Vol.2012-MBL-64 No.18
Vol.2012-ITS-51 No.18
2012/11/16

グリッドグラフを利用した位置推定手法

久保健[†1]　田上敦士[†1]　長谷川輝之[†1]　長谷川亨[†1]　Jean Walrand[†2]

本論文では，アフィン変換を利用した新しいタイプの位置推定手法を提案する．本手法は Range-free localization に分類され，各ノードがトポロジ情報のみを用いて位置推定を行う分散アルゴリズムである．各ノードはグリッド状のトポロジを持つサブグラフをセンサーネットワークから抽出し，x-y 座標系をそのサブグラフ上に割り当てる．ノードは，サブグラフ上に含まれる 3 つのアンカーの（物理）位置と x-y 座標のマッピングに基づいて，自分の位置を推定する．この推定手順がアフィン変換によって記述される．既存の多辺測量を利用した手法では，非凸包なノード配置，すなわちノードが全く存在しない非配置領域を含むような配置の場合に位置推定誤差が大きくなる問題があるが，提案手法ではそのような配置でも誤差が少ない．本論文では，提案手法のアルゴリズムの概要および基本性能の評価結果について報告する．

Range-Free Localization using Grid Graph

TAKESHI KUBO[†1]　ATSUSHI TAGAMI[†1]　TERUYUKI HASEGAWA[†1]
TORU HASEGAWA[†1]　Jean Walrand[†2]

This paper proposes a new type of range-free localization method based on affine transformation. In the proposed method, nodes estimate their positions using topology information in a decentralized manner. Nodes extract a subgraph with grid topology and they assign x-y coordinates to themselves. The nodes on the subgraph estimates their position based on the mapping between the physical positions and the x-y coordinates of three anchors on the subgraph. This mapping process is described as an affine transformation. In the existing multilateration-based methods, the estimation error tends to be large in the case of a non-convex hull deployment, such as a terrain with big regions without sensors. However, the proposed method works well in such a deployment. This paper presents the key parts of the proposed method and some simulation results.

1. はじめに

センサーデバイスの小型，長寿命，低価格化に伴い，大量のセンサーを設置した環境センシングやオブジェクトトラッキングといったアプリケーションが現実味を帯び始めている．大量のセンサーを用いることで，より抜け目が少ない詳細な情報収集が可能になるため，例えば，あらゆるモノにセンサータグを貼り付けて，遺失物を発見する[3]などの新しいアプリケーションが期待できる．

センサーは環境情報などを取得するデバイスであるが，上記のようなアプリケーションを実現するためには，取得した情報がどの場所の情報であるかを知る術が必須である．つまり，センサーの位置推定，すなわち各センサーが自身の位置を認識する技術が必要不可欠である．

センサーの位置推定技術は，Range-based 方式と Range-free 方式の 2 種類に分類することができる．Range-based 方式は，各センサーがセンサー間の距離や受信角度など，物理的な量を測定する何らかの手段を備え，それを利用して位置推定を行う方式である．一方，Range-free 方式は，物理量を用いずセンサー同士の接続関係（トポロジ）のみを利用して位置推定を行う方式である．

本論文では，センサーが超小型化し，あらゆるモノがセンシングやコンピューティングの能力を備えるような将来的な利用形態を想定する．その形態の場合，Range-free 方式が，センサーのサイズや価格が小さく抑えられる可能性が高く有用であると思われる．Range-free 方式の既存研究には，多辺測量 (multilateration)の考え方を応用した手法[6][7]がある．元来，多辺測量とはある点から複数の基準点までの物理距離を測定し，物理距離と基準点の座標から 2 次方程式群を作り，位置を推定する手法である．これに対して[6][7]の方式では，物理距離を直接測定する代わりにホップ数から物理距離を推定する．しかし，このような方法では，センサーが配置されていない非配置領域が存在する場合に，位置推定精度が低下してしまう．それは，基準点から推定を行うノードまでのパスが当該領域を迂回せざるを得ないため，本来あるべきホップ数よりも大きくなる（実際よりも長い物理距離を推定しまう）ためである．このようなセンサーの非配置領域は現実にあり得る問題であり，例えばショッピングモールの吹き抜けのように物理的にセンサーの配置が困難な場所や，建物内の曲がりくねった廊下に設置されたセンサーが壁で電波が遮蔽されてしまうような状況が考えられる．

センサーが密にばらまかれている領域の中に上記のような非配置領域が存在するような空間は，数学的には非凸包 (non-convex hull) な空間と見なすことができる．本論文は，このような非凸包な空間においても，高い位置推定精度を実現する手法を提案する．

提案手法は，ネットワーク全体の一部のセンサー（以後

†1 株式会社 KDDI 研究所
KDDI R&D Laboratories, Inc.
†2 University of California, Berkeley

単にノードと呼ぶ）群の位置をまず推定し，その結果を用いて残りのノード群の位置を推定する 2 段階手法であり，集中サーバを一切必要としない分散アルゴリズムである．そして，第 1 段階の位置推定手法が提案の核となる．

提案手法の第 1 段階では，密に配置されたノードが構成するネットワークトポロジをユークリッド空間に見立て，座標変換の演算によって位置推定を行う．座標変換を行うために，ネットワーク全体から格子状のサブグラフ（本論文ではこれをグリッドグラフと呼ぶ）を発見し，グリッドグラフ上の各ノードに x-y 座標を割り振ることで座標系を確立する．この格子状のサブグラフは，直感的にはユークリッド空間に敷かれた方眼紙と見なすことができ，方眼紙の各格子点がグリッドグラフ上のノードに対応する．物理位置がわかっている格子点から，方眼紙の x，y 方向に何マス（何ホップ）ずつ離れているか，および 1 マス分の物理距離がわかれば，方眼紙の格子点の物理位置は算出できる．このような位置推定は，非配置領域がある非凸包な空間であっても，方眼紙を敷くことができれば可能であるため，高い位置推定精度を実現できる．

