社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－MBL－40 (24)
2007－UBI－13 (24)
2007／2／23

歩行情報解析に向けたデータ圧縮方式の設計

上原雄一† 森雅智† 白石陽‡ 戸辺義人†

† 東京電機大学 工学部 情報メディア学科
‡ 東京大学 空間情報科学研究センター

歩行情報解析では，短い間隔で取得した多くの足圧力データが必要となる．さらに，無線センサネットワーク環境においては，歩行情報を抽出する際にセンサの電力量，メモリサイズやCPUなどの制限された資源を考慮する必要がある．そこで，我々は以前，設計した健康モニタリングシステムAlways-on Karteに歩行の特性を考慮したデータ圧縮方式を適用することで，データ量および電力消費を低減させる．さらに，本圧縮方式では，AoKスリッパ内で取得した圧力値をグループ分類し，そのグループに基づいて圧力値の特性に合わせたデータ圧縮を行うことで圧縮効果を向上させる．

Design of Data Compression for Analyzing Walking Pattern

Yuichi Uehara†, Masato Mori†, Yoh Shiraishi‡ and Yoshito Tobe†

† Department of Information and Media Design, Tokyo Denki University
‡ Center for Spatial Information Science, the University of Tokyo

We have developed AoK mule system that obtains the distribution of pressures data at feet, transmits the data to a server through intermediate nodes, and analyzes the gait of the person wearing the mule. Extracting human's walking pattern using wireless sensors impose a challenge in the balance between data storage and CPU power. Taking available resources on nodes into consideration, we apply data compression based walking pattern to Always-on Karte. Our compression algorithm selects more efficient data compression algorithm for analyzing walking pattern according to acquired data.

1. はじめに

現在の日本の医療では，発病したことを自分で認識した後に専門医にかかるスタイルが主流となっている．しかし，専門知識無しで，病気の発病を自身で判断するということは容易ではない．特に加齢により判断能力が衰える高齢者に対してはさらに大きな問題となる．内閣府の高齢社会白書[7]によると，65歳以上の高齢者人口は，総人口に占める割合が19.5%に達したとしている．こうしたことから，自分で病気を認知し，通院するのではなく，自動的に健康状態の異常を察知する情報技術の重要性が増している．そこで，以前我々はAlways-on Karte[5]を設計した．本システムは，人の身体に様々なセンサ(足圧力，血圧，体温など)を取り付け，そのセンサから得たデータをサーバへ送信し，データ管理することで健康管理を行うシステムである．その人の身体に取り付けるセンサのプロトタイプとして，圧力センサおよびMicaZ Mote[3]を取り付けたスリッパ(以後，AoKスリッパと呼ぶ)を開発した．AoKスリッパは，両足から足圧力分布を取得し，ノードを中継して，サーバへ送信する．本システムにおける問題は，生データをサーバへ送信する頻度である．もし，さらに多くのセンサを取り付け，測定点が増えた場合，より多くのデータが発生し，送信頻度が高くなる．しかし，我々はAoKスリッパにおいて，電力量やメモリサイズなどの資源は高性能なものが実装されていると想定していないため，単に逐次データを送信する手法を適用することはできない．そこで，制限された環境で多くのデータを高頻度でより効率よく送信する手法が必要である．この問題を対処するために，我々は歩行特性を考慮した圧縮方式をAlways-on Karteに適用する．

本稿では，2章でAlways-on Karteについて述べ，3

章で圧縮方式に関して述べる．4 章にて性能評価実験について述べ，5 章で関連研究を述べる．最後にまとめと今後の課題について述べる．

2. Always-on Karte

本章では，提案する圧縮方式の適用先である Always-on Karte について述べる．

2.1 システム概要

Always-on Karte は，人々の健康状態をモニタリングすることにより健康管理を行うシステムである．図 1 に目標とするシステム図を示す．Always-on Karte は，AoK probe, AoK recipients, AoK_Server から成る．AoK probe は人からデータを収集する無線機能を備えたデバイスである．AoK probe は常に足圧力，体温などの生体情報を取得する．生体情報には，「通常歩行」，「座っている」のような生データ以外の情報も含む．また，複数の AoK probe を身に着けると，AoK probe は相互に通信を行い，新たな情報を生成する．AoK recipients は AoK probe から集められたデータを転送するために，屋内にネットワークを形成する．生体情報は，AoK probe から得られ，AoK recipients を通して AoK_Server に送られる．AoK_Server では，Karte と呼ばれる生体情報データベースを構築する．また，AoK_Server は，インターネット等の外部ネットワークと接続され，外部ネットワークと通して，医者や遠隔地に住んでいる家族に個人のデータを送信する．家族からの健康状態に関する問い合わせや医者から診断書(Karte)を受け取る．

