STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor: 10. Elektrotechnika, elektronika a telekomunikace

Laserový projektor

Šimon Hrouda

Brno 2024

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

LASEROVÝ PROJEKTOR

LASER PROJECTOR

AUTOR	Šimon Hrouda
ŠKOLA	Gymnázium Brno-Řečkovice
KRAJ	Jihomoravský
INTERNÍ KONZULTANT	Mgr. Kateřina Vídenková
EXTERNÍ KONZULTANT	Tomáš Rohlínek
OBOR	10. Elektrotechnika, elektronika a telekomunikace

Brno 2024

Prohlášení

Prohlašuji, že svou práci na téma *Laserový projektor* jsem vypracoval/a samostatně pod vedením Tomáše Rohlínka a Mgr. Kateřiny Vídenkové a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Dále prohlašuji, že tištěná i elektronická verze práce SOČ jsou shodné a 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 změně některých zákonů (autorský zákon) v platném znění.

V Brně dne: _____

_____

Šimon Hrouda

Poděkování

Děkuji svému externímu konzultantovi Tomáši Rohlínkovi a své interní konzultantce Mgr. Kateřině Vídenkové za obětavou pomoc, podnětné připomínky a nekonečnou trpělivost, kterou mi během práce poskytovali.

Tato práce byla provedena za finanční podpory Jihomoravského kraje.

Anotace

Klíčová slova

Annotation

Keywords

Obsah

Note!

[**TODO 3. osoba - Práce se zaměřuje**] https://www.sciencedirect.com/search?qs=galvanom
muzu rict, ze jsem neco nezvladl dohledat :)

Úvod

V této práci se zaměřuji na návrh a výrobu laserového projektoru, který bude za pomoci páru zrcátek připevněných na galvanometrech rychle měnit směr laserového paprsku a tím vykreslovat obraz na promítací plochu.

Technologie rychle se pohybujícího laserového paprsku (laser scanning) je využívána v mnoha oblastech od jednoduchého promítání, efektů na diskotékách a Heads Up Displejů v letadlech či autech [1], přes čtení čárových kódů [2] a 3d tisk [3] po 3D skenování modelů [4] i Zemského povrchu [5].

Bohužel ale neexistují žádné uživatelsky přívětivé open-source platformy, kde by se s touto technologií mohli seznámit zájemci o její rozvíjení.

V této práci jsem se proto rozhodl pro tuto technologii vytvořit jednoduché uživatelské prostředí, ve kterém si i začínající kutil může vyzkoušet jak funguje.

definice pojmů a zkratek

CLGS	Closed Loop Galvanometer System	systém galvanometru se zpětnou vazbou
OLGS	Open Loop Galvanometer System	systém galvanometru bez zpětné vazby
SPI	Serial Peripheral Interface	sériové periferní rozhraní

Kapitola 1

hardware

1.1 Raspberry Pi

1.2 Galvanometr a zrcátko

- Galvanometry, často nazývané galva, jsou elektronické součástky používané k měření intenzity a směru elektrického proudu. [6]

  Můžeme je rozdělit mezi galvanometry s uzavřenou smyčkou zpětné vazby (CLGS) a galvanometry bez uzavřené smyčky zpětné vazby (OLGS). [7]

  [8] ale postupně nachází uplatnění ve více a více odvětvích práce s lasery. Oproti jiným možnostem nabízí flexibilitu, rychlost a přesnost za nízkou cenu.

  V této práci jsou ale využívány CLGS, které jsou lépe zdokumentované.

  V CLGS jsou potřeba 3 hlavní prvky,

  Nejmodernější galvanometrové polohovací systémy jsou založené na principech elektromotorů s permanentními magnety, kde [8]

- https://en.wikipedia.org/wiki/Galvanometer
  https://en.wikipedia.org/wiki/Laser_scanning
  https://en.wikipedia.org/wiki/Mirror_galvanometer
  https://en.wikipedia.org/wiki/Laser_lighting_display

  A mirror galvanometer is an ammeter that indicates it has sensed an electric current by deflecting a light beam with a mirror. The mirror galvanometer consists of a long fine coil of silk-covered copper wire. In the heart of that coil, within a little air-chamber, a small round mirror is hung by a single fibre of floss silk, with four tiny magnets cemented to its back The small weight of the mirror and magnets which form the moving part of this instrument, and the range to which the minute motions of the mirror can be magnified on the screen by the reflected beam of light, which acts as a long impalpable hand or pointer, render the mirror galvanometer marvellously sensitive to the current, especially when compared with other forms of receiving instruments.

