BPC-PRP Projekt

Filip Botlo
Fakulta informačních technologií
Vysoké učení technické v Brně
Brno, Česká republika
filip.botlo@gmail.com

Jakub Kolář
Fakulta elektrotechniky a komunikačních technologií
Vysoké učení technické v Brně
Brno, Česká republika
247395@vutbr.cz

***Abstrakt*—Článek se zabývá implementací algoritmů pro autonomního robota, jehož cílem je vidět z bludiště.**

***Klíčová slova*—robotika, BPC-PRP, lidar, IMU, PID**

I. Úvod

Článek se zaměřuje na programové vybavení autonomního robota, jehož úkolem je vyjet z bludiště sestaveného z buněk 40 na 40 cm o maximální rozloze 8 na 8 buněk. K řízení robota byl využit framework ROS 2, který usnadňuje integraci komponent, jako jsou senzory a řízení pohybu. ROS 2 podporuje vícejádrové procesory a komunikaci přes middleware DDS, což ho činí vhodným pro real-time a distribuované aplikace.

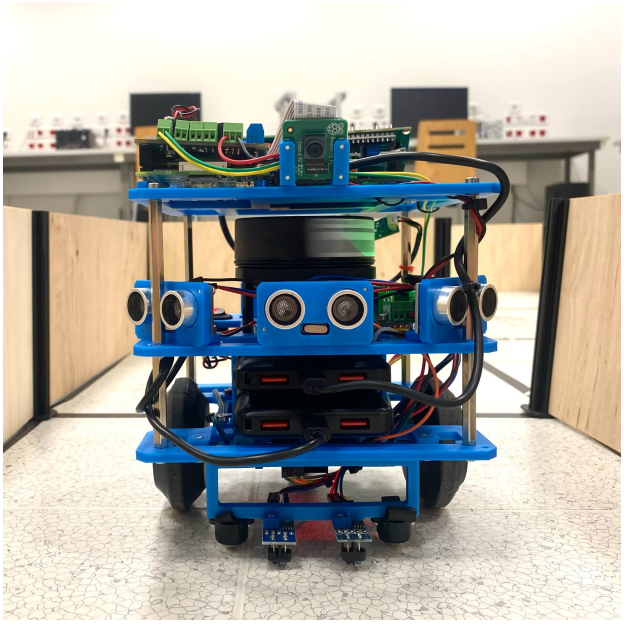

Fig. 1. Robot

Z konstrukce robota vyplývá, že je neholonomní, což znamená, že stupeň volnosti pohybu robota (pohyb v jedné ose a rotace) je menší než stupeň volnosti prostoru (3 DOF). O pohyb robota se starají kola poháněná motory na stranách.

Pro řízení robota používáme vybrané komponenty, které umožňují detekovat okolní prostor a orientaci robota v něm. Lidar je používán k detekci překážek (okolních stěn), díky kterým je možné řízení pohybu robota v prostoru. IMU v projektu slouží k řízení otáčení robota v křižovatkách. O detekci Aruco tagů, které poskytují informaci o nejkratší cestě z bludiště, se stará kamera.

II. Řízení robota v koridoru

V počátečním stavu se robot nachází ve stabilní pozici, kdy probíhá kalibraci IMU. Teprve až po stisknutí prostředního tlačítka začne robot vykonávat řízený pohyb. V rámci řízení pohybu v node nazvaný *pid* robot přepíná mezi 2 stavy, a to DRIVE_FORWARD a TURNING.

Řízení robota ve stavu DRIVE_FORWARD spočívá v nastavování rychlostí otáčení do levého a pravého kola. Při regulaci v koridoru jsou tyto rychlosti nastavovány PD regulátorem. Na základě pozice robota v prostoru se mění relativní odchylka robota od ideální pozice (*error*), pomocí níž se vypočítá akční zásah do levého a pravého kola. Odchylka je pro filtraci brána jako průměr ze dvou po sobě jdoucích hodnot (*error_avg*) a ve výpočtu akčního zásahu (*speed_diff*) násobena proporcionální konstantou $Kp = 6$. Akční zásah se dále skládá z derivační složky určené rozdílem dvou posledních odchylek *d_error* násobené derivační konstantou $Kd = 4$ (rovnice 1).

$$speed\_diff = Kp * error\_avg + Kd * d\_error \quad (1)$$

Pokud se robot nachází blízko u jedné ze zdí (vzdálenost ode zdi je pod úrovní 0,20), akční zásah (*speed_diff* s příslušným znaménkem) se nastaví na hodnotu 2, aby se robot odchýlil ode zdi.

