Constrained Motif Discovery in Time Series

Yasser Mohammad and Toyoaki Nishida

Graduate School of Informatics, Kyoto University, Japan

yasser@ii.ist.i.kyoto-u.ac.jp, nishida@i.kyoto-u.ac.jp

Received 25 May 2009

Abstract

The goal of motif discovery algorithms is to efficiently find unknown recurring patterns. In this paper we focus on motif discovery in time series. Most available algorithms cannot utilize domain knowledge in any way which results in quadratic or at least super-linear time and space complexity. In this paper we define the Constrained Motif Discovery problem which enables utilization of domain knowledge into the motif discovery process. The paper then provides two algorithms called MCFull and MCInc for efficiently solving the constrained motif discovery problem. We also show that most unconstrained motif discovery problems be converted into constrained ones using a change-point detection algorithm. A novel change-point detection algorithm called the Robust Singular Spectrum Transform (RSST) is then introduced and compared to traditional Singular Spectrum Transform using synthetic and real-world data sets. The results show that RSST achieves higher specificity and is more adequate for finding constraints to convert unconstrained motif discovery problems to constrained ones that can be solved using MCFull and MCInc. We then compare the combination of RSST and MCFull or MCInc with two state-of-the-art motif discovery algorithms on a large set of synthetic time series. The results show that the proposed algorithms provided four to ten folds increase in speed compared the unconstrained motif discovery algorithms studied without any loss of accuracy. RSST+MCFull is then used in a real world human-robot interaction experiment to enable the robot to learn free hand gestures, actions, and their associations by *watching* humans and other robots interacting.

Keywords Constrained Motif Discovery in Time Series, Change-Point Detection, Robust Singular Spectrum Transform, Mining Human Physiological Data, Human-Robot Interaction

§1 INTRODUCTION

The problem of unconstrained motif discovery from time series can be informally defined as efficiently finding unknown recurring patterns in long time series with no prior knowledge about their locations. The research in this problem have led to many techniques including the PROJECTIONS algorithm [1], PERUSE [2], Gemoda [3] among many others (e.g. Lin et al.[4],Minnen et al. [5], and Tang et al. [6]). With the exception of Gemoda which is quadratic in time and space complexities, these algorithms aim to achieve sub-quadratic time complexity by first looking for candidate motif *stems* using some heuristic method and then doing exhaustive motif detection instead of motif discovery which is linear in time.

The most used method for finding these stems is the PROJECTIONS algorithm [1] which requires discretization of the data using the SAX [7] algorithm. After discretization, a collision matrix is built that specifies the distance between every two subsequences in the time series. The size of this collision matrix is quadratic in the length of the time series. To avoid calculating the accurate distance between every two pairs of subsequences (as done in Gemoda [3]), random projections [1] is used to approximate this distance. The subsequences with high similarity are then used as the stems for subsequent motif detection. One common problem to all the algorithms based on the PROJECTIONS algorithm [1] is the need to construct and keep the collision matrix which is in general quadratic in the length of the SAX word describing the time series. Given that the optimal word size can be very short for short motifs in signals with high frequency components, the size of this matrix can grow quadratic with the length of the time series. In the two real world examples given in this paper (mining human physiological data in section 5.4, and mining hand and robot motions in section 7), the length of the time series involved was between 2×10^5 and 3×10^6 points with motifs as short as few hundred points.

Another common problem of most of the previously proposed techniques is the need to specify an exact or at least roughly correct motif length.

Catalano et al. [8] suggested a very efficient algorithm for locating variable length patterns in data series using random sampling that allows it to run in linear time and constant memory. The main problem of this approach is that it relies on random sampling which can lead to poor performance for long time series with infrequent embedded motifs including long records of human activities and physiological data which

are among our main targeted application areas.

All of the methods proposed for motif discovery that we are aware of assume no prior knowledge of the probable locations of the motifs which leads to this explosion in the processing time or space needed. This paper introduces the Constrained Motif Discovery problem in which we try to find motifs utilizing information about their probable locations in the data series. This leads to a linear time algorithm with constant space complexity. The introduction of constraints also allows interactive implementations which can utilize the human eye as the ultimate pattern recognizer to discover more *meaningful* motifs. Other forms of constraints will be considered in future research.

In the following section we provide the definition of the constrained motif discovery problems. In section 3, Catalano et al's [8] algorithm for solving the unconstrained motif discovery problem is introduced followed by two alternative modifications to it for solving constrained motif discovery problems. Section 4 provides a theoretical comparison between the accuracy of the proposed algorithms and the original Catalano et al.'s algorithm. Section 5 introduces the technique of using a change point discovery algorithm for calculating the constraint when no prior knowledge about the time series is available. Section 6 provides detailed evaluation of the proposed MCFull and MCInc algorithms on a large synthetic data set. Section 7 then provides a complete application of the MCFull and RSST algorithms in learning actions, commands and command-action associations during human-robot interactions. The paper is then concluded.