  Zrcadlový galvanometr je měřič proudu, který reaguje na měřený proud vychýlením světelného paprsku zrcátkem připevněném na jeho konci.

  Skládá se z cívky a [9]

- following text generated by ai (google bard) on 2023-10-20

> popiš mi jak je sestaven a jak funguje galvanometr se zrcátkem k odrážení světelného paprsku

Galvanometr se zrcátkem je sestaven z následujících komponent: * **Cívka:** Cívka je navinuta na jádře z feromagnetického materiálu, jako je železo nebo nikl. Cívkou prochází elektrický proud, který vytváří magnetické pole. * **Ukazatel:** Ukazatel je připojen k cívce a je vyroben z lehkého materiálu, jako

je hliník nebo plast. Ukazatel je vychýlen magnetickým polem vytvářeným cívkou. * **Zrcátko:** Zrcátko je připojeno k ukazateli a odráží světelný paprsek. * **Okulár:** Okulár je umístěn tak, aby bylo možné pozorovat odraz světelného paprsku.
Jak funguje:
Když prochází cívkou elektrický proud, vytváří magnetické pole. Toto magnetické pole působí na ukazatel a vychýlí ho. Velikost výchylky ukazatele je přímo úměrná velikosti proudu procházejícího cívkou.
Zrcátko je připojeno k ukazateli a odráží světelný paprsek. Když se ukazatel vychýlí, odraz světelného paprsku se také vychýlí.
Okulár je umístěn tak, aby bylo možné pozorovat odraz světelného paprsku. Velikost vychýlení světelného paprsku je přímo úměrná velikosti proudu procházejícího cívkou.
Výhody:
* Galvanometr se zrcátkem je citlivější než galvanometr bez zrcátka. * Galvanometr se zrcátkem lze použít k měření velmi malých proudů.
Nevýhody:
* Galvanometr se zrcátkem je dražší než galvanometr bez zrcátka.
Použití:
Galvanometr se zrcátkem se používá k měření velmi malých proudů. Používá se například v elektrotechnice, lékařství a fyzice.

ovládá se variabilním proudem

1.3 hlavice

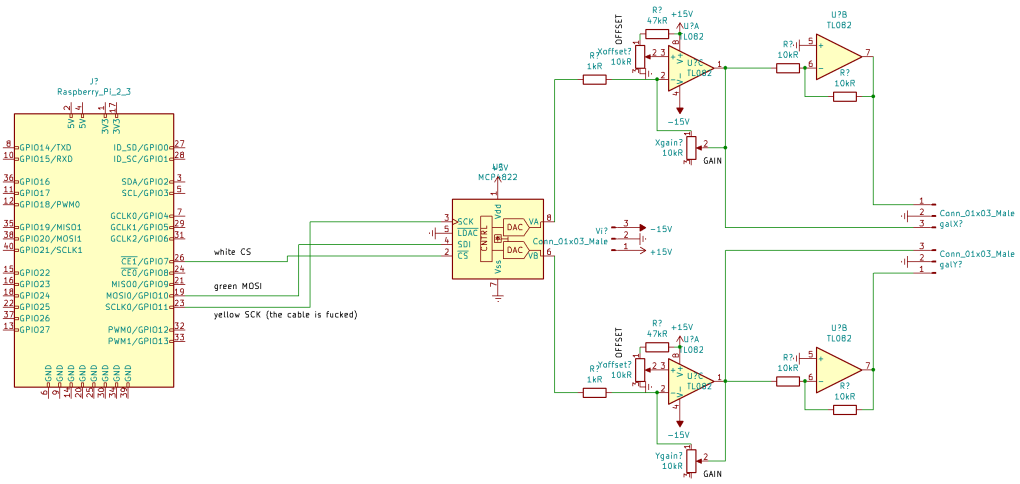
The mirrors, mounted perpendicularly on the engines, move the laser beam along the X and Y axes according to the input received from the motor. The

big advantage of these devices is that they can reach a very high acceleration and speed of well asi patří do sekce galvanometr

1.4 moje deska na napětí

Galvanometry v obou osách pohybu potřebujeí analogový vstupní signál v rozpětí −15 V až +15 V udávající vychýlení galvanometru v daném směru.