Na levé kolo je pak nastavena základní rychlost *base_speed* = *157* mínus hodnota *speed_diff*, na pravém kole je tato hodnota k základní rychlosti přičtena. Hodnota *speed_diff* je programem saturována na interval $\langle -2, 2 \rangle$.

Nejjednodušší a nejspolehlivější typ řízení, které robot využívá, je řízení v koridoru. V tomto případě robota obklopují z obou stran stěny (obr. 2), takže je relativní odchylka určena rozdílem výsečí ve výhledu lidaru vlevo a vpravo. Robot se při tomto řízení snaží vycentrovat svou pozici vhledem k okolním stěnám.

Pokud robot kolem sebe nemá obě stěny, orientuje se za pomoci pouze jedné stěny. Konkrétně pouze té, která je blíž. To demonstruje obr. 3. V tomto případě se robot řídí bang bang regulací. Při překročení vzdálenosti 0,23 od stěny je nastavena hodnota *speed_diff* na 2, tak aby robot provedl pohyb směrem ke stěně. Při překročení minimální vzdálenosti od stěny (0,20)

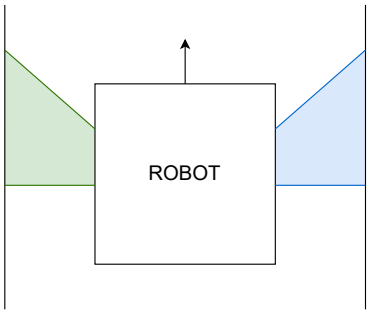

Fig. 2. Relativní pozice robota v koridoru při řízení na základě obou stěn

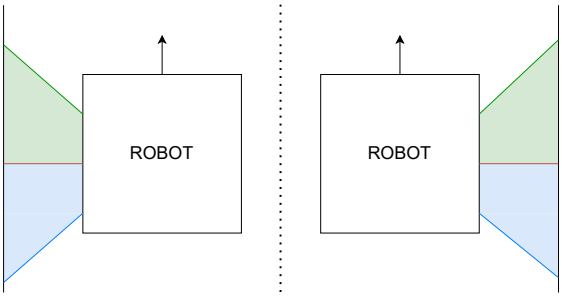

Fig. 3. Relativní pozice robota v koridoru při řízení na základě jedné stěny

se uplatňuje předešlého mechanismu akčního zásahu směrem od stěny. Této regulace se využívá i ve stavu stavu po otočení robota, který narazil do stěny. Při rotaci doleva se následně robot řídí pravou stěnou, při rotaci doprava levou stěnou. Tento typ regulace je pevně nastaven po 0,75 sekund po otočce.

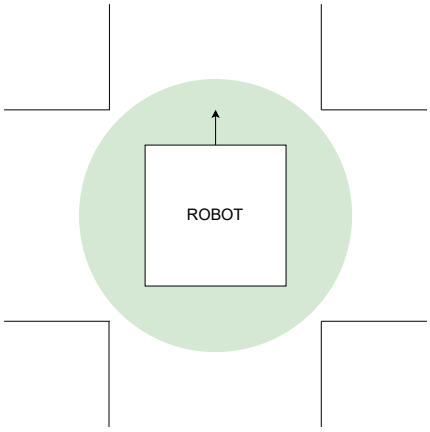

Fig. 4. Relativní pozice robota v křižovatce +

Poslední situace, ve které se robot může nacházet, je křižovatka typu ”+” (obr. 4). V této fázi robot při vjezdu spoléhá na vyregulování pozice v předchozím koridoru a dále se již spoléhá pouze na jízdu rovně, protože nevidí záchytný bod, kterého by se mohl řídit.

III. Chování robota při změně směru jízdy

Pohyb robota v bludišti krom regulace jízdy rovně dále zahrnuje otáčení. Robot se při rotaci řídí na základě dat z inerciály. Při počátečním postavení se provede kalibrace, kdy se zaznamená úhel v této pozici. Při požadované rotaci robota v bludišti se pak každá tato rotace vztahuje k počátečnímu nakalibrovanému úhlu. Povolené rotace jsou pouze v násobcích $k \cdot 90°$ od kalibrovaného úhlu.

