SLAM - rešerše
Projekt 4

Jakub Kratochvíl

Akademický rok 2017/2018

Obsah

1 Úvod **2**

2 SLAM **3**

3 Pravděpodobnostní definice **4**

4 Graph-Based SLAM **6**
4.1 Definice 7
4.1.1 Front-end 7
4.1.2 Back-end 8

5 Vizuální SLAM **8**
5.1 Základní struktura vSLAM 9
5.2 Dodatkové moduly vSLAM 9
5.3 Metody řešení 10
5.3.1 S použitím významných bodů 10
5.3.2 RGB-D 10
5.3.3 Přímé metody 11

1 Úvod

SLAM je zkratka pro simultánní lokalizaci a mapování, jeden ze základních problémů autonomních robotů. Jeho řešením by měl být robot, schopný na neznámém místě v neznámém prostředí vytvořit mapu tohoto prostředí a zároveň se v ní sám během pohybu lokalizovat.

Do povědomí se SLAM dostal v roce 1986 na konferenci IEEE Robotics and Automation, která se konala v San Francisku v Kalifornii [4]. V tomto období se v robotice a umělé inteligenci začaly objevovat metody založené na pravděpodobnostních principech a tak vyvstala otázka možnosti použití odhadových metod na mapovací a lokalizační problémy. Po následujících diskuzích se problém ukázal jako velice zajímavý a konzistentní pravděpodobnostní mapování stalo jedním ze základních problémů a výzev robotiky.

Aplikací SLAM můžeme najít mnoho, počínaje autonomním domácím vysavačem nebo sekačkou na trávu přes robotický průzkum opuštěných nebo člověku nebezpečných prostor, navigaci ponorek kolem podmořských přírodních překážek, řízení bezpilotních letounů a dronů až po v poslední době hodně diskutované samořídící automobily nebo dokonce planetární rovery brázdící povrch Marsu.

Tato práce si klade za úkol čtenáře seznámit se základními vlastnostmi a popisem problému SLAM. Dává nahlédnout do jeho původní pravděpodobnostní definice a v další části se podrobněji zabývá metodou Graph SLAM.

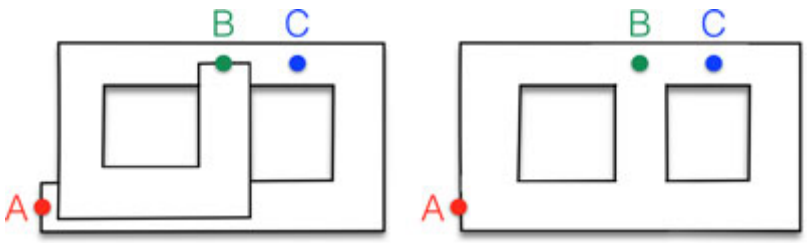
2 SLAM

SLAM je problém týkající se otázky, zda je možné najít polohu nějakého zařízení vzhledem k jeho okolí a současně mapovat strukturu prostředí.

Landmarky nebo také majáky jsou nezaměnitelné a snadno rozpoznatelné orientační body v prostředí, které mohou být v některých aplikacích dopředu známy. Například autonomní vozík pohybující se ve výrobní hale může mít předem nadefinovanou mapu landmarků. Nebo robot pro práci pod širým nebem používající pro svou orientaci GNSS (globální družicový polohový systém). V takových případech, kdy lze stroj lokalizovat vzhledem ke známým bodům, nemusí být SLAM vyžadován. Ovšem v husté zástavbě, v podzemí a uvnitř budov je použití GNSS omezené nebo úplně nemožné. V neznámém prostředí zase nelze využít předem připravenou mapu a přichází nutnost použití jiného řešení, kterým bývá nejčastěji právě SLAM. Navíc v mnoha vojenských i civilních aplikacích není cílem lokalizace, ale právě robotem vytvořená mapa, kterou poté dále zpracovává lidský operátor.