2.2 AoK スリッパ

AoK probe の実装として，我々はスリッパを選択した．スリッパはモニタリングするためのものではなく，日常に使用するものであるため，人々はモニタリングされていると感じずに使用可能となる．AoK スリッパでは，圧力センサから圧力値を取得し，AoK_Server へ送信する機能を持つ．AoK スリッパは，スリッパ，MicaZ Mote[3](以下，Mote と呼ぶ)，およびスリッパの表面上の 2 つの圧力センサから構成される．圧力センサの出力は Mote を通して，デジタルデータに変換される．スリッパの表面に 2 つの圧力センサを配置することによって，圧力分布を観察することが可能となる．図 2 にプロトタイプを示す．図 2 に示した通り，圧力センサは表面の最前部および最後部に配置される．デジタル化された圧力データは，Mote の無線通信を通して AoK_Server へ送信される．

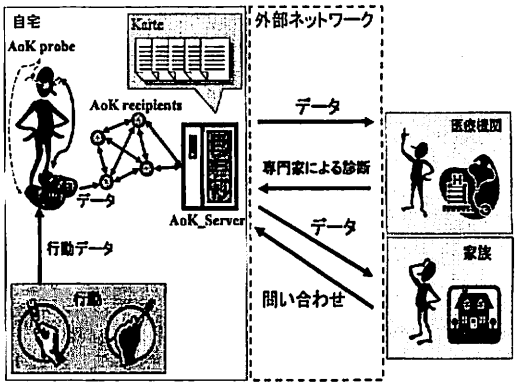

図 1：Always-on Karte の最終図

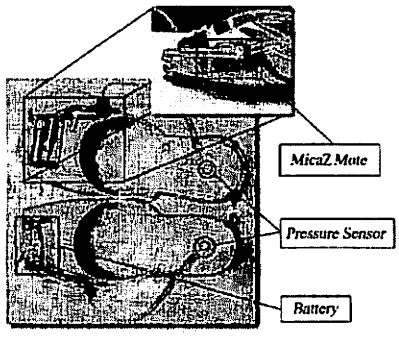

図 2：AoK スリッパ

本稿では，圧力分布を取得した後，取得したデータを圧縮する機能を AoK スリッパへ適用する．取得した後すぐに送信せず，データを圧縮し送信することで，データ量およびデータ送信回数を減らし，消費電力を低減させる．データ圧縮方式に関しては，次章にて詳述する．

3. データ圧縮方式

本章では，歩行特性を考慮したデータ圧縮方式に関して述べる．

3.1 課題

データ圧縮において，2 つの課題が存在する．1 つ目は，使用する無線センサデバイスにおける課題，2 つ目は，歩行データ取得における課題である．

- 無線センサデバイスにおける課題

AoK スリッパでは，取得したデータを直接 AoK recipients を通して AoK_Server へ送信する．しかし，直接データを送信することは，データ量および消費電力の増加を引き起こす．さらに，AoK スリッパ内でデ

ータを圧縮する際に問題が存在する．本システムで想定しているセンサデバイスは，高性能のものではなく，電力量，CPUやメモリサイズに制限がある．この制限により，本システムで目的としている継続的な解析を行うことが困難となる．例えば，データ圧縮方式に高速フーリエ変換などのような多くのデータを用いて複雑な計算をする必要がある手法を用いることは難しい．そのため，単純で少量のデータから圧縮可能な圧縮方式が望まれる．このように，資源の制限されたセンサネットワーク特有の環境において，センサデバイスの電力や圧縮時の計算方法は考慮しなければならない課題である．

- 歩行データ取得における課題

歩行データに対して非可逆圧縮を行うことで，データ量は減るが，その一方で本来生データが持っていた情報をいくらか失うことになる．しかし，健康モニタリングにおいては，取得したデータを基に詳細な人間の状態や行動の分析を行うため，情報量が減ることは極力抑えたい．そのため，歩行データ圧縮において，情報量の減少を抑えるという課題が存在する．

