

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 10: Elektrotechnika, elektronika a telekomunikace

Lidarový skener

Tadeáš Fabián

Jihočeský kraj

Tábor, 2024

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 10: Elektrotechnika, elektronika a telekomunikace

Lidarový skener

Autor: Tadeáš Fabián

Škola: Střední škola spojů a informatiky, Bydlinkého 2474,
390 02 Tábor

Kraj: Jihočeský kraj

Konzultanti: Ing. Dana Almášiová, Ing. Vladimír Čebiš

Tábor, 2024

Čestné prohlášení

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

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

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 Táboře dne 7. dubna 2024 Tadeáš Fabián

Anotace

Tato práce se věnuje popisu lidarových zařízení, zhotovení vlastního lidarového skeneru a vytvoření grafické aplikace pro obsluhu tohoto skeneru.

Klíčová slova

LIDAR, TOF, ATmega328PB, UART, detekce překážek, skenování prostoru, 2D

Obsah

1	Co je to LiDAR	8
2	Klíčové technologie	9
2.1	Metody měření vzdálenosti	9
2.1.1	Direct ToF	9
2.1.2	AMCW ToF	10
2.1.3	FMCW ToF	10
2.1.4	Triangulace	10
2.2	Metody skenování	11
2.2.1	Mechanické	11
2.2.2	MEMS	11
2.2.3	OPA	12
2.2.4	Vrstvení senzorů	12
2.2.5	Flash	12
2.3	Dodatečné měření	12
3	Analýza dostupných řešení	13
4	Elektronická konstrukce	15
4.1	Komponenty	15
4.2	Zapojení	15
4.3	DPS	16
5	Mechanická konstrukce	17
5.1	Základna	17

5.2	Otočný skener	18
6	Firmware pro zařízení	21
6.1	Komunikace lidarů s externím zařízením	23
6.1.1	Formát paketů	23
6.2	Nahrávání firmwaru na mikrokontrolér	25
6.2.1	ISP	25
6.2.2	Bootloader	26
7	Klient	27
7.1	Terminálový klient	27
7.2	Grafický klient	27
7.2.1	Funkce grafického klienta	28

Úvod

S nástupem digitálního věku vznikla potřeba transformovat podobu fyzického světa do podoby, kterou dokáže zpracovat počítač. Existuje celá řada způsobů, jak toho dosáhnout, ale v případě potřeby přesných a spolehlivých měření této oblasti dominuje lidar.

Cílem této práce je navrhnout, naprogramovat a vytvořit funkční lidar, který lze snadno začlenit do budoucích projektů, a vytvořit počítačový program schopný demonstrovat některé schopnosti lidarů.

Kapitola 1

Co je to LiDAR

LiDAR (*Light Detection And Ranging*) je metoda určování vzdálenosti vyzařováním pulsků světla a měřením doby nutné pro jejich odrazení a návrat.

Výstupem lidaru je point cloud. Point cloud je množina bodů v prostoru, reprezentující libovolný tvar nebo objekt. Podle typu lidaru může výsledný point cloud obsahovat jeden bod, představovat dvourozměrnou část prostoru nebo trojrozměrný prostor. Lidar také dokáže snímat některé vlastnosti pozorovaných objektů.

Lidar lze použít v těchto odvětvích:

Totální stanice a digitální metry využívají lidar pro měření vzdálenosti.

robotika a autonomní vozidla využívá lidar pro detekci překážek.

mapování terénu využívá lidar pro vytvoření trojrozměrné mapy terénu.

3D rekonstrukce využívá lidar pro skenování trojrozměrných objektů.

SLAM je kombinace robotiky s mapováním terénu. Pohyblivý robot s lidarem projíždí neznámým terénem a zaznamenává data lidaru, ze kterých poté vytvoří mapu terénu.

Kapitola 2

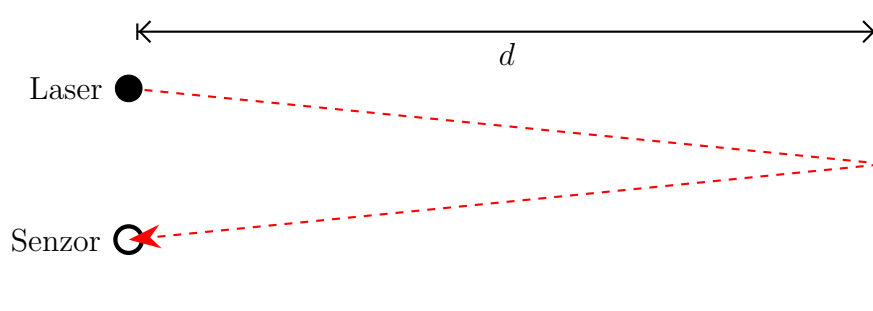
Klíčové technologie

Lidary lze rozdělit podle technologií, které používají.

2.1 Metody měření vzdálenosti

2.1.1 Direct ToF

Time of Flight je čas, který světelný paprsek potřebuje k překonání neznámé vzdálenosti. Tento časový údaj lze využít k výpočtu vzdálenosti.