§2 Constrained Motif Discovery Problem

In this section, we introduce some definitions needed to describe the Constrained Motif Discovery problem.

Definition 2.1 (Time Series)
A time series of length n (X^n) is defined as an ordered set of values $x(t)$ where $1 \leq t \leq n$ and $x(t) \in U$. U is called the *domain* of X^n.

Definition 2.2 (Distance Function)
A distance function d over a domain U is a function that implements the transformation: $d\left({y_1}^{l_1}, {y_2}^{l_2}\right) : U^{l_1} \times U^{l_2} \rightarrow \Re^+$ for any two time series y^{l_1} and y^{l_2} defined over the domain U. The transformation function must be nonzero for different ${y_1}^{l_1}$ and ${y_2}^{l_2}$, reflective and transitive.

Definition 2.3 (Subsequence set)
Given a time series X^n of length n, the subsequence set of X of length range l_{min} : l_{max} ($X_{sub}^{l_{\min}:l_{\max}}$) is defined as the set of all unique time series x_{sub}^{l} that satisfy:

1. $l_{min} \leq l \leq l_{max}$
2. $x_{sub}^{l}(t) = x(t+i)$ for all $1 \leq t \leq l$ and some integer $i \geq 0$

Definition 2.4 (Trivial Match Set)
Given a subsequence x_{sub} of a time series X, the trivial match set of x_{sub} ($triv(x_{sub})$) is defined as the set of all time subsequences T_{sub} that intersects the subsequence x_{sub} in one or more points.

Definition 2.5 (Motif)
Given a time series X^n, and a distance function d and two length limits ($l_{\min}$ and $l_{\max}$), a motif M is defined as a set of time series called motif occurrences (m_i) that satisfy:

1. $m_i \in X_{sub}^{l_{\min}:l_{\max}}$
2. $\arg\min_x (d(m_i, x)) \in M, x \in \left\{x_{sub}^{l_{\min}:l_{\max}}\right\} - triv(m_i)$
3. $$d(m_i, m_j) < d(m_i, y)$$ where $m_j \in M, y \in X_{sub} - M - triv(m_i)$
4. For every $m_i \in M$; m_i is maximal. This means that m_i is not a subsequence of any larger motif occurrence satisfying the previous conditions.
5. There is no subsequence $x \notin M$ that can be added to M without breaking the previous conditions.

It can be easily shown that condition 2 of the *Motif* definition entails the following lemma:

Lemma 2.1
There is a real value T_M that satisfies:

1. $d(m_i, m_k) \leq \mathrm{T_M}$ for every two subsequences in a motif M
2. $d(m_i, x) > \mathrm{T_M}$ where $l_{min} \leq len(x) \leq l_{max}$ and $x \notin M$

Most available motif discovery algorithms assume some fixed value for T_M (usually called the *range*) and find the motifs that satisfy *Lemma 2* directly for this value. An exception is the algorithm by Minnen et al. [9] that automatically determines a sensible value for this range by finding the *knee in the curve* representing distances between motif stems and other motif occurrence candidates.

This definition of a motif is more general than previously suggested definitions because unlike the definition by Oates in [2] it assumes no specific model of the motif, and unlike the definitions by Jensen et al. in [3] it requires no hard coded range threshold T_M. Minnen [9] provided a similar formulation of the unconstrained motif discovery problem that casted it as a density estimation problem.

Definition 2.6 (Constrained Motif Discovery Problem)
Given a time series X^n, another time series P^n of the same length called a constraint (where $0.0 \leq p(t) \leq 1.0$), two real valued limits l_{min} and l_{max}, and a distance function d find all motifs ($M^{l_{min}:l_{max}}$) of length l where $l_{min} \leq l \leq l_{max}$ by utilizing P^n.

Semantically, the constraint $p(t)$ can be interpreted as specifying the probability that a motif occurrence ends at (or *near*) the time step t. Section 3 provides two alternative ways to utilize this constraint P^n to speed up the process of motif discovery in the time series X^n.

The unconstrained motif discovery problem solved by available motif discovery algorithms can be considered a special case of the constrained motif discovery problem where $p(t) = 1/n$ for all $1 \leq t \leq n$.

In the remaining of this paper we will assume without loss of generality that the domain of the time series (U) is the set of real numbers ($\Re$). A transitive version of Discrete Time Wrapping (DTW) will be used as the distance function.