Při rotaci se nastaví rychlosti otáčení jednotlivých motorů na *128 - turn_speed* a *128 + turn_speed* tak, aby se kola otáčela na opačnou stranu. Pokud je odchylka od cíle rotace větší než 0,3 rad, *turn_speed* je nastavena na hodnotu 15, v opačném případě na hodnotu 3. Po překročení odchylky 0,09 rad od požadované hodnoty úhlu natočení se rotace zastaví a přechází se zpět do stavu DRIVE_FORWARD.

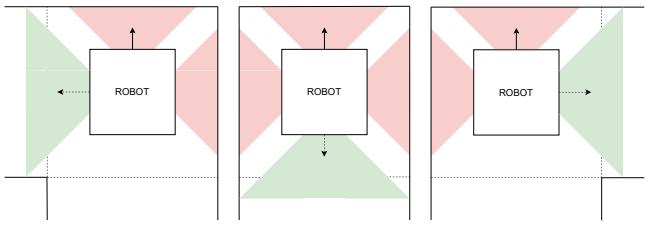

Fig. 5. Robot v povinné zatáčce

A. Povinné zatáčky

Prvním typem otáček je takzvané povinné otáčení, které nastává, jestliže robot před sebou lidarem detekuje překážku (ve vzdálenosti menší než 0,26 předního paprsku lidaru) a má možnost se otočit pouze na jednu ze stran. V tomto případě se robot rozhoduje o otočce na základě dat z bočních paprsků lidaru. V případě, že robot detekuje místo ve vedlejší buňce na jedné ze stran a na druhé ne, zatočí právě tam, kde je volno o 90°. Pro slepé místo v bludišti robot nedetekuje prostor ani v jedné z vedlejších buněk, a tak rovnou provede otočku o 180°.

B. Zpracování ArUco značiek

Pro zpracování křižovatek využíváme kameru, která detekuje ArUco značky a používá je pro rozhodnutí o směru. Konkrétně jsou zpracovány a uloženy ve funkci *aruco_callback*, která nám umožňuje výběr mezi značkami s ID (vedoucími k východu) a ID (vedoucími k pokladu). Následně se ID ukládá do proměnné *last_aruco_id*, jejíž hodnotu program uchovává, dokud není použita na některé z křižovatek.

C. Zatáčky s výběrem

Druhým typem otáček jsou otáčky s výběrem, kde se robot řídí předem nasnímanými ArUco značkami. V první možnosti se robot zastaví před stěnou a má na výběr mezi pravou a levou stranou (obr. 6), přičemž si stranu zvolí podle výše zmíněné proměnné. Ve druhé možnosti (obr. 7)) robot jede v koridoru a když mine volnou stěnu, také si podle proměnné vyhodnotí, zda bude pokračovat rovně, nebo odbočí.

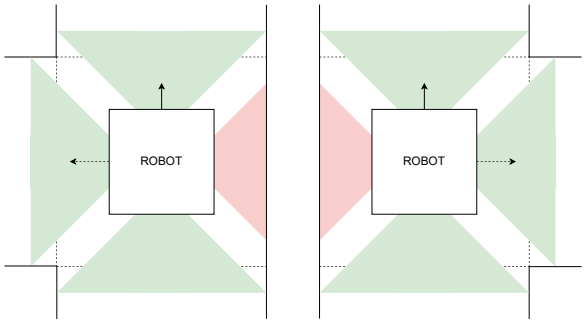

Fig. 6. Robot v křižovatce s volitelným výběrem směru

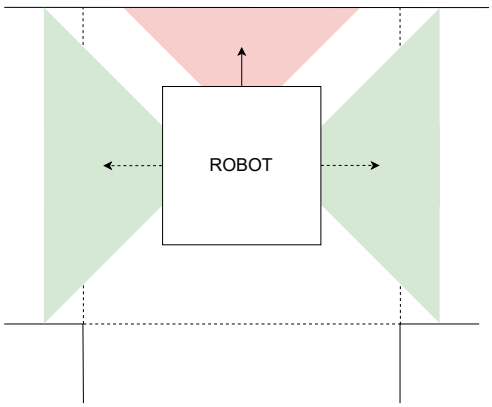

Fig. 7. Robot v křižovatce s volitelným výběrem směru

IV. Shrnutí projektu

Robot má pomocí IMU, lidaru a kamery implementovaný systém, který mu umožňuje navigovat se a úspěšně uniknout z bludiště ve 3D prostoru.