人工知能学会 インタラクティブ
情報アクセスと可視化マイニング研究会(第18回)
SIG-AM-18-02

多次元軌跡データに対する類似部分軌跡検索の高速化

Fast Similarity Search of Subtrajectories in Multidimensional Trajectory Databases

岡部 臨　　浦本 明　　尾崎 知伸 *
Nozomu Okabe　Akira Uramoto　Tomonobu Ozaki

日本大学 文理学部
College of Humanities of Sciences, Nihon University

Abstract: In the analysis of team sports, searching for simultaneous movements by a plurality of players specified by the user is one of basic operations. In this paper, we formulate this operation as a similarity search problem of subtrajectories in multidimensional trajectory data, and develop fast algorithms to solve the problem. The proposed algorithms are evaluated from the viewpoint of computation time and quality of obtained subtrajectories using real trajectory datasets on nine matches in Japanese professional football league.

1　はじめに

チームスポーツの分析において，利用者がクエリとして与えるフィールド上での動き（移動軌跡）と類似する場面を検索すること，すなわち移動軌跡の類似検索は基本的な操作の一つであり，近年盛んに研究が行われている．例えば Sha らは，バスケットボールにおけるパス・ショットなどの意味を含めた部分移動軌跡の検索を対象に，ハッシュ技術を用いた高速検索手法を提案している [1]．また Ohashi らは，動的属性を考慮した移動軌跡の類似検索手法を提案している [2]．動的属性とは，移動速度や周囲との位置関係など，時間経過に合わせて変化する（位置座標そのもの以外の）属性を表す．動的属性を考慮することで，より柔軟な類似検索を実現している．

既存研究の多くは，一つの移動軌跡をクエリとすることを前提としているが，チームスポーツにおける類似検索を考えた場合，複数人の移動軌跡から一人の移動軌跡を検索するだけでは，必ずしも十分であるとは言えない．なぜなら，チームスポーツにおいては，選手個人個人の動きだけではなく，しばしば複数人の選手が絡む連携プレーが重要となるからである．この問題を解決し，複数人の連携する動きを検索するには，クエリとして複数人の移動軌跡を与えると同時に，クエリに合致するプレイヤの組み合わせを考慮することが必要となる．

本論文ではこの問題を，多次元移動軌跡データに対する類似部分移動軌跡検索問題，すなわち長大な N 次元移動軌跡データからクエリである M 次元の移動軌跡と類似する部分を抽出する問題として定式化し，その高速アルゴリズムを提案する．また，サッカーにおける選手の移動軌跡データを対象に，計算時間と精度の面から提案手法を評価する．

*連絡先：日本大学文理学部情報科学科
〒 156-8550 東京都世田谷区桜上水 3-25-40
E-mail: tozaki@chs.nihon-u.ac.jp

2　準備

本章では準備として，移動軌跡に関する記法や定義を導入し，本論文で扱う問題を形式的に示す．

オブジェクト o に関する移動軌跡 tr^o を，時刻 t における o の位置座標 (x_t^o, y_t^o) の系列

$$tr^o = (x_1^o, y_1^o), (x_2^o, y_2^o), \cdots, (x_t^o, y_t^o), \cdots, (x_T^o, y_T^o)$$

と定義する．また系列長 T を tr^o の長さと呼ぶ．なお本論文では，時刻 t は 1 から始まり T まで連続していることを仮定する．移動軌跡データ tr^o における時刻 i から $j\,(1 \leq i \leq j \leq T)$ の連続する部分移動軌跡を

$$tr_{i,j}^o = (x_i^o, y_i^o), \cdots, (x_j^o, y_j^o)$$

と表記する．

長さ L の移動軌跡を要素とするサイズ M の配列

$$Q = [tr^{q_1}, tr^{q_2}, \cdots, tr^{q_M}]$$

をクエリと呼ぶ．また配列 Q の第 i 要素を $Q[i]$ と表記する．一方，$N\,(N \geq M)$ 個のオブジェクト $d_1, d_2, \cdots, d_N$

に関する長さ $T\,(T >> L)$ の移動軌跡データの集合

$$D = \left\{ \begin{array}{rcl} tr^{d_1} & = & (x_1^{d_1}, y_1^{d_1}), \cdots, (x_T^{d_1}, y_T^{d_1}), \\ & \vdots & \\ tr^{d_N} & = & (x_1^{d_N}, y_1^{d_N}), \cdots, (x_T^{d_N}, y_T^{d_N}) \end{array} \right\}$$