Obrázek 2.1: princip ToF

Vysíláním krátkých světelných paprsků a měřením času τ potřebného pro jejich návrat, můžeme pomocí rychlosti světla c vypočítat vzdálenost d od sledovaného objektu.

$$d = \frac{c}{2} \cdot \tau \quad (2.1)$$

2.1.2 AMCW ToF

S menšími vzdálenostmi je přímé měření ToF poměrně složité a nákladné. Přímému měření ToF se můžeme vyhnout použitím **Amplitude Modulated Continuous Wave**.

Při použití AMCW vysíláme kontinuální laserový signál modulovaný určitou frekvencí f . Když se odraz tohoto signálu vrátí, můžeme porovnat jeho frekvenci s originální frekvencí. Tímto zjistíme že došlo k fázovému posunu $\Delta\phi$. Pomocí fázového posunu $\Delta\phi$ a modulační frekvence f lze vypočítat dobu letu τ následně.

$$\tau = \frac{\Delta\phi}{2\pi} \cdot \frac{1}{f} \quad (2.2)$$

Vzdálenost d se tedy vypočítá takto.

$$d = \frac{c}{2} \cdot \frac{\Delta\phi}{2\pi} \cdot \frac{1}{f} \quad (2.3)$$

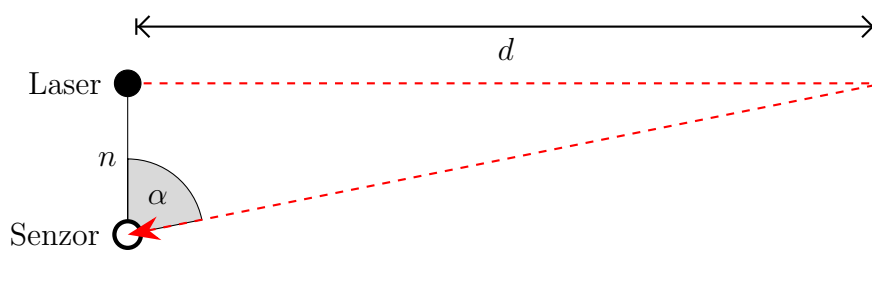
2.1.3 FMCW ToF

Frequency Modulated Continuous Wave využívá dopplerova jevu pro získání relativní rychlosti sledovaných objektů. Toto však funguje pouze v případě, že se objekty pohybují směrem k nebo od lidaru, pokud se objekty pohybují do stran, není možné touto metodou získat jejich rychlost.

2.1.4 Triangulace

Triangulace je metoda, při které se neměří ToF, ale měří se úhel vracejících se světelných paprsků.

d je vzdálenost kterou pozorujeme, n je vzdálenost mezi senzorem a laserem, α je úhel pod kterým se vracejí světelné paprsky.



Obrázek 2.2: prnip triangulace

Po změření úhlu můžeme pomocí jednoduché trigonometrie vypočítat vzdálenost sledovaného objektu.

$$d = n \cdot \tan \alpha \quad (2.4)$$

2.2 Metody skenování

K nasměrování paprsků lidarů do různých směrů potřebujeme metodu skenování.

2.2.1 Mechanické

Nejjednodušší a nejrozšířenější metoda. Fyzicky pohybuje lidárovým senzorem v požadovaných směrech. Nejčastěji se jedná o motor, který nepřetržitě otáčí senzorem o 360° .

2.2.2 MEMS

Micro-ElectroMechanical Systems jsou malá zařízení, která obsahují mechanické i elektronické součásti. V tomto případě se jedná o drobné elektromechanicky pohyblivé zrcátko. Místo mechanického pohybu lidárovým senzorem pouze přeměrujeme vyzářené a příchozí světelné paprsky. Touto metodou lze dosáhnout většího rozlišení snímání, ale je podstatně obtížnější snímat celý 360° záběr

2.2.3 OPA

Optical Phased Array je řada emitorů schopná měnit směr vysílaného signálu změnou fázového posunu jednotlivých emitorů. OPA potřebují nízké řídicí napětí, ale maximální úhel snímání je pouze $\pm 10^\circ$.

2.2.4 Vrstvení senzorů

Pokud vedle sebe umístíme několik lidarových senzorů, můžeme sledovat více různých směrů současně. Toto se často používá v kombinaci s mechanickým skenováním k získání trojrozměrného obrazu.

2.2.5 Flash

Flash lidar funguje podobně jako fotoaparát s bleskem. Osvětluje celou snímanou plochu jedním zábleskem a poté zachycuje vracející se světlo pomocí světlocitlivé plochy. Na rozdíl od fotoaparátu však jednotlivé pixely světlocitlivé plochy nezachycují intenzitu světla, ale místo toho zachycují ToF světla.

Výhodou zábleskových lidarů je, že neobsahují žádné pohyblivé části. Mají však omezený rozsah měření vzdálenosti. Navíc je rozlišení fotosenzitivní plochy značně omezeno velikostí pixelů měřících ToF.

2.3 Dodatečné měření

Při odrazu světelného paprsku vysílaného lidarem od sledovaného objektu dochází ke změnám některých vlastností tohoto světelného paprsku. Zkoumáním těchto vlastností po návratu paprsku můžeme získat doplňující informace o vlastnostech sledovaného objektu.

Podle intenzity navraceného paprsku je možné určit optické vlastnosti objektu. Například odrazivost nebo absorpci.