Další z impulzů vývoje simultánní lokalizace a mapování byl špatný odhad pohybu získaný z odometrie kol, čímž se rozumí například počet otáček, úhel natočení apod. [3]. Tento odhad se navíc s ujetou vzdáleností zhoršuje (tzv. drift) a tím znemožňoval použití pro správnou a přesnou lokalizaci i tvorbu mapy. Nicméně dnešní algoritmy dokáží snížit drift na přijatelnou hodnotu 0,5% a méně, takže v tomto ohledu není nutno využívat SLAM.

V čem je ale stále nenahraditelný, je schopnost tzv. uzavírání smyček. Při použití dat pouze z odometrie robot vnímá svět jako "nekonečný koridor", ve kterém neustále zkoumá nové oblasti (obr.1 vlevo). SLAM ale dokáže robotem vytvořenou smyčku uzavřít, protože porovná aktuální landmarky s těmi dříve objevenými a tím dokáže správně porozumět topologii prostředí a v mapě určit, že tyto dvě chodby se protínají (obr.1 vpravo).



Obrázek 1: Levá mapa je vytvořená pomocí odometrie a znázorňuje jeden dlouhý koridor od bodu A do B. Body, které jsou ve skutečnosti blízko (B a C) mohou být podle této mapy libovolně daleko. Pravá mapa je vytvořená pomocí SLAM s využití uzavírání smyček, kde je znázorněná správná topologie prostředí, neboť robot správně našel "zkratku"mezi dvěma koridory. [3]

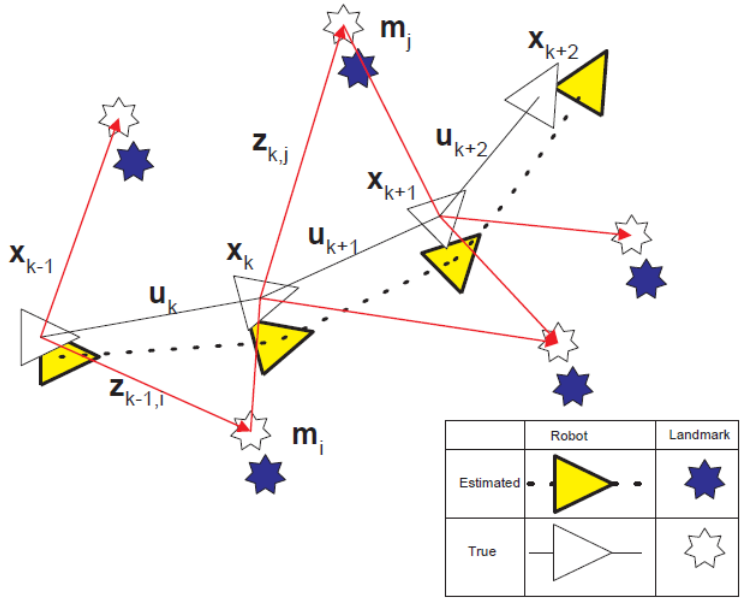
SLAM je ze své podstaty problém typu slepice-vejce a je tedy do značné míry netriviální. Robot pro svoji lokalizaci potřebuje mapu terénu, avšak k sestavení mapy musí znát svou vlastní polohu. Ovšem existují algoritmy, které, i přes tento rozpor, uspokojivě fungují a nasnadě je tedy otázka "Je SLAM vyřešen?". Ano i ne, otázku je nutno položit pro konkrétní konfiguraci robota, prostředí a požadavků, kde může figurovat mnoho kombinací, např. z následujících možností.

- Robot: dynamika, maximální rychlost, dostupné senzory, výpočetní výkon
- Prostředí: 2D/3D, přírodní/umělé, přítomnost dynamických prvků, množství a typ landmarků, množství symetrie
- Požadavky: přesnost odhadu stavu robota, přesnost a typ mapy, míra úspěšnosti, latence odhadu, maximální velikost mapované oblasti...

Například mapování vnitřního 2D prostředí s robotem vybaveným snímačem kol a laserovým senzorem s dostatečnou přesností a robustností lze považovat z velké části za vyřešené. Na druhou stranu další kombinace robot/prostředí/požadavky si stále zaslouží velké množství výzkumu. Aktuální algoritmy mohou snadno selhat, jestliže je pohyb robota nebo prostředí příliš náročný.