を移動軌跡データベースと呼ぶ．移動軌跡データベース D において，時刻 i から長さ L，すなわち時刻 i から $i+L-1$ までの部分移動軌跡の集合を

$$D_{(i,L)} = \left\{ tr_{i,i+L-1}^{d} \mid tr^d \in TR \right\}$$

と表記する．また，$d_1 \sim d_N$ から M 個の異なる要素を選択し並べることで得られる配列を

$$\pi = [\, d_{x_1}, d_{x_2}, \cdots, d_{x_M} \,]$$

とし，π の各要素 $\pi[j]$ に関する $D_{(i,L)}$ 中の部分移動軌跡を並べた配列を

$$D_{(i,L)}^{\pi} = [\, tr_{i,i+L-1}^{\pi[1]}, \cdots, tr_{i,i+L-1}^{\pi[M]} \,]$$

と表記する．

長さ L の部分移動軌跡の配列である $D_{(i,L)}^{\pi}$ に対し，x 軸 y 軸それぞれの平均値と標準偏差

$$\begin{array}{l} \overline{X} = \frac{1}{L \times M} \sum_{i \le t \le i+L-1, 1 \le k \le M} x_t^{\pi[k]} \\ \overline{Y} = \frac{1}{L \times M} \sum_{i \le t \le i+L-1, 1 \le k \le M} y_t^{\pi[k]} \\ \sigma(X) = \sqrt{\frac{1}{L \times M} \sum_{i \le t \le i+L-1, 1 \le k \le M} (x_t^{\pi[k]} - \overline{X})^2} \\ \sigma(Y) = \sqrt{\frac{1}{L \times M} \sum_{i \le t \le i+L-1, 1 \le k \le M} (y_t^{\pi[k]} - \overline{Y})^2} \end{array}$$

を用いて，各座標位置を平均 0，分散 1 に標準化することで得られる正規化済み移動軌跡の配列を

$$n\_D_{(i,L)}^{\pi} = [\, n\_tr_{i,i+L-1}^{\pi[1]}, \cdots, n\_tr_{i,i+L-1}^{\pi[M]} \,]$$

where

$$\begin{array}{l} n\_tr_{i,i+L-1}^{d} = \left(\left(x_i^d - \overline{X} \right) / \sigma(X), \left(y_i^d - \overline{Y} \right) / \sigma(Y) \right), \\ \cdots, \left(\left(x_{i+L-1}^d - \overline{X} \right) / \sigma(X), \left(y_{i+L-1}^d - \overline{Y} \right) / \sigma(Y) \right) \end{array}$$

と表記する．同様に，移動軌跡の配列であるクエリ Q に対する正規化済み移動軌跡の配列を n_Q と表記する．

正規化済みの移動軌跡データ配列 $n\_D_{(i,L)}^{\pi}$ とクエリ配列 n_Q との非類似度（距離）を，

$$dist(\, n\_D_{(i,L)}^{\pi},\, n\_Q \,) = \sum_{j=1}^{M} dtw(\, n\_D_{(i,L)}^{\pi}[j],\, n\_Q[j] \,)$$

と定義する．ここで dtw は動的時間伸縮法 [3] に基づく距離（DTW 距離）であり，2 つの移動軌跡

$$\begin{array}{l} tr^d = (x_1^d, y_1^d), \cdots, (x_{L_d}^d, y_{L_d}^d) \\ tr^q = (x_1^q, y_1^q), \cdots, (x_{L_q}^q, y_{L_q}^q) \end{array}$$

