Vol.2017-CG-168 No.21
Vol.2017-DCC-17 No.21
Vol.2017-CVIM-209 No.21
2017/11/8

追加SLAMデバイスを用いたロボットナビゲーション

石川　涼一[1,a)]　大石　岳史[1,b)]　池内　克史[2,c)]

概要：
本稿では, ロボットに取り付けられた外部 SLAM デバイスを用いたロボットナビゲーション技術を提案する. そのために必要となるロボット-デバイス間のキャリブレーションにおいて, 必要な動きや情報を明らかにしたうえで計算方法を示す.　加えてロボットの形状と外界の情報を用いたオンラインでのポジション修正手法を提案する. 本稿ではまずセンサとロボットの間のキャリブレーション方法を二つ示す. 一つは未知のパラメータを, デバイスが取り付けられたロボットフレームの二方向の回転遷移から解く方法である. もう一つはロボットフレームが roll 軸, pitch 軸周りに回転する機構を持たない場合のための, 水平方向のロボットの動きとセンサの位置を用いたキャリブレーション手法である. オンラインでのポジション修正手法では, ロボットとデバイスの相対位置姿勢をデバイス, ロボット, 外界情報の整合性が合うように調整する. 提案するオンライン修正では床の法線ベクトルを用いている.
我々の手法は様々な種類のロボットに簡単に用いることができることであり, オフラインのキャリブレーションとオンラインの修正で, デバイスの取り付けが簡易的でも十分な精度のローカリゼーションが可能である. 実験では二つのオフラインキャリブレーションによるパラメータの確認を行い, オンラインの修正手法の有効性を, ロボットの激しい動きの間のローカライズされた位置の誤差を見ることで示す. その後 SLAM device を用いたナビゲーション手法のデモンストレーションを示す.

1. Introduction

ロボットナビゲーションは, 例えば家庭や災害救助といった様々な環境下で多くのタスクを行うために非常に重要な技術である. ナビゲーションには外界環境の情報や, その環境下におけるロボットの位置や向きが必要となる. しかしながらレーザセンサやデプスセンサのような能動的な測定センサ, そして正確なローカリゼーションシステムなしでナビゲーションを行うことは難しい. 一方で, 単眼カメラ [2], [15] や RGB-D カメラ [1] を用いた SLAM (Simultaneous Localization and Mapping) 技術は近年発達してきており, Augmented Reality (AR) や Mixed Reality (MR) 向けの HMD のようなデバイスにも応用されてきている. 本稿ではこのような実時間の三次元センシングデバイスを”SLAM デバイス”と呼ぶことにする. 本研究では Fig. 1 に示すように追加の SLAM デバイスをローカリゼーションに用いたロボットナビゲーションを扱う.

ナビゲーションにおけるローカリゼーションの方法としては, あらかじめナビゲーションさせたい場所のフロアマップを用意しておいてそのマップに対してローカリゼーションする方法がある. そのために ICP アルゴリズム [13] や 2D の Monte Carlo Localization (MCL)[4], 3D の MCL[8], ビジョンベースのアプローチ [9], RGB-D カメラとパーティクルフィルタ [21] などを用いたものが提案されている. 位置姿勢推定のために RFID（Radio-frequency identifier）[18] や二次元のバーコード [6] のようなランドマークを用いた方法も同様に提案されている. また, LiDAR, あるいはレーザレンジファインダー [11], [14] や RBG-D camera[16], [20] といった様々なセンサに依存した SLAM ベースの方法も用いられている.

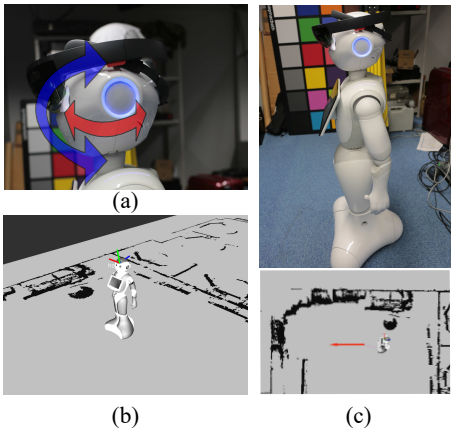

図 1　(a) Attaching SLAM device on robot and calibrating relative pose by moving robot head, (b)After calibration, SLAM device is located in front of robot head (c) Running navigation

1　東京大学　生産技術研究所
2　Microsoft Research Asia
a)　ishikawa@cvl.iis.u-tokyo.ac.jp
b)　oishi@cvl.iis.u-tokyo.ac.jp
c)　katsuike@microsoft.com

Vol.2017-CG-168 No.21
Vol.2017-DCC-17 No.21
Vol.2017-CVIM-209 No.21
2017/11/8

外部センサをロボットに取り付けるためにはロボット-センサキャリブレーションが必要となってくる．カメラとロボットのアームや IMU とのキャリブレーションは hand-eye calibration としてよく知られている．[17], [19] ではロボットのパーツとカメラ間の回転と併進移動を含んだ未知の 4×4 行列のパラメータを $\mathbf{X}$, ロボットを動かしたときの位置姿勢遷移を表す既知の 4×4 行列のパラメータを $\mathbf{A}$ と $\mathbf{B}$ としたときにできる, $\mathbf{AX} = \mathbf{XB}$ という式の数学的な解法が示されている．Fassi らは同様に $\mathbf{AX} = \mathbf{XB}$ の解法を幾何的な視点から議論している [3]．カルマンフィルタを用いたカメラと IMU のキャリブレーション手法も [7], [10] にて提案されている．

本稿では SLAM デバイスをロボットのナビゲーションシステムに適用する手法を提案する．通常の Hand-eye calibration に比べると, 地上を走行するロボットを扱うため, 移動自由度の制限が大きい．様々な種類のロボットに対してキャリブレーションを行えるようにするため, ロボット-デバイスキャリブレーションにおいて必要な動きや情報を明らかにし, その解法を示す．本稿では二つのキャリブレーション手法を扱う．一つは二方向の回転による位置姿勢遷移を用いており, もう一つは水平方向のみの位置姿勢遷移とデバイスの床からの高さを用いている．後者はロボットのフレームに取り付けられたデバイスを roll や pitch 軸回りに回転できないロボットのためにデザインされている．