Obvod, který se stará o vytváření tohoto signálu je založený na obvodu ze zdroje [10]. Vytváření tohoto signálu je rozděleno do dvou částí. Nejdříve DAC (digital-to-analog converter, D/A převodník) připojený k RPi vytvoří signál v rozpětí 0 až 5 V a následně je tento signál pomocí operačního zesilovače převeden na požadované rozpětí, tj. −15 V až +15 V. Jednotlivé části tohoto obvodu jsou blíže popsány v následujících kapitolách. Celé zapojení je vidět na obrázku 1.1. [**unreadable text, make schem more compact**] Note!



Obrázek 1.1: Zapojení DAC a zesilovačů k RPi a řídící desce galvanometrů

1.4.1 dac

K generování signálu v rozpětí 0–5 V jsem využil DAC MCP4822 od firmy Microchip Technology Inc. [**TODO tečka? ("\-= explicitni mezera)**] Tento čip podporuje komunikaci přes rozhraní SPI, pracuje s napájecím napětím 5 V a s 12bitovým rozlišením (je schopen vygenerovat 4 096 různých napětí) na dvou kanálech. Note!

RPi komunikuje s čipem pomocí rozhraním SPI, toto rozhraní využívám pomocí knihovny ze serveru https://github.com[1] [**TODO tečka?**] [**TODO more spec**] Tato knihovna poskytuje následující funkce, se kterými pracuji v mém kódu. Note! Note!

- `bool mcp4822_initialize();`
- `bool mcp4822_set_voltage(mcp4822_channel_t channel, uint16_t value_mV );`
- `mcp4822_deinitialize();`

1.4.2 amps

K rozšíření signálu z DAC jsem využil dva operační zesilovače TL082 od firmy Texas Instruments Incorporated. Každý z nich je připojený na jeden kanál DAC čipu mcp4822. [**TODO more spec**] Tyto čipy mi napěťové rozpětí zvýší z 0–5 V na −15 V až +15 V. Note!

zesilovac - cteni baterek https://is.muni.cz/el/sci/jaro2017/F5090/um/E17_P8.pdf

[1] https://github.com/abelectronicsuk/ABElectronics_CPP_Libraries/tree/master/ADCDACPi; staženo 2. 1. 2024

1.5 laser

1.6 if rgb: 3 dacs

Note!

[TODO cos udelal svyho vlastne a jak to facha]

1.7 napájení

Note!

[TODO ay tak co, zvladls to dat na baterky?]

Kapitola 2

software

Tento laserový projektor se skládá ze dvou částí. Jednou je software pro řízení galvanometrů a druhou je software pro interakci s uživatelem. [**TODO: zkrátit věty**] Note!

O řízení galvanometrů se stará program lasershow, který je psaný v jazyce c++ pro maximální rychlost. Tento program běží na pozadí a čeká na příkazy od programů určených k interakci s uživatelem. Na tento program se zaměřuje kapitola lasershow. [**TODO: odkaz**] Note!

Dále jsou tu programy, které se starají o interakci s uživatelem. Tyto programy přijímají příkazy od uživatele a posílají je programu lasershow. Navíc od lasershow získávají výstup, který následně zprostředkovávají uživateli; důkladněji popsáno v kapitole 2.1.

Mezi tyto programy patří programy UI, web_ui a discord_bot. Program UI spravuje OLED displej, přijímá od uživatele vstup rotačním enkodérem a je psaný v c++ pro jednodušší interakci s hardwarem. Program web_ui využívá runtime Node.js, ve kterém je nenáročné vytvořit http server dostupný z lokální sítě. [**(A)?**] Program discord_bot, také využívající Node.js, přijímá příkazy z chatovací aplikace discord a je přístupný i přes internet. Note!

Nakonec je tu program wifi_manager, ten spravuje wifi připojení RPi, je psaný v Node.js a komunikuje s programy, které interagují s uživatelem stejně jako program lasershow.

2.1 komunikace mezi programy

Všechny tyto programy jsou propojeny síťovými sockety zprostředkovanými knihovnou ZeroMQ, která nabízí frontu[1] zpráv, bez potřeby samostatně běžícího brokeru.[**TODO: divna jednicka**] Note!