提案手法の第 2 段階では，第 1 段階で位置推定したノードから推定結果をブロードキャストし，それを取得したノードは，受信した推定結果の重心を自身の位置とする．これは，Centroid と呼ばれる手法[1]と同様である．

筆者らは[5]でこの手法を提案している．本稿では，提案手法の概要と，シミュレーションによる基本性能の評価結果について報告する．

2. 関連研究

Range-free 方式にはいくつかのタイプがあり，多辺測量，Centroid，Multi-Dimensional Scaling (MDS)を利用した手法が挙げられる．

多辺測量を応用した手法[6][7]では，物理位置が判明している一部のノード（これをアンカーと呼ぶ）を基準位置として，自ノードと複数のアンカーとの距離を用いて 2 次方程式を立て，自分の位置を最尤推定する．このとき，各アンカーが自身の物理位置情報をフラッディングすることで，各ノードはアンカーまでのホップ数を知り，さらにはアンカー同士でホップ数と物理距離の情報を交換することで 1 ホップの平均距離を算出し，ネットワーク全体で共有する．ホップ数と 1 ホップあたりの距離の情報から，各ノードはアンカーまでの距離を求め，位置推定を行う．しかし，冒頭でも述べたように，非凸包の空間では迂回経路が発生してしまうために位置推定精度が低下する．

Centroid を応用した手法[1][4]では，各ノードは隣接ノードにアンカーが存在した場合に，それらのアンカー群の位置の重心を自身の位置と推定する手法である．本手法の位置推定精度は，存在するアンカーの数が多いほど向上する．逆に，アンカーが少なければそもそも位置推定できないノード数が増加，すなわちカバレッジが低下する．アンカーの配置にはコストが余計にかかるため（ノードに GPS を持たせる，人手でノードに位置情報を書き込むなど），アンカーの数は少ないほどよい．提案手法は，アンカーの数を Centroid よりも低減しつつ，第 2 段階で Centroid を導入することで，高い推定精度を実現する．

Multi-Dimensional Scaling (MDS)とは，高次元データを可視化するための次元圧縮手法の一つである．MDS を応用した手法[8]では，アンカーが存在しなくても，相対位置を推定することができ，絶対位置を推定するためにも最低で 3 つのアンカーがあればよく，非凸包の空間での位置推定精度も高い．しかし，MDS は集中型アルゴリズムであり，ネットワーク全体のトポロジ情報を一カ所に集めて最尤推定を行う必要がある．ノードが広範囲に配置されてノード数が膨大になると，ネットワークトポロジの収集のためのトラフィックおよび計算の負荷が膨大になると考えられる．[9]は MDS のアルゴリズムを分散化した手法であるが，ネットワーク全体のトポロジ情報が必要という点では本質的に MDS と同様であり，ノード数が膨大になると，ノード間で受け渡しすべき情報量および計算量が膨大になる．一方提案手法は，ノード間で受け渡しすべき情報量と計算量は，各ノードの隣接ノード数のみに依存するため，どれだけ広範囲にノードが配置されてもその影響を受けない．

3. 提案手法の概要

3.1 前提条件

各ノードは無線通信によって他のノードと通信可能であり，直接通信可能なノードを隣接ノードと呼ぶ．本論文では，問題をシンプルにするため，各ノードの無線通信範囲は真円かつ同一半径であるものとする．つまり，ネットワークトポロジは，Unit Disc Graph であるものとする．また，ノードは 2 次元平面に配置されているものとする．Range-free 方式であるため，物理距離測定，受信角度の測定に関わるようなハードウェア補助は一切想定しない．

3.2 位置推定手法の概要

提案手法は，グリッドグラフを抽出してそこに含まれるノードのみが位置推定を行う第 1 段階と，その結果を用いて残りのノードが位置を推定する第 2 段階から構成される．

第 1 段階で行われるタスクは次の 2 つである．**(i)**ノードがネットワークからグリッドグラフを抽出し，そのグラフの各ノードに x-y 座標を割り当てる．**(ii)**割り当てた x-y 座標を物理位置に変換する．図 1 は，タスク(i)で抽出されるグリッドグラフの例を示している．ノード A～Z の 26 ノードからなるネットワークであり，実線または点線で結ばれているノード同士が隣接関係にある．また，オレンジ色のノード（A，F，H）はアンカー，すなわち自身の物理位置を知っているノードである．全ノードのうち 12 ノード（A～L）がグリッドグラフを構成する．本論文ではグリッド

Vol.2012-MBL-64 No.18
Vol.2012-ITS-51 No.18
2012/11/16

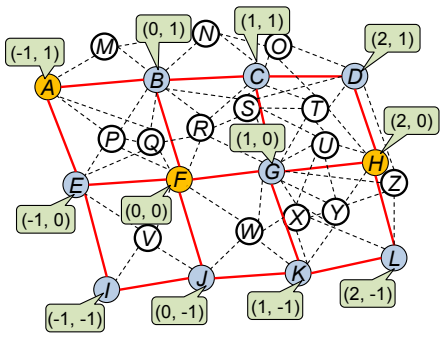


図 1　グリッドグラフの抽出例
Figure 1　Example of a grid graph extraction

グラフに含まれるノードをグリッドノードと呼び，それ以外のノードは非グリッドノードと呼ぶ（なお，単にノードと呼ぶ場合は，グリッド，非グリッドを意識しない）．1 章で述べたように，グリッドグラフが方眼紙に，グリッドノードが格子点に相当する．

