FEI STU

Manuál

K predmetu mobilné robotické systémy

Martin Dekan
15. 2. 2023

Obsah

1. Úlohy na cvičeniach

Cieľom predmetu je naučiť sa základy riadenia pohybu mobilných robotov najmä z hľadiska senzorového vybavenia a k tomu určených algoritmov lokalizácie, navigácie, mapovania a plánovania pohybu robota v prostredí. Úlohy budete programovať na počítači, ktorý sa bude pripájať na diaľku k mikropočítaču Raspberry Pi, ktoré je umiestnené na každom robote. Raspberry Pi preposiela údaje z robota a laserového diaľkomeru (v ďalšom texte sa môže objaviť označenie lidar, alebo len skrátene laser.) pomocou protokolu UDP.

Na počítačoch v laboratóriu (alebo na https://github.com/dekdekan/demoRMR-all ak budete robiť na vlastnom notebooku) sa nachádza demo program, ktorý má v sebe urobenú základnú komunikáciu ako s robotom tak aj s lidarom. Demo program je spravený vo vývojovom prostredí QT a jazyku C++. Ak vám tento jazyk/prostredie nevyhovuje, môžete robiť v čomkoľvek inom, ale komunikáciu a parsovanie dát si musíte spraviť samy. Rovnako nie je nutné aby váš výsledný program vyzeral ako tento demo program, bez problémov akceptujeme ako výsledok konzolovú aplikáciu.

Na cvičeniach budete mať za úlohu vypracovať projekt pozostávajúci z čiastkových úloh rozdelených do štyroch blokov. Z každého bloku si môžete vybrať buď úlohu, ktorá je vysvetlená v tomto manuáli (a bude podrobnejšie prejdená aj na cvičení) alebo niektorú z metód vysvetlených na prednáškach (s bodovým ohodnotením podľa náročnosti).

Celkovo je možné na cvičeniach získať **35 bodov** za úlohy a maximálne **5 bodov za súťaž**. Body budú pridelené na základe **predvedenia fungovania algoritmu a odovzdanej dokumentácie**. Dokumentácia musí obsahovať užívateľskú príručku (informácie o tom, ako program spustiť, čo robia jednotlivé gombíky/ aké príkazy váš program podporuje) a popis fungovania jednotlivých algoritmov (slovne, pseudo kód, vývojový diagram)

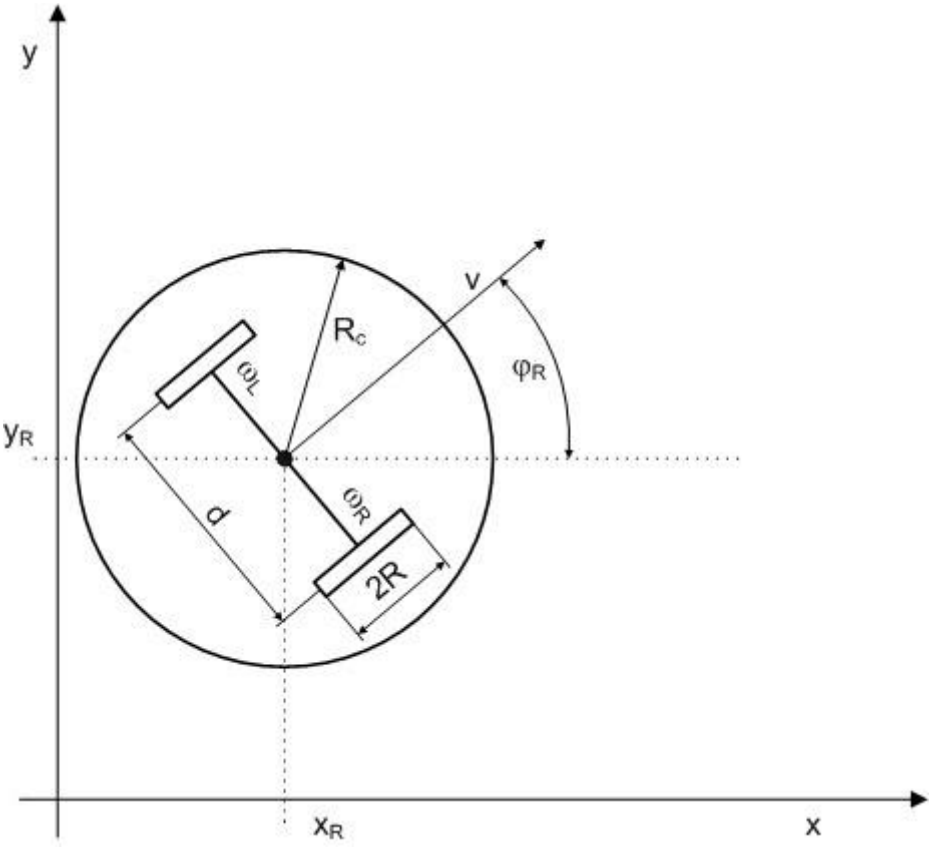
Súťaž (0-5b)