Kapitola 3

Analýza dostupných řešení

YDLIDAR X4 je 360° dvourozměrový LiDAR, založený na technologiích triangulace a mechanického skenování.

Rychlost skenování je nastavitelná v rozmezí 36-72 rpm. Triangulační senzor provádí své měření vzdálenosti rychlostí 5000Hz, a dokáže měřit vzdálenosti v rozsahu 0,12-10 metrů.

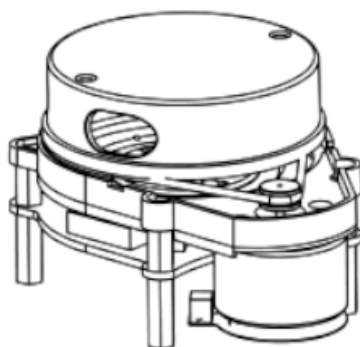
Inzerované způsoby využití jsou:

- Navigace robotů a vyhýbání se překážkám
- Výuka a výzkum robotických systémů ROS
- Skenování prostředí a 3D rekonstrukce
- Navigace a vyhýbání se překážkám u domácích robotů/robotů vysavačů

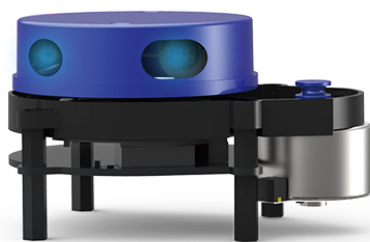
Základna zařízení obsahuje motor pro pohon otočného skeneru a základní desku lidarů. Přes ložisko je k základně připevněn otočný skener který obsahuje triangulační senzor.

Napájení otočné části lidarů je prováděno elektromagnetickou indukcí. Přenos dat mezi otočným skenerem a základnou je zajištěn opticky.

Lidar disponuje sériovým rozhraním po kterém, po připojení externího systému, posílá point cloud data.



Obrázek 3.1: schéma YDLIDAR X4



Obrázek 3.2: YDLIDAR X4

Kapitola 4

Elektronická konstrukce

4.1 Komponenty

Jako senzor vzdálenosti jsem zvolil TFmini-S. TFmini-S je AMCW ToF senzor se schopností měřit vzdálenosti v rozsahu 0,1-12 metrů. Naměřené hodnoty posílá sériovým portem.

Pro připojení TFmini-S ke zbytku zařízení budu používat kluzný kroužek.

Pro ovládání všech součástí lidarů využívám mikrokontrolér ATmega328PB. Mikrokontroléry stejné řady využívají Arduino jednodeskové počítače a je mi tedy tento čip známý. Další výhodou tohoto čipu je přítomnost dvou sériových rozhraní.

K otáčení konstrukce budu používat kartáčový motor s napětím 0,5 - 3 volty.

Pro detekci dokončení otočky budu používat Hallův senzor.

4.2 Zapojení

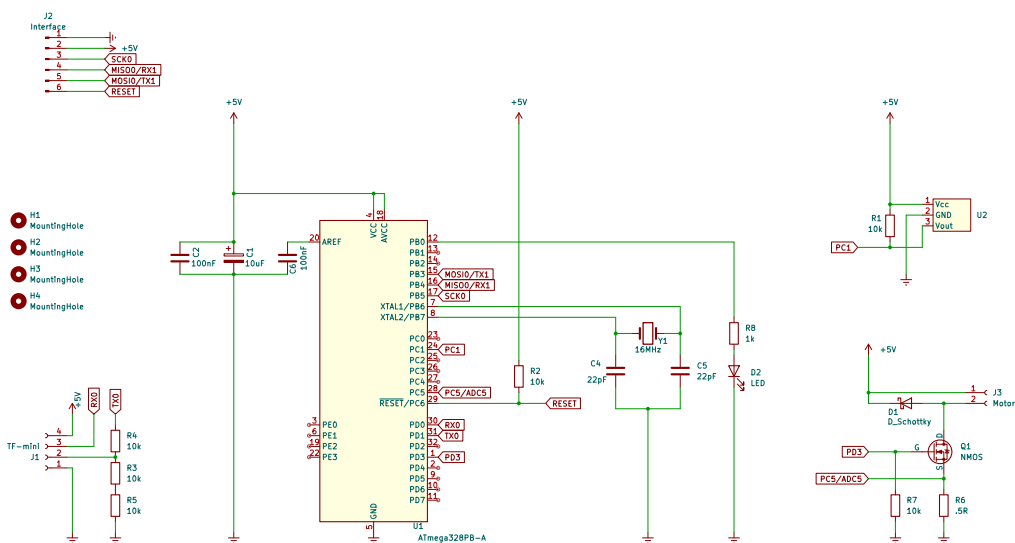
Pro hodiny čipu jsem použil standardní 16Mhz krystal.

Jako blokovací kondenzátory jsem použil jeden $10\mu\text{F}$ elektrolytický kondenzátor a jeden 100nF keramický kondenzátor.

Senzor TFmini-S vyžaduje napájení a sériové připojení. Napájení senzoru vyžaduje 5 voltů, ale sériové rozhraní sensor pracuje na 3,3 voltech. Z toho důvodu je k sériovému vývodu přidal dělič napětí.

Otáčky motoru jsou řízeny pomocí MOSFETu a do série s MOSFETem je přidán rezistor malé hodnoty, aby bylo možné měřit proudový odběr motoru.