グリッドグラフの抽出は，アンカーが起点となって行われる．アンカーは自ノードの x-y 座標を(0,0)として，+x, -x, +y, -y 方向の隣接ノード，すなわち(1,0)，(-1,0)，(0,1)，(0,-1)となるグリッドノードを選び，それを通知する．図 1 では，ノード F が原点となり，ノード G，E，B，J を選ぶ．通知を受けたノード G，E，B，J はさらに次のグリッドノードを選び，通知していく．このような手順を繰り返すことで，グリッドグラフが次第に拡大していく．

上述のように，グリッドグラフの抽出と x-y 座標の割り当ては同時に進行する．ただし，物理的な位置や方向とここで割り当てられる x-y 座標は全く独立である．そこで，グリッドグラフの中に含まれているアンカーの物理位置とその x-y 座標の対応関係を用いて，タスク(ii)の x-y 座標と物理位置の変換を行う．具体的には，グリッドグラフに直線上にない 3 つのアンカーが含まれていれば，それらの物理位置と x-y 座標から基底ベクトルを作り，1 次変換の行列を求める（実際には原点が平行移動するためアフィン変換となる）．この作業は各グリッドノードが個別に行うことができ，求めた行列と自身の x-y 座標から，物理位置を算出できる．

なお，ここまではある 1 つのノード（図 1 のノード F）を起点として抽出された 1 つのグリッドグラフのみに着目して説明してきた．しかし本手法では，実際には全てのアンカーが起点となってグリッドグラフが抽出される．グリッドグラフはそれぞれ独立に抽出され，一つのノードが複数のグリッドグラフに属す可能性がある．その場合，そのグリッドノードは，それぞれのグリッドグラフごとに x-y 座標を保持し，物理位置を算出する．そこで，最終的な推定位置は，算出した複数の物理位置の重心とする．

位置推定を行ったグリッドノード(およびアンカー)は，その情報をブロードキャストする．これによって，本手法第 2 段階の，Centroid 手法による位置推定が可能となる．図 1 では例えば，ノード W は，ノード F，G，J，K から位置情報を受信し，これら 4 ノードの重心を自位置とみなす．

以上のように，提案手法は各ノードが隣接ノードとのみ通信を行うことで位置推定を行う分散アルゴリズムである．また，タスク(i)において各ノードは 2 ホップ先までしか考慮しないため，各ノードの通信量および計算量は隣接ノード数にしか依存しない．

4. グリッドグラフの定義と位置推定

本節では，まずタスク(i)で抽出されるグリッドグラフを定義する．次に，タスク(ii)で実施される位置推定演算の方法および位置推定誤差の発生要因について述べる．

4.1 グリッドグラフの定義

ネットワーク全体をグラフ $\boldsymbol{G}=\{\boldsymbol{V},\boldsymbol{E}\}$, $\boldsymbol{V}$をノード集合，$\boldsymbol{E}$ をエッジ集合とする．$e_{i,j}\in\boldsymbol{E}$ をノード $v_i\in\boldsymbol{V}$ から $v_j\in\boldsymbol{V}$ への有向エッジとする．また，エッジの始点と終点のノードを意識しない場合は e^1, e^2 などのように表す．

(1) エッジ属性

エッジは向きを表す属性を持つ．提案手法では属性値として，+x，-x，+y，-y，「未割当」の 5 つを定義し，それぞれ(1,0)，(-1,0)，(0,1)，(0,-1)，ϕ というベクトルで表す．また，関数 $f(e^1)$はエッジ e^1 の属性値を返すものとする．

(2) 凸サイクル

グリッドグラフを定義する前に，その基本コンポーネントである「凸サイクル」を定義する．凸サイクルはそれ自身が最小のグリッドグラフである．4 つのノードと 8 個のエッジからなるサブグラフ $\boldsymbol{G_c}\subset\boldsymbol{G}$, $\boldsymbol{G_c}=\{\boldsymbol{V_c},\boldsymbol{E_c}\}$, $\boldsymbol{V_c}=\{v_1, v_2, v_3, v_4\}$，$|\boldsymbol{E_c}|=8$ を考える．このサブグラフ $\boldsymbol{G_c}$ が以下の条件を満たすとき，$\boldsymbol{G_c}$ は凸サイクルである．

C1: あらゆる $v_i\in\boldsymbol{V_c}$ について $deg(v_i)=4$ である
C2: あらゆる $e^i\in\boldsymbol{E_c}$ について $f(e^i)\neq\phi$である
C3: あらゆる $e_{i,j}\in\boldsymbol{E_c}$ について $e_{j,i}\in\boldsymbol{E_c}$ である
C4: あらゆる $e_{i,j}, e_{j,k}\in\boldsymbol{E_c}$ について $e_{i,k}\notin\boldsymbol{E}$ である
C5: あらゆるエッジの組 $e_{i,j}, e_{j,i}\in\boldsymbol{E_c}$ について $f(e_{i,j})+f(e_{j,i})=(0, 0)$である．
C6: エッジ集合 $\boldsymbol{E_c}$ について $f(e_{i,j})+f(e_{j,k})+f(e_{k,l})+f(e_{l,i})=(0, 0)$を満たす．
C7: あらゆるノード $v\in\boldsymbol{V_c}$ の入エッジ e^1 および出エッジ e^2 について，$|f(e^1)+f(e^2)|=\sqrt{2}$を満たす．

これらの条件の中でも特に重要なのは定義 **C4** で，4 つのノードによって構成されるサイクルが物理空間上で凸四角形になることを保証するための条件である[5]．凸サイクルは，方眼紙における各「マス目」に相当し，全てのマス目が凸四角形であることによって，1 対 1 写像（すなわち行列を用いたアフィン変換）による位置推定が可能となる[2]．