に対し，

$$dtw(\, tr^d,\, tr^q \,) = \gamma(\, L_d,\, L_q \,)$$

where

$$\gamma(i,j) = \left\{ \begin{array}{ll} 0 & i = 0, j = 0 \\ \infty & i \neq 0, j = 0 \\ \infty & i = 0, j \neq 0 \\ d(i,j) + \min \begin{pmatrix} \gamma(i-1,j) \\ \gamma(i-1,j-1) \\ \gamma(i,j-1) \end{pmatrix} & \text{otherwise} \end{array} \right.$$

$$d(i,j) = \sqrt{(x_i^d - x_j^q)^2 + (y_i^d - y_j^q)^2}$$

と定義される．

本研究では，移動軌跡データベースに対する類似度上位 K 検索を考える．すなわち，長さ L の移動軌跡を保持する要素数 M のクエリ配列 Q と，$N\,(N \geq M)$ 個の各オブジェクトに関する長さ $T\,(T >> L)$ の移動軌跡の集合 D に対し，$dist(n\_D_{(i,L)}^{\pi}, n\_Q)$ の値が上位 K 番目以内である部分移動軌跡 $D_{(i,L)}^{\pi}$ を求める問題を対象とする．より形式的には，$K\,(K > 0)$ をパラメタとし，条件

$$\left| \{ (i', \pi') \mid dist(n\_D_{(i,L)}^{\pi}, n\_Q) > dist(n\_D_{(i',L)}^{\pi'}, n\_Q) \} \right| < K$$

を満たす $D_{(i,L)}^{\pi}$ を獲得する．

原理的には，移動軌跡データベースに対し，時刻 1 から $T-L+1$ までの各時刻から長さ L の部分移動軌跡を取り出し，さらに N 個あるオブジェクトから異なる M 個を選択した上で，実際に動的時間伸縮法を用いて距離を計算する，という操作により，この問題の解を得ることが可能である．しかしこの場合，一回の動的時間伸縮法の計算量は 2 つの系列の長さの積となるので，全体として計算量は

$$\mathcal{O}\left((T - L + 1) \times {}_N P_M \times L^2 \right)$$

となり，計算コストが非常に大きいことが問題となる．

3 最類似部分時系列の検索

これまでに，1 次元の時系列データを対象としたクエリに対する高速な類似検索手法として，動的時間伸縮法に基づくストリーム検索アルゴリズム The UCR suite[4] [1] が提案されている．本研究では，このアルゴリズムを出発点とし，多次元移動軌跡データへの拡張を行う．

[1] http://www.cs.ucr.edu/~eamonn/UCRsuite.html

SimilaritySearch(D, Q)

$\langle bsf, loc\rangle := \langle\infty, -1\rangle$
$n\_Q :=$ normalize(Q)
for i in $\{1, \cdots, T-L+1\}$
  $n\_D_{(i,L)} :=$ normalize($D_{(i,L)}$)
  if $lb_{kim}LF(n\_D_{(i,L)}, n\_Q) > bsf \vee$
    $lb_{keogh}EQ(n\_D_{(i,L)}, n\_Q) > bsf \vee$
    $lb_{keogh}EC(n\_D_{(i,L)}, n\_Q) > bsf$
    then continue
  $dist := dtw(n\_D_{(i,L)}, n\_Q)$
  if $dist < bst$ **then** $\langle bsf, loc\rangle := \langle dist, i\rangle$
end for
return $D_{(loc,L)}$

図 1: The UCR suite の疑似コード

The UCR suite は，ストリームアルゴリズムの 1 種であり，大きく (1) 打ち切りを伴う標準化，(2) 下界計算による枝刈り，(3) 打ち切りを伴う DTW 計算から構成される．

検索対象である長さ T の時系列データ D とクエリである長さ L の時系列データ Q を対象とした，上位 1 位検索に対する The UCR suite の疑似コードを図 1 に示す．疑似コード中において，normalize は正規化を表す．また $lb_{Kim}LF$ は，$\mathcal{O}(1)$ で計算可能な DTW 距離の下界 [5]，$lb_{Keogh}EQ$ と $lb_{Keogh}EC$ は $\mathcal{O}(L)$ で計算可能な DTW 距離の下界 [6, 7] をそれぞれ表す．疑似コードが示す通り，開始位置を 1 から $T-L+1$ まで変化させながら，そのそれぞれでデータ $D_{(i,L)}$ の正規化と 3 種の下界計算及び DTW による距離計算を行い，最も類似する部分系列を特定する．このとき，複数の下界を計算量の小さい順に計算することで，少ない計算時間での枝刈り（DTW 計算の回避）を実現している．なお本疑似コードは概念レベルでの動作を表すものであり，実際のアルゴリズムではストリーム化及び下界計算を考慮したデータの部分的な正規化など，更なる最適化手法が利用されている．より詳細なアルゴリズムは原著論文 [4] を参照されたい．