そこで，我々は送信するデータの情報量減少をできる限り抑えつつ，データ量を減少する手法として，Always-on Karteに対して歩行特性を考慮したデータ圧縮方式を適用する．

3.2 予備実験

我々は，歩行時における足圧力データの特徴を掴むため，予備実験を行った．

3.2.1 実験環境

AoKスリッパを用い，数種類の歩き方で歩行圧力データを取得し，歩行データの特性を分析した．AoKスリッパを履き，サンプリング周期 50ms でセンシングを行うことで，両足の前後の足圧力を取得する．実験環境では，通常歩行，すり足歩行，前傾歩行の3種類の歩行状態のデータを取り，解析を行った．前傾歩行は，体が前に傾いた状態での歩行であり，すり足歩行とは，足を捻挫したときのような片足を引きずった歩行である．本実験においてすり足は，左足を引きずって歩いたデータを用いた．すり足歩行および前傾歩行は，転倒を誘発するような危険な歩行であるため，今回，実験に加えた．

3.2.2 実験結果と考察

予備実験における歩行時の圧力データから，通常歩行における重心変化は足後部から足前部という順序で現れ，1 歩の間には足が地につかず，圧力が加わらない時間が存在するという結果が得られた．すり足歩行においては，常に足全体に力がかかっており，重心の変化に規則性が見られない．特に足裏の圧力が強く，足が地についておらず圧力がかからない時間がほぼ存在しない．前傾歩行時には前部へ常に力がかかっているため，重心はほとんど変化がなかった．また，常に前部に力がかかり，圧力がかからない時間がほぼ存在しなかった．実験結果が示す通り，歩行とは，連続した重心の移動である．そこで，本研究では，これらの歩行時の重心情報を基にデータ圧縮を行うことでより圧縮効果を高めることを目指す．

3.3 データ圧縮方式の設計

本データ圧縮方式の基本的な動作概要を以下に示す．

1. Mote により圧力センサから足圧力データを取得する．
2. 取得した圧力データを基に 1 歩毎の最大圧力点を足の前後部の両方において算出する．その算出した各最大圧力点を基に 1 歩分の圧力データ群をグループ分類する．
3. 1歩分の圧力データ群を分類されたグループに基づき，データ圧縮を行う．

3.3.1 グループ分類方法

本節では，圧力データのグループ分類に関して述べる．このグループ分類結果に基づき，1 歩分の圧力データ群の圧縮を行う．図 5 にグループ分類方法のアルゴリズムを示す．

図 3 に通常歩行における重心圧力データを示す．これは，取得した片足における前後部の圧力データにおいて，前部の圧力データと後部の圧力データの差から算出したものである．縦軸において，正の方向に圧力がかかるほど，重心は前部へと傾いているということを示し，負の方向に圧力がかかるほど，重心は後部へ傾いているということを示す．前部に重心が傾いている際の最大圧力値と後部に重心が傾いている際の最大圧力値を見ると，通常歩行においては，図 3 に示した通り，前後に重心が大きく振れていることがわかる．この最大圧力値をパターン分類の指標として抽出することで，圧力データをパターン分類する．

正常歩行，すり足歩行，前傾歩行時に関しての 120 歩分の前後部の最大圧力データを，図 4 に示す．横軸が後部に重心が傾いている際の最大圧力値であり，縦軸が前部に重心が傾いている際の最大圧力値である．それぞれの歩行方法は図 4 のように分類される．この

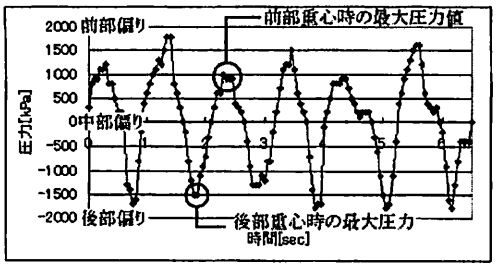

図3：通常歩行における重心データ

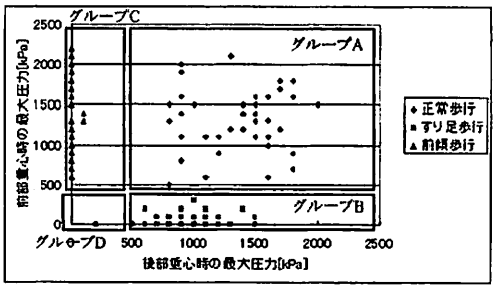

図4：前後の最大圧力に基づいた歩行分類

ように，足の重心の傾きを考慮した分類方法を用いてデータを分類し，分類されたそれぞれのデータに圧縮を適用する．

3.3.2 データ圧縮

前節でも述べた通り，歩行特性を考慮し，図4における4つのグループに分類を行う．分類したグループに基づいて1歩分のデータ群に対してデータ圧縮を行う．図6に圧縮のアルゴリズムを示す．