Pro usnadnění vývoje firmwaru jsem přidal LED diodu.



Obrázek 4.1: schéma obvodu

4.3 DPS

DPS má rozměry 20 mm × 45 mm. U rohů ma 4 úchytné dírky pro M3 šrouby.

Hallův senzor je umístěn na DPS tak, aby se magnet, který je na otočném skeneru, mohl dostat do jeho dosahu.

Na stranu DPS jsem umístil konektorovou lištu s roztečí 2,54 mm. Tato lišta umožňuje jednoduché připojení externích zařízení k sériovému rozhraní mikrokontroléru ATmega328PB.

DPS jsem si nechal vyhotovit u JLCPCB.

Kapitola 5

Mechanická konstrukce

Cílem návrhu jsou dvě části, které lze snadno vytisknout na 3d tiskárně a které jsou konstrukčně robustní a vizuálně příjemné. V jedné části bude umístěn motor a DPS. Druhá část bude otočná a bude v ní umístěn senzor ToF.

5.1 Základna

V základně zařízení je umístěn motor a řídicí DPS.

Motor je k základně připevněn dvěma šrouby procházejícími rámem základny tak, aby jeho hřídel vyčnívala do horní části základny. Tam pomocí gumového řemene otáčí skenerem.

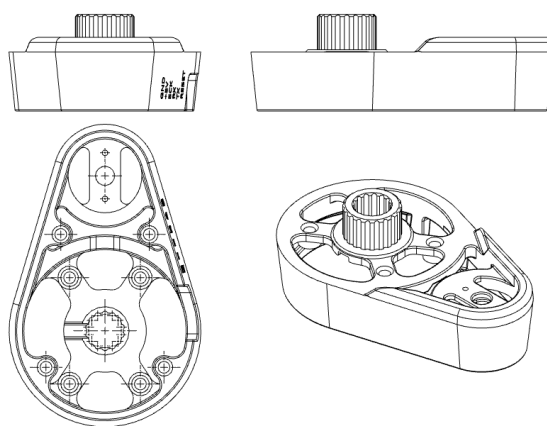
Řídicí DPS je přišroubována do 4 závitových vložek, tepelně vtlačených do základny. Také jsem do rámu základny umístil výřez pro Hallův snímač. Na boku základny je otvor pro snadný přístup k vývodům na DPS. Vedle otvoru jsem přidal popisky vývodů DPS.

Po vnitřním obvodu základny je umístěn žlábek pro vodiče motoru a kluzného kroužku.

V horní části základny se nachází výstupek určený k držení ložiska a kluzného kroužku. Ložisko se nasune na výstupek a poté se do něj zasune

kluzný kroužek. Pro zajištění těsného uchycení dílů nasazených na výstupek jsem přidal odlehčovací drážky[1].

Na dno základny jsem umístil 4 výstupky, každý s otvorem o průměru 4 mm. Účelem výstupků je upevnění zařízení. Do otvorů lze umístit závitové vložky M3, šrouby M4 nebo gumové držáky ventilátorů sloužící jako nožičky. Výstupky jsou umístěny tak, že sundání skeneru umožňuje přístup k výstupkům shora.



Obrázek 5.1: schéma základny

Ačkoli současná geometrie základny vyžaduje, aby byla vytištěna s podpěrami, všechny povrchy, které se dotýkají podpěr, jsou na vnitřní straně, a nemají tak žádný vliv na vizuální stránku základny.

5.2 Otočný skener

Aby bylo možné skener sundat. Je třeba, aby bylo možné odpojit ToF senzor od kluzného kroužku. K tomu slouží BLS konektor, připevněný v horní části skeneru. Pro umožnění přístupu ke konektoru je potřeba odnímatelné víčko. Po odpojení BLS konektoru lze celou sestavu skeneru sundat z ložiska.

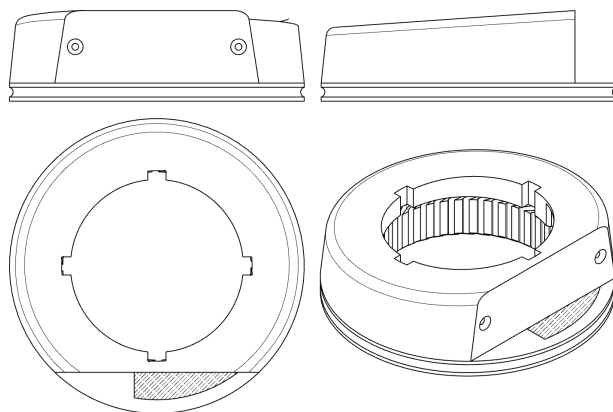
Skener je k ložisku přichycen třecími ploškami. To umožňuje opakované sundání skeneru, aniž by se snížila síla uchycení k ložisku.

Ve spodní části skeneru je umístěn otvor pro neodymový magnet. Do

otvoru jsem umístil odlehčovací drážky[1], aby magnet vždy těsně zapadl do otvoru a držel tam i bez lepidla.