4 類似部分移動軌跡の検索

本論文で対象とする，長さ T の N 次元移動軌跡データ D に対する長さ L の M 次元クエリ Q の類似部分移動軌跡検索問題では，距離 $dist(\, n\_D^{\pi},\, n\_Q\,)$ の計算が必要とされる [2]．距離 $dist(\, n\_D^{\pi},\, n\_Q\,)$ が DTW 距離の合計であることに着目し，The UCR suite に対して以下の 3 点の拡張を行うことで，多次元移動軌跡データに対する類似部分移動軌跡検索アルゴリズムを開発する．

1. 各 DTW 距離 $dtw(\, n\_D^{\pi}[j],\, n\_Q[j]\,)$ とその下界の計算を（1 次元の）時系列から（2 次元の）移動軌跡へと変更する
2. DTW 距離の合計である $dist(\, n\_D^{\pi},\, n\_Q\,)$ に対する下界を導入する
3. N 次元の移動軌跡データ D から，異なる M 個を選択する順列 π をすべて考慮する

以下では，上記の拡張に基づく基本アルゴリズム及びヒューリスティクスに基づく高速化アルゴリズムを導入する．

4.1 基本アルゴリズム

本論文における The UCR suite の拡張の一つである，時系列データから移動軌跡データへの変更に関しては自明な点も多く，また下界 lb_{Keogh} に関してはその多次元化が既に報告されている [8]．従って，DTW 距離及びその下界計算に関しては，時系列データと移動軌跡データとで本質的な変更の必要はなく，本論文でも特に表記を分けることは行わない．

一方，$dist(\, n\_D^{\pi},\, n\_Q\,)$ が各要素における DTW 距離の合計であることと，不等式

$$0 \le lb_{Kim}LF \le \max(\, lb_{Keogh}EQ, LB_{Keogh}EC\,) \le dtw$$

が成り立つことより，LB を DTW 距離の下界とし，また $S = \{1, 2, \cdots, M\}$，$S \supseteq S' \supseteq S''$ としたとき，

$$dist(S', n\_D^{\pi}, n\_Q) = \sum_{j \in S'} dtw(\, n\_D^{\pi}[j],\, n\_Q[j]\,)$$

$$dist(S', S'', n\_D^{\pi}, n\_Q) = \sum_{j \in S''} dtw(\, n\_D^{\pi}[j],\, n\_Q[j]\,) + \sum_{j \in S' \setminus S''} LB(\, n\_D^{\pi}[j],\, n\_Q[j]\,)$$

に対して，不等式

$$\begin{aligned} dist(n\_D^{\pi}, n\_Q) &\ge dist(S', n\_D^{\pi}, n\_Q) \\ &\ge dist(S', S'', n\_D^{\pi}, n\_Q) \end{aligned}$$

が成り立つ．この不等式より，$dist(S', n\_D^{\pi}, n\_Q)$ と $dist(S', S'', n\_D^{\pi}, n\_Q)$ は，$dist(n\_D^{\pi}, n\_Q)$ に対する下界を与えることとなるので，両者を $dist(n\_D^{\pi}, n\_Q)$ 計算に対する枝刈り基準として利用する．

この枝刈り基準を利用し，クエリの各要素に対して下界の計算と DTW 距離の計算を繰り返す上位 1 位類似部分移動軌跡検索のための基本アルゴリズムを図 2

[2] 簡略化のため以降ではデータ n_D^{π} に関する添字 (i, L) を省略する．

Algorithm1(D, Q)