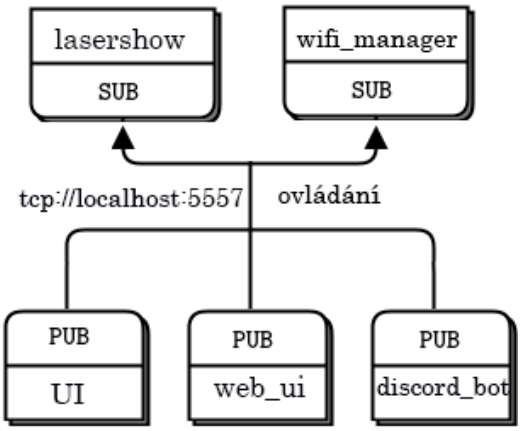
Tato knihovna je využita k vytvoření dvou socketů, jedním lasershow přijímá příkazy od uživatele prostřednictvím ostatních programů (vstupní socket na portu 5557, viz obr. 2.1) a do druhého posílá informace ostatním programům (výstupní socket na portu 5556, viz obr. 2.2), aby je zprostředkovaly uživateli. Do prvního zmíněného posílají progamy interagující s uživatelem příkazy pro programy lasershow a wifi_manager. Do druhého posílá lasershow informace o stavu a změnách nastavení a také wifi_manager informace o stavu a změnách v nastavení WiFi.

Příkazy pro programy lasershow a wifi_manager vypadají následovně [**TODO: příklady příkazů pro lasershow a wifi_manager z** https://github.com/phuid/laser_projector/blob/master/README.md] Note! [**TODO: příklady status infos od lasershow a wifi_manager z** https://github.com/phuid/laser_projector/blob/master/README.md] Note!

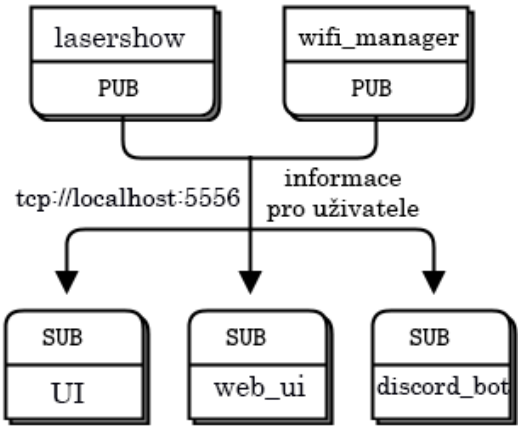
2.2 lasershow

Program lasershow je psaný v jazyce c++, který je kompilovaný a obecně považovaný za jeden z nejrychlejších jazyků. Druhé zmíněné se hodí, jelikož chceme vykreslovat co možná nejrychleji.

[1] Ve frontě jsou zprávy seřazeny od té nejdříve odeslané.



Obrázek 2.1: komunikace mezi programy vstupním socketem na portu 5557



Obrázek 2.2: komunikace mezi programy výstupním socketem na portu 5556

Tento program zaregistruje vstupní TCP socket na portu 5557 a knihovnou ZeroMQ se na něm přihlásí k odběru zpráv, které do něj publikují ostatní programy. Zárověň podobně zaregistruje výstupní socket na portu 5556, do kterého později bude posílat zprávy pro programy, které interagují s uživatelem.

Následně se připojí k DAC a čeká na zprávy od ostatních programů. Jakmile zprávu obdrží, zpracuje ji a pokud je požadována změna nastavení, okamžitě ji provede a aktuální nastavení si uloží do souboru, jestliže je požadováno vy-

kreslení obrazu ze souboru, začne obraz vykreslovat. Při tom průběžně posílá informace o stavu vykreslování do výstupního socketu. I při vykreslování obrazu tento program zpracovává zprávy a pokyny ze vstupního socketu.

Program byl původně převzat z projektu https://github.com/tteskac/rpi-lasershow[2], následně byl ale přepsán skoro ve všech ohledech a z původního programu zbylo asi 20 řádků. [**TODO: odkud jsem to vzal a prepsal a jak moc jsem toho udelal a s jakymy vysledky**] Note! Note!

[**TODO: diagram programu**] Note!