Vol.2012-MBL-64 No.18
Vol.2012-ITS-51 No.18
2012/11/16

(3) **グリッドグラフ**

サブグラフ $\boldsymbol{G_g} \subset \boldsymbol{G}$, $\boldsymbol{G_g} = \{\boldsymbol{V_g}, \boldsymbol{E_g}\}$を考える．このサブグラフ $\boldsymbol{G_g}$ が以下の条件を満たすとき，$\boldsymbol{G_g}$ はグリッドグラフである．またノード $v \in \boldsymbol{V_g}$ をグリッドノードとよび，それ以外を非グリッドノードと呼ぶ．

G1: 全てのグリッドノードは，入エッジに対してエッジ属性値をたかだか 1 種類ずつ持つ．また，出エッジについても同様である．

G2: 全てのグリッドノードは，少なくとも 1 つの凸サイクルに含まれる．

G3: サブグラフ $\boldsymbol{G_g}$ のあらゆる閉路は，void 領域を含まない．なお，void 領域とはその内部に凸サイクルを一つも含まない領域である（図 2）．ただし，凸サイクルの内部は除く．

定義 **G1** と **G2** から，各グリッドノードは双方向通信可能なたかだか 4 つの隣接ノードしか持ない．また，定義 **G3** から，グラフ $\boldsymbol{G_g}$ 上の全てのグリッドノードは，それぞれユニークな x-y 座標を 1 つだけ持つ[5].

4.2 グリッドグラフを用いた位置推定演算

タスク(i)が完了すると，前節で定義したようなグリッドグラフが抽出され，各グリッドノードがそれぞれユニークな x-y 座標を保持する．つまり，グリッドグラフが座標系を構成しており，それが図 1 のように物理空間上にオーバレイされていると考えることができる．提案手法では，グリッドグラフが構成する座標系を 2 次元ユークリッド空間とみなし，物理空間（これも 2 次元ユークリッド空間である）と 1 対 1 写像で関連づける．この関連づけの操作はアフィン変換と呼ばれる[2].

提案手法のタスク(ii)で用いるアフィン変換は次のように表される．あるグリッドグラフ上の 3 つのアンカーL_1, L_2, L_3 を考える．これらの物理座標をそれぞれ(X_0, Y_0), (X_1+X_0, Y_1+Y_0), (X_2+X_0, Y_2+Y_0)とし，同じくこれらのグリッドグラフ上での x-y 座標を(x_0, y_0), (x_0+x_1, y_0+y_1), (x_0+x_2, y_0+y_2)とする．そのグリッドグラフ上のあるグリッドノード g の x-y 座標が(x, y)である場合，ノード g の物理位置(X, Y)を次式で推定する．

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} X_1 & X_2 \\ Y_1 & Y_2 \end{pmatrix} \begin{pmatrix} x_1 & x_2 \\ y_1 & y_2 \end{pmatrix}^{-1} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} X_0 \\ Y_0 \end{pmatrix} \qquad (1)$$

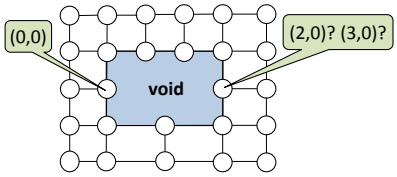


図 2　void 領域を含むサブグラフの例
Figure 2　Example of a subgraph with a void region

ただし，アンカーL_1, L_2, L_3 は直線上にあってはならない．これは，x-y 座標上で直線上にある場合，第 1 項の逆行列が定義できない．また物理空間上で直線上にある場合は，式(1)の計算結果が常に直線上のいずれかの点にしか射影されないためである．

式(1)を計算するためには，必ず 3 つのアンカーが必要である．グリッドグラフがアンカーを 2 つ以下しか含まない場合は，例えタスク(i)でグリッドグラフを抽出できたとしても，タスク(ii)が実施できないため，グリッドノードは位置推定できない．

4.3 位置推定誤差の発生要因

前節で述べたように，提案手法ではグリッドグラフが構成する座標系を 2 次元ユークリッド空間とみなす．しかし，実際には必ずしも正確なユークリッド空間にならず，それが原因で位置推定誤差が発生する．図 3(a)，(b)はその様子を示している．図 3(a)では，ノード A～I がグリッドノードであり，そのうちノード D，E，F がアンカーである．図 3(a)から直観的にもわかるとおり，トポロジは格子状になってはいるものの，方眼紙と呼ぶには少し歪な形状となっている．以下では，式(1)が意味することを考察し，誤差発生のメカニズムを説明する．

式(1)を正確に表現すると，3 つのアンカーによって作られる 2 つの基底ベクトルが張る 2 次元ユークリッド空間と，物理空間との写像の式である．基底ベクトルとは，例えばベクトル L_1L_2 とベクトル L_1L_3 などである．各グリッドノードが格子点になるように，基底ベクトル同士を適当に合成して基底変換を行うと，式(1)は，図 3(b)の赤線で示すようなグリッド線を想定していることがわかる．つまり，各グリッドノードは，自分がグリッド線の交点（すなわち格子点）であると信じて式(1)を計算し，物理位置(X, Y)を得るということである．しかし図 3(b)に示すように，実際の位置は想定した格子点からずれている場合があり，そのずれの大きさが位置推定誤差となる（図 3(b)の矢印が誤差を表す）．

つまり位置推定誤差の根本的な原因は 4.1 節で定義した凸サイクル（方眼紙のマス目に相当）の物理空間上での形状が一定ではないことにある．これは元々のノードの配置のされ方に依存する部分が大きい．しかしそれ以外に，位