1: $\langle bsf, loc, S\rangle := \langle \infty, -1, \emptyset\rangle$
2: $n\_Q :=$ normalize(Q)
3: **for** i in $\{1, \cdots, T-L+1\}$
4: **for** π in $Perm(D, Q)$
5: $n\_D^{\pi}_{(i,L)} :=$ normalize$(D^{\pi}_{(i,L)})$, $dist := 0$
6: **for** j in $\{1, \cdots, M\}$
7: **if** $dist + lb_{kim}LF(n\_D^{\pi}_{(i,L)}[j], n\_Q[j]) > bsf \vee$
8: $dist + lb_{keogh}EQ(n\_D^{\pi}_{(i,L)}[j], n\_Q[j]) > bsf \vee$
9: $dist + lb_{keogh}EC(n\_D^{\pi}_{(i,L)}[j], n\_Q[j]) > bsf$
10: $dist + dtw(n\_D^{\pi}_{(i,L)}[j], n\_Q[j]) > bsf$
11: **then** $dist := \infty$, **break**
12: $dist := dist + dtw(n\_D^{\pi}_{(i,L)}[j], n\_Q[j])$
13: **end for**
14: **if** $dist < bst$ **then** $\langle bsf, loc, S\rangle := \langle dist, i, \pi\rangle$
15: **end for**
16: **end for**
17: **return** $D^{S}_{(loc,L)}$

図 2: 類似部分移動軌跡検索の基本アルゴリズム 1

に示す．なお図中において $Perm(D, Q)$ は，クエリ Q の要素数である M 個の異なる要素をデータ D から選択する順列の全体集合である．また，アルゴリズム中で保持する解を上位 K 件とすることで，同様の方法により上位 K 位検索を実現することが可能である．

基本アルゴリズム（図 2）では，クエリの要素毎に下界の計算と DTW 距離の計算を繰り返す．しかし，計算の順序を変え，計算量の小さい下界の計算を優先的に行うことで，全体の計算時間を短縮することが期待できる．この考えに基づき，クエリの全要素に対して下界の計算を行った後に DTW 計算を行うアルゴリズムを図 3 に示す．

4.2 ヒューリスティクスの導入

本節では，チームスポーツへの応用の観点から，高速化のための 2 つのヒューリスティクスを導入する．

一つ目のヒューリスティクスは「似たようなプレイはしばしば似たような場所で起こる」という考えに基づき，考慮する順列 π の順序（図 2, 図 3 の 4 行目）を制御するというものである．具体的には,（正規化前の）データ $D^{\pi}_{(i,L)}$ とクエリとの重心を計算し，その距離の昇順にデータの組み合わせ π を考慮する．なお正規化済みデータ $n\_D^{\pi}_{(i,L)}$ を求めるためには $D^{\pi}_{(i,L)}$ の重心計算が必要となるため，このヒューリスティクスを利用する場合に係る追加コストは，順列集合 $P(D, Q)$ 内の要素を並べ替えるコストだけである．

Algorithm2(D, Q)

1: $\langle bsf, loc, S\rangle := \langle \infty, -1, \emptyset\rangle$
2: $n\_Q :=$ normalize(Q)
3: **for** i in $\{1, \cdots, T-L+1\}$
4: **for** π in $Perm(D, Q)$
5: $n\_D^{\pi}_{(i,L)} :=$ normalize$(D^{\pi}_{(i,L)})$, $dist := 0$
6: **for** j in $\{1, \cdots, M\}$
7: $dist := dist + lb_{kim}LF(n\_D^{\pi}_{(i,L)}[j], n\_Q[j])$
8: **if** $dist > bsf$ **then** $dist := \infty$, **goto** 22
9: **end for**
10: **for** j in $\{1, \cdots, M\}$
11: $dist := dist - lb_{kim}LF(n\_D^{\pi}_{(i,L)}[j], n\_Q[j])$
$+lb_{keogh}EQ(n\_D^{\pi}_{(i,L)}[j], n\_Q[j])$
12: **if** $dist > bsf$ **then** $dist := \infty$, **goto** 22
13: **end for**
14: **for** j in $\{1, \cdots, M\}$
15: $dist := dist - lb_{keogh}EQ(n\_D^{\pi}_{(i,L)}[j], n\_Q[j])$
$+lb_{keogh}EC(n\_D^{\pi}_{(i,L)}[j], n\_Q[j])$
16: **if** $dist > bsf$ **then** $dist := \infty$, **goto** 22
17: **end for**
18: **for** j in $\{1, \cdots, M\}$
19: $dist := dist - lb_{keogh}EC(n\_D^{\pi}_{(i,L)}[j], n\_Q[j])$
$+dtw(n\_D^{\pi}_{(i,L)}[j], n\_Q[j])$
20: **if** $dist > bsf$ **then** $dist := \infty$, **goto** 22
21: **end for**
22: **if** $dist < bst$ **then** $\langle bsf, loc, S\rangle := \langle dist, i, \pi\rangle$
23: **end for**
24: **end for**
25: **return** $D^{S}_{(loc,L)}$