Súťaž prebehne na konci semestra. Podmienkou účasti je schopnosť presunúť robota z bodu A do bodu B (bez prekážok). Cieľom súťaže je dostať robota z bodu A do bodu B v priestore s prekážkami. Roboty sa budú pohybovať v čiastočne známej mape, t.j. dostanete neúplnú mapu prostredia. Pri súťaži sa vyhodnocuje čas, za ktorý sa dostanete do cieľa a presnosť s akou sa dostanete do bodu B. Najlepší získajú 5b najhorší 3b. Pre účastníkov ktorý nedôjdu do cieľa ale ich robot sa vyhol aspoň jednej prekážke sú rezervované 2 body. Za účasť je 1 bod.

Úloha 1. Lokalizácia a polohovanie robota v prostredí (10b)

Úlohou lokalizácie je povedať, kde v priestore sa mobilný robot nachádza. Najjednoduchší spôsob lokalizácie mobilného kolesového robota s diferenciálnym podvozkom (t.j. taký ako sa používa na cvičeniach) je odometria. Odometria je typ relatívnej lokalizácie (určuje polohu voči

predchádzajúcej polohe), ktorá prírastok polohy určuje na základe otočenia kolies. Keďže robot Kobuki má v sebe aj gyroskop môžete natočenie robota spresniť a vytvoriť takzvanú gyroodometriu.



Obrázok 1 Kinematická schéma diferenciálneho podvozku mobilného robota

Na výpočet odometrie pre robot zobrazený na obrázku 1, je nutné aplikovať tieto vzorce:

$$v_K = (\omega_{L_K} + \omega_{R_K})\frac{R}{2}$$

$$\varphi_{R_{K+1}} = \varphi_{R_K} + \Delta t(\omega_{R_K} - \omega_{L_K})\frac{R}{d}$$

$$x_{R_{K+1}} = x_{R_K} + \frac{d(v_r + v_l)}{2(v_r - v_l)}(\sin\varphi_{R_{K+1}} - \sin\varphi_{R_K})$$

$$y_{R_{K+1}} = y_{R_K} - \frac{d(v_r + v_l)}{2(v_r - v_l)}(\cos\varphi_{R_{K+1}} - \cos\varphi_{R_K})$$

Pre prípad ak sú obe rýchlosti rovnaké:

$$x_{R_{K+1}} = x_{R_K} + \Delta t v_K \cos\varphi_{R_K}$$

$$y_{R_{K+1}} = y_{R_K} + \Delta t v_K \sin \varphi_{R_K}$$

Úlohou polohovania je dostať mobilný robot na želané súradnice. To je pre robot Kobuki možné dosiahnuť riadením rýchlosti jednotlivých kolies. Na to, aby ste dosiahli želanú polohu presne, je nutné navrhnúť regulátor (P,PI,PID). Najjednoduchšie riešenie polohovania, je rozdeliť pohyb robota na transláciu a rotáciu (t.j. robot sa buď hýbe dopredu alebo sa točí na mieste) a pre každý z týchto pohybov navrhnúť regulátor samostatne. Pseudo-kód diskrétneho regulátora vyzerá takto:

```
previous_error = 0
integral = 0
start:
  error = setpoint - measured_value
  integral = integral + error*dt
  derivative = (error - previous_error)/dt
  output = Kp*error + Ki*integral + Kd*derivative
  previous_error = error
  wait(dt)
  goto start
```