[**TODO: priklad zmq**]

```
1  //  Clone server Model One
2
3  #include "zmq.hpp"
4  #include "zmq_addon.hpp"
5  #include <chrono>
6  #include <thread>
7  #include <iostream>
8  #include <string>
9
10 int main(void)
11 {
12     //  Prepare our context and publisher socket
13     zmq::context_t ctx(1);
14
15     zmq::socket_t publisher(ctx, zmq::socket_type::pub);
16     publisher.bind("tcp://*:5556");
17
18     zmq::socket_t command_receiver(ctx, zmq::socket_type::sub
       );
19     command_receiver.bind("tcp://*:5557");
20     command_receiver.set(zmq::sockopt::subscribe, "");
21
22     uint64_t wahoo = 0;
```

[2]staženo 28. 12. 2023

```
    std::cout << "start done" << std::endl;

    while (true)
    {
        zmq::message_t received;
        command_receiver.recv(received, zmq::recv_flags::none);

        std::cout << "received: \"" << received.to_string() << "\"" << std::endl;

        std::string msg_string = "nice, thank you bro, i got this from you \"" + received.to_string() + "\"";

        zmq::message_t msg(msg_string.c_str(), msg_string.length() + 1);
        publisher.send(msg, zmq::send_flags::none);

        wahoo++;
    }
    return 0;
}
```

```
//  Clone client Model One

#include "zmq.hpp"
#include "zmq_addon.hpp"
#include <iostream>
#include <thread>
#include <chrono>

int main(void)
{
    //  Prepare our context and updates socket
    zmq::context_t ctx(1);

    zmq::socket_t subscriber(ctx, zmq::socket_type::sub);
    subscriber.connect("tcp://localhost:5556");
```

```
16      subscriber.set(zmq::sockopt::subscribe, "");
17
18      zmq::socket_t command_sender(ctx, zmq::socket_type::pub);
19      command_sender.connect("tcp://localhost:5557");
20
21      std::cout << "start done" << std::endl;
22
23      while (true)
24      {
25          std::this_thread::sleep_for(std::chrono::milliseconds
        (20));
26
27          zmq::message_t received;
28          while(subscriber.recv(received, zmq::recv_flags::
        dontwait)) {
29              std::cout << "received: \"" << received.to_string
        () << "\"" << std::endl;
30          }
31
32          // read user input
33          std::string u_in;
34          getline(std::cin, u_in);
35
36          zmq::message_t msg(u_in.c_str(), u_in.length());
37          command_sender.send(msg, zmq::send_flags::none);
38      }
39      return 0;
40  }
```

2.3 UI

Program UI je také psaný v jazyce c++ a využívá knihovnu WiringPi, která umožňuje jednoduchou komunikaci s GPIO piny Raspberry Pi. Tento program ovládá OLED displej, který je připojený na Raspberry Pi pomocí rozhraní I2C, a přijímá vstup od uživatele čtením rotačního enkodéru s tlačítkem.

Program se při začátku exekuce pomocí knihovny ZeroMQ přihlásí ke vstupnímu socketu a k odběru zpráv z výstupního TCP socketu, kam publikuje zprávy o stavu vykreslování program lasershow. Dále si pomocí knihovny wiringPi zaregistruje zpracovávání přerušení z enkodéru a tlačítka na něm a čeká buď na interakci s uživatelem, který by skrz něj poslal zprávy programu lasershow, nebo na zprávy od lasershow, které by zobrazil uživateli. Note!

[TODO: diagram programu]

2.4 web_ui

Narozdíl od předchozích dvou zmiňovaných programů je program web_ui psaný v jazyce javascript, ten nepatří mezi nejrychlejší, ale díky runtime Node.js a knihovnám http a formidable v něm bylo časově nenáročné vytořit http web server.

Tento server běží na portu 3000 a je dostupný z lokální sítě (tzn. přímo z Raspberry Pi na adrese http://localhost:3000 nebo z jakéhokoliv zařízení na stejné lokální síti na adrese http://IP_ADRESA_RPI:3000). Program je využíván pro jednoduchou interakci s uživatelem, který může pomocí webového prohlížeče ovládat laserový projektor pár kliknutími i zadávat vlastní příkazy klávesnicí. Note!