図 3: 類似部分移動軌跡検索の基本アルゴリズム 2

二つ目のヒューリスティクスは「大きさがクエリと極端に異なる検索結果は，実用上有益ではない」との考えに基づき，クエリと大きさが大幅に異なるデータ $D^{\pi}_{(i,L)}$ を検索対象から外すというものである．具体的には，クエリ Q に対する幅 W_Q と高さ H_Q 及び $D^{\pi}_{(i,L)}$ に対する幅 W_D と高さ H_D，すなわち

$$
\begin{aligned}
W_Q &= \max_{1\le k\le L, 1\le j\le M}(x_k^{q_j}) - \min_{1\le k\le L, 1\le j\le M}(x_k^{q_j}) \\
H_Q &= \max_{1\le k\le L, 1\le j\le M}(y_k^{q_j}) - \min_{1\le k\le L, 1\le j\le M}(y_k^{q_j}) \\
W_D &= \max_{i\le k\le i+L-1, 1\le j\le M}(x_k^{\pi[j]}) - \min_{i\le k\le i+L-1, 1\le j\le M}(x_k^{\pi[j]}) \\
W_Q &= \max_{i\le k\le i+L-1, 1\le j\le M}(y_k^{\pi[j]}) - \min_{i\le k\le i+L-1, 1\le j\le M}(y_k^{\pi[j]})
\end{aligned}
$$

に対し，$\alpha\,(0 \le \alpha \le 1)$ をパラメタとし，条件

$$
\begin{aligned}
(1-\alpha)W_Q \le W_D \le (1+\alpha)W_Q, \\
(1-\alpha)H_Q \le H_D \le (1+\alpha)H_Q
\end{aligned}
$$

を満たさない $D^{\pi}_{(i,L)}$ を計算対象から除外する．

表 1: 上位 10 位検索の計算時間（秒）

$Q_{M,L}$	A_1	A_2	A_2^G	A_2^R	A_2^{GR}
$Q_{2,2}$	746	215	214	127	94
$Q_{2,3}$	5994	1011	1014	674	237
$Q_{3,2}$	–	1420	1753	1894	1821
$Q_{3,3}$	–	3584	3963	3974	2627

表 2: 検索精度（K-適合率のマイクロ平均）

K	C_1	C_2	C_3	C_4
10	0/100	2/100	0/100	1/100
20	0/200	2/200	0/200	1/100
30	1/300	2/300	0/300	1/100
40	3/400	2/400	0/400	2/400
50	3/500	2/500	2/500	2/500

5 評価実験

提案手法を評価するため，Java 言語を用いて各アルゴリズムを実装し，Windows PC（OS:Windows7 Pro, CPU:Intel Core-i3 2.40GHz, 主記憶:4GB）を用いて評価実験を行った．また実験には，J リーグのリーグ戦に関するデータ [3]（移動軌跡のサンプリングレートは 1 秒間 25 回）を利用した．

5.1 実行速度の評価

次元数（人数）M 人，長さ L 秒のクエリ $Q_{M,L}$ を用いて 5 試合を対象にそれぞれ上位 10 位検索を行い，その検索時間の平均値を計測した．実行結果を 表 1 に示す．表中において，A_1，A_2 はそれぞれ基本アルゴリズム 1（図 2）と 2（図 3）を表す．また，A_2 の上付き文字 G と R は，それぞれ重心 (center of gravity) によるヒューリスティクスと範囲 (range) によるヒューリスティクス（パラメタ $\alpha = 0.3$）を適用したことを表す．なお，“–” はタイムアウト（4 時間）を表す．

実験結果より，A_1 と比較し A_2 の方が高速であることが分かる．これは，計算量の少ない下界計算を優先させた効果によるものであると考えられる．また，A_2 と重心に基づく高速化を加えた A_2^G を比較すると，$Q_{2,x}$ ではほぼ同じ，$Q_{3,x}$ では A_2 の方が高速であり，必ずしも導入したヒューリスティクスの効果が得られているわけではないことが分かる．その一方で，範囲に基づく高速化を用いた A_2^R と，2 つの高速化手法を用いた A_2^{GR} を比較すると，A_2^{GR} の方が高速であり，組み合わせることにより，重心を考慮するの効果が得られることが分かる．