Úloha 2 Navigácia (15b)

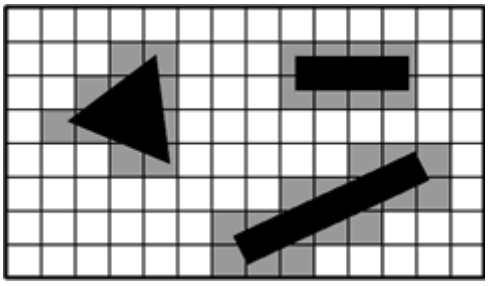
Navigácia mobilného robota zabezpečuje bezkolízny prechod robota prostredím. Na svoju činnosť využíva aktuálne údaje zo snímačov a nepotrebuje poznať prostredie vopred (mapu). K najjednoduchším algoritmom navigácie patria hmyzie algoritmy (bug algorithm), z toho len niektoré na svoje fungovanie využívajú laserový diaľkomer. Takýmto algoritmom je hmyzí algoritmus - typ dotyčnica (tangent bug algorithm).

Princíp algoritmu je:

1. Ak sa dá ísť na cieľ a nie si v móde sledovanie steny, choď na cieľ.
2. Ak je v ceste prekážka a nie si v móde sledovania steny, tak smeruj na jej hranu tak, aby bola euklidovská vzdialenosť, ktorú má robot prejsť čo najmenšia. Zapamätaj si aktuálnu vzdialenosť do cieľa.
3. Ak sa niekedy vzdialenosť od cieľa začne zväčšovať prepni sa na mód sledovanie steny.
4. Ak sleduješ stenu, tak ju sleduj až do prípadu až kým vzdialenosť do cieľa nie je menšia ako najkratšia zapamätaná vzdialenosť do cieľa.
5. Zastav, ak si dosiahol cieľ .

Úloha 3 Mapovanie (5b)

Úlohou mapovania je vytvoriť reprezentáciu prostredia na základe údajov zo snímačov robota. Inak povedané, mapovanie je fúzia (spájanie) informácií o polohe robota s informáciami o prekážkach získaných z laserového diaľkomera. Najjednoduchšia reprezentácia prostredia je mriežka obsadenia. Mriežka obsadenia rozdelí priestor na konečný počet prvkov (buniek), kde každý prvok obsahuje informáciu o jeho obsadenosti (t.j či sa tam nachádza prekážka).



Prerátanie údajov získaných z laserového diaľkomera do súradníc robota je možné na základe homogénnej transformácie. V prípade, že zanedbáme pohyb v osi Z (t.j hýbeme sa v rovine), ako aj rotácie okolo osi X a Y (robot sa pri pohybe nenakláňa do žiadneho smeru) je možné použiť homogénnu transformáciu pre 2 rozmery. 2D transformáciu je možné zapísať ako:

$$\begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\varphi_k & -\sin\varphi_k & 0 \\ \sin\varphi_k & \cos\varphi_k & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_l \\ y_l \\ 1 \end{bmatrix} + \begin{bmatrix} x_k \\ y_k \\ 1 \end{bmatrix}$$