このオフラインキャリブレーションに加えて, 外界環境とロボットの形状を用いて, デバイス-ロボット-外界環境情報の整合性が合うようにロボットとデバイスの相対位置姿勢を調整することによる, オンラインの位置修正手法も提案する．オフラインキャリブレーションによって得られたパラメータは誤差を含んでいたり, ナビゲーション中にずれてしまう可能性がある．さらにはロボットの関節のエンコーダも誤差を含んだり, 激しく関節を動かしたときにタイムラグが生じる可能性もある．このキャリブレーションやエンコーダの誤差は大きなローカリゼーションの誤差を含む可能性があるため, ロボットは垂直に立っており, ロボットのベースと世界座標系で垂直方向のベクトルは一致するという前提のもと, これらのベクトルが整合性を持つようにキャリブレーションパラメータの回転成分をオンラインで調整する．

本手法は様々な種類のロボットに簡単に適用することができる．我々の SLAM デバイス-ロボット間のキャリブレーション手法はロボットが二方向以上に回転できない場合でも用いることができる．オンラインのキャリブレーションパラメータ調整により誤差を軽減することで, 屋内ナビゲーションに十分な精度も得られている．

1.1 Notation

以後の記法として, 座標系 A から B への変換を表す回転行列と併進移動ベクトルを $\mathbf{R}_A^B, \mathbf{t}_A^B$ とあらわし, 座標系 B における点 $\mathbf{p}_B$ は下記の式のよって座標系 A に変換される．

$$\mathbf{p}_A = \mathbf{R}_A^B \mathbf{p}_B + \mathbf{t}_A^B. \quad (1)$$

我々のシステムではいくつかの座標系を用いている．世界座標系を”map”, SLAM デバイス (モーショントラッキングセンサ) の座標系を”ts”, SLAM デバイスが取り付けられたロボットのフレームの座標系を”head”, そして床との接地面を”foot”とする．便宜上 SLAM デバイスが取り付けられたロボットフレームの座標系を”head”と呼んではいるが, ローバのように head の機構がなくても本手法は用いることができる

ロボットの foot 位置を世界座標系内にローカライズするために, ”map” - ”ts” - ”head” - ”foot”という座標系変換をたどる．このうち $\mathbf{R}_{map}^{ts}, \mathbf{t}_{map}^{ts}$ については SLAM デバイスから与えられ, $\mathbf{R}_{head}^{foot}, \mathbf{t}_{head}^{foot}$ についてはロボットから与えられる．したがって未知のパラメータは $\mathbf{R}_{ts}^{head}, \mathbf{t}_{ts}^{head}$ となる．以降の章では, どのようにしてこの $\mathbf{R}_{ts}^{head}, \mathbf{t}_{ts}^{head}$ をオフラインのキャリブレーションとオンライン修正によって求めるかを説明する．

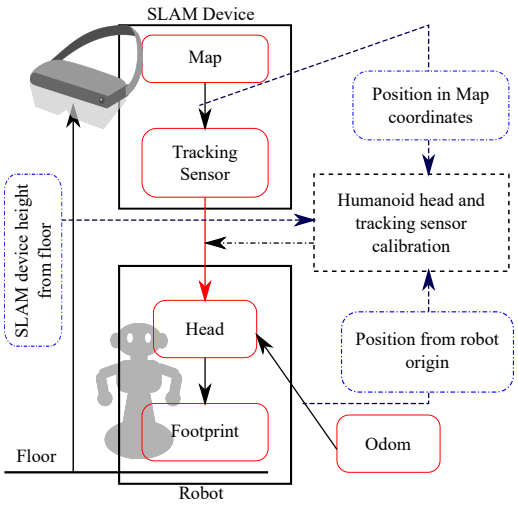

図 2 Overview of the off-line calibration method.

2. Off-line robot-sensor calibration

本章ではオフラインキャリブレーション手法について説明する．通常の Hand-eye calibration とは違い, 用いるロボットによってはロボットの動作に制限がかかる可能性がある．そこで Fig. 3 に示すように二方向に回転することのできるロボットに適用することができる回転遷移を用いたキャリブレーション手法と, 水平方向（x 方向, y 方向, yaw 軸回りの回転）にしか動くことのできないロボットに向けたキャリブレーション手法を示す．後者は最低限前進移動と回転による方向転換の遷移ができればよく, 位置姿勢遷移のほか, 入力として 3D 環境地図から得られる SLAM デバイスの床からの高さを用いている．Figure. 2 にオフラ

Vol.2017-CG-168 No.21
Vol.2017-DCC-17 No.21
Vol.2017-CVIM-209 No.21
2017/11/8

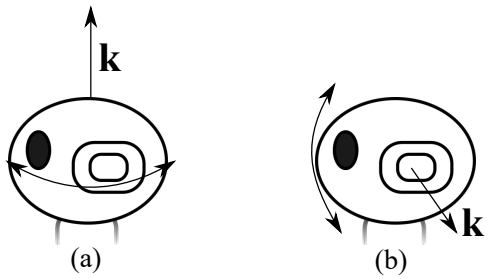

図 3 (a) Rotation around *yaw* axis (e.g. twisting neck) fixes *roll* and *pitch* (b) Rotation around *pitch* axis fixes *roll* and *yaw*.

インキャリブレーションのオーバービューを示す.

SLAM デバイスが環境地図を持ち, その座標系における三次元位置をトラッキングしている, また, ロボットも IMU やモータエンコーダなどを持ち, ロボット内部における原点"odom"からの"head"の位置姿勢パラメータを与えているものとする. 本キャリブレーションにおける問題は $\mathbf{R}_{ts}^{head}, \mathbf{t}_{ts}^{head}$ を計算することであり, これを解くために $\mathbf{R}_{odom}^{head}, \mathbf{t}_{odom}^{head}$ と $\mathbf{R}_{map}^{ts}, \mathbf{t}_{map}^{ts}$ を入力としその位置姿勢遷移を計算する. さらに SLAM デバイスの位置姿勢情報も, ロボットの動作に制限がありロボットの位置姿勢遷移のみではすべてのパラメータに拘束がかからない場合に用いている.