§3 MC Algorithms

Catalano et al. [8] proposed an efficient algorithm for locating recurring motifs in time series that works in linear time and constant memory. The main idea behind the algorithm is to randomly sample a user specified number of large candidate windows of data, sub-window and store them, then these sub-windows are periodically compared to the sub-windows of comparison windows from the time series. The steps of the algorithm can be summarized as:

1. Select a subsequence s^w of length $w \geq l_{max}$.
2. Select w values randomly from X and concatenate them to form a noise sequence n^w.
3. Select a set of n_c comparison subsequences of length w from X ($\{{c^w}_i\}$).
4. Find the set $S^{\hat{w}}$ of subsequences of X of length $\hat{w}$ where $\hat{w} \leq l_{min}$ for s^w (call this $\left\{s_k^{\hat{w}}\right\}$), and n^w (call this $\left\{n_k^{\hat{w}}\right\}$) where $1 \leq k \leq w - \hat{w} - 1$. Then normalize all of the resulting subsequences to have unit mean square.
5. For the candidate subsequence of s^w ($s_k^{\hat{w}}$) do the following:

a. Randomly select $w - \hat{w} - 1$ subsequences from the set of all subsequences of the comparison windows (${c^w}_i$). Call this set the comparison set $\hat{c}_j^{\hat{w}}$.
b. Find the distances $d_{kj} = d\left(s_k^{\hat{w}}, {\hat{c}^{\hat{w}}}_j\right)$.
c. Group the set $\hat{c}_j^{\hat{w}}$ with their parent subsequence ${c^w}_i$ and for every group select $\hat{c}_j^{\hat{w}}$ that has the minimum distance d_{kj}. This leads to a set of R subsequences $\tilde{c}_r^{\hat{w}}$ where $R \leq n_c$.
d. Keep only the $\hat{R}$ subsequences of $\tilde{c}_r^{\hat{w}}$ with least d_{kr}.
e. Repeat the previous three steps for subsequences of the noise subsequence n^w. This leads to another set of $\hat{R}$ subsequences called $\tilde{n}_r^{\hat{w}}$.

6. Remove all candidate subsequences $s_k^{\hat{w}}$ that have average d_{kr} less than a threshold γ. To calculate γ, sort $\tilde{n}_r^{\hat{w}}$ by their distances and call these distances $dn\,(u)$. γ is selected as $dn\,(\lfloor(\alpha n)\rfloor)$ where α is a parameter of the algorithm. Repeat the steps above using this reduced set. α is a parameter of the algorithm. Repeat this reduction for n_r times.
7. If the final set $s_k^{\hat{w}}$ is not empty output each of them as a motif seed M_s after concatenating any continuous subset of them.

There are six parameters of this algorithm: w, $\hat{w}$, $\hat{R}$, n_r, n_c, α. Catalano et. al's paper [8] suggested that w should be selected larger than l_{max} (we select it as $l_{max} + 1$). They have also shown that a choice of α between 0.05 and 0.2 gives good results. n_r is decided based on the time available to run (the more the better). n_c is decided based on the space available. We select $\hat{w}$ to be less than $l_{min}/2$ in order to be sure that the smallest pattern has more than one subwindow. $\hat{R}$ is difficult to select but we used $\hat{R} = n_c/2$ which gave good results in all our evaluations. In case motif length limits l_{min} and l_{max} are not available, it is necessary to estimate them by visualizing parts of the data or trying different values for them.

One of the main advantages of this algorithm is that it can distinguish between patterns and noise by comparing the distribution of distances between a candidate subwindow and its best matches in the comparison set with its best matches in the noise set created by concatenating noncontiguous samples of the time series. This is a logical choice of the noise window when no information about the possible locations of the patterns (motif occurrences) is available as any long enough window will have some high probability of containing a motif occurrence if the motif was frequent in the data. The problem with this approach is that it is not clear how can this *randomly* selected set of values represent noise in the time series. Utilizing the constraint $p\,(t)$ it is possible to find a more meaningful *random window* by selecting and concatenating a set

of subsequences of some length less than $\hat{w}$ around the minimums of this constraint. This way the *random window* does not contain only random values selected from a distribution defined by the time series (as they are in the previous algorithm), but the subsequences that are relatively less likely to occur in interesting motifs according to the input constraint. One advantage of this approach is that for long time series with varying dynamics, it is possible to compile a random window locally by local minima around the candidate window and this random window will carry information about the *local* subsequences that are not likely to be found in real motifs rather than *global* information. This is the first modification we propose to Catalano et al.'s algorithm.

Assuming that the constraint $p(t)$ is specifying the probability that a motif occurrence ends *at* or *near* the time step t, we can also modify the first three steps of the original algorithm to speed up the motif discovery process. This is the second modification proposed to the algorithm.

These modifications result in two modified Catalano algorithms called the MCFull and MCInc algorithms as given below.