3 Pravděpodobnostní definice

Uvažujme mobilního robota, který se pohybuje v neznámém prostředí a pomocí svých senzorů pořizuje relativní pozorování okolních landmarků (obr.2).



Obrázek 2: Skutečné umístění landmarků ani robota není nikdy známo, jen jejich vzájemná poloha [4].

V okamžiku k definujeme následující veličiny a zároveň vektory jejich historie od počátku snímání až do aktuálního časového kroku:

- $\mathbf{x}_k$; $\mathbf{X}_{0:k} = \{\mathrm{x}_0, \mathrm{x}_1, \cdots, \mathrm{x}_k\} = \{\mathbf{X}_{0:k-1}, \mathbf{x}_k\}$: Stavový vektor popisující polohu a orientaci robota, respektive historii všech jeho stavů.
- $\mathbf{u}_k$; $\mathbf{U}_{0:k} = \{\mathrm{u}_1, \mathrm{u}_2, \cdots, \mathrm{u}_k\} = \{\mathbf{U}_{0:k-1}, \mathbf{u}_k\}$: Vektor řízení použitý v čase k-1 pro dosažení stavu x_k v čase k, resp. historie řízení.
- $\mathbf{m}_i$; $\mathbf{m} = \{\mathrm{m}_1, \mathrm{m}_2, \cdots, \mathrm{m}_n\}$: Vektor popisující polohu i-tého landmarku, resp. vektor popisující polohy všech landmarků.
- $\mathbf{z}_{ik}$; $\mathbf{Z}_{0:k} = \{\mathrm{z}_1, \mathrm{z}_2, \cdots, \mathrm{z}_k\} = \{\mathbf{Z}_{0:k-1}, \mathbf{z}_k\}$: Pozorování o poloze i-tého landmarku získané z robota, resp. soubor všech pozorování.

Skutečný robot bude vždy zatížen nějakou nepřesností použitých měřících zařízení a tak se přímo nabízí využití pravděpodobnosti pro formulaci úlohy SLAM. Tento přístup je jedním z nejběžnějších a také nejstarších. K vyřešení problému SLAM je potřeba určit sdruženou aposteriorní hustotu pravděpodobnosti landmarků $\mathbf{m}$ a stavů robota x_k pro všechny časy k.

$$p(\mathbf{x}_k, \mathbf{m} \,|\, \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) \tag{1}$$

K výpočtu je nutno znát vektor pozorování, vektor řízení a počáteční polohu robota. Výpočet hustoty se provádí pro každý časový okamžik k. Vhodné je použít rekurzivní algoritmus, který získá hustotu pravděpodobnosti v časovém okamžiku k pomocí hustoty v okamžiku $k-1$.

$$p(\mathbf{x}_{k-1}, \mathbf{m} \,|\, \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}) \tag{2}$$

Tento výpočet vyžaduje definování modelu pozorování a pohybového modelu.

1. **Model pozorování** popisuje s jakou pravděpodobností získáme pozorování z_k, pokud známe polohu robota i landmarků.

$$p(\mathbf{z}_k \,|\, \mathbf{x}_k, \mathbf{m}) \tag{3}$$

2. **Pohybový model** popisuje s jakou pravděpodobností se robot nachází ve stavu x_k, pokud známe předchozí stav x_{k-1} a aplikované řízení u_k.

$$p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{u}_k) \tag{4}$$

Stavový přechod pohybového modelu je Markovský proces, neboť stav x_k závisí pouze na předchozím stavu x_{k-1} a aplikovaném řízení u_k a je nezávislý na pozorováních i mapě. SLAM algoritmus je nyní implementován ve standardní dvoustupňové (predikce-korekce) rekurzivní formě.