V přední části skeneru jsem umístil výřez určený pro umístění ToF senzoru. Senzor drží na místě dva šrouby m2x12. Ve spodní části výřezu jsem umístil kanálek pro vodiče senzoru. Kanálek vede ze spoda od vývodů senzoru přes boční stěnu skeneru do horní části skeneru, kde jsou vodiče zakončeny konektorem BLS.

Po obvodu skeneru je žlábek pro gumový řemen.



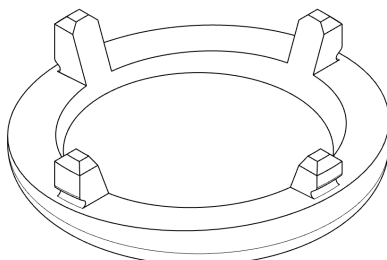
Obrázek 5.2: schéma otočného skeneru

Aby byl šikmý horní povrch hladký a vizuálně přitažlivý. Skener jsem vytiskl vzhůru nohama. Tím se zubatá textura, kterou 3d tiskárny vytvářejí na šikmých plochách, přesunula na spodní část skeneru, která je skrytá.

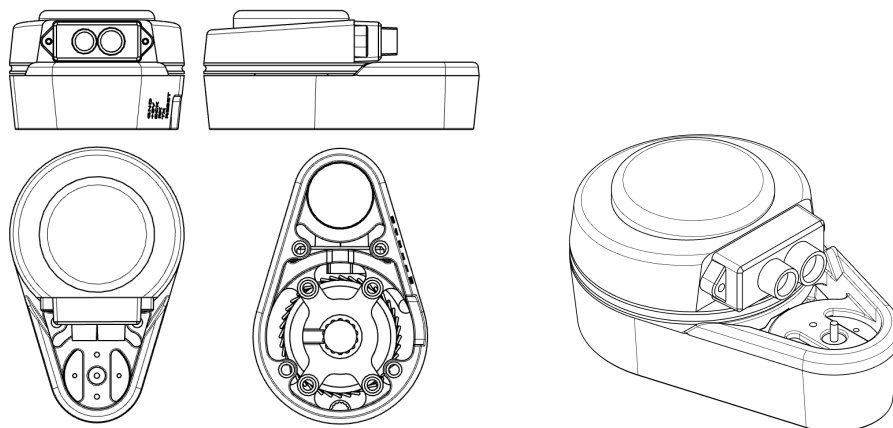
Tisk vzhůru nohama však přinesl jiný problém, a to ten, že okraj ve výřezu pro ToF senzor je nyní tištěn ve vzduchu a pod příliš malým úhlem, na to aby mohl být vytištěn bez podpěr. Podpěry v kombinaci se žlábkem pro dráty ToF senzoru velmi ztěžuje tisk této oblasti. Proto jsem do otvoru žlábků přidal obětní vrstvu. Která usnadňuje tisk tím, že umožňuje tiskárně přemostit z jedné strany výřezu na druhou v jedné souvislé linii, místo tvoření párů linií začínajících a končících ve vzduchu. Tato obětní vrstva se poté vyřízne. Ve schématu 5.2 je obětní vrstva vyšrafována.

Skener v současné podobě není vyvážený, a kdybychom s ním začali

otáčet, začal by vibrovat. Proto jsem při slicování nastavil určitou část zadní strany skeneru na 100% výplň. K tomu jsem použil funkci *těžiště* v PrusaSliceru.



Obrázek 5.3: schéma víčka



Obrázek 5.4: schéma celku

Kapitola 6

Firmware pro zařízení

Firmware je napsán v C++.

Protože ATmega328PB je stejný mikrokontrolér jako ten, který se používá v Arduino jednodeskových mikrokontrolérech, mohu použít Arduino C++ knihovnu, která výrazně zjednoduší proces programování.

Entry point normálních C++ programů je většinou funkce `main`, tu ovšem implementuje Arduino knihovna a nahrazuje ji funkcemi `setup` a `loop`.

Funkce `setup`

Zavolá se bezprostředně po tom, co mikrokontrolér obdrží napájení. Provede tedy všechny operace nutné pro start lidaru.

- Nastaví hodnotu dělitele druhého časovače mikrokontroléru. Tento časovač určuje frekvenci PWM signálu na pinu ke kterému je připojen motor. Výchozí hodnota dělitele časovače je 64 a frekvence PWM signálu je tedy 490.20 Hz. Pokud bychom motor poháněli s touto frekvencí, vydával by slyšitelný pískavý zvuk. Nastavením hodnoty dělitele na 1, se změní frekvence PWM signálu na 31372.55 Hz. Po nastavení již tato frekvence není slyšet.
- nastaví režim všech pinů buď na vstup, nebo na výstup.
- inicializuje sériová rozhraní s vhodnými přenosovými rychlostmi.

```

1  void setup() {
2      TCCR2B = (TCCR2B & B11111000) | B00000001;
3
4      pinMode(P_LED, OUTPUT);
5      pinMode(P_HALL, INPUT);
6      // ...
7
8      ITF_serial.begin(ITF_BAUD);
9      TFM_serial.begin(TFM_BAUD);
10 }

```

Funkce loop

Zavolá se ihned po dokončení funkce setup. A po jejím dokončení je volána opakovaně.

Tato funkce při každém zavolání provádí několik operací.