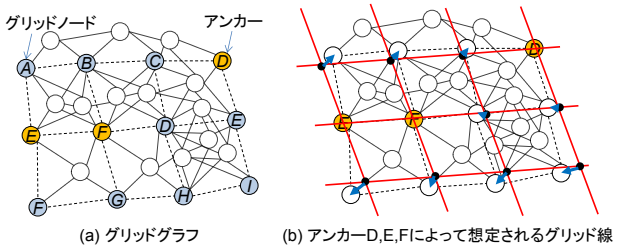


図 3　理想格子点からのずれ
Figure 3　Slippage from the ideal intersections

置推定誤差は基底ベクトルの選び方にも依存する. つまり，与えられたグリッドグラフに 4 つ以上のアンカーが含まれる場合には，どの 3 つを選ぶかによって，位置推定誤差が異なる可能性がある．図 3(b)に示すような理想格子点は，アンカーの位置で位置推定誤差が 0 になるように定められる．しかし，マス目の形が歪であると，アンカーから離れるほど，理想格子点からの乖離，すなわち位置推定誤差が増大する（位置推定誤差がアンカーからのホップ数 n に対して O(n)のオーダーとなる[5]）．従って，各グリッドノードは自身からなるべく近いアンカーを 3 つ選んで式(1)を計算すべきである．また，選択したアンカーの間にあるマス目の形に従って図 3(b)のグリッド線が想定されるため，図 4(a)に示すようにアンカー同士が近すぎると（アンカーa, b, c），偶然存在した歪な形状のマス目の影響を受けてしまう．逆に，図 4(b)のようにアンカー同士がある程度離れていてれば（アンカーa，d，e），マス目の歪さが平準化されるため特異なマス目の影響を軽減できると考えられる．なお，図 4(a)および(b)は全く同じノード配置である．

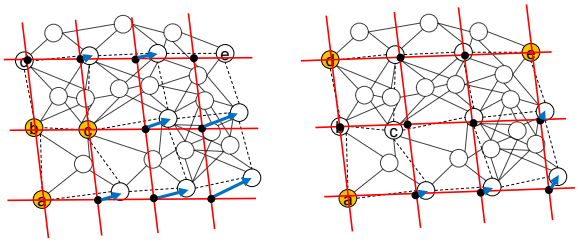
上述の条件をうまく満たすアンカーがない場合には，いつ推定誤差が増大する可能性があるため，アプリケーションによっては位置推定を行わないという選択もありうる．

5. グリッドグラフ抽出アルゴリズム

本節では，前節で定義したグリッドグラフを抽出するための分散アルゴリズムについて述べる．なお，位置推定は 3.2 節の後半や 4.2 節で述べたように，非グリッドノードは Centroid 手法と同様の重心計算により位置推定を行うが，手順が単純であるため本論文では説明を省略する．分散アルゴリズムや位置推定手順の詳細は[5]を参照されたい.

5.1 グリッドグラフ抽出の流れ

グリッドグラフの抽出は，アンカーから開始される．つまり, アンカーごとに 1 つのグリッドグラフが抽出される. アンカーは，自身の x-y 座標を原点(0,0)として，隣接格子点すなわち(1,0)，(-1,0)，(0,1)，(0,-1)の 4 点に対応するノードを自身の隣接ノードの中から選ぶ．選ばれたノードはそのグリッドグラフのグリッドノードとなる．新しく選ばれたグリッドノードは更に原点から遠い方の格子点, 例えば, (1,0)のグリッドノードは，(2,0)，(1,1)，(1,-1)に対応する新たなグリッドノードに選ぶ．本論文では，あるグリッドノードに対して，そのノードよりも原点に 1 ホップ近いノードを上流グリッドノード，原点から 1 ホップ遠いノードを下流グリッドノードと呼ぶ．つまり，グリッドノードは下流グリッドノードを新しく選択する. なお, 特定の方向 (+x, -x，+y，-y 方向）について選択できるノードがない場合はその方向は存在しないものとして扱う．このような動作を繰り返し，新しく選ばれるグリッドノードが無くなるまで拡大することで，x-y 座標付きのグリッドグラフが抽出される.



(a) 近いアンカー同士を選んだ場合　(b) 遠いアンカー同士を選んだ場合

図 4　アンカーの選び方による位置推定誤差の違い

Figure 4　Estimation errors for different anchor selections

グリッドグラフ抽出の過程で，アンカーが新しいグリッドノードとして選ばれる場合がある．その場合，そのアンカーは，自身の x-y 座標と物理位置の組をすでに抽出済みのグリッドグラフ内にフラッディングする. これによって，グリッドグラフ上の全てのグリッドノードがその上に存在するアンカーの情報を知ることができるため，4.2 節と 4.3 節で述べたように，そこから 3 つのアンカーを選んで式(1)を計算して位置を推定する.

5.2 方向ベクトルとユニットグリッド

前節で各グリッドノードは，+x，-x，+y，-y 方向の隣接ノードを選択すると述べた．図 1 では，ノード F が原点となり，ノード G，E，B，J を選ぶ．本論文では，選んだノードを並べてベクトル形式で表記したもの，すなわちベクトル(G,E,B,J)を方向ベクトルと呼ぶ. 要素の並びは+x, -x, +y，-y 方向の順とする．なお，選べるノードが存在しない場合は，その方向の要素はΦと表記する.

一方，4.1 節で定義したグリッドグラフの基本構成要素は凸サイクルである．そこで，凸サイクルを最大 4 つ組み合わせたユニットグリッド（図 5）を導入し，方向ベクトルと 4.1 節のグリッドグラフの定義を紐づける．ユニットグリッドとは,

図 5(b)に示すように，3×3 のノードから構成されるグリッドグラフである．なお，図 5(b)には，4 つの凸サイクル B-C-A-E，C-D-F-A，A-F-I-H，E-A-H-G が含まれている．各ノードは自身を中心とする全てのユニットグリッドを調べ，その中の一つから方向ベクトルを取り出す．図 5(c)では，ノード A が(F,E,C,H)という方向ベクトルを選択することを示しているが，これは図 5(b)のユニットグリッドからノード A の隣接ノードのみを取り出したものとなっている.