3.1 MCFull Algorithm

If calculating the constraint P is not computationally expensive or if it cannot be calculated for any required single point of the series X using only local information, we can speed up Catalano et al.'s algorithm [8] described in previous version by modifying the first three steps as follows:

1. Apply a Gaussian smoothing filter ($N(0, \sigma^2)$) to the original P constraint which results on the smoothed constraint $\tilde{P}$, then normalize $\tilde{P}$ so that $\sum_{t=1}^{n} \hat{p}(t) = 1$ and $0 \leq \hat{p}(t) \leq 1$.
2. Randomly select a subsequence s^w of length $w \geq l_{max}$ using $\hat{P}$ as the probability distribution.
3. Randomly select a set of w values from X by concatenating short subsequences of length $\hat{w}$ using $1 - \hat{P}^m$ where $m \geq 1$. This is the noise sequence n^w.
4. Randomly select a set of n_c comparison subsequences of X ($\{{c^w}_i\}$) each of length w using $\hat{P}$ as the probability distribution.

The rest of the algorithm goes exactly as the original algorithm. The smoothing step is required to account for any inaccuracy of the constraint. This way we understand the constraint as specifying the probability that a motif occurrence ends *near* and not

necessary *at* every time step. The only parameter added to the set of Catalano's parameters is the smoothing variance σ. If an estimation of the locality variance of the constraint is available (how accurate is it in finding the exact starting/ending point of motifs) then it can be used.

3.2 MCInc Algorithm

If calculating the constraint P is computationally expensive and more over if $p(t)$ can be calculated using only local information around t, then an incremental version of MCFull can be constructed (named MCInc hereafter) by modifying the MCFull steps as follows:

1. Randomly select a time step $1 \leq \tau \leq n$ using a uniform distribution.
2. Calculate $p(t)$ for $\tau-\delta \leq t \leq \tau+\delta$ and calculate $\hat{p}(\tau) = \frac{1}{2 \times \delta + 1} \sum_{t=\tau-\delta}^{\tau+\delta} p(t)$.
3. Repeat steps 1 and 2 as long as $\hat{p}(\tau) \leq th$.
4. Select the subsequence s^w of length $w \geq l_{max}$ that ends at τ.
5. Randomly select a set of $\hat{w}$ values from X and concatenate them to form a noise sequence n^w.
6. Repeat steps 1,2,3 for n_c times to select the comparison subsequences of X ($\{{c^w}_i\}$) each of length w.

The only added parameters by this algorithm are δ and τ. δ is selected based on the locality variance of the constraint (similar to σ for MCFull). τ should be decided experimentally by finding a tradeoff between increasing the number of iterations (low τ) and increasing the number of random samplings (high τ).

§4 Accuracy Analysis

In this section we analyze the relative accuracy of proposed algorithms and Catalano's algorithm. For this analysis we assume that we have m occurrences ($\{O_i\}$) of a motif (where $1 \leq i \leq m$) in a time series X of n points. Each motif occurrence (O_i) is characterized by a tuple $\langle t_i, l_i \rangle$ where t_i is the location of the motif (its beginning) and l_i is its length. Motif lengths are controlled by a probability distribution P_l with an expected value ($E(P_l)$) equals $\bar{l}$. For Catalano's Algorithm, we define the probability of finding a motif occurrence O_i in a subwindow of length w as P_i^c. As the motif can fall any place within the window: $P_i^c = \frac{(w - l_i + 1)}{n}$ It is straight forward to show that the probability of finding any occurrence of the motif (P^c) in a subwindow

is: $P^c = \dfrac{m\left(w-\bar{l}+1\right)}{n}$. This means that the probability of discovering a motif in Catalano's algorithm ($\overset{c}{\Pr}$) is:

$$\overset{c}{\Pr} = \eta n_c \left(\frac{m\left(w-\bar{l}+1\right)}{n}\right)^2 \tag{1}$$

Where η is the probability of discovering the motif occurrence given that it appears in the candidate window and at least one of the n_c comparison windows. Here we assumed that finding the motif in the candidate window and one of the comparison windows are independent events.

Analyzing proposed algorithms is more complicated by the fact that we use a nonuniform probability distribution for the location of the motif. To facilitate the analysis we define the $r\,(i)$-compact distribution of a given probability distribution P (C_r^P) as $\sum_{j=1}^{r(i)} P\left(j+(i-1)\,r\,(i)\right)$ where $\sum r\,(i) = n$.