- Nastaví pin LED na opačnou hodnotu, než je hodnota pinu Hallova senzoru. Tím je zajištěno, že se LED dioda rozsvítí vždy, když lidar dokončí otočku.
- Zavolá funkce pro kontrolu sériových rozhraní zda jsou v jejich bufferech k dispozici nepřechtené pakety, a pokud ano, přečtou a zpracují se.
- Zavolá funkci pro kontrolu dokončení otočky. Tato funkce provede základní debouncing a následně vyše paket ukončení rotace.
- Zavolá funkci pro kontrolu odběru energie motoru. Tato funkce měří proudový odběr motoru, a pokud je odběr příliš vysoký, sníží napětí na motoru.

```

1  void loop() {
2      digitalWrite(P_LED, !digitalRead(P_HALL));
3
4      check_tfm();
5
6      check_itf();

```

```
7
8     check_rotation();
9
10    check_speed();
11 }
```

6.1 Komunikace lidarů s externím zařízením

Nejsložitější částí firmwaru je komunikační rozhraní.

Pro komunikaci lidarů s externím zařízením budu používat asynchronní sériovou komunikaci (UART). Je totiž poměrně jednoduchá, podporovaná mikrokontrolérem ATmega328PB a již rozšířená u podobných hobby LiDARů.

6.1.1 Formát paketů

Pakety jsem rozdělil do tří kategorií. hodnoty jsou zapsány po bajtech hexadecimálně. Více-bajtová čísla jsou formátována little-endiánsky¹

ukončení rotace

Tento packet se vyšle pokaždé co LiDAR dokončí svou rotaci, tj. dorazí do úhlu 0°.

První byte obsahuje hlavičku s hodnotou 0x45.

Jelikož úhly změřených bodů se vyhodnocují až po dokončení rotace, je stěžejní aby se otočný skener točil konstantní rychlostí. Toho však nelze dosáhnout, proto druhý byte obsahuje hodnotu spolehlivosti rotace. Tato hodnota se rovná 0x47 pokud absolutní odchylka mezi trváním dokončené rotace a průměrným trváním jedné rotace je menší než 32 milisekund. Pokud je tato podmínka nesplněna hodnota bude 0x42.

¹Endiánita definuje pořadí bajtů čísla. little-endian: bajty se ukládají od nejméně významného po ten nejvýznamnější

Poté je na uživateli jestli body změřené v rozsahu této rotace využije nebo ne.

byte	název	hodnota
byte 0	hlavička	0x45
byte 1	spolehlivost rotace	0x47(G) nebo 0x42(B)

Obrázek 6.1: formát packetu ukončení rotace

měřený bod

ToF senzor TFmini-S kontinuálně vysílá packety obsahující změřené body. LiDAR tyto packety přeposílá v lehce pozměněném formátu.

První bajt obsahuje hlavičku s hodnotou 0x44.

Druhý a třetí bajt obsahují změřenou vzdálenost v centimetrech.

Čtvrtý a pátý bajt obsahují sílu signálu. Síla signálu se mění s reflektivitou a vzdáleností pozorovaného povrchu, a také s intenzitou okolního světla.

Šestý bajt obsahuje nejméně významný bajt kontrolního součtu.

byte	název	hodnota
byte 0	hlavička	0x44
byte 1	vzdálenost L	-
byte 2	vzdálenost H	-
byte 3	síla signálu L	-
byte 4	síla signálu H	-
byte 5	kontrolní součet	sum

Obrázek 6.2: formát packetu měřeného bodu

příkaz

Poslední kategorii packetů tvoří příkazy. Příkazů jsem vytvořil několik, některé slouží k posílání instrukcí z externího zařízení do LiDARu, některé pouze k posílání diagnostických informací z LiDARu, všechny však splňují následující formát.

První bajt obsahuje hlavičku s hodnotou 0x43.

Velikost packetu je libovolná, proto druhý bajt obsahuje velikost packetu.

Třetí bajt obsahuje identifikační číslo příkazu.

Čtvrtý až předposlední bajt obsahují hodnotu příkazu.

Poslední bajt obsahuje nejméně významný bajt kontrolního součtu.

byte	název	hodnota
byte 0	hlavička	0x43
byte 1	velikost packetu	n
byte 2	ID příkazu	-
byte 3 - n	payload	-
byte n	kontrolní součet	sum

Obrázek 6.3: formát packetu příkazu

příkaz PING Pošle zpět přijatý packet. Id je 0x01

příkaz FORWARD_TFM Přeposílá packety do a z ToF senzoru. Id je 0x02

příkaz SET_SPEED Nastaví rychlost otáčení. Id je 0x03

příkaz GET_DIAGNOSTICS Vyžádá diagnostická data. Id je 0x04

6.2 Nahrávání firmwaru na mikrokontrolér

6.2.1 ISP

In-System Programming je způsob programování mikrokontrolérů, umožňující programování bez nutnosti vložení čipu do zařízení k tomu určenému. K využití je potřeba přístup k pinům *MOSI*, *MISO*, *SCK* a *RESET*. Tyto piny se poté připojí do ISP programátoru kterým můžeme přes USB z počítače čip programovat. Jako ISP programátor používám *Arduino UNO* s *Arduino as ISP* programem[2].

Protože používám *PlatformIO* je ještě nutné pozměnit konfigurační soubor podle oficiální *PlatformIO* dokumentace[15].