5.2 検索精度の評価

提案する枠組みにおける検索精度を検証するため，9 試合を対象に，サッカーにおける代表的なプレイ（オーバーラップやワンツーパスなど）を人手により抽出し，正解データセット C を作成した．各正解データセット C に対し，一つの要素 $q \in C$ をクエリとし，残りの $C \setminus \{q\}$ を正解集合とした場合の上位 K 位検索を 10 回繰り返し，K-適合率のマイクロ平均を算出した．

クエリの次元数を M とした場合，検索対象となる部分移動軌跡の総数はおよそ

$$25(Hz) \times 60(\text{秒}) \times 90(\text{分}) \times {}_{22(\text{人})}P_M \times 9(\text{試合})$$

と非常に大きくなるので，今回の実験では正解判定を緩く設定し，(1) 選手の組み合わせ π が一致し，(2) 時間に重なりがある場合に正解であると判定する．具体的には，条件 $i-L+1 \leq j \leq i+L-1$ を満たす $D^{\pi}_{(j,L)}$ が正解集合に含まれている場合，$D^{\pi}_{(i,L)}$ を正解と判断する．実験結果を表 2 に示す．表中において，C_1 はオーバーラップに関するプレイ（正解数 : 218），C_2 は前方からの守備に関するプレイ（正解数 : 114），C_3 は後方からの守備に関するプレイ（正解数 : 309），C_4 はワンツーパスに関するプレイ（正解数 : 132）であり，クエリの次元数（人数）はすべて 2 である．

実験結果より，必ずしも高い精度が得られたわけではなく，移動軌跡だけではなく，ボールの位置や方向，他プレイヤとの距離など，外部の動的要因を考慮する必要があることが伺える．

6 まとめと今後の課題

本論文では，チームスポーツへの応用を念頭に，多次元移動軌跡データベースに対する上位 K 部分移動軌跡検索問題を定式化するとともに，DTW 距離を用いた検索アルゴリズムを提案した．また，実際のサッカー選手の移動軌跡データを用いてその性能を評価した．

今後の課題としては，更なる高速化アルゴリズムの検討があげられる．また，動的属性 [2] の導入や時間差の考慮，選手の属性（チームやポジション）の反映なども重要な課題である．

参考文献

[1] L. Sha, P. Lucey, Y. Yue, P. Carr, C. Rohlf, and I. Matthews : Chalkboarding: A New Spatiotem-

[3] データスタジアム株式会社 (http://www.datastadium.co.jp) より

poral Query Paradigm for Sports Play Retrieval, *Proc. of the 21st International Conference on Intelligent User Interfaces*, pp.336-347 , 2016.

[2] H. Ohashi, T. Shimizu, and M. Yoshikawa : Flexible Similarity Search for Enriched Trajectories, *IEICE Transactions on Information and Systems* Vol. E100-D, No.9, pp.2081–2091, 2017.

[3] H. Sakoe and S. Chiba : Dynamic programming algorithm optimization for spoken word recognition, *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol.26, No.1, pp.43–49, 1978.

[4] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh : Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping, *Proc. of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.262-270, 2012.

[5] S. Kim, S Park, and W. Chu : An index-based approach for similarity search supporting time warping in large sequence databases, *Proc. of the 17th International Conference on Data Engineering*, pp. 607–614, 2001.

[6] A. Fu, E. Keogh, L. Lau, C. Ratanamahatana, and R. Wong : Scaling and time warping in time series querying, *The International Journal on Very Large Data Bases*, Vol.17, No.4, pp.899–921, 2008.

[7] E. Keogh, L. Wei, X. Xi, M. Vlachos, S.H. Lee, and P. Protopapas : Supporting exact indexing of arbitrarily rotated shapes and periodic time series under Euclidean and warping distance measures, *The International Journal on Very Large Data Bases*, Vol.18, No.3, pp.611–630 , 2009.

[8] T. M. Rath and R. Manmatha : Lower-Bounding of Dynamic Time Warping Distances for Multivariate Time Series, Technical Report MM-40, Center for Intelligent Information Retrieval, University of Massachusetts Amherst, 2002.