The ground truth probability distribution P_g is defined as:

$$P_g\,(j) = \begin{cases} 1/m & j = t_i \\ 0 & otherwise \end{cases}$$

This distribution represents the actual locations of the motifs with uniform distribution among them. Now we calculate the $(r\,(i))$-compact distributions of $\hat{P}$ (see section 3.1) and P_g ($C_r^{\hat{P}}$ and $C_r^{P_g}$ respectively). We also assume that P_g is rearranged so that $C_r^{P_g}\,(i) = 1/m$ for $1 \le i \le m$ and zero otherwise. The same rearrangement is done for $\hat{P}$.

For MCFull, $r\,(i)$ is defined as $w - l_i + 1$ for $1 \le i \le m$ and is defined as $m\left(n - m\bar{l}\right)/n$ otherwise. This simply means that we reduce the *resolution* of the probability distributions to n/m points around the m locations of the motifs and uniformly around the rest of the n points. For MCInc, $r\,(i)$ is defined as 2δ for $1 \le i \le n/\,(2\delta)$.

Using the above definitions:

$$P_i^{mc} = \sum_{j=0}^{w-l_i} \hat{P}\left(t_i + j - \frac{w-l_i+1}{2}\right) = C_r^{\hat{P}}\,(i)$$

$$\overset{mc}{\Pr} = \eta n_c \left(\sum_{i=1}^{m} C_r^{\hat{P}}\,(i)\right)^2 \tag{2}$$

To find the relative accuracy of the two algorithms we need to quantify the accuracy of $C_r^{\hat{P}}$ in reflecting $C_r^{P_g}$. A common information theoretic measure of the difference between two probability distributions is the *relative entropy* defined as:

$$D\left(p||q\right) = \sum_i p\left(i\right) \log_2 \left(p\left(i\right)/q\left(i\right)\right)$$

Applying this to measure the difference between $C_r^{\hat{P}}$ and $C_r^{P_g}$ leads to:

$$D\left(\hat{P}||P_g\right) = \sum_{i=1}^{m} \hat{P}\left(i\right) \log_2 \left(\hat{P}(i)/P_g\left(i\right)\right) + \sum_{i=m+1}^{n/m} \hat{P}\left(i\right) \log_2 \left(\hat{P}(i)/P_g\left(i\right)\right)$$

Using $C_r^{P_g} = {}^1\!/_m$ for $1 \leq i \leq m$ and zero otherwise and with few manipulations we get: $\sum_{i=1}^{m} C_r^{\hat{P}}\left(i\right) = \frac{1}{\log_2 m}\left(D\left(C_r^{\hat{P}}||C_r^{P_g}\right) + H\left(C_r^{\hat{P}}\right)\right)$ where $H\left(p\right)$ is the entropy of the probability distribution p defined as $-\sum_i p\left(i\right)\log_2 p\left(i\right)$.

Substituting in equation 2 and dividing by equation 1 we get:

$$\frac{\Pr^{mc}}{\Pr^{c}} = \left(\frac{n}{m\log_2 m} \times \frac{D\left(C_r^{\hat{P}}||C_r^{P_g}\right) + H\left(C_r^{\hat{P}}\right)}{w - \bar{l} + 1}\right)^2 \tag{3}$$

Equation 3 shows the ratio between the probability of finding a motif using the proposed algorithms and this probability using Catalano's algorithm. As n/m is made larger $\left(n/m\right)^2$ in the equation takes over every thing else increasing this ratio. Even with small values of n/m the proposed algorithms have higher priority of detecting motifs given the same number of trials as long as $D\left(C_r^{\hat{P}}||C_r^{P_g}\right) + H\left(C_r^{\hat{P}}\right) > n\left(w - \bar{l} + 1\right)/m\log_2 m$. In the limit when $\hat{P} = P_g$, the ratio becomes:

$$\frac{\Pr^{mc}}{\Pr^{c}} = \left(\frac{n}{m\left(w - \bar{l} + 1\right)}\right)^2 > 1$$

On the other hand if $\hat{P}$ is selected so that $C_r^{\hat{P}}\left(i\right) = 0$ for $1 \leq i \leq m$ and m/n otherwise then the proposed algorithms will never find any motifs. To achieve best performance of the proposed algorithms, the constraint should be selected to maximize $D\left(C_r^{\hat{P}}||C_r^{P_g}\right) + H\left(C_r^{\hat{P}}\right)$. The following section proposes a heuristic to achieve this.

§5 Calculating the constraint

The Constrained Motif Discovery problem as specified in section 2 does not define any specific way to calculate the constraint (P^n), because this constraint can come from domain knowledge. In this section we describe a method for calculating the constraint (P^n) given only the time series X^n. Using this technique it is possible to apply algorithms designed to solve the constrained motif discovery problem (e.g. MCFull and MCInc) even when there is no enough domain knowledge to calculate the constraint. In section 6 we will use this technique to compare the proposed algorithms with other motif discovery algorithms. We will also show that in many cases this indirect method of solving the unconstrained motif discovery problem can be even more efficient than using traditional approaches to directly solve it.