1. **Predikce**

$$p(\mathbf{x}_k, \mathbf{m} \,|\, \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) \;= \tag{5}$$

$$= \int p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{u}_k) \;\times\; p(\mathbf{x}_{k-1}, \mathbf{m} \,|\, \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0) d\mathbf{x}_{k-1}$$

2. **Korekce**

$$p(\mathbf{x}_k, \mathbf{m} \,|\, \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) \;=\; \frac{p(\mathbf{z}_k \,|\, \mathbf{x}_k, \mathbf{m}) p(\mathbf{x}_k, \mathbf{m} \,|\, \mathbf{Z}_{0:k-1, \mathbf{U}_{0:k}, \mathbf{x}_0})}{p(\mathbf{z}_k \,|\, \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})} \tag{6}$$

Rovnice (5) a (6) poskytují rekurzivní postup pro výpočet hustoty pravděpodobnosti $p(\mathbf{x}_k, \mathbf{m} \,|\, \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_k)$ stavu robota $\mathbf{x}_k$ a mapy $\mathbf{m}$ v čase k na základě všech pozorování $\mathbf{Z}_{0:k}$ a všech řízení $\mathbf{U}_{0:k}$.

Problém budování mapy lze formulovat jako výpočet podmíněné hustoty pravděpodobnosti $p(\mathbf{m} \,|\, \mathbf{X}_{0:k}, \mathbf{Z}_{0:k}, \mathbf{U}_{0:k})$. To předpokládá, že umístění robota $\mathbf{x}_k$ je známo (nebo alespoň deterministické) po celou dobu, pod podmínkou znalosti počátečního umístění. Mapa $\mathbf{m}$ se potom vytvoří spojením pozorování z různých míst. Naopak problém lokalizace může být formulován jako výpočet pravděpodobnostní distribuce $p(\mathbf{x}_k \,|\, \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{m})$. To předpokládá, že je známo umístění landmarků a cílem je vzhledem k nim vypočítat odhad umístění robota. Avšak model pozorování $p(\mathbf{z}_k \,|\, \mathbf{x}_k, \mathbf{m})$ závisí jak na stavu robota, tak na umístění landmarků. Z toho vyplývá, že pravděpodobnostní hustota nelze takto rozdělit

$$p(\mathbf{x}_k, \mathbf{m} \,|\, \mathbf{z}_k) \;\neq\; p(\mathbf{x}_k \,|\, \mathbf{z}_k) p(\mathbf{m} \,|\, \mathbf{z}_k).$$

Takové rozdělení by vedlo k nekonzistentnosti odhadů a nepoužitelným výsledkům.

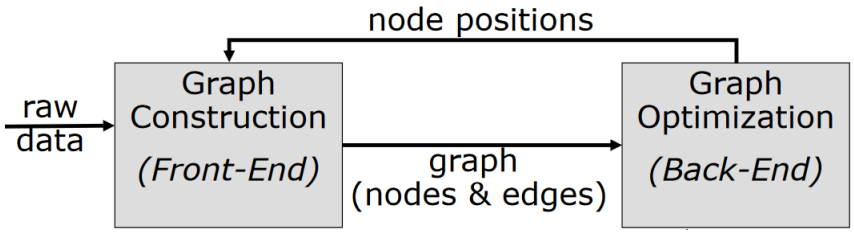
4 Graph-Based SLAM

Při tomto přístupu se k ukládání dat používá graf. Uzel grafu reprezentuje stav robota a měření provedená z tohoto stavu. Hrany mezi uzly představují jejich vzájemnou polohu. Cílem algoritmu je najít takové rozložení uzlů, které minimalizuje čtvercovou chybu rozdílu pozorování a "virtuálního pozorování"[1]. Tím se získá nejlepší odhad mapy. Optimalizace neprobíhá po každém novém pozorování, čímž se Graph SLAM řadí mezi offline metody řešení.

Formulace SLAM pomocí grafu byla poprvé představena v roce 1997, avšak kvůli vysoké výpočetní náročnosti trvalo ještě několik let, než se začala používat v reálných aplikacích. V dnešní době patří Graph SLAM k nejmodernějším technikám, co se rychlosti a přesnosti týče.

4.1 Definice

Pro správný chod graph SLAMu musí fungovat souhra mezi jeho dvěma částmi front-end a back-end. Front-end se obecně stará o sestrojení samotného grafu a back-end o jeho optimalizaci a tedy i optimalizaci mapy.