[TODO: příklad http serveru]

```
1  const http = require('http');
2  const fs = require('fs');
3  const path = require('path');
4
5  const server = http.createServer((req, res) => {
6    // Get the file path from the request URL
7    const filePath = path.join(__dirname, req.url);
8
9    // Check if the file exists
10   fs.access(filePath, fs.constants.F_OK, (err) => {
```

```
    if (err) {
      // File not found
      res.statusCode = 404;
      res.end('File not found');
    } else {
      // Read the file and send it as the response
      fs.readFile(filePath, (err, data) => {
        if (err) {
          // Error reading the file
          res.statusCode = 500;
          res.end('Internal server error');
        } else {
          // Set the appropriate content type and send the
   file data
          const ext = path.extname(filePath);
          let contentType = 'text/plain';
          if (ext === '.html') {
            contentType = 'text/html';
          } else if (ext === '.css') {
            contentType = 'text/css';
          } else if (ext === '.js') {
            contentType = 'text/javascript';
          }

          res.setHeader('Content-Type', contentType);
          res.end(data);
        }
      });
    }
  });
});

const port = 3000;
server.listen(port, () => {
  console.log(`Server running on port ${port}`);
});
```

Stejně jako program UI za pomoci knihovny ZeroMQ tento program odebírá

z výstupního socketu zprávy o průběhu vykreslování od programu lasershow a odesílá mu pokyny uživatele na vstupní socket. Note!

[**TODO: příklad přihlášení k socketům v js**] Note!

[**TODO: xterm + ssh**]

2.5 discord bot

Posledním programem, který je využíván k interakci s uživatelem je discord_bot, který je také psaný v jazyce javascript v runtime Node.js, stejně jako předchozí programy se přihlásí k socketům knihovnou zmq, ale na rozdíl od nich tento program může interagovat s uživatelem přes internet ať už je kdekoliv na světě. Pomocí knihovny discord.js se přihlásí k předem vytvořenému bot účtu, který může na předem vytvořeném discord serveru čekat na zprávy od uživatele, ty posílat do vstupního socketu a posílat uživateli zpětnou vazbu, kterou příjme z výstupního socketu.

2.6 wifi_manager

V rámci této práce byl vyvinut ještě jeden program, který se přímo nepodílí ani na projekci, ani na interakci s uživatelem.

Program wifi_manager je také napsaný v jazyce JavaScript s využitím runtime Node.js. Registruje se ke stejným socketům jako lasershow, přijímá příkazy týkající se nastavení WiFi na Raspberry Pi TCP socketem na portu 5557 a odesílá zpětnou vazbu na TCP socket s portem 5556. Note!

[**TODO: jak se komunikace s lasershow odlisuje od wifi_managera**] Note!

[**TODO: ukazka(idk what)**]

Hlavním úkolem tohoto programu je správa a konfigurace WiFi připojení na Raspberry Pi. Přijímá příkazy od ostatních programů a nastavuje WiFi

parametry na základě těchto příkazů. Tím umožňuje uživatelům snadno a pohodlně nastavit WiFi připojení na svém zařízení.

Stejně jako lasershow, wifi_manager také posílá zpětnou vazbu ostatním programům, aby informoval o stavu a změnách v nastavení WiFi. Tímto způsobem je zajištěna komunikace a synchronizace mezi všemi programy v laserovém projektoru.

Celkově wifi_manager přispívá k plynulému a efektivnímu provozu laserového projektoru tím, že umožňuje snadnou správu a konfiguraci WiFi připojení na Raspberry Pi. Note!

[**TODO udelals to vubec dobre? porovnej se s ostatnima**]

Kapitola 3

Diskuze

3.1 ruzne technologie

Note!