To find the needed estimation of the motif locations for cases in which domain knowledge cannot be utilized we propose to use a change-point detection algorithm to find change-points in the time series under the assumption that motifs are corresponding to changes in the dynamics of the time series or the generating processes. Although pathological cases that do not follow this assumption can be designed, in most real-world time series mining we are interested in motifs that cause or result from some change either in the time series itself or related variables. In all such cases the proposed technique can be of value in providing an adequate constraint for discovering the interesting motifs much faster than currently available techniques. The main idea is to find the *change score* at every point which is defined as the probability that there is a *change* in the dynamics of the time series and then to utilize this score as the constraint value at this point.

The research in change point (CP) discovery problem have resulted in many techniques including CUMSUM [10], wavelet analysis [11], inflection point search [12], autoregressive modeling [13], Discrete Cosine Transform, and Singular Spectrum Analysis SST [14]. Most of these methods with the exception of SST either discover a single kind of change (e.g. CUMSUM discovers only mean shifts), require ad-hoc tuning for every time series (e.g. wavelet analysis), or assume a restricted generation process (e.g. Gaussian mixtures). In this section we describe a variation of SST transform that keeps its advantages (i.e. no restricting assumptions about the nature of change or generation process, and no ad-hoc tuning) while having higher noise resistance and requiring only two easily specifiable parameters rather than five in case of SST.

5.1 RSST Algorithm

The Robust Singular Spectrum Transform is a change point detection algo-

rithm that has the advantages of requiring only two easily specified parameters, generality as it can be applied to nearly any time series with no ad-hoc tweaking, and resistance to the effects of noise on the specificity of the traditional Singular Spectrum Transform [15]. The essence of the $RSST$ transform is to find for every point $x(i)$ the difference between a representation of the dynamics of the few points before it (i.e. $x(i-p):x(i)$) and the few points after it (i.e. $x(i+g):x(i+f)$). This difference is normalized to have a value between zero and one and named $x_s(i)$.

The dynamics of the points before and after the current point are represented using the Hankel matrix which is calculated in two steps:

1. A set of subsequences $seq(t)$ are calculated as:
$$seq(t-1) = \{x(t-w), ..., x(t-1)\}^T \tag{4}$$
2. The Hankel matrix is calculated as the concatenation of n overlapping subsequences:
$$H(t) = [seq(t-n), ..., seq(t-1)] \tag{5}$$

Singular Value Decomposition (SVD) is then used to find the singular values and vectors of the Hankel Matrix by solving:

$$H(t) = U(t) S(t) V(t)^T \tag{6}$$

where $S(i-1,i-1) \leq S(i,i) \leq (i+1,i+1)$.

Only the first $l(t)$ left singular vectors ($U_l(t)$) are kept to represent the past change pattern as the hyperplane defined by them. Ide and Inoue [14] showed that this hyperplane encodes the major directions of change in the signal. In RSST the value of $l(t)$ is allowed to change from point to point in the time series depending on the complexity of the signal before it. To calculate a sensible value for $l(t)$ we first sort the singular values of $H(t)$ and find the corner of the accumulated sum of them ($l_{inf}(t)$) [the point at which the tangent to the curve has an angle of $\pi/4$].

To find a first guess of the change score around every point, RSST tries to utilize as much information as possible from the future Henkel Matrix ($G(t)$) by using the $l_f(t)$ Eigen vectors of $G(t) G(t)^T$ with highest corresponding Eigen values ($\lambda_{1:l_f}$). The value of $l_f(t)$ is selected using the same algorithm for selecting $l(t)$. This dynamic calculation of $l(t)$ and $l_f(t)$ is the first difference between RSST and traditional SST which fixes $l(t)$ to a user specified value and fixes $l_f(t)$ to one.

$$G(t) G(t)^T u^g = \mu u^g \tag{7}$$

$$\beta_i(t) = u_i^g,\ i \leq l_f(t)\ and \lambda_{j-1} \leq \lambda_j \leq \lambda_{j+1}\ for\ 1 \leq j \leq w \tag{8}$$

If the dynamics captured by the future Hankel Matrix $G(t)$ are similar to the dynamics of the past Hankel Matrix $H(t)$, it is expected that $\beta_i(t)$ will lie on or very near to the hyperplane defined by $U_l(t)$. To quantify this measure of similarity, each one of these $l_f(t)$ directions is then projected onto the hyperplane defined by $U_l(t)$ using:

$$\alpha_i(t) = \frac{U_l^T \beta_i(t)}{\left\| U_l^T \beta_i(t) \right\|}, i \leq l_f \tag{9}$$