2.1 Obtaining pose transition

まず位置姿勢遷移のパラメータの計算方法について説明する. ロボットとデバイス間の相対位置姿勢を得るために, 位置姿勢遷移を行い, 繊維前と遷移後の二つの位置姿勢パラメータの差分をとる. ロボットのヘッドとセンサの初めの姿勢を pose 1 とし, その時の位置姿勢を表す 4×4 行列を $\mathbf{M_1}_{map}^{ts}$ と $\mathbf{M_1}_{odom}^{head}$ と定義する. そして遷移後の姿勢を pose 2 とし 4×4 位置姿勢行列を $\mathbf{M_2}_{map}^{ts}$ と $\mathbf{M_2}_{odom}^{head}$ とする. 遷移前, 遷移後の二つの位置姿勢の差分は下記の式で与えられる

$$\mathbf{A} = \mathbf{M_1}_{odom}^{head\ -1}\mathbf{M_2}_{odom}^{head} \tag{2}$$

$$\mathbf{B} = \mathbf{M_1}_{map}^{ts\ -1}\mathbf{M_2}_{map}^{ts}. \tag{3}$$

$\mathbf{X}$ を下記の式で与えられるロボットとセンサの相対位置姿勢を表す未知の 4×4 行列とする.

$$\mathbf{X} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{4}$$

ただし,

$$\mathbf{R} = \mathbf{R}_{ts}^{head-1} \tag{5}$$

$$\mathbf{t} = -\mathbf{R}_{ts}^{head-1}\mathbf{t}_{ts}^{head}. \tag{6}$$

ここで pose 1 における"head"の座標フレームから pose 2 における"ts"の座標フレームへの変換を考えたとき, pose 2 の"head"を経由する変換と pose 1 の"ts"を経由する変換が存在する. 前者を行列計算で表すと $\mathbf{AX}$ となり, 後者は $\mathbf{XB}$ と表される. どちらの座標変換でも結果は変わらないため $\mathbf{AX} = \mathbf{XB}$ が成り立つ.

2.2 Solving AX=XB with Two Directional Rotation

本節ではロボット-センサキャリブレーションに必要な姿勢遷移について論じる. $\mathbf{AX} = \mathbf{XB}$ の解法については [3], [17], [19] などで論じられており, 初期値の計算ののち最小二乗法の最適化を用いることで $\mathbf{X}$ を計算することができる. ここでは SLAM デバイスとロボットのキャリブレーションの問題を $\mathbf{AX} = \mathbf{XB}$ の問題に落とし込んだ時, この行列を精度よく推定できるような動作を考える

$\mathbf{R}_A, \mathbf{R}_B$ をそれぞれ $\mathbf{A}, \mathbf{B}$ における 3×3 回転行列成分とし, $\mathbf{t}_A, \mathbf{t}_B$ を三次元の併進移動ベクトル成分とする. また $\mathbf{k}_A, \mathbf{k}_B$ を $\mathbf{R}_A, \mathbf{R}_B$ における回転軸を示す単位ベクトルとする. $\mathbf{AX} = \mathbf{XB}$ から下記の二つの式が成り立つ [19].

$$\mathbf{k}_A = \mathbf{R}\mathbf{k}_B, \tag{7}$$

$$\mathbf{R}_A\mathbf{t} + \mathbf{t}_A = \mathbf{R}\mathbf{t}_B + \mathbf{t}. \tag{8}$$

まず Eq. 7 から得られる拘束について考える. 一組の $\mathbf{k}_A, \mathbf{k}_B$ が与えられたとき $\mathbf{R}$ に含まれる 3 自由度の回転パラメータのうち $\mathbf{k}_A$ ベクトル回り以外の 2 自由度の回転パラメータが決定される. 従って Eq. 7 から $\mathbf{R}$ を決定するには少なくとも二方向の回転が必要となる. $\mathbf{k}$ については Fig. 3 に示す通り, ロボットが水平方向に回転したとき垂直方向を向き, 垂直方向に回転したとき（例えばヒューマノイドがうなずくような動作）横方向を向く

続いて Eq. 8 から得られる拘束について考慮する. Eq. 8 を変形して

$$(\mathbf{I} - \mathbf{R}_A)\mathbf{t} = \mathbf{t}_A - \mathbf{R}\mathbf{t}_B. \tag{9}$$

Eq. 9 は, $(\mathbf{I} - \mathbf{R}_A)$ はランクが 2 であるため, 一つの式で 2 パラメータに対して拘束がかかる. 位置姿勢遷移の併進移動成分 $\mathbf{t}_A, \mathbf{t}_B$ は回転パラメータ $\mathbf{R}$ に対して拘束をかけ, 回転成分 $\mathbf{R}_A$ は $\mathbf{t}$ を求めるために必要となる. $\mathbf{t}$ を $\mathbf{k}_A$ とたがいに直行する単位ベクトル $\mathbf{t}_1, \mathbf{t}_2$ の成分に分解したときを考えると, $(\mathbf{I} - \mathbf{R}_A)\mathbf{k}_A = \mathbf{0}$ であるため,$\mathbf{k}_A$ 方向に自由度を持ち, $\mathbf{t}_1, \mathbf{t}_2$ 方向に拘束を持つ. 以上より, 回転方向の異なる二度の位置姿勢遷移があれば $\mathbf{R}$ も $\mathbf{t}$ も両方求めることができる.

ここまでで回転方向の異なる二度の位置姿勢遷移が十分条件であることを示した. ここで Eq. 8 において $\mathbf{t}_A, \mathbf{t}_B$, も回転行列 $\mathbf{R}$ に拘束をかける要素になりうるため, この位置姿勢遷移に併進移動成分が含まれる場合について考察する. $\mathbf{t}$ は, Eq. 9 より, 回転遷移を行うことによってできるロボットの"head"の併進移動成分 $\mathbf{t}_A$ とセンサの併進移動

Vol.2017-CG-168 No.21
Vol.2017-DCC-17 No.21
Vol.2017-CVIM-209 No.21
2017/11/8

成分 $\mathbf{Rt}_B$ の差をもとに計算される. さて, 位置姿勢遷移においてロボットを前方に推進させるなどの併進移動を行った場合, 特に IMU や encoder の不正確さが理由でこの $\mathbf{t}_A$ と $\mathbf{Rt}_B$ の間に蓄積誤差がたまることが考えられる. この蓄積誤差は $\mathbf{t}_A$ と $\mathbf{Rt}_B$ の差分をもとに計算する $\mathbf{t}$ の誤差に直結してしまう. そのため, この蓄積誤差が出ないよう, なるべく併進移動を短くした回転のみの遷移を用いたほうが精度の良いキャリブレーションを期待できる.

