Semestrální projekt LAR

Matěj Pinkas, Martin Erben, Filip Hudec

27.3.2024

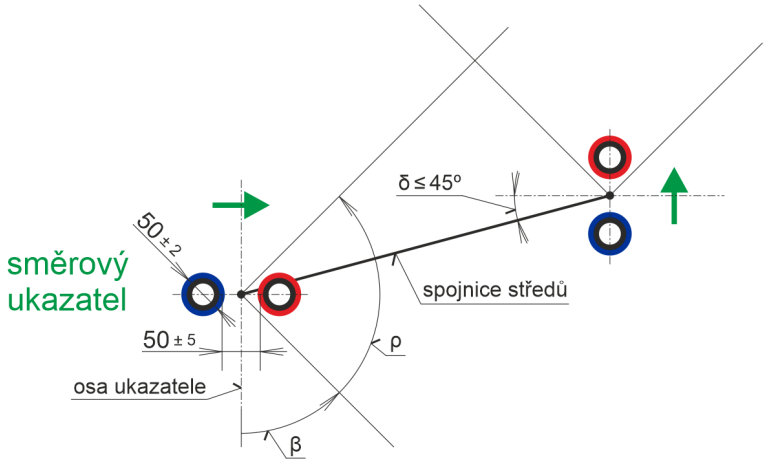
Obsah

1 Úvod

Tento projekt slouží jako výstup z předmětu LAR. Cílem je projet stanovenou dráhu s robotem v časovém limitu, kde se bude pohybovat zcela autonomně podle barevných směrovek. Pro splnění tohoto úkolu používáme platformu TurtleBot. Jedná se ve stručnosti o hnaný dvoukolový podvozek, který je doplněn o nástavbu s osazenými senzory (více v sekci 4.1.1). Směrovky jsou tvořeny dvojicí barevných sloupků.
Na výběr jsme dostali ze tří zadání podle úrovně obtížnosti. Náš tým si vybral 2. možnost - průjezd tratě bez překážek s tím, že přidáme zastavení u poslední dvojice zelených sloupků.

2 Rozbor problému

Robot je nejdříve umístěn na startovní pozici, ve výseči první směrovky $\pm 45°$, která má vrchol ve středu směrovky. Vzdálenost start – první ukazatel leží v intervalu $(a; b) = (500; 3000)$ mm. Robot může být libovolně natočen, takže si musí první směrovku najít s tím, že volí tu nejblíže k němu.
Poté, co spustíme na robotovi program a cvičící stiskne tlačítko pro zahájení pohybu, již nemůžeme robota externě ovládat a musí se rozhodovat autonomně.
Pro případ, že by došlo k chybě (tj. nárazu do jakékoliv překážky) je robot vybaven nárazníkem, který pošle callback řídícímu programu a ten je okamžitě ukončen.
Naše použité směrovky jsou tvořeny modrým a červeným sloupkem. Jejich rozestup je specifikován na 50 ± 5 mm.
Při navigaci od aktuální k následující směrovce musí spojnice středů těchto směrovek s osou následující směrovky vždy svírat úhel $\delta \leq 45°$. Následující směrovka leží vždy vpravo nebo vlevo od aktuální a to ve výseči $\rho = 90°$, která začíná od úhlu $\beta = 45°$ měřeno od osy aktuální směrovky (viz Obr. 1).
Časový limit na kompletní průjezd je v této úloze stanoven na 5 min.