6.2.2 Bootloader

Druhá možnost programování čipu je pomocí bootladeru. Bootloader je program nahraný na čipu který umožňuje jednodušší programování. Eliminuje totiž potřebu ISP programátoru, k programování stačí pouze převaděč z USB do sériového spojení. Takovýmto způsobem fungují *Arduino* jednodeskové počítače. Jelikož jsem ale takovýto převaděč do svého zařízení nezakomponoval a musel bych takovýto převaděč používat externí, nepřinese mi tato metoda žádné výhody.

Kapitola 7

Klient

7.1 Terminálový klient

Pro zjednodušení debugování a zobrazování nezpracovaných dat lidarů, jsem vytvořil jednoduchou C++ aplikaci pro příkazový řádek.

Spuštěním aplikace bez argumentů nebo s argumentem `-l` se vypíše dostupné sériové porty.

Pokud je aplikace spuštěna s argumentem `-p [název portu]`, naváže spojení s příslušným sériovým portem. Poté aplikace okamžitě začne vypisovat všechny příchozí pakety.

Pokud je aplikace spuštěna s argumentem `-c [payload příkazu]`, vyšle paket s příkazem obsahující specifikovaný payload.

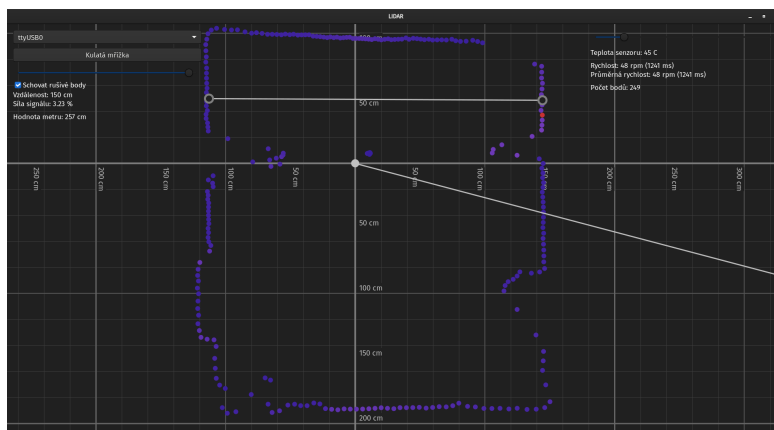
Pokud je aplikace spuštěna s argumentem `-f`, bude vypisovat pouze pakety příkazů a dokončení rotace.

7.2 Grafický klient

Pro zobrazování dat nasnímaných lidarem, jsem vytvořil grafickou aplikaci v C++. Jako GUI knihovnu jsem si vybral wxWidgets.

Hlavním cílem aplikace je zobrazit point cloud shromážděný lidarem. Ale

přidal jsem také spoustu dalších funkcí. Některé slouží k ovládání a monitorování samotného lidarů a některé k manipulaci se shromážděnými daty.



Obrázek 7.1: Grafický klient

7.2.1 Funkce grafického klienta

Body

Body, které představují data naměřená lidarem, se aktualizují pokaždé, když lidar dokončí svou rotaci. Celý point cloud lze otočit tak, aby odpovídal orientaci lidarů.

Barva bodů vyjadřuje sílu signálu hlášenou TFmini senzorem v dané poloze. Více modré barvy znamená nízkou sílu signálu a více červené barvy znamená silnou sílu signálu. Pokud bod nemá žádné blízké sousedy, jeho barva se změní na šedou. Díky tomu je graf přehlednější. Pokud jsou šedé body stále na obtíž, přidal jsem možnost je zcela skrýt.

Mřížka

Pro snazší orientaci mezi body jsem přidal mřížku. Uživatel může body i mřížku posouvat a přibližovat. Uživatel může přepínat mezi hranatou a kulatou mřížkou. Úroveň detailu mřížky se dynamicky mění v závislosti na úrovni přiblížení.

Čísla představující vzdálenosti zůstávají na středních osách mřížky, ale pokud uživatel pohne mřížkou tak, že se čísla dotýkají okraje obrazovky, čísla se také posunou, aby zůstala vždy na očích.

Metr

Také jsem přidal dva způsoby měření vzdáleností.

První je způsob měření vzdálenosti mezi bodem a lidarem. Ze středu grafu vychází čára, která vždy ukazuje ve směru kurzoru myši uživatele. Po namíření čáry na bod, který chce uživatel změřit, se text v levém horním rohu obrazovky změní na hodnoty tohoto bodu.

Pokud chce uživatel změřit vzdálenost mezi dvěma body, může stisknoutout pravé tlačítko myši a tím vytvořit nový bod na mřížce. V levé horní části obrazovky se poté zobrazí text informující o vzdálenosti mezi tímto novým bodem a kurzorem myši. Opětovným kliknutím pravého tlačítka myši se objeví další bod a text se změní tak, aby hlásil vzdálenost mezi těmito dvěma body. Dalším kliknutím pravého tlačítka myši může uživatel oba body odstranit.

Rozbalovací menu pro výběr sériového portu

V případě, že je k počítači připojeno více sériových zařízení, je dobré mít možnost si mezi nimi vybrat ke kterému se chceme připojit. Za tímto účelem jsem přidal rozbalovací nabídku v levém horním rohu obrazovky.