2.3 Solving AX=XB with SLAM device height

ローバなどのある種のロボットについては, 特に *roll* 軸または *pitch* 軸回りに回転させることが難しいために, センサが取り付けられたフレームを二方向に動かすことが難しい場合がある. そのようなロボットに向けて, 水平方向のみの動作とトラッキングセンサから得られる高さ情報からキャリブレーションを行う手法を提案する

まず Eq. 7 と Eq. 8 からロボットの水平方向の遷移 (x, y, yaw) のみで求めることができるパラメータについて検討する. まずロボットが *yaw* 軸回りに回転した場合 Eq. 7 の $\mathbf{k}_A$, $\mathbf{k}_B$ は Fig. 3 (a) に示す通り垂直方向に向く. また Fig. 3 (b) に示すような垂直方向の回転は得ることができない. 従って *roll* 軸と *pitch* 軸回りの 2 自由度の成分については求めることができるが *yaw* 軸回りについては求めることができない. 同様の理由で Eq. 8 については, $\mathbf{t}$ に含まれる 3 自由度の成分のうち, 回転軸となる垂直方向の成分については求めることができない.

ロボットが水平 (x, y) 方向への併進移動を行った場合, Eq. 8 から $\mathbf{t}_A, \mathbf{t}_B$ が水平方向を向く. そのため $\mathbf{Rt}_B$ 回りの回転方向, すなわち併進移動した方向を軸としたもの以外の回転成分に対して拘束がかかる. ここで *yaw* 軸回りの回転遷移では拘束がかからなかったキャリブレーションパラメータ *yaw* に対しても拘束がかかるようになる.

キャリブレーションパラメータのうち, 残る z 方向, 垂直方向の成分に対しては, 水平方向の位置姿勢遷移だけでは拘束をかけることができない. そこでトラッキングセンサの床からの高さを z パラメータを計算するために用いる. 床の法線ベクトルを $\mathbf{n}$ としトラッキングセンサの床からの高さを h とする. $\mathbf{n}$, h は SLAM デバイスが持つ環境地図から求めることができる. $\mathbf{o}$ を床と平行な方向のベクトルとしたとき下記の式が成り立つ

$$h\mathbf{n}_{ts} + \mathbf{t}^{ts}_{head} + \mathbf{R}^{ts}_{head}\mathbf{t}^{head}_{foot} = \mathbf{o} \tag{10}$$

Eq. 10 から下記の式が成り立つ.

$$\mathbf{n}_{ts} \cdot \mathbf{o} = \mathbf{n}_{ts} \cdot (h\mathbf{n}_{ts} + \mathbf{t}^{ts}_{head} + \mathbf{R}^{ts}_{head}\mathbf{t}^{head}_{foot}) = 0 \tag{11}$$

この Eq. 11 は, 床の法線ベクトル方向, すなわち求めたい 6 自由度のパラメータのうち, 残る垂直方向の併進移動パラメータ z に対して拘束をかけることができる.

さて, ここでキャリブレーションに適した位置姿勢遷移について考察する. まず最低限必要な位置姿勢遷移の回数であるが Eq. 9 で x, y, yaw の 3 パラメータに拘束をかけているが, $(\mathbf{I} - \mathbf{R}_A)$ はランクが 2 であるため少なくとも一つの式で 2 パラメータに対してしか拘束がかからない. 従って少なくとも 2 度の位置姿勢遷移が必要となる. 回転パラメータ $\mathbf{R}$ についてはキャリブレーションパラメータ yaw に対して拘束をかけるために併進移動の遷移をさせることが必要となる. 一方で Sec. 2.2 で述べた通り, $\mathbf{t}$ は $\mathbf{t}_A$ と $\mathbf{Rt}_B$ の差分によって計算され, その結果は $\mathbf{t}_A$ と $\mathbf{Rt}_B$ の蓄積誤差の影響を受けやすい. 従って x, y, yaw に対して Eq. 8 を用いて同時に拘束をかけると, 併進移動の距離によって x, y と yaw の精度にトレードオフが出るといえる. そこで位置姿勢遷移については x, y に拘束をかけるための回転のみの遷移と, yaw に拘束をかけるための併進移動のみの遷移を別々に分けたほうが良いと考えられる.

ここまででキャリブレーションパラメータの持つ 6 自由度に対して拘束をかけられることを示したので, 実際の計算手法について述べる. 位置姿勢遷移に少なくとも回転のみの遷移と併進移動のみの遷移がひとつづつ含まれているものとして, まず初期値の線形な計算方法について下記のステップを示す.

(1) $\mathbf{k}_A = \mathbf{R}_1\mathbf{k}_B$ を満たす回転行列 $\mathbf{R}_1$ を, $\mathbf{k}_A \times \mathbf{k}_B$ を回転軸とするように求める.

(2) $\mathbf{t}_1$ を下記の式より求める.

$$\mathbf{t}_1 = n_{ts} \cdot (h\mathbf{n}_{ts} + {\mathbf{R}_1}^{-1}\mathbf{t}^{head}_{foot})\mathbf{n}_{ts}. \tag{12}$$

(3) Eq. 8 において位置姿勢遷移が回転を含まない場合, 下記の式が成り立つ.

$$\mathbf{t}_A = \mathbf{R}^{ts}_{head}\mathbf{t}_B. \tag{13}$$

そこで回転を含まない遷移を一つ選んで下記の式を満たすような $\mathbf{R}_2$ を step 1 と同様に計算する.

$$\frac{\mathbf{t}_A}{|\mathbf{t}_A|} = \mathbf{R}_2\mathbf{R}_1\frac{\mathbf{t}_B}{|\mathbf{t}_B|} \tag{14}$$

その後, $\mathbf{R}$ を下記の式により計算する,

$$\mathbf{R} = \mathbf{R}_2\mathbf{R}_1. \tag{15}$$

(4) $\mathbf{t}_1$, $\mathbf{t}_2$, $\mathbf{t}_3$ がたがいに直行する条件を満たす任意の $\mathbf{t}_2$, $\mathbf{t}_3$ を決める. 回転のみを含む遷移から, 下記の式が成り立つ.

$$(\mathbf{I} - \mathbf{R}_A)(a\mathbf{t}_2 + b\mathbf{t}_3) = \mathbf{t}_A - \mathbf{Rt}_b. \tag{16}$$

この式を連立方程式として解いて, a, b を計算し, その後 $\mathbf{t}$ を下記の式により求める.