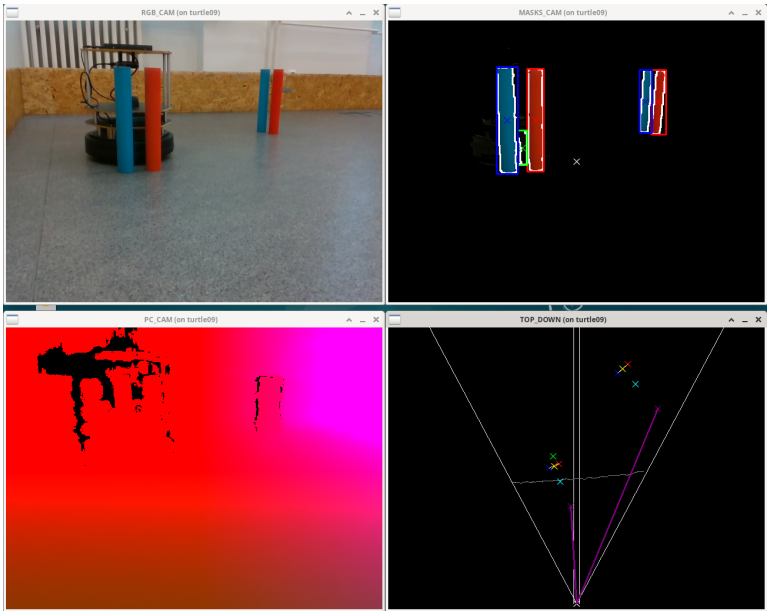


Obr. 1: Parametry vyznačené tratě

2.1 Návrh řešení

Při spuštění programu si robot inicializuje RGB a 3D hloubkovou kameru. Na obrazovce počítače, ze kterého spouštíme řídicí program zobrazujeme 4 okna (viz Obr. 2): obraz z hloubkové kamery; obraz z RGB kamery; obraz z RGB kamery s vymaskovanými směrovkami a našemi navigačními body; mapa zobrazující výseč prostoru, kterou robot snímá (top-down view).
Robot si uloží 2 nebližší směrovky, které vidí, podle dat z RGB a hloubkové kamery. Vybere z nich tu bližší a spočítá si 3 virtuální body (M, T, P), podle kterých se bude navigovat k nejbližší směrovce.
Bod M leží v polovině spojnice středů dvou sloupků. Bod T (*target*) leží na kolmici ke spojnici středů sloupků (kde bod M je průsečík těchto přímek) ve vzdálenosti 50 ± 5 mm od sloupků a je to bod na který se snažíme dostat tak, aby ještě robot neshodil sloupky. Bod P potom leží na stejné kolmici jako T a M, ale leží ve vzdálenosti 100 mm od sloupků.

Robot si uloží z odometrie počáteční bod (resetuje odometrii) a orientuje se nejdříve na bod P, ke kterému se kolmo natočí ze své pozice a potom se k němu pohybuje již po přímce. Zde se natočí kolmo ke sloupkům a pomocí drobných translačních a rotačních pohybů se snaží dostat na požadovaný bod T. Robot se otočí podle směrovek o 90°a celý proces iteruje dokud se nedostane k zelené směrovce.



Obr. 2: Obraz z pohledu robota

2.2 Očekávaná funkčnost

Robot by neměl mít problém se změnou osvětlení prostředí, protože při výpočtu masek pro RGB obraz naše funkce *drawMasks* upravuje výchozí přednastavený rozsah barev dynamicky podle dat získaných z RGB kamery. Největším problémem kterému jsme čelili, je relativně velká nepřesnost odometrických dat, získaných z motorů robota, což nás při řízení pohybu nutilo k použití experimentálně zjišťovaných konstant, které přesně neodpovídali našim teoretickým očekávaným výsledkům. Podobně bylo třeba upravovat i data získaná z hloubkové kamery, aby jsme docílili požadovaného chování.

3 Řešení problému

3.1 Matematický model

3.2 Algoritmizace

4 Implementace

4.1 Hardware & programovací jazyk

4.1.1 Platforma TurtleBot

Jak už jsme zmínili, používáme open-source platformu TurtleBot 2.0 s podvozkem od firmy Kobuki. Podvozek je vybaven dvěma hnanými koly, z nichž odečítá odometrická data (natočení kol). Dále má na čelní straně blatník, který se dá zmáčknout ve třech segmentech (Right, Left, Center) a v případě srážky s překážkou pošle callback, kterým můžeme zastavit chod řídicího programu.
V nástavbě se schovává počítač Intel NUC, na kterém běží operační systém Linux, a v němž spouštíme náš řídicí program prostřednictvím virtualizace Singularity. Dále zde najdeme senzor Intel RealSense D435/D435i, který obsahuje RGB kameru s rozlišením 640x480 px a 3D hloubkovou kameru se stejným rozlišením.

4.1.2 Python

Celý řídicí program je implementovaný v programovacím jazyce Python. Jedná se o interpretovaný jazyk, což požaduje instalaci interpreteru v místě, kde chceme spouštět program. S robotem komunikujeme voláním jeho třídy *TurtleBot*, která je implementována v dodaném balíčku *robolab_turtlebot*. Pro zpracování obrazu používáme knihovnu *cv2* a pro matematické operace knihovnu *NumPy*.

4.2 Detaily implementace

5 Závěr

6 Literatura