- グループA

グループAは，足の重心が前後に大きく振れているデータ群である．人間の歩行時の足圧力は，このように大きく重心が振れている場合「歩行中において踵の着地」および「踵の離床」，「足が地面をけり離れること」が行われ，前後に大きく圧力が加わっている．そのため，重心の変化点を得ることでグループAの歩行時の波形を把握できる．重心の変化点は，足の前後部の圧力データの差から算出される重心データの波形を用いて導出する．

そのことから，重心の変化点以外は，データにおいて冗長な部分とし考慮しないことで，データ量を減らし，圧縮を行う．

- グループB

グループBでは，前部にはほとんど重心がかからず，後部に主に圧力がかかっているデータ群である．この

```
Algorithm 1 Classification of Pressure Data on one step Algorithm.

1:procedure classification
2:  var
3:    frontData:array[dataNum] of integer; // front foot pressure data
4:    rearData:array[dataNum] of integer; // rear foot pressure data
5:    gravity:array[dataNum] of integer; // center of gravity data
6:    frontPoint:array[pointNum] of integer;
                          // maximum Center of Gravity point
7:    rearPoint:array[pointNum] of integer;
                          // minimum Center of Gravity point
8:  for i:=0 to dataNum-1 do gravity[i]:=frontData[i] - rearData[i];
9:  for i:=0 to pointNum-1
      do function frontPoint[i]:=extractFrontMax(gravity[i]);
         // extract maximum Front Foot Center of Gravity point
10: for i:=0 to pointNum-1
      do function rearPoint[i]:=extractRearMax(gravity[i]);
         // extract maximum Rear Foot Center of Gravity point
11: repeat
12:   if frontPoint[i] > threshold for classification
        and rearPoint[i] > threshold for classification then Group A;
13:   else if frontPoint[i] > threshold for classification
        and rearPoint[i] < threshold for classification then Group B;
14:   else if frontPoint[i] < threshold for classification
        and rearPoint[i] > threshold for classification then Group C;
15:   else if frontPoint[i] < threshold for classification
        and rearPoint[i] < threshold for classification then Group D;
16:   i:=i+1;
17: until i equals to pointNum-1
18: end procedure
```

図5：グループ分類方法のアルゴリズム

際の前部へ圧力がかからず，重心が前部へ移らない原因として2つのことが考えられる．1つ目は，前部の圧力が低いことである．2つ目は，前部および後部に同時に圧力がかかっていることにより，足の前後部の圧力差を算出した際に圧力値が小さくなったことである．このことから，圧縮の際に前部の圧力が低い要因がどちらであるかを検討する．検討結果に応じて，後部および前部の圧力波形の特徴を掴んだ圧縮を行う．

前部の圧力が弱い場合は，グループAの場合と同様に，重心の変化点を得て，圧縮を行う．しかし，前後部に同時に圧力がかかっている場合は，重心の変化点を取得するだけでは，前後部の圧力波形の概観を得ることができず，情報量が大きく減少してしまう．そのため，歩行の圧力データの情報量を極力減らさないよう前後部の圧力値を得る必要がある．そこで，足の前後部両方の圧力変化点を取得することで圧縮を行う．

- グループC

グループCは，後部にはほとんど重心がいかず，前部に主に圧力がかかっているデータ群である．グループBと同様に，圧縮の際に後部の圧力が低い要因となる事項を検討し圧縮を行う．検討結果に応じて，後部および前部の圧力波形を把握できるように圧縮を行う．