5.3 グリッドグラフ抽出アルゴリズム

あるグリッドグラフが抽出される過程において，グリッドノードは 2 種類に分類される．一つ目は，x-y 座標が x 軸上または y 軸上，すなわち座標値が y=0 または x=0 となるノードである．これを軸グリッドノードと名付ける．軸グリッドノードの特徴は上流グリッドノードが 1 つしかないことである．残りの一つは，x-y 座標の座標値が，$x\neq0$ かつ $y\neq0$ となるノードである. これを一般グリッドノードと名付ける．一般グリッドノードの特徴は上流グリッドノ

Vol.2012-MBL-64 No.18
Vol.2012-ITS-51 No.18
2012/11/16

ードが 2 つあることである．上流グリッドノード数の違いから，軸グリッドノードと一般グリッドノードで異なるアルゴリズムが動作する．以下，各種別について，アルゴリズムの概要を述べる．

(1) **軸グリッドノード**

ここでは，図 5(a)のノード F が，ノード A から方向ベクトル(F,E,C,H)を受け取り，x-y 座標(1,0)の軸グリッドノードになったと想定して，以後の処理をステップごとに記述する．

(a) ノード F は自身を中心とする全てのユニットグリッドから方向ベクトルを取り出し，それら全てを広告する．広告には，「+y フラグ」を付加する．+y フラグとは，ノード F の下流グリッドノードのうち，y の座標値が自分より 1 だけ大きい，すなわち(1,1)となるノードだけが受信した方向ベクトルを処理すべきであることを表す．

(b) ノード F は x-y 座標(1,1)のグリッドノードから方向ベクトルを受信するのを待つ．ここでは，(M,C,P,F)という方向ベクトルをノード D から受け取るとする．

(c) ノード F はステップ(a)で見つけたユニットグリッドの中から，受信した方向ベクトルと合致するものだけを残し，残りを破棄する．そして残ったユニットグリッドから方向ベクトルを取り出して，それを広告する．ここでは，「-y フラグ」を付加する．+y フラグと同様，x-y 座標が(1,-1)となるノードだけが処理すべきであることを表す．

(d) ノード F は x-y 座標(1,-1)のグリッドノードから方向ベクトルを受信するのを待つ．ここでは，(T,H,F,Φ)という方向ベクトルをノード D から受け取るとする．

(e) ノード F はステップ(c)と同様に，受信した方向ベクトルと合致するユニットグリッドだけを残す．そして，残ったユニットグリッドの中から一つを選んで方向ベクトルを取りだす．ここで選んだ方向ベクトル（例えば，(K,A,D,I)とする）が，確定情報となる．ノード F はフラグをつけずにその方向ベクトルを広告する．

なお，ステップ(c)や(e)で，受信方向ベクトルとユニットグリッドの合致判定が行われるが，ノード F がノード D から受信した方向ベクトルについてその判定基準を図 6 に示す．

(2) **一般グリッドノード**

ここでは，図 5(a)のノード D に着目して説明する．

(a) ノード D はノード F から方向ベクトルの広告を受信する（広告には複数の方向ベクトルが含まれる）．ここでは，受信広告には(K,A,D,I)という方向ベクトルが含まれており，+y フラグが付加されているものとする．送信元の x-y 座標が(1,0)であり，ノード D が方向ベクトルの 3 番目の要素にあることから，自身の x-y 座標は(1,1)になることがわかる．さらに，受信広告に+y フラグが付加されているため，その広告を受領する．ノード D は一般グリッドノードであり，上流グリッドノードが 2 つあるため，もう一方の上流から広告を受け取るまで待つ．

(b) ノード D はノード C から方向ベクトルの広告を受信する．ここでは，受信広告に(D,B,L,A)が含まれているものとする．

(c) ノード D は受信した 2 ノードからの広告から，自身がグリッドノードになるべきかをチェックする．グリッドノードになるためには，2 つのノードから受信した方向ベクトルを一つずつ取り出し，その 2 つのベクトルの両方に自分自身が含まれており，さらにもう一つ別のノードも両方に含まれていなければならない．ステップ(a)，(b)で受けた方向ベクトルの例では，ノード D およびノード A が両方に含まれており，この条件を満たす．

(d) 自身がグリッドノードになると決めたノード D は，その判定に用いた方向ベクトルと合致するようなユニットグリッドだけを選び，方向ベクトルを取り出し，広告する．なお，広告する方向ベクトル 1 つとは限らないが，少なくとも上流グリッドノードは確定している（ノード D の上流はノード C および F である）．

(e) 以後，ノード D は下流グリッドノード（例えばノード M や P）から上記ステップ(d)で自身が広告したときと同様に，ノード D が上流グリッドノードであるような方向ベクトルの広告を受け取るため，下流グリッドノードが何であるかを知ることができる．