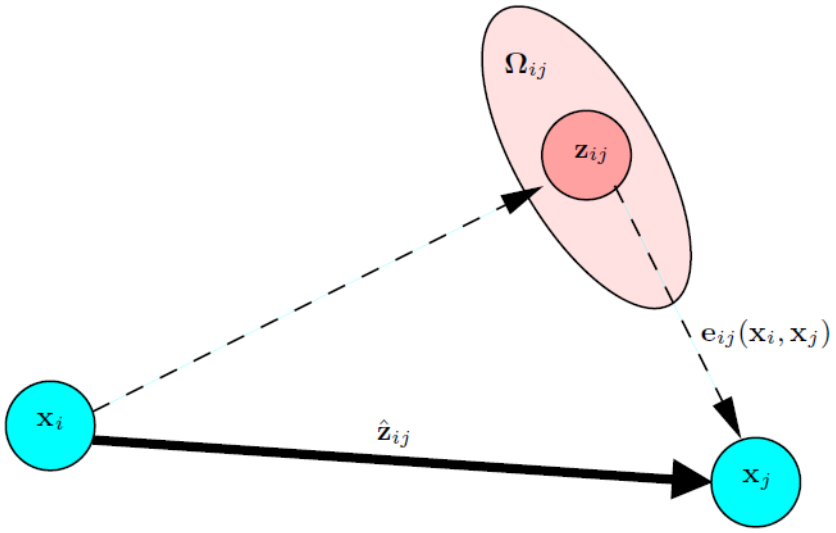
Obrázek 3: Schéma převzaté z [2] znázorňuje průchod dat metodou Graph SLAM. Surová data (pozorování, odometrie) nejdříve zpracuje front-end a vytvoří graf. Ten po daném okamžiku putuje do back-end, kde se optimalizací získá nejlepší odhad mapy prostředí a poté předá opět do první části.

4.1.1 Front-end

První část vytváří graf, kterým reprezentuje mapu prostředí. Každý uzel x_i obsahuje informaci o stavu robota a o pozorování provedeném z tohoto stavu v čase i. Hrana mezi sousedními uzly x_i a x_{i+1} odpovídá zaznamenané odometrii při pohybu robota. Pokud je hrana mezi různými uzly x_i a x_j, pak je to tzv. virtuální měření. To se zaznamená ve chvíli, kdy robot naměří podobná data (tzn. ocitne se ve stejné části prostředí) jako v některém z předešlých měření.

Využívá se scan matching, který porovnává dvě pozorování z uzlů x_i a x_j z různých časových okamžiků a v případě zjištění podobnosti je vypočítána relativní poloha uzlů x_i a x_j. To je ve skutečnosti transformace pozorování z x_i taková, aby se maximálně překrývalo s pozorováním uzlu x_j. Tuto transformaci nazýváme virtuálním pozorováním.

Znázornění problému je v obrázku 4, kde x_i je aktuální uzel, x_j je uzel, ve kterém bylo provedeno podobné pozorování jako to aktuální, $\hat{z}_{ij}$ je predikce virtuálního měření reprezentující "jak uzel i vidí uzel j", z_{ij} je virtuální měření, Ω_{ij} je informační matice virtuálního měření a $e_{ij}(x_i, x_j)$ je funkce, která vyjadřuje rozdíl virtuálního a predikovaného měření.



Obrázek 4: Znázornění virtuálního měření a jeho predikce [5].

4.1.2 Back-end

Druhá část se stará o optimalizaci rozložení uzlů v grafu.

Definujme funkci

$$\mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j) \;=\; \mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j),$$

která vyjadřuje chybu mezi predikcí měření $\hat{\mathbf{z}}_{ij}$ a měřením $\mathbf{z}_{ij}$ a je také znázorněna v obrázku 4. Optimalizace rozložení všech uzlů dosáhneme minimalizací funkce $\mathbf{F}(\mathbf{x})$ pro všechna virtuální měření

$$\mathbf{F}(\mathbf{x}) \;=\; \sum_{i,j} \mathbf{F}_{ij} \;=\; \sum_{i,j} \mathbf{e}_{ij}^T \Omega_{ij} \mathbf{e}_{ij}.$$

Nejlepší odhad mapy tedy musí splňovat podmínku

$$\mathbf{x}^* \;=\; \operatorname{argmin} \mathbf{F}(\mathbf{x}).$$

Numericky lze tento problém řešit metodou nejmenších čtverců, konkrétně na odhad rozmístění uzlů z front-end aplikovat Gauss-Newtonův algoritmus, který minimalizuje chybovou funkci $\mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j)$.