$$\mathbf{t} = \mathbf{t}_1 + a\mathbf{t}_2 + b\mathbf{t}_3. \tag{17}$$

Vol.2017-CG-168 No.21
Vol.2017-DCC-17 No.21
Vol.2017-CVIM-209 No.21
2017/11/8

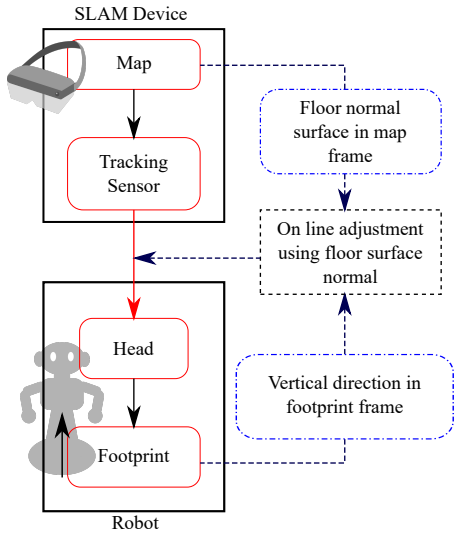

図 4 Overview of the on-line adjustment method.

次に非線形最適化について説明する. 非線形最適化のためのコスト関数については Eq. 7, Eq. 8, Eq. 11 をもとに各項を設定する. Eq. 7, Eq. 11 から下記の 2 項を設定できる,

$$f_1(\phi) = |\mathbf{k}_A \times \mathbf{R}\mathbf{k}_B| \tag{18}$$

$$f_2(\phi, \mathbf{t}) = |\mathbf{n}_{ts} \cdot (\mathbf{t}^{ts}_{head} + \mathbf{R}^{ts}_{head}\mathbf{n}_{foot} + h\mathbf{n}_{ts})|. \tag{19}$$

一方で Eq. 8 からは, 下記の項を設定できる,

$$f_3(\phi, \mathbf{t}) = \frac{|(\mathbf{R}_A\mathbf{t}^{ts}_{head}+\mathbf{t}_A)\times(\mathbf{R}^{ts}_{head}\mathbf{t}_B+\mathbf{t}^{ts}_{head})|}{|\mathbf{R}_A\mathbf{t}^{ts}_{head}+\mathbf{t}_A|\cdot|\mathbf{R}^{ts}_{head}\mathbf{t}_B+\mathbf{t}^{ts}_{head}|}. \tag{20}$$

この Eq. 20 は角度をメトリックとしている. これは前述したとおり $\mathbf{t}$ が蓄積誤差の影響を受けやすいためであり, 角度をメトリックとした場合, $\mathbf{t}_A$ と $\mathbf{R}\mathbf{t}_B$ 間のスケール方向の誤差に対しては距離をメトリックとした場合に比べて頑健であるからである. 6 自由度のキャリブレーションパラメータは f_1, f_2, f_3 の 3 項で構成されたコスト関数を最小化することで最適化される

3. On-line position adjustment

本章ではロボットが地面に垂直に立っているという前提にのっとったオンラインの位置修正手法について説明する. SLAM デバイスとロボット間の相対位置姿勢はオフラインのキャリブレーションによって求められているが, ローカライズされたベースの位置を不正確にするいくつかの要因が存在する. まず外部 SLAM デバイスを用いることでローカライズは簡単になるが, ロボット-センサキャリブレーションそのものに誤差が含まれることが考えられる. さらにナビゲーションを行っている最中にセンサがずれてしまう可能性もある. またロボットの IMU やエンコーダの精度が誤差の原因になったり, あるいは関節を動かしたときのエンコーダのタイムラグも問題になることも考えられる. このような誤差のうち, Fig. 5 に示すように特に *roll* 軸,*pitch* 軸回転の誤差はデバイスが高いところに取り付けられていた場合に大きなローカライズ誤差につながってしまう.

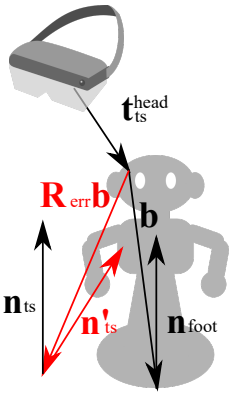

図 5 Diagram when calibration error is caused. Vector $\mathbf{b}$ is rotated including error and rotation error around *roll* or *pitch* axis caused large position error. Transformed $\mathbf{n}_{foot}$ in tracking sensor frame and $\mathbf{n}_{ts}$ should be in the same direction.

オンラインの修正手法では, 外部環境の情報とロボット形状, デバイスの位置が整合性を持つようにセンサ-ロボット間のパラメータを修正する. Figure. 4 にオーバービューを示す. 具体的な計算では外部環境の情報としてロボットが立っている床の情報を使い, ロボットが床に垂直に立っているという前提にのっとってロボットのベースにおける鉛直方向のベクトルと, 床の法線ベクトルの方向が一致するようにする. それによって 2 自由度に拘束をかける部分的なループクロージャを作り, それが成り立つようにロボット-センサ間に誤差を修正する回転行列を挿入する.

まず回転行列を挿入したときどのように誤差が軽減されるかについて説明する. $\mathbf{b}_A$ を座標フレーム A において"head"から"foot"に向かうベクトルとする. トラッキングセンサからロボットの"foot"に向かうベクトル $\mathbf{t}^{foot}_{ts}$ は下記のように表現できる

$$\mathbf{t}^{foot}_{ts} = \mathbf{R}^{foot}_{ts}\mathbf{b}_{foot} + \mathbf{t}^{head}_{ts}. \tag{21}$$

R^{foot}_{ts} を変換行列の真値とし, $\mathbf{R}_{err}$ を R^{foot}_{ts} に対して積算される誤差の回転とする. この $\mathbf{R}_{err}$ は前述したキャリブレーションのずれやエンコーダの誤差に起因するものとする. 観察されたセンサに対するロボットの足の位置 $\mathbf{t}'^{foot}_{ts}$ は

$$\mathbf{t}'^{foot}_{ts} = \mathbf{R}_{err}\mathbf{R}^{foot}_{ts}\mathbf{b}_{foot} + \mathbf{t}^{head}_{ts}, \tag{22}$$

となり, 真値からの誤差は

$$\begin{aligned}\mathbf{e} &= \mathbf{t}'^{foot}_{ts} - \mathbf{t}^{foot}_{ts} \\ &= \mathbf{R}_{err}\mathbf{R}^{foot}_{ts}\mathbf{b}_{foot} - \mathbf{R}^{foot}_{ts}\mathbf{b}_{foot}.\end{aligned} \tag{23}$$

となる. ここで, 床の法線を $\mathbf{n}$ とすると以下の式が成り立つ,

$$\mathbf{n}_{ts} = \mathbf{R}^{foot}_{ts}\mathbf{n}_{foot}. \tag{24}$$

Vol.2017-CG-168 No.21
Vol.2017-DCC-17 No.21
Vol.2017-CVIM-209 No.21
2017/11/8

誤差を含む場合は

$$\mathbf{n'}_{ts} = \mathbf{R}_{err}\mathbf{R}_{ts}^{foot}\mathbf{n}_{foot}. \tag{25}$$

となる.