[kam pokracovat: api p ro random programy - aby si mohli pokrocilejsi kutilove taky hrat a randomly posouvat laser ig]

3.2 další zpracování tématu

udelal jsem to dobre? vybral jsem si dobry technky? like byl by lepsi ten harddrive z yt? nebo fakt to melo byt napajeny z baterek a ne ze zasuvky?

ze hej ze typek z vut udelal kinda kurva podobnej HW jak ja, ale ja to mam trochu jinak, cuz jsem o tom nevedel, ale ofc moje je lepsi :)) also to delala hromada dalsich lidi na internetu ten hw, also od gh.com/tteskac mam executable, kterou jsem ale totalne ze rozsiril a taky jsem pridal vsechno moje genialni ui muhahahah

ze este dalsi zpracovani: (19.10.2023 vsechny dostupne)

1. used/modified code

- https://github.com/marcan/openlase/blob/master/tools/svg2ild.py
- https://github.com/tteskac/rpi-lasershow
- https://github.com/sabhiram/raspberry-wifi-conf/blob/master/app/wifi_manager.js
- http://www.electronicayciencia.com/wPi_soft_lcd/
- typek z vut

2. dalsi zpracovani stejny projekty
   - https://www.instructables.com/Arduino-Laser-Show-With-Real-Galvos/
   - https://github.com/tteskac/rpi-lasershow
   - https://www.instructables.com/DIY-STEPDIR-LASER-GALVO-CONTROLLER/
   - borec na yt hard-drive text gut
3. other useful thingies
   - https://hackaday.io/project/172284-galvo-laser-cutterengraver
   - https://hackaday.io/project/172284/instructions
   - https://learn.adafruit.com/mcp4725-12-bit-dac-with-raspberry-pi/hooking-it-up
   - https://www.ilda.com/resources/StandardsDocs/ILDA_IDTF14_rev011.pdf
   - cool demos https://marcan.st/projects/openlase/
   - https://www.youtube.com/watch?v=u9TpJ-_hBR8

4. read

    - https://www.laserworld.com/en/glossary-definitions/90-t/2797-ttl-modulation-en.html

Závěr

Note!

[FIXME proc vsichni maji zaver v obsahu jako section, kdyz pak vypada, ze je pod posledni kapitolou??] [TODO závěr][TODO muj projekt je dostupný na githubu] [tim ze mam linux lasershow obcas zpomali, obcas zrychli, samozrejme mam osetreny, aby nezrychlily framy, ale pointy ano]

Note!
Note!
Note!

Literatura

1. MAROTO, Marcos; CAÑO, Enrique; GONZÁLEZ, Pavel; VILLEGAS, Diego. Head-up Displays (HUD) in driving. *arXiv preprint arXiv:1803.08383*. 2018.
2. EASTMAN, Jay. Brief history of barcode scanning. In: *OSA Century of Optics*. Optical Society, 2015, s. 128–133.
3. CHAROO, Naseem A.; ALI, Sogra F. Barakh; MOHAMED, Eman M.; KUTTOLAMADOM, Mathew A.; OZKAN, Tanil; KHAN, Mansoor A.; RAHMAN, Ziyaur. Selective laser sintering 3D printing – an overview of the technology and pharmaceutical applications. *Drug Development and Industrial Pharmacy*. 2020, roč. 46, č. 6, s. 869–877. Dostupné z DOI: 10.1080/03639045.2020.1764027. PMID: 32364418.
4. EDL, MMTJ; MIZERÁK, Marek; TROJAN, Jozef. 3D laser scanners: history and applications. *Acta Simulatio*. 2018, roč. 4, č. 4, s. 1–5.
5. PFEIFER, Norbert; BRIESE, Christian. Laser scanning–principles and applications. In: *Geosiberia 2007-international exhibition and scientific congress*. European Association of Geoscientists & Engineers, 2007, cp–59.
6. FERROVIAL. *What is a galvanometer?* [online]. [cit. 2023-11-29]. Dostupné z: https://www.ferrovial.com/en/stem/galvanometer.
7. LASERFX. *How Laser Shows Work - Scanning System* [online]. [cit. 2023-10-19]. Dostupné z: http://www.laserfx.com/Works/Works3S.html.

8. AYLWARD, Redmond P. Advanced galvanometer-based optical scanner design. *Sensor Review*. Září 2003, roč. 23, č. 3, s. 216–222. Dostupné z DOI: 10.1108/02602280310481968.

9. WIKIPEDIA CONTRIBUTORS. *Mirror galvanometer* [online]. [cit. 2023-10-19]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Mirror_galvanometer&oldid=1170954323.

10. INSTRUCTABLES, DELTAFLO. *Arduino Laser Show With Real Galvos* [online]. [cit. 2024-01-22]. Dostupné z: https://www.instructables.com/Arduino-Laser-Show-With-Real-Galvos.

Seznam obrázků

Seznam tabulek