5 Vizuální SLAM

Vizuální SLAM, Visual-based SLAM nebo také vSLAM je přístup k simultánní lokalizaci a mapování, při kterém jsou jako vstup algoritmu využívány obrazová data. Tato metoda se začala používat až po roce 2000, neboť zpracování obrazu z kamery je značně výpočetně náročné.

Na druhou stranu sebou přináší dodatečné informace o vzhledu, barvě, jasu a textuře prostředí. To umožňuje začlenění dalších úkolů jako například detekce a rozpoznávání

obličeje, věcí nebo konkrétních míst. Pro vSLAM také nahrává cenová dostupnost kamer, jejich velikost a nižší spotřeba energie oproti často používaným laserovým skenerům.

Bohužel při použití snímačů obrazových dat mohou vzniknout chyby pokud kamera nemá dostatečné rozlišení, v prostředí je málo nebo moc landmarků, snímané povrchy nemají nedostatečnou texturu, málo osvětlení a jeho změny nebo pořízení rozmazaných snímků při pohybu. Navíc nezávisle na použitém algoritmu musí být provedena kalibrace kamer.

5.1 Základní struktura vSLAM

Většina vSLAM algoritmů používá stejnou obecnou strukturu, která se skládá ze tří hlavních částí:

- Kalibrace
- Sledování
- Mapování

Nezávisle na přesném typu algoritmu musí proběhnout kalibrace kamer. Ta se dá rozdělit na vnitřní, která odpovídá geometrii kamery (ohnisková vzdálenost) a vnější, která záleží na pozici kamery v prostoru (rotace a translace s respektováním nějakého souřadnicového systému). Provádí se s použitím několika různých obrazů s motivem šachovnice. Jejich nasnímáním se vnitřní souřadnicový systém nastaví tak, aby odpovídal reálnému světu.

V další části je sledovaný obraz zasazen do rekonstruované mapy k odhadu pozice robota. K tomu je potřeba nejdříve najít společné významné body obrazu s mapou a poté vypočítat odhad pozice kamery.

Mapováním se rozumí rozšiřování existující mapy v případě, kdy kamera zachytí dosud neznámou oblast.

5.2 Dodatkové moduly vSLAM

Následující dvě části jsou často zahrnuty do algoritmu pro stabilnější a přesnější běh, avšak v závislosti na konkrétním použití je lze vynechat:

- Relokalizace
- Globální optimalizace mapy

V případě, kdy robot ztratí přehled o své pozici, například kvůli rychlému pohybu kamery, je zapotřebí jej znovu lokalizovat. Pokud by vSLAM systém neobsahoval tuto funkci, nemohl by dále pokračovat po ztrátě informace o své pozici.

Mapa zpravidla obsahuje kumulativní chybu odhadu, která se zvětšuje s ujetou vzdáleností. Když robot navštíví již dříve objevenou oblast, použitím techniky zvané uzavírání smyček, dojde k porovnání aktuálního obrazu s dříve nasnímanými a v případě shody může být dopočítána a následně odstraněna chyba odhadu.

5.3 Metody řešení

Existují tři základní přístupy v získávání a/nebo zpracování obrazových dat. První dva rozlišujeme podle práce s obrazovými daty, kdy jeden přístup je založen na detekci a zpracování významných bodů, zatímco druhý pracuje přímo se získaným obrazem. Třetí možností je použití RGB-D kamer jako je například Microsoft Kinect, který je výhodný v možnosti získávání nejen obrazu prostředí, ale i informaci o jeho hloubce.