次に, 下記の式に示す通り $\mathbf{n'}_{ts}$ を $\mathbf{n}_{ts}$ と同じ方向に向くような追加の回転行列 $\mathbf{R}_{add}$ を定義する,

$$\mathbf{n}_{ts} = \mathbf{R}_{add}\mathbf{n'}_{ts}. \tag{26}$$

この $\mathbf{R}_{add}$ が最終的に誤差を補正するために追加される回転行列となる.

補正行列を追加した場合の観察値 $\mathbf{t''}_{ts}^{foot}$ は

$$\mathbf{t''}_{ts}^{foot} = \mathbf{R}_{add}\mathbf{R}_{err}\mathbf{R}_{ts}^{foot}\mathbf{t}_{body} + \mathbf{t}_{ts}^{head}, \tag{27}$$

となり真値からのずれは

$$\begin{aligned}\mathbf{e'} &= \mathbf{t''}_{ts}^{foot} - \mathbf{t}_{ts}^{foot} \\ &= \mathbf{R}_{add}\mathbf{R}_{err}\mathbf{R}_{ts}^{foot}\mathbf{b}_{foot} - \mathbf{R}_{ts}^{foot}\mathbf{b}_{foot},\end{aligned} \tag{28}$$

となる.

ここで, $\mathbf{b}_{foot}$ を $\mathbf{n}_{foot}$ ベクトル方向と, $\mathbf{n}_{foot}$ に対して直行するベクトル $\mathbf{m}_{foot}$ に分解する;

$$\mathbf{b}_{foot} = \alpha\mathbf{n}_{foot} + \mathbf{m}_{foot}, \tag{29}$$

ただし

$$\alpha = \mathbf{b}_{foot} \cdot \mathbf{n}_{foot}. \tag{30}$$

Eq. 30 を Eq. 28 に代入して,

$$\mathbf{t''}_{ts}^{foot} = \mathbf{R}_{add}\mathbf{R}_{err}\mathbf{R}_{ts}^{foot}\mathbf{t}_{body} + \mathbf{t}_{ts}^{head}. \tag{31}$$

この時, 誤差は下記の式で与えられる,

$$\begin{aligned}\mathbf{e'} &= \mathbf{R}_{add}\mathbf{R}_{err}\mathbf{R}_{ts}^{foot}(\alpha\mathbf{n}_{foot} + \mathbf{m}_{foot}) \\ &\quad - \mathbf{R}_{ts}^{foot}(\alpha\mathbf{n}_{foot} + \mathbf{m}_{foot}), \qquad (32) \\ &= \alpha\mathbf{n}_{ts} + \mathbf{R}_{add}\mathbf{R}_{err}\mathbf{R}_{ts}^{foot}\mathbf{m}_{foot} \\ &\quad - \alpha\mathbf{n}_{ts} + \mathbf{R}_{ts}^{foot}\mathbf{m}_{foot} \qquad (33) \\ &= \mathbf{R}_{add}\mathbf{R}_{err}\mathbf{m}_{ts} - \mathbf{m}_{ts}. \qquad (34)\end{aligned}$$

この式は追加回転行列 $\mathbf{R}_{add}$ が $\mathbf{n}$ ベクトル回りの回転成分以外の 2 自由度の回転成分を打ち消していることを示している. この打ち消される 2 自由度の回転誤差成分はもともと α に比例した距離の誤差を生み出しており, この α は特にヒューマノイドのように背の高いロボットに取り付けた場合大きくなる. 従ってこの床の法線ベクトルを用いて回転を補正する手法はヒューマノイドなどの背の高いロボットに対して特に有効となる.

最後に $\mathbf{R}_{add}$ の求め方について述べる. $\mathbf{R}_{add}$ は一意には決定できないが, $\mathbf{n}_{ts} \times \mathbf{n'}_{ts}$ を回転軸とし, $\mathbf{n}_{ts}$ と $\mathbf{n'}_{ts}$ がなす角を回転角とした回転行列を計算することで求められる.

4. Experimental results

我々のナビゲーションシステムのデモにおいて, ロボットに SoftBank Pepper[*1], SLAM デバイスに Microsoft HoloLens[*2]を用いた. Robot Operating System (ROS)[*3]をホストシステムとして用いている. HoloLens は Fig. 1 (a) に示すように Pepper の head の部分に取り付けられている. HoloLens は外部環境の 3D map を作成し, その 3D map の任意の点に設定でき, 空間の基準点として用いることができる空間アンカーを持つ. フロア上の障害物などの情報を簡単にシステムに取り入れられるように, ナビゲーションはあらかじめ作られた 2D のマップを用いて行った.

実装について, HoloLens を ROS の世界座標系の中にローカライズするため, あらかじめ作られた 2D フロアマップに対して HoloLens から得られた三次元マップをトップからの直交視点からレンダリングした画像によって位置合わせを行う. この位置合わせの結果から空間アンカーを ROS のシステム座標系の中に設置し, HoloLens を地図上にローカライズすることができる. 一度空間アンカーを設置して HoloLens のローカライズを行っても, その空間アンカーから距離が離れてしまうと, ホロレンズ内の環境地図とフロアマップの誤差が原因で, 実世界の HoloLens の位置とローカライズされた HoloLens のずれが大きくなってしまう傾向がある. この誤差を避けるため, HoloLens が現在基準としている空間アンカーから離れていったら, 再度現在の HoloLens の位置を初期位置とした位置合わせとローカライズを行い蓄積誤差を解消している.

手法評価のため, 初めに二通りのオフラインキャリブレーションにおいて得られるパラメータの値を確認する. その後オンライン位置調整手法の有効性を, ロボットの関節を激しく動かしたときのローカライズ位置の誤差を確認することによって示す. 最後に SLAM デバイスを用いたロボットナビゲーションシステムをデモンストレーションする.