The change scores defined by $\beta_i(t)$s and $\alpha_i(t)$s are then calculated as the cosine of the angle between each $\beta_i(t)$ and its projection ($\alpha_i(t)$) using:

$$cs_i(t) = 1 - \alpha_i(t)^T \beta_i(t) \tag{10}$$

The first guess of the change score at the point t is then calculated as the weighted sum of these change point scores where the Eigen values of the matrix $G(t)$ are used as weights.

$$\hat{x}(t) = \frac{\sum_{i=1}^{l_f} \lambda_i \times cs_i}{\sum_{i=1}^{l_f} \lambda_i} \tag{11}$$

The second difference between RSST and traditional SST is this utilization of multiple projections from the future Hankel Matrix's Eigen vectors weighted by the Eigen vectors.

After applying the aforementioned steps we get a first estimate $\hat{x}(t)$ of the change score at every point t of the time series. RSST then applies a novel filtering step that has no counterpart in traditional SST to attenuate the effect of noise on the final scores. The main insight of this filter is that the reduction of SST specificity in noisy signals happens in the sections in which noise takes over the original signal in the time series (see section 5.2). The response of SST at these sections can be modeled by a random walk around a high average for uncorrelated white noise. The filter used by RSST discovers these sections in which the average and the variance of $\hat{x}(t)$ remains nearly constant and attenuates them. The filter first calculates the mean (μ) and variance

(σ^2) of the signal before and after each point using a subwindow of size w. The guess of the change score at every point is then updated by:

$$\tilde{x}(t) = \hat{x}(t) \times |\mu_a(t) - \mu_b(t)| \times \left| \sqrt{\sigma_a(t)} - \sqrt{\sigma_b(t)} \right| \tag{12}$$

where $\mu_b(t) = \frac{\sum_{i=0}^{w-1} \hat{x}(t-i)}{w}$, $\sigma_b^2(t) = \frac{\sum_{i=1}^{w} (\hat{x}(t-i) - \mu_b(t))^2}{w-1}$, $\mu_a(t) = \frac{\sum_{i=1}^{w} \hat{x}(t+i)}{w}$, and $\sigma_a^2(t) = \frac{\sum_{i=1}^{w} (\hat{x}(t+i) - \mu_a(t))^2}{w-1}$

$\tilde{x}(t)$ is then normalized to get $\hat{x}(t)$ which represents the final change score of RSST.

RSST is a single dimension change point detection algorithm. To apply it to multidimensional data the algorithm is first applied to every dimension of the data and then the resulting transforms are then combined using:

$$\hat{x}(t) = \max(\hat{x}_i(t)) \tag{13}$$

This combination rule corresponds to accepting the change point if some change is detected in *any* dimension of the data. Other combination rules can be used as well.

RSST has two parameters, namely, w and n. These parameters determine the length of the subsequence used to measure the dynamics (calculate the Eigen vectors) before and after every point. When a tight upper bound of the motif length of available (l_{max}), then w and n should be selected so that $l_{max} \approx w \times n$. This ensures that at the boundaries of the motif, its whole length is used to calculate the dynamics. Furthermore if it is known that the motif is a *nearly* periodic function (e.g. in ECG signals), and an estimate of its period is available T (where $T < l_{max}$) then w and n should be selected so that $T \approx w \times n$. In both these cases w and n should be selected so that $w \approx n$ and $w \leq n$.

If no tight upper bound of the motif length is available; then the rules $w \approx n$ and $w \leq n$ still hold and can be used to calculate n once w is selected. In this case the value of w (and hence n) can be used as a *focusing* factor. Increasing w means that we look at changes in long subsequences (zooming out) while decreasing w is equivalent to zooming in and focusing on the details of the change.

5.2 Comparison Between SST and RSST

The main differences between RSST and traditional SST can be summarized as follows. First, RSST requires the specification of only two rather than five parameters.

Second, RSST automatically calculates the optimal number of Eigen vectors to use both for past and future Hankel matrices rather than fixing them as in SST (Equation 12). Third, RSST utilizes more information from the time series by using multiple Eigen vectors of the future Hankel matrix (Equation 11). Finally RSST uses a post-processing step to reduce the effect of noise on the final change score reported (Equation 12).