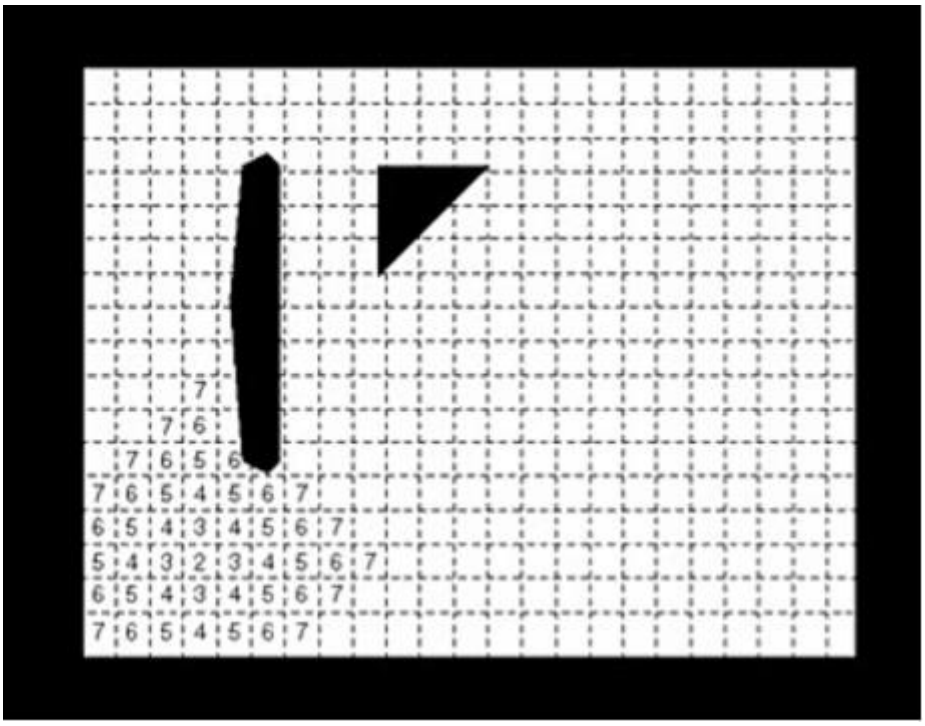
Blok 4 Plánovanie dráhy (5b)

Úlohou plánovania dráhy je vygenerovať optimálnu trasu pre mobilný robot na základe existujúcej mapy prostredia. Na tento účel sa využije typ mapy z predchádzajúcej úlohy a použitý algoritmus na plánovanie dráhy je záplavový algoritmus (flood fill algorithm). Jeho princíp je nasledovný:

1. Cieľová bunka je ohodnotená na 2. Všetky bunky susediace s touto bunkou sú ohodnotené na 3.

2. Nulové bunky susediace s bunkami s hodnotami 3 sú ohodnotené na 4.

3. Procedúra pokračuje dovtedy, kým nie je spojený štart a cieľ.



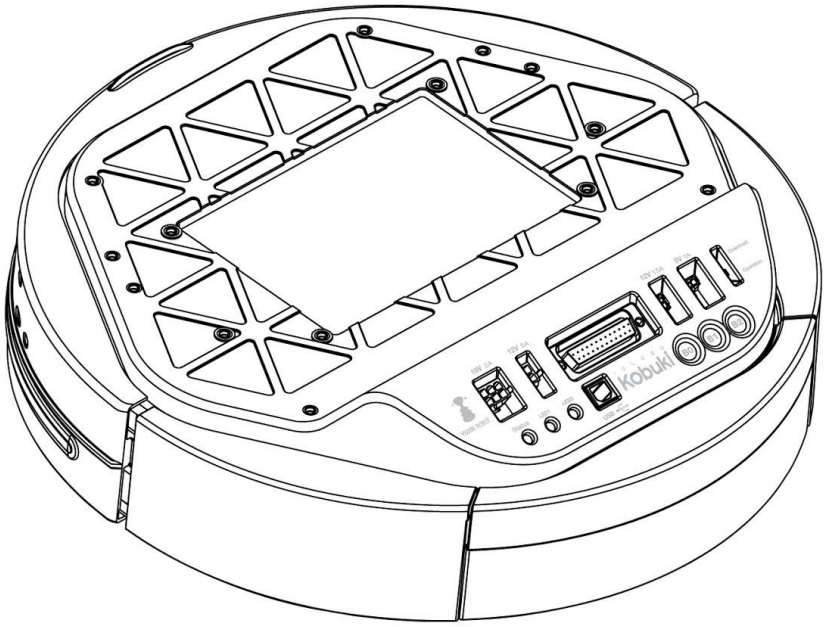
Bonusové úlohy

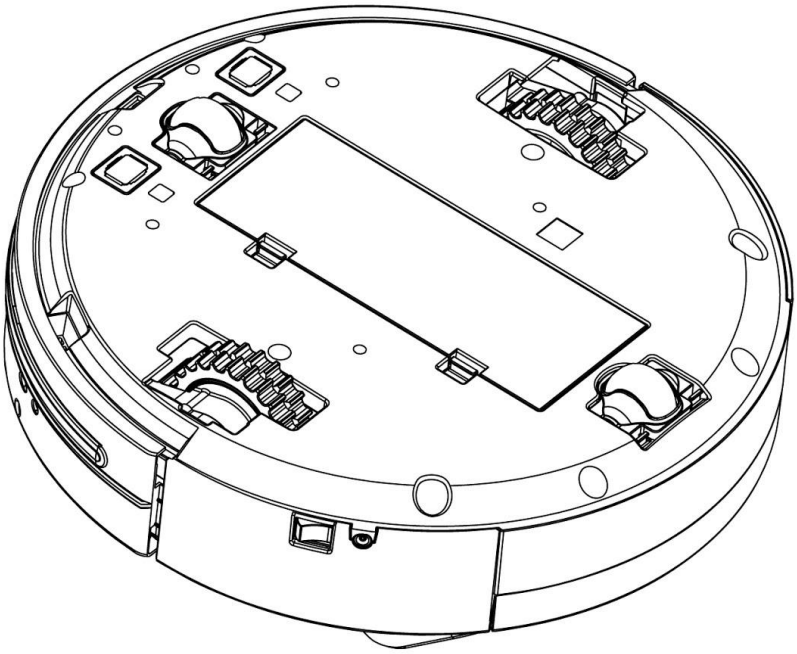
Bonusové úlohy sú úlohy, ktoré nie je možné zaradiť do predchádzajúcich blokov, ale napriek tomu patria k úlohám, ktoré sa riešia v mobilnej robotike.

V prípade záujmu, je možné získať bonusové body za prácu na „modrom“ a „čiernom“ robote. Podmienkou je získanie aspoň 20 bodov za štandardné úlohy. Maximum bonusových bodov, ktoré je možné takto získať je 20.

3. Hardvér

Robotický podvozok Kobuki





Podrobnú špecifikáciu protokolu môžete nájsť tu:

http://yujinrobot.github.io/kobuki/enAppendixProtocolSpecification.html

Čo potrebujete vedieť, ako vyzerá štruktúra príkazu ktorý posielate robotu(ako napríklad rýchlosť kolies).

Každá správa začína hlavičkou pozostávajúca z dvoch bytov 0xAA 0x55

Potom nasleduje veľkosť celej správy okrem hlavičky a checksum bytu. V správe sa môže nachádzať viacero príkazov (ak chcete v krátkom okamžiku posielať viac príkazov, spojte ich do jedného, neposielajte ich ako samostatné správy). Checksum je poslendý byte správy a je vyrátaný ako XOR všetkých bytov okrem hlavičky.

Každý príkaz má tiež svoju hlavičku, ktorá sa skladá z identifikátora a počtu dátových bytov za ktorými nasledujú dátové byty.

Pre príkaz pohybu rovno s rýchlosťou 250 mm/s by správa vyzerala takto:

0xAA 0x55 0x06 0x01 0x04 0xFA 0x00 0x00 0x00 0xFF

Dajte si však pozor, každá správa musí obsahovať príkaz na pustenie 12V vetvy, inak sa vypne raspberry pi. Pre istotu je súčasťou všetkých demo správ príkaz na zapnutie všetkých napájaní (ak by si chcel niekto pripojiť ďalšie zariadenie na niektorú z týchto vetiev)

0x0c 0x02 0xf0 0x00

Čiže predchádzajúca správa ktorá má poslať robot rýchlosťou 250 mm/s by vyzerala takto:

0xAA 0x55 0x0A 0x0c 0x02 0xf0 0x00 0x01 0x04 0xFA 0x00 0x00 0x00 0x01

Zoznam možných príkazov je v nasledujúcej tabuľke:

Názov	Identifikátor	Počet dátových bytov
Ovládanie motorov	0x01	0x04
Hranie noty	0x03	0x03
Hranie sekvencie zvukov	0x04	0x01
Vyžiadanie doplnkových informácií	0x09	0x02
Ovládanie periférií	0x0C	0x02
Nastavenie PID regulátora rýchlosti	0x0D	0x0D
Získanie aktuálnych hodnôt PID regulátora	0x0E	0x01

Údaje zo snímačov robot posiela v pravidelných intervaloch(50Hz). Podobne ako príkazové správy, správy zo snímačov sú vyskladané z viacerých packetov. Podrobne si ich môžete naštudovať na stránke špecifikácie protokolu.