4.1 Off-line Calibration

初めに二方向回転を用いたキャリブレーションと水平方向の位置姿勢遷移+HoloLens の高さを用いたキャリブレーションで得られたパラメータの値を比較する. Table. 1 に各キャリブレーション方法を 5 回行った時のパラメータの平均値と標準偏差を示す. 二方向回転のキャリブレーションでは首の *pitch* 角と *yaw* 角をコントロールし, 一度のキャリブレーションで二度水平方向の回転遷移を行い二度垂直方向の回転遷移を行う. 水平方向の位置姿勢遷移を用いる場合は Pepper のホイールを操作し初めに二度回転の遷移を行いその後二度前進遷移を行った. その間

[*1] https://www.softbank.jp/en/robot/
[*2] https://www.microsoft.com/hololens
[*3] http://www.ros.org/

Vol.2017-CG-168 No.21
Vol.2017-DCC-17 No.21
Vol.2017-CVIM-209 No.21
2017/11/8

表 1 Calibration parameter value obtained by normal and horizontal calibration method

		x (m)	y (m)	z (m)	angle(rad)	axis x	axis y	axis z
normal calibration	mean	0.083285	0.031271	0.129084	1.66594	0.511814	-0.49433	-0.70262
	standard deviation	0.000531	0.000906	0.00169	0.000391	0.000877	0.000786	0.000232
horizontal calibration	mean	0.101651	0.031098	0.11713	1.642449	0.52515	-0.47133	-0.70838
	standard deviation	0.004845	0.002489	0.002671	0.010782	0.015054	0.00668	0.007034

HoloLens の高さと床の法線ベクトルを 5 箇所で記録してキャリブレーションに用いた.

二方向回転のキャリブレーションについては併進移動の標準偏差が $2mm$ 以下であり, 角度は $4.0 \times 10^{-4} rad$ 以下となっている. 水平方向の位置姿勢遷移を用いたキャリブレーションの標準偏差は二方向回転のキャリブレーションとでの値と比べると大きくなってしまってはいるがナビゲーションには十分な精度といえる.

4.2 On-line Adjustment

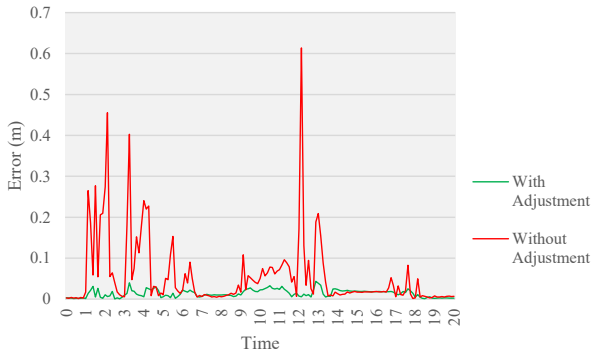

図 6 Localized position error during motion shaking Pepper head wearing HoloLens drastically. During the movement, adjusted position has less than $5cm$ error (Green line) whereas there are more than $60cm$ error in the result of before adjusted position (Red line).

次にオンライン位置修正手法の有効性を示す. キャリブレーションでは二方向回転の方法を用いて先ほどと同様二度ずつ水平方向と垂直方向に回転をさせた. キャリブレーションののちロボットの関節を激しく動かしたときの, 位置修正なしとあり, 両方のローカライズされた位置を二次元上で記録する. ロボットの動作中, 関節は動くがロボットの足の位置は変化していないため, プロットされる位置の真値は二次元上で変化しない. しかしながらロボットの関節を激しく動かすことでタイムラグによるエンコーダの誤差が生じ, それを原因として, HoloLens を介してローカライズされた観測値には誤差が生じると予想される.

Figure. 6 にロボットの関節の動作中, ローカライズされたペッパーの foot の初期位置からの誤差をプロットした結果を示す. 動作中, 修正なしでは最大で 60cm 以上の誤差が記録されているのに対し, 修正された位置のプロットは 5cm 以下で収まっている. これより提案された位置修正手法はキャリブレーションの回転誤差やロボットのエンコーダの回転誤差を吸収できていることを示した.

4.3 Navigation

Pepper と HoloLens を用いたナビゲーションシステムをデモンストレーションするためのいくつかの手順を示す. まず HoloLens を Pepper の頭に取り付ける, 次にホストシステムと Pepper 及び HoloLens との通信を確立する, その後ナビゲーションとローカリゼーションのプログラムを走らせる. そののち Pepper の head と HoloLens を姿勢遷移を用いてキャリブレーションを行う (Fig. 1 (a) and (b)). HoloLens の初期位置を二次元のフロアマップ上で GUI を通してマニュアルに指定することで位置合わせが行われ空間アンカーが設置される. 最後に二次元マップ上で目的地を指定することでナビゲーションが実行される.(Fig. 1 (c))

ナビゲーションにおいては ROS の NavFn[*4] パッケージにて実装されているダイクストラ法を用いて大域な経路計画を行い, コストマップ [12] を作成しローカルな経路計画には Dynamic Window Approach[5] を用いた. なおフロアマップは 3D のレーザスキャナを用いて測定し生成している. Figure. 7 に長い廊下を進んで部屋に入るナビゲーションの連続写真を示す.

5. Conclusion

本稿ではロボットナビゲーションのための外部 SLAM デバイスを用いたローカリゼーションとキャリブレーションの手法を提案した. デモンストレーションにはヒューマノイド型のロボットに SLAM デバイスを取り付けたものを用いているが, 本手法は IMU やエンコーダを持つ様々なロボットに対して簡単に適用することができる. 実験ではあらかじめ作成された 2D のフロアマップを用いているが, 事前地図なしのナビゲーションや三次元のナビゲーションにも拡張することができる.

Acknowledgment

本研究に協力いただいた Microsoft の the Strategic prototyping group, 特に Yutaka Suzue 氏に深く感謝する. 本研究の一部は JSPS Research Fellow No.16J09277 によった.