5.3.1 S použitím významných bodů

První monokulární SLAM nazvaný MonoSLAM byl vyvinutý v roce 2003. V tomto algoritmu probíhá lokalizace i mapování souběžně a k odhadu se používá Kalmanův filtr. Pohyb kamery a pozice významných bodů prostředí jsou reprezentovány stavovým vektorem. Při objevení nového významného bodu je tento přidán do stavového vektoru, v důsledku čehož se při použití ve velkém prostředí dimenze vektoru zvětší natolik, že výpočetní náročnost souběžné lokalizace a mapování přestává být únosná a je těžké dosáhnout potřebných odhadů v reálném čase.

Řešením problému výpočetní náročnosti je lokalizaci a mapování spustit každé na svém vlastním procesorovém vlákně tak, jako v metodě zvané PTAM (tj. Parallel Tracking and Mapping). Tyto vlákna běží paralelně, tím nedochází k omezování lokalizace výpočetní náročností odhadu mapy. Navíc je do mapování zahrnuta i průběžná optimalizace, takže je robot v reálném čase lokalizován a navíc vytvářena přesná mapa prostředí. PTAM je první metodou, která zahrnuje optimalizaci do algoritmu v reálném čase a většina novějších algoritmů následuje tento vícevláknový přístup [6].

PTAM, narozdíl od MonoSLAM, dokáže v réálném čase fungovat i v případě rozlehlého prostředí, kde stavový vektor obsahuje několik tisíc významných bodů.

5.3.2 RGB-D

RGB-D SLAM se začal nejvíce používat po příchodu kamery Microsoft Kinect, která započala malou „revoluci“ díky své ceně, velikosti a jednoduchosti použití. Má zabudovanou RGB i infrakameru a dokáže tak kromě obrazu navíc získávat informaci o hloubce prostředí. Většina spotřebitelských RGB-D kamer je však určena pouze pro použití v interiéru, neboť v exteriéru je obtížné zachytit odražené IR záření a kromě toho mají malý dosah v řádu jednotek metrů.

Pomocí RGB-D kamer lze získat přímo 3D strukturu prostředí, většina přístupů však mapu rekonstruuje z kombinací více hloubkových map. K odhadu pozice kamery je hojně využívaný iterativní algoritmus nejbližšího bodu (ICP).

5.3.3 Přímé metody

Přímé metody zpracovávají vstupní obraz bez použití doplňků extrahujících určité části obrazu a používají tak jeho fotometrickou konzistenci. Metody s detekcí významných bodů využívají geometrické konzistence obrazu.

Jedním příkladem je metoda DTAM, kde při lokalizaci porovnáváme vstupní obraz s uměle generovaným obrazem z rekonstruované 3D mapy. Tento algoritmus bývá implementován na grafické kartě, čímž dosáhneme vyšší efektivity výpočtu a zároveň oddělení lokalizace a mapování. Mapování se provádí s využitím několika sérií pozorování z různých míst a následnou optimalizací. Tímto způsobem lze získat všechny 3 souřadnice každého pixelu a sestavit tak 3D mapu prostředí. DTAM je navržený pro rychlé a online 3D modelování pro použití s mobilním telefonem.

LSD-SLAM

Reference

[1] P. Abbeel. GraphSLAM. https://people.eecs.berkeley.edu/ pabbeel/cs287-fa13/slides/GraphSLAM.pdf, 11 2013. UC Berkeley EECS.

[2] W. Burgard. Graph-based SLAM. http://ais.informatik.uni-freiburg.de/teaching/ss18/robotics/slides/16-graph-slam.pdf, 07 2018. UNI Freiburg.

[3] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard. Simultaneous localization and mapping: Present, future, and the robust-perception age. *CoRR*, abs/1606.05830, 2016.

[4] H. Durrant-Whyte and T. Bailey. Simultaneous localisation and mapping (slam): Part i the essential algorithms. *IEEE ROBOTICS AND AUTOMATION MAGAZINE*, 2006.

[5] G. Grisetti, R. Kuemmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *Intelligent Transportation Systems Magazine, IEEE*, 2(4):31–43, 2010.

[6] S. I. Takafumi Taketomi, Hideaki Uchiyama. Visual slam algorithms: a survey from 2010 to 2016. *IPSJ Transactions on Computer Vision and Applications*, 2017.