なお，ステップ(c)で複数のノードがグリッドノードになる

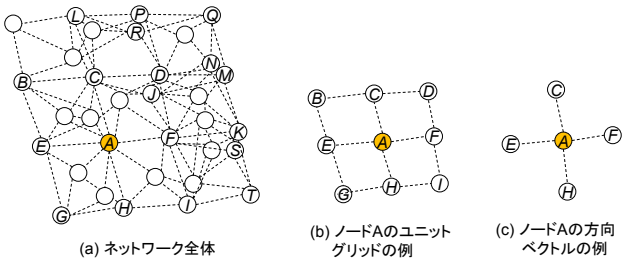


図 5　方向ベクトルとユニットグリッドの例
Figure 5　Example of direction vector and unit grid

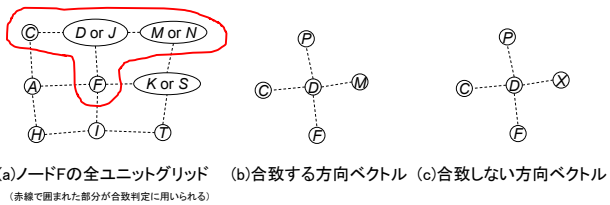


図 6　方向ベクトルとユニットグリッドの合致判定
Figure 6　Matching between unit grids and a direction vector

べきと判定される可能性がある．これを防ぐためにグリッドスコアというユニットグリッドの良さを表す指標を用いるが，本論文では紙面の都合で割愛した．詳しくは[5]を参照されたい．

以上，軸グリッドノードおよび一般グリッドノードがステップ(a)～(e)を実施することで，1 つのグリッドグラフが抽出される．

6. シミュレーション実験

提案手法の有効性を検証するためにマルチエージェントシミュレータを実装し，既存手法として DV-hop [6]と Centroid [1]との性能比較を行った．評価した性能は，位置推定誤差，カバレッジ，位置推定完了までにやりとりされるパケット数である．なおカバレッジは次のように定義する．全ノード数を N，ランドマーク数を L とし，位置推定ができなかったノード数を w としたとき，カバレッジ C は $C = (N\text{-}L\text{-}w)/(N\text{-}L)$である．

6.1 グリッドグラフの例

まず，ある一つのアンカーを起点に抽出されたグリッドグラフの例を示す．図 7 の四角はアンカーを×印はノードを表しており，ノード密度がなるべく均一になるように配置した．なお，各ノードの通信半径は 0.8 で平均次数は 40 である．中央付近のアンカーからグリッドグラフ抽出を開始し，26 ステップ（ノードは 1 ステップにたかだか一回パケットを送信する）で図 7 に示したようなグリッドグラフが抽出された．なお，図 7 では例示のためアンカーを 1 つしか配置していないため，4.2 節で述べたように，各グリッドノードは位置推定を行うことはできない（アンカーに隣接している非グリッドノードは，アンカーと同一位置であると推定する）．

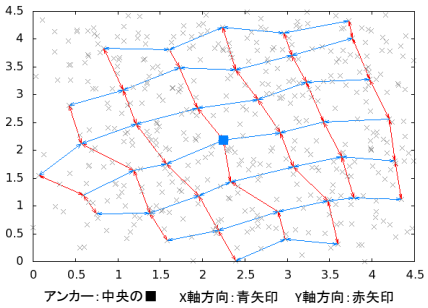
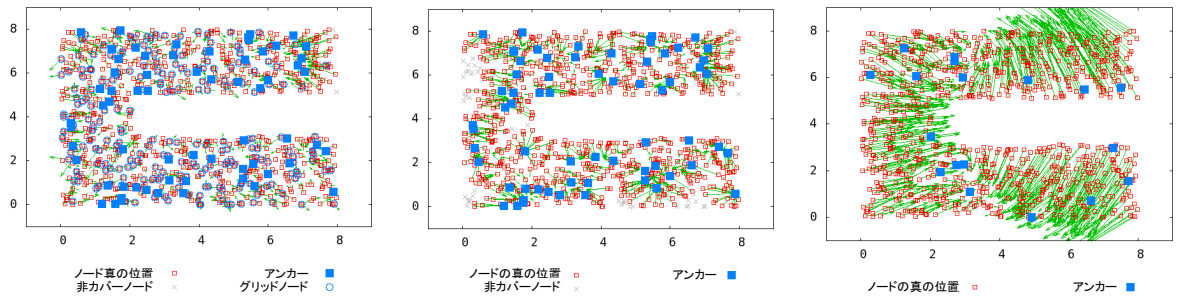


図 7　抽出されたグリッドグラフの例

Figure 7　Example of an extracted grid graph

6.2 実験結果

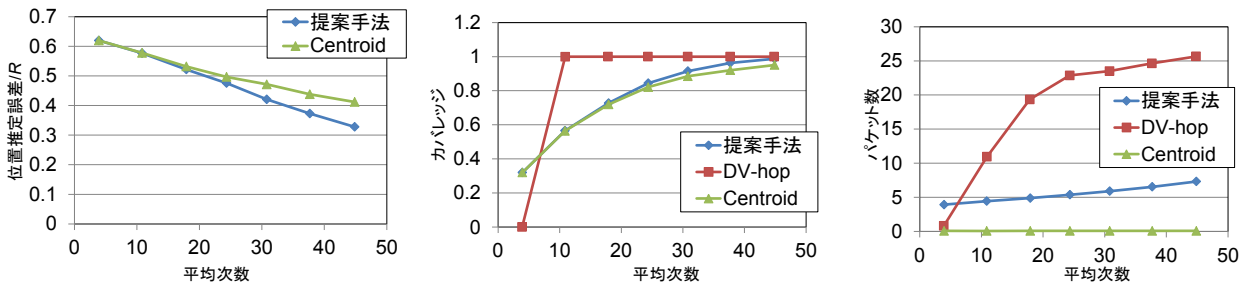
非凸包なノード配置における既存手法と提案手法の位置推定誤差とカバレッジを図 8 に示す．実験では，8×8 のシミュレーションフィールドの中央付近に void 領域を設定し，void 領域以外の部分にノードをランダムに配置した．各ノードの通信範囲は 0.8 で，平均次数は 44 である．また，提案手法と Centroid では，全ノードの 7%をアンカーとし，DV-hop では全ノードの 2%をアンカーとした（提案手法などと割合が異なるのは，DV-hop では 2%の時に位置推定誤差が最小になったためである）．図 8 (a)，(b)，(c)がそれぞれ提案手法，Centroid，DV-hop を示しており，いずれもノ