*4 http://wiki.ros.org/navfn

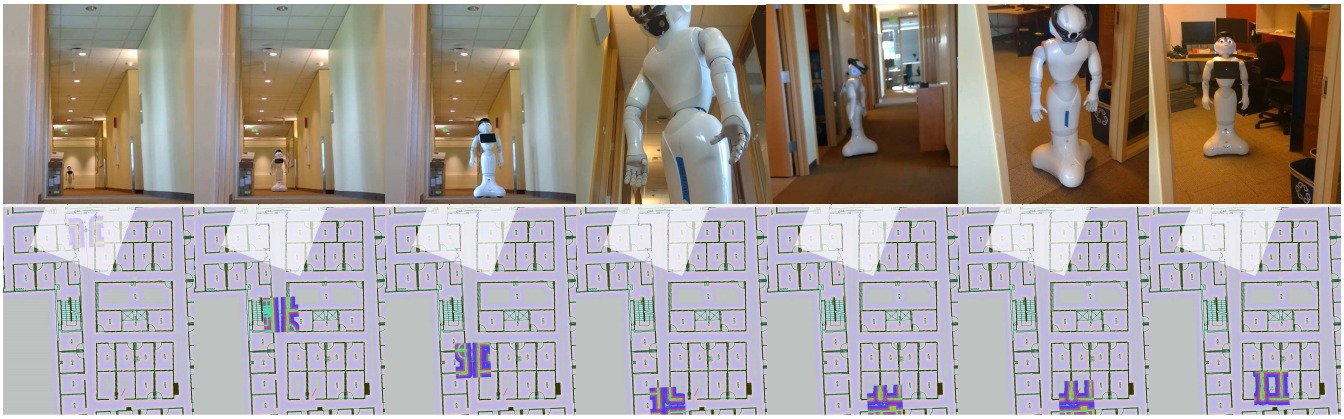

図 7 Navigation with our system using external SLAM device. Lower sequence is floor map in GUI. First, Pepper walks long corridor and then turn at the corner. Finally, Pepper enters a room and reaches to the destination.

参考文献

[1] Endres, F., Hess, J., Sturm, J., Cremers, D. and Burgard, W.: 3-D mapping with an RGB-D camera, *IEEE Transactions on Robotics*, Vol. 30, No. 1, pp. 177–187 (2014).

[2] Engel, J., Schöps, T. and Cremers, D.: LSD-SLAM: Large-scale direct monocular SLAM, *European Conference on Computer Vision*, Springer, pp. 834–849 (2014).

[3] Fassi, I. and Legnani, G.: Hand to sensor calibration: A geometrical interpretation of the matrix equation AX= XB, *Journal of Field Robotics*, Vol. 22, No. 9, pp. 497–506 (2005).

[4] Fox, D., Burgard, W., Dellaert, F. and Thrun, S.: Monte carlo localization: Efficient position estimation for mobile robots, *AAAI/IAAI*, Vol. 1999, No. 343-349, pp. 2–2 (1999).

[5] Fox, D., Burgard, W. and Thrun, S.: The dynamic window approach to collision avoidance, *IEEE Robotics & Automation Magazine*, Vol. 4, No. 1, pp. 23–33 (1997).

[6] George, L. and Mazel, A.: Humanoid robot indoor navigation based on 2D bar codes: Application to the NAO robot, *Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on*, IEEE, pp. 329–335 (2013).

[7] Hol, J. D., Schön, T. B. and Gustafsson, F.: Modeling and calibration of inertial and vision sensors, *The international journal of robotics research*, Vol. 29, No. 2-3, pp. 231–244 (2010).

[8] Hornung, A., Oßwald, S., Maier, D. and Bennewitz, M.: Monte Carlo localization for humanoid robot navigation in complex indoor environments, *International Journal of Humanoid Robotics*, Vol. 11, No. 02, p. 1441002 (2014).

[9] Ido, J., Shimizu, Y., Matsumoto, Y. and Ogasawara, T.: Indoor navigation for a humanoid robot using a view sequence, *The International Journal of Robotics Research*, Vol. 28, No. 2, pp. 315–325 (2009).

[10] Kelly, J. and Sukhatme, G. S.: Fast relative pose calibration for visual and inertial sensors, *Experimental Robotics*, Springer, pp. 515–524 (2009).

[11] Klančar, G., Teslić, L. and Škrjanc, I.: Mobile-robot pose estimation and environment mapping using an extended Kalman filter, *International Journal of Systems Science*, Vol. 45, No. 12, pp. 2603–2618 (2014).

[12] Lu, D. V., Hershberger, D. and Smart, W. D.: Layered costmaps for context-sensitive navigation, *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, IEEE, pp. 709–715 (2014).

[13] Lu, F. and Milios, E.: Robot pose estimation in unknown environments by matching 2d range scans, *Journal of Intelligent and Robotic systems*, Vol. 18, No. 3, pp. 249–275 (1997).

[14] Misono, Y., Goto, Y., Tarutoko, Y., Kobayashi, K. and Watanabe, K.: Development of laser rangefinder-based SLAM algorithm for mobile robot navigation, *SICE, 2007 Annual Conference*, IEEE, pp. 392–396 (2007).

[15] Mur-Artal, R., Montiel, J. M. M. and Tardos, J. D.: ORB-SLAM: a versatile and accurate monocular SLAM system, *IEEE Transactions on Robotics*, Vol. 31, No. 5, pp. 1147–1163 (2015).

[16] Oliver, A., Kang, S., Wünsche, B. C. and MacDonald, B.: Using the Kinect as a navigation sensor for mobile robotics, *Proceedings of the 27th conference on image and vision computing New Zealand*, ACM, pp. 509–514 (2012).

[17] Park, F. C. and Martin, B. J.: Robot sensor calibration: solving AX= XB on the Euclidean group, *IEEE Transactions on Robotics and Automation*, Vol. 10, No. 5, pp. 717–721 (1994).

[18] Park, S. and Hashimoto, S.: Autonomous mobile robot navigation using passive RFID in indoor environment, *IEEE Transactions on Industrial Electronics*, Vol. 56, No. 7, pp. 2366–2373 (2009).

[19] Shiu, Y. C. and Ahmad, S.: Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form AX= XB, *ieee Transactions on Robotics and Automation*, Vol. 5, No. 1, pp. 16–29 (1989).

[20] Wang, S., Li, Y., Sun, Y., Li, X., Sun, N., Zhang, X. and Yu, N.: A localization and navigation method with ORB-SLAM for indoor service mobile robots, *Real-time Computing and Robotics (RCAR), IEEE International Conference on*, IEEE, pp. 443–447 (2016).

[21] Winterhalter, W., Fleckenstein, F., Steder, B., Spinello, L. and Burgard, W.: Accurate indoor localization for RGB-D smartphones and tablets given 2D floor plans, *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, IEEE, pp. 3138–3143 (2015).