használt Direct3D szerint. Origónak a láb tövét lehet tekinteni, ami a robot testéhez képest mindig ugyanott van, így eltolással a testen bárhova áthelyezhető. A láb végpontjainak koordinátái *x*, *y* és *z*. Ez első csukló (a képen 1-gyel jelölve) állása könnyen meghatározható:

A további számításokhoz érdemes bevezetni *r*-t, ami a láb végének távolsága *x*, *z* irányokat tekintve az origótól:

Továbbá legyen:

Ezzel 2 dimenzióra lett redukálva a probléma.

A *C* és *D* pontok *y* koordinátája megegyezik. Az *DAC* derékszögű háromszög *C* sarkánál lévő szög:

Ahol:

Ilyenkor *BAC* háromszögre koszinusz tételt felírva:

Ahol:

Itt a második csukló állása, hogy vízszintestől mennyit emelte fel a lábat.

Ebből:

Az *ABC* háromszögre felírva a koszinusz tételt megkapható a harmadik csukló állása (ábrán 3-mal jelölve):