Po výběru sériového portu se na tento port každou sekundu odešle ping paket, pokud se ping paket vrátí, aplikace si může být jistá, že je opravdu připojen lidar. A že tomu tak je, oznámí uživateli zobrazením nastavení rychlosti otáčení a diagnostických údajů lidaru v pravém horním rohu obrazovky.

Diagnostická data

Diagnostickými údaji lidaru jsou teplota senzoru TFmini, průměrná a skutečná rychlost otáčení a počet bodů naměřených v zobrazeném point cloudu.

Závěr

Výsledkem mé práce je funkční lidar schopný měřit vzdálenosti mezi 10 centimetry a 12 metry. A to s přesností ± 6 centimetrů v rozsahu menším než 6 metrů a s přesností $\pm 1\%$ v rozsahu větším než 6 metrů. Rychlost otáčení lidarů je nastavitelná v rozmezí 40-120 otáček za minutu. Během jedné otáčky zachytí lidar v závislosti na rychlosti otáčení od 110 do 340 bodů. Lidar lze ovládat jakýmkoli externím zařízením se sériovým rozhraním.

Také jsem vytvořil program, který zobrazuje data shromážděná lidarem, ovládá rychlost lidarů a má příjemné uživatelské prostředí.

Nabízí se řada možných úprav do budoucna. Vylepšení parametrů zařízení. Přidání dalších funkcí do programu. Vytvoření driverů pro operační systém pro roboty ROS.

Zdroje

- [1] Slant 3D. *Design Better Holes — Improve Tolerances — Reduce Sagging — Design for Mass Production 3D Printing*. 2023. URL: <https://www.youtube.com/watch?v=Bd7Yyn61XWQ> (cit. 2023).
- [2] Arduino. *Arduino as ISP and Arduino Bootloaders*. 2018. URL: <https://docs.arduino.cc/built-in-examples/arduino-isp/ArduinoISP/> (cit. 2024).
- [3] arduino. *arduino schema*. URL: https://www.arduino.cc/en/uploads/Main/Arduino_Nano-Rev3.2-SCH.pdf (cit. 2024).
- [4] *Arduino-PWM-Frequency - ArduinoInfo*. [Online; accessed 26. Mar. 2024]. Ún. 2021. URL: <https://arduoinfo.mywikis.net/wiki/Arduino-PWM-Frequency>.
- [5] *ATmega328PB — PlatformIO v6.1 documentation*. [Online; accessed 26. Mar. 2024]. Břez. 2024. URL: <https://docs.platformio.org/en/stable/boards/atmelavr/ATmega328PB.html>.
- [6] Benewake. *Product Manual of TFmini-S*. SJ-PM-TFmini-S A01 Product Manual.
- [7] Benewake. *TFmini-S LiDAR Module*. SJ-GU-TFmini-S-01 A00.
- [8] Jeremy Cohen. „(Mindmap) A Hardcore Look at 9 types of LiDAR systems“. In: *Welcome to The Library!* (Zář. 2023). URL: <https://www.thinkautonomous.ai/blog/types-of-lidar>.
- [9] Jeremy Cohen. „Understanding the magnificent FMCW LiDAR“. In: *Welcome to The Library!* (Srp. 2023). URL: <https://www.thinkautonomous.ai/blog/fmcw-lidar>.

- [10] *cppreference.com*. [Online; accessed 26. Mar. 2024]. Břez. 2024. URL: <https://en.cppreference.com/w>.
- [11] GreatScott. *Is LIDAR easy to use for hobbyists? DIY Roomba? Obstacle Avoidance System for Robotics*. 2019. URL: https://www.youtube.com/watch?v=_--8K1BW_04 (cit. 2024).
- [12] Geoffrey Hunter. *Linux Serial Ports Using C/C++*. 2017. URL: <https://blog.mbedded.ninja/programming/operating-systems/linux/linux-serial-ports-using-c-cpp/> (cit. 2023).
- [13] *Introduction to LiDAR Technology*. [Online; accessed 26. Mar. 2024]. Břez. 2024. URL: <https://cz.farnell.com/introduction-to-lidar-technology>.
- [14] Nanxi Li et al. „A Progress Review on Solid-State LiDAR and Nanophotonics-Based LiDAR Sensors“. In: *Laser Photonics Rev.* 16.11 (lis. 2022), s. 2100511. ISSN: 1863-8880. DOI: 10.1002/lpor.202100511.
- [15] PlatformIO. *Atmel AVR Upload using Programmer*. URL: <https://docs.platformio.org/en/latest/platforms/atmelavr.html#upload-using-programmer> (cit. 2024).
- [16] Microchip Technology. *ATmega328PB datasheet*. 40001907A.
- [17] Dingkang Wang, Connor Watkins a Huikai Xie. „MEMS Mirrors for LiDAR: A Review“. In: *Micromachines* 11.5 (dub. 2020), s. 456. ISSN: 2072-666X. DOI: 10.3390/mi11050456.
- [18] *wxWidgets: Documentation*. [Online; accessed 26. Mar. 2024]. Břez. 2024. URL: <https://docs.wxwidgets.org/3.2>.
- [19] YDLIDAR. *YDLIDAR X4 DATA SHEET*. YDLIDAR X4.