Ak budete používať demo príklad tak vás parsovanie nemusí trápiť, je už spravené. To s čím budete pracovať je štruktúra dát vo formáte:

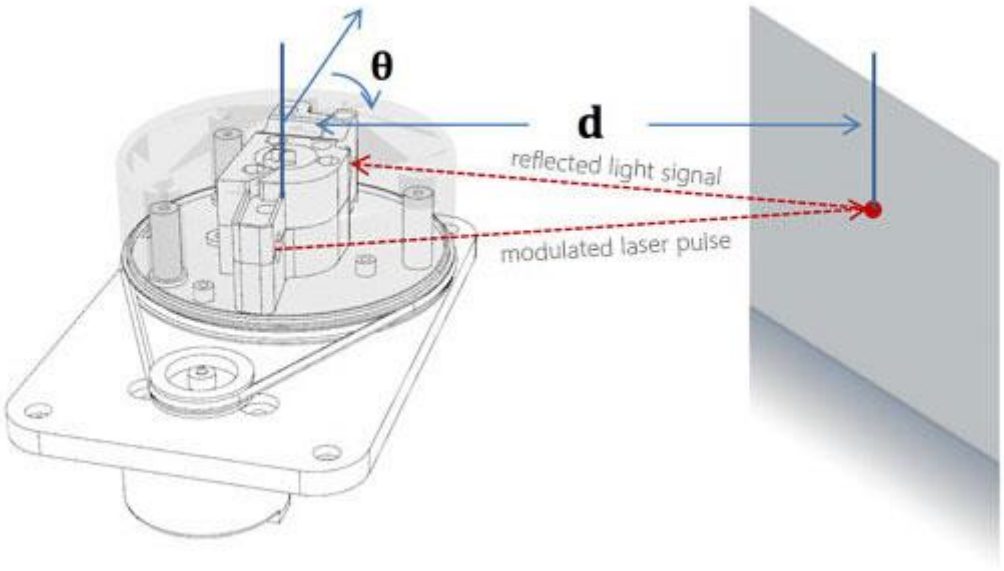
```
typedef struct
{
        //---zakladny balik
        unsigned short timestamp;
        //narazniky
        bool BumperLeft;
        bool BumperCenter;
        bool BumperRight;
        //cliff
        bool CliffLeft;
        bool CliffCenter;
        bool CliffRight;
        // padnutie kolies
        bool WheelDropLeft;
        bool WheelDropRight;
        //tocenie kolies
        unsigned short EncoderRight;
        unsigned short EncoderLeft;
        unsigned char PWMright;
        unsigned char PWMleft;
        //gombiky
        unsigned char ButtonPress;// 0 nie, 1 2 4 pre button 0 1 2 (7 je ze
vsetky tri)
        //napajanie
        unsigned char Charger;
        unsigned char Battery;
        unsigned char overCurrent;
        //---docking ir
        unsigned char IRSensorRight;
        unsigned char IRSensorCenter;
        unsigned char IRSensorLeft;
        //---Inertial Sensor Data
        signed short GyroAngle;
        unsigned short GyroAngleRate;
        //---Cliff Sensor Data
        unsigned short CliffSensorRight;
        unsigned short CliffSensorCenter;
        unsigned short CliffSensorLeft;
        //---Current
        unsigned char wheelCurrentLeft;
        unsigned char wheelCurrentRight;
        //---Raw Data Of 3D Gyro
        unsigned char frameId;
        std::vector<TRawGyroData> gyroData;
        //---General Purpose Input
        unsigned short digitalInput;
        unsigned short analogInputCh0;
        unsigned short analogInputCh1;
        unsigned short analogInputCh2;
        unsigned short analogInputCh3;
        //---struktura s datami ktore sa nam tam objavia iba na poziadanie
        TExtraRequestData extraInfo;
```