後部の圧力が小さい場合は，重心の変化点を取るこ

Algorithm 2 Data Compression in Group Algorithm.

```
1:procedure Group A
2:   for i:=0 to dataNum-1 do function calculate gravity();
3:   repeat // search changing point of gravity
4:      if this center of gravity value is changing point
          // compressed pressure data is changing point of gravity
          then compressedData:=gravity[i];
               timeStamp:=i;
5:             i:=i+1;
6:      until the number of repeat is more than compress point num;
7:end procedure

1:procedure Group B, C, D
2:   var
3:      selectAlgorithm:boolean; // select algorithm from 2 algorithm
4:   // case of rear foot pressure data or front pressure data is small.
      selectAlgorithm:=false;
5:   //if group is B
6:   if case of rear foot pressure data is large.
       then selectAlgorithm:=true;
7:   //if group is C
8:   if case of front foot pressure data is large.
       then selectAlgorithm:=true;
9:   //if group is D
10:  if case of rear foot pressure data and front pressure data is large.
       then selectAlgorithm:=true;
11:  if selectAlgorithm is false then // search changing point of gravity
       for i:=0 to dataNum-1 do function calculate gravity();
12:    repeat // search changing point of gravity
13:      if this center of gravity value is changing point
         // compressed pressure data is changing point of gravity
           then compressedData:=gravity[i];
                timeStamp:=i;
14:             i:=i+1;
15:     until the number of repeat is more than compress point num;
16:  else if selectAlgorithm is true then
       // search changing point of front data and rear data
18:    repeat
19:      if this front foot presure value is changing point
           then compressedData of front foot:=front Data[i];
                timeStamp of front foot:=i;
22:      if this rear foot presure value is changing point
           then compressedData of rear foot:=rear Data[i];
                timeStamp of rear foot:=i;
24:             i:=i+1;
25:     until the number of repeat is more than compress point num;
26: end procedure
```

図 6：データ圧縮形式のアルゴリズム

とで圧縮を行う．足の前後部に同時に圧力がかかることで圧力値が小さい場合は，足の前後部における圧力変化点をそれぞれ取得することで圧縮を行う．

- グループ D

グループ D では，前後部ともに圧力がかかっていないデータ群である．そのため，圧縮の際に前後部の圧力が低い要因となる事項を検討し，圧縮を行う．前部および後部の圧力が小さい場合は，重心の変化点を得る．前部と後部が同時期に圧力がかかることで圧力値が小さい場合は，足の前後両方の圧力波形の変化点をそれぞれ取得することで圧縮を行う．

3.3.3 データ形式

図 7 および図 8 に本圧縮方式における入力データの

Front Foot Pressure Data (16 byte)	*Rear Foot Pressure Data (16 byte)*	*Time Stamp (8 byte)*

図 7：入力データ形式

Compressed Data (16 byte)	*Time Stamp (8 byte)*

(a) 重心の変化点を得て，圧縮を行った場合のデータ形式

Compressed Data of Front foot part (16 byte)	*Compressed Data of Rear foot part (16 byte)*	*Time Stamp (8 byte)*

(b) グループ B,C,D における前後部に同時期に圧力がかかっている場合のデータ形式

図 8：出力データ形式

形式と出力データの形式を示す．Mote 特有のヘッダ部分などは省略し，重要となるデータ部分のみを示す．

4. 性能評価実験

我々は，データ圧縮方式を AoK スリッパ上の MicaZ Mote に実装した．本章では，データ圧縮方式を適用した AoK スリッパによる性能評価実験に関して述べる．図 9 に本実験で用いた AoK スリッパおよびシステムのプロトタイプを示す．

実験手順は，最初に被験者に AoK スリッパを左右の足に履いてもらい，その状態で被験者が歩行している間の 4 点の圧力値を 0.1 秒間隔に取得する．その圧力値を Mote 上で圧縮し，その圧縮データを AoK_Server へ送る．AoK_Server 内で圧縮データを復号し，生データと比較を行う．本実験では，8 人の被験者の圧力データを取得し，評価を行った．性能評価として，データの圧縮率と生データとの誤差に関して述べる．

4.1 実験結果と考察

本節では，実験を行った結果に対して，圧縮率と生データとの誤差に関して評価を行う．

- 圧縮率

図 10 に，圧縮を行わずに生データを送信した場合および圧縮を行い送信した場合における歩数毎の平均データ量を示す．歩数毎のデータ量はすべての場合において圧縮を行った場合の方が少ない．また，本実験で取得した全データを圧縮した結果，圧縮率は約 26%となった．1 歩毎にデータを圧縮した場合においても，約 36%の圧縮率となり，少ないデータでも十分な圧縮率となった．