(a) 提案手法　(b) Centroid　(c) DV-hop

図 8　非凸包な配置における位置推定誤差

Figure 8　Position estimations in a non-convex hull deployment



(a) 平均位置推定誤差　(b) カバレッジ　(c) 収束までの送信パケット数

図 9　ノードの平均次数に対する各種パフォーマンス

Figure 9　Performance of the three methods

ードの配置の仕方は同一である．

図 8 からもわかるとおり，Centroid ではカバレッジが提案手法よりも小さく，DV-hop では位置推定誤差が非常に大きくなっている．Centroid は，アンカーと隣接するノードしか位置推定できないが，提案手法はグリッドノードが推定結果を広告することにより，擬似的にアンカーの数を増加させる効果があるため，必ずカバレッジが Centroid より向上する．また DV-hop は，アンカーまでのホップ数を物理距離に変換するが，中央の void 領域を迂回するためにホップ数が実際の物理距離に比べて増大し，位置推定誤差が大きくなる．

図 9 に，ノードの次数，すなわちノードの配置密度に対する位置推定誤差（通信半径で正規化），カバレッジおよびアルゴリズム終了までに一つのノードが送信するパケット数の結果を示す．各手法，各平均次数について，100 回ずつ異なるノード配置を発生させ，ランダムにランドマークを選んで（ランドマーク数は図 8 の場合と同じ）これらのパフォーマンスを測定し，平均を取ったものをプロットしている．

図 9(a)では，DV-hop の位置推定誤差が最小でも次数 50 の時で 1.4 であるため，グラフ上には現れない．提案手法と Centroid を比較すると，次数が大きくなるとグリッドグラフが抽出できる可能性が高まり，グリッドノードがより正確に位置推定できることから，提案手法の位置推定誤差が改善する．

カバレッジについては，図 9(b)に示すように，DV-hop では次数が最小の時以外常に 1 となっている（次数が最小の時は，ネットワークが分断され，3 つ以上のアンカーが含まれなくなるため位置推定できない）．ただし，優れたカバレッジと引き替えに，位置推定誤差を犠牲にしている．提案手法と Centroid を比較すると，提案手法の方が次数が増えたときに位置推定誤差，カバレッジとも改善量が大きいが，これはグリッドノードが疑似的にランドマークとして働くためである．

アルゴリズム終了までに各ノードが送信するパケット数は，図 9(c)に示すように Centroid では非常に小さい．これはアンカーのみ 1 回だけ位置情報を広告すれば良いためである．一方提案手法では，非グリッドノードはユニットグリッド構成のために 3 回のパケット送信を行う．グリッドノードは，それに加えて一つのグリッドグラフに組み込まれる際に 2 回の送信，さらにそのグリッドグラフのアンカーの数だけパケットを送信する．そのため Centroid よりは送信パケット数が大きい．しかし，DV-hop では各アンカーが 3 回ずつネットワーク全体にフラッディングを行うため，アンカー数が多くなるほど送信パケット数が増加する．

7. おわりに

本論文では，アフィン変換を利用した新しいタイプの位置推定手法を提案した．提案手法は，物理空間に方眼紙をオーバレイして位置推定を行う方法から着想を得ており，センサーノードが自律分散的に方眼紙に対応するサブグラフ，すなわちグリッドグラフを抽出する．グリッドグラフには x-y 座標を割り当てておき，グリッドグラフ上の 3 つのアンカーの物理位置と x-y 座標のマッピングを利用することで，グリッドノードが位置を推定する．この計算プロセスがアフィン変換で記述される．

今後の課題は，ノードの故障や不均一な電波強度のような実際の状況に近い想定で提案手法の評価を行い，有効性を検証することである．

謝辞 日ごろご指導いただく，KDDI 研究所中島所長に深く感謝いたします．

参考文献

1) N. Bulusu, J. Heidemann and D. Estrin, “GPS-less Low Cost Outdoor Localization For Very Small Devices,” IEEE Personal Communications, Vol. 7, Issue 5, pp. 28-34, 2000.
2) H. Coxeter, and S. Greitzer, “Geometry Revisited,” Washington, DC: Math. Assoc. Amer., pp. 101, 1967.
3) M. Gorlatova, P. Kinget, I. Kymissis, D. Rubenstein, X. Wang and G. Zussman, “Challenge: Ultra-Low-Power Energy-Harvesting Active Networked Tags (EnHANTs),” In Proc. of ACM MobiCom 2009, pp. 253-260, 2009.
4) T. He, C. Huang, B. M. Blum, J. A. Stankovic and T. Abdelzaher, “Range-Free Localization Schemes for Large Scale Sensor Networks,” In Proc. of ACM Mobicom’03, 2003.
5) T. Kubo, A. Tagami, T. Hasegawa, T. Hasegawa and J. Walrand, “Range-free Localization using Grid Graph Extraction,” In Proc. of IEEE ICNP’12, 2012.
6) R. Nagpal, H. Shrobe and J. Bachrach, “Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network,” In Proc. of ACM IPSN’03, 2003.
7) D. Niculescu and B. Nath, “Ad Hoc Positioning System (APS),” In Proc. of IEEE GLOBECOM 2001, pp. 2926-2931, 2001.
8) Y. Shang, W. Ruml and Y. Zhang, “Localization from Mere Connectivity,” In Proc. of ACM MobiHoc’03, 2003.
9) Y. Shang and W. Ruml, “Improved MDS-Based Localization,” In Proc. of IEEE Infocom’04, 2004.