}TKobukiData;

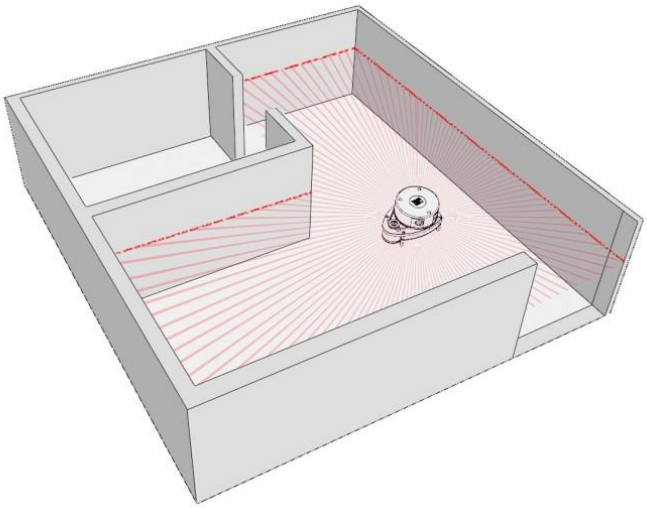
```
typedef struct
{

        unsigned short x;
        unsigned short y;
        unsigned short z;

}TRawGyroData;

typedef struct
{
        //Hardware Version
        unsigned char HardwareVersionMajor;
        unsigned char HardwareVersionMinor;
        unsigned char HardwareVersionPatch;
        //Firmware Version
        unsigned char FirmwareVersionMajor;
        unsigned char FirmwareVersionMinor;
        unsigned char FirmwareVersionPatch;

        //Unique Device IDentifier(UDID)
        unsigned int UDID0;
        unsigned int UDID1;
        unsigned int UDID2;
        //Controller Info
        unsigned char PIDtype;
        unsigned int PIDgainP;
        unsigned int PIDgainI;
        unsigned int PIDgainD;
}TExtraRequestData;
```

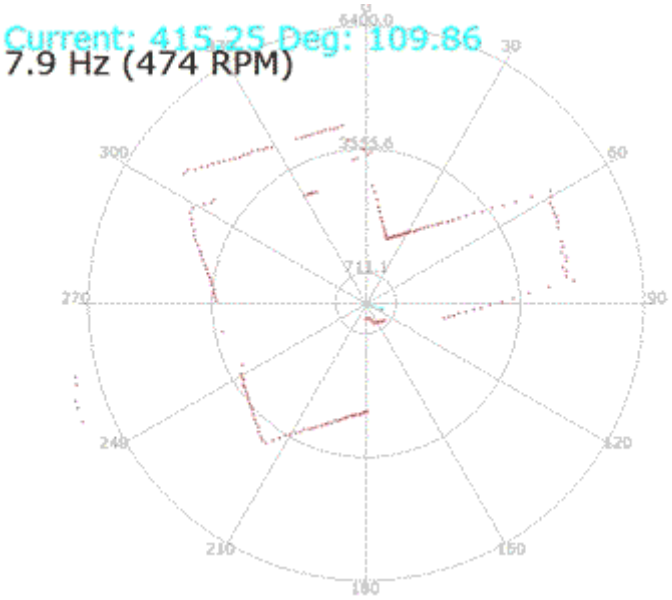
Laserový diaľkomer RPLidar.

Laserový diaľkomer RPlidar je diaľkomer, ktorý meria vzdialenosť od robota k prekážkam v jednej rovine. Princíp fungovania je na základe triangulácie zosnímaného odrazu vysielaného laserového lúča.



Pokrytie celej plochy v okolí robota je dosiahnuté rotáciou hlavy laserového diaľkomera. Uhlové rozlíšenie snímača, je závislé od rýchlosti otáčania, na cvičeniach je nastavené na 0.8° až 1.3°.





Lidar poskytuje údaje o meraní vo forme stream-u meraní, kde každý blok obsahuje informácie o uhle (natočenie hlavy) , nameranej vzdialenosti a intenzite (množstvo energie, ktoré sa odrazí od povrchu).