- 生データとの誤差率

本圧縮方式により圧縮したデータを復号した波形と

図 9：システム構成

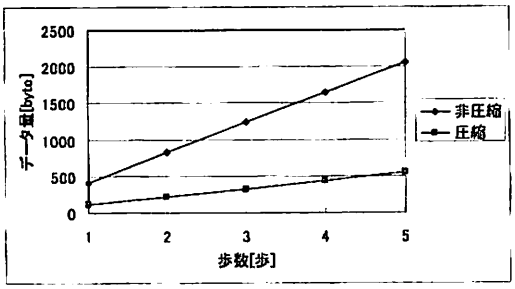

図 10：歩数毎の平均データ量

元データによる波形との誤差の評価を行った．誤差の評価として以下の式を用いた．

$$\text{誤差率}E = \sum_{i=0}^{n-1}\left|c(i)-r(i)\right| \Big/ \sum_{i=0}^{n-1} r(i)$$

$c(i)$：i番目の圧縮後データ
$r(i)$：i番目の圧縮前データ

本実験で取得した全データを評価した結果，誤差率は約 24.5%となった．また，1 歩毎の誤差率は，平均して約 24%となり，歩行時の波形を把握するという点では問題ないと考えられる．

5. 関連研究

IT シューズ[6]では，靴に足圧力センサを取り付け，階段や坂道の上り下りを検知することや，足の各部にかかる圧力を測定することで，転倒を検知することができる．また，GPS を内蔵し，高齢者の徘徊といった問題にも臨んでいる．

Oulu 大学[2]では，EMFi(ElectroMechanical Film)を部屋の床に設置し，人の歩行データを測定するシステムを提案している．このシステムでは，隠れマルコフモデルの拡張である SSMM(Segmental Semi-Markov Model)を用いて歩行時の圧力パターンをモデル化することで，人のトラッキングと認証を行うことができる．

Niki+ iPod[4]では，走った時間や走った距離，消費カロリ，走っているペースを iPod の画面で確認することができる．Nike+シューズに無線センサを入れることで，iPod へセンサから情報が送信される．受信機を取り付けた iPod では，その情報を取得し，表示を行う．

Adidas 1[1]は，スニーカーにマイクロコンピュータ，センサが埋め込まれており，使用者が歩くことで圧力データを取得する．その情報をマイクロコンピュータに送る．そして，疲労のレベル，地形，ペース，使用者の大きさから使用者への影響を計算する．その計算を基に，かかとのクッションを調節し，走りやすい環境を作る．データのサンプリングは毎秒数百回と取得を行う．

本研究では，生データのみの取得ではなく，情報量をより減らさずに生データに近い状態に復元できるように圧縮を行う．

6. まとめと今後の課題

本研究では，Always-on Karte における AoK スリッパへデータ圧縮方式を適用することで，データ量の低減および省電力化を行った．また，本圧縮方式を MicaZ Mote 上に実装し評価を行い，十分な圧縮率および問題が無い範囲での誤差率であることを確認し，有用性を示した．今後，年代などを変えてより多くの人数による評価を行っていきたいと考えている．

参考文献

[1] Adidas1, http://www.personal.psu.edu/users/z/l/zlr102/
[2] Koho K, Suutala J, Seppanen T & Roning J., "Footstep pattern matching from pressure signals using segmental semi-Markov models," 12th European Signal Processing Conference (EUSIPCO 2004), September 6-10, 2004, Vienna, Austria.
[3] MicaZ, http://www.xbow.com/products/productsdetails.aspx?sid=101
[4] Nike+ ipod, http://www.apple.com/ipod/nike/
[5] Uehara, Y., Uchiyama, T., Mori, M., Saito, H., and Tobe, Y., "Always-on Karte: A System for Elderly People's Healthcare Using Wireless Sensors," 3rd International Conference on Networked Sensing Systems (INSS 2006), Chicago, May 2006
[6] 板生淸，保坂寛，佐々木健，山内規義，矢作直樹，高橋龍太郎，田島孝，嶌田聡，塩手良知，加納史朗，佐藤光，漆原育子，浅井直樹，佐藤明男，"ウェアラブルセンサを用いた健康情報システム，" 情報処理進行事業協会 2002 年度成果報告(2003)
[7] 高齢者白書，http://www8.cao.go.jp/kourei/whitepaper/w-2005/zenbun/17index.html