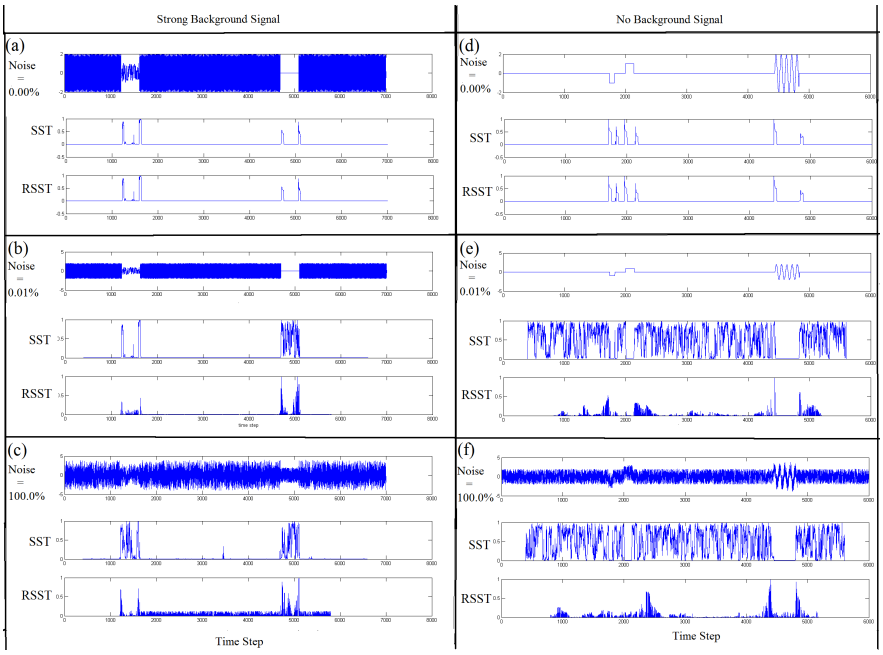


Fig. 1 The Effect of noise Level on the performance of SST and RSST algorithms.

Fig. 1 shows visually the effect of adding white noise to the signal on the performance of SST and the proposed RSST. For every condition the original signal, response of the SST transform and response of the RSST transform are shown. Fig. 1.a and Fig. 1.d show the response of both SST and RSST to two changes in dynamics of a data series with strong background signal and with no background signal. Both algorithms perform well in this condition. Fig. 1.b and Fig. 1.e show the effect of adding uniform random noise of range -0.5%:0.5% of the peak to peak (P-P) value of the original signal. Even with this very low noise level, the SST transform gives high response in all the locations in which the noise level is higher than the original signal value. The performance of SST transform is especially unacceptable when there is no background signal. The RSST transform on the other hand performs much better under this very low noise level. Fig. 1.c and Fig. 1.f show the effect of adding uniform

random noise of range -50%:50% of the peak to peak (P-P) value of the original signal. Again the performance of RSST is visually much better than the performance of SST under this very noisy condition. This difference in performance will be quantified by extensive tests using synthetic data with adjustable noise levels and background signal strength in section 5.3.

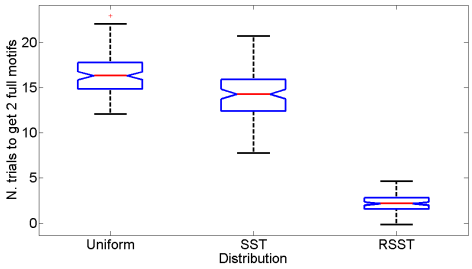


Fig. 2 Difference in the number of trial needed to get a full motif in both the candidate window and one comparison windows depending on the window sampling distribution

Fig. 2 shows the average number of trials needed to get a complete motif occurrence in both the comparison and candidate windows to be used with MCFull algorithm (section 3.1) and utilizing a uniform distribution for the constraint (traditional Catalano et al's algorithm), and the results of applying SST and RSST to the signal. In this experiment a single candidate window and three comparison windows were used. The window size was selected as double the length of the *embedded pattern*. The averages over 5760 different synthetic time series that will be introduced in the following section are computed and are shown in Fig. 2. There is no significant difference in the number of trials needed to have a complete motif in the candidate window and at least one comparison window between using uniform random sampling used by Calalano et al.'s algorithm [8] and using SST. Nevertheless using the normalized change scores of RSST reduced the mean number of trials from around 16 to less than two trials. The t-test was used to test this hypothesis and it confirmed that the difference between RSST and both SST and Uniform distributions is significant with $p < 0.0001$. The difference between SST based and uniform sampling was not statistically significant.

5.3 Evaluation of RSST with Synthetic Data

To compare the performance of SST and RSST both in speed and accuracy, 5760 different synthetic time series with controlled embedded changes were produced by changing various features of the time series. Every time series was composed of a re-

curring pattern called the *background signal* with *embedded patterns* embedded at random locations with controllable numbers. Uniform noise was added to the time series before applying the transforms to it. The dependent variables in this experiment were the background signal strength (peak-to-peak value of the *background signal* divided by the peak-to-peak value of the *embedded pattern*), the noise level, and the generating processes. The length of the time series was fixed at 8000 points and the length of the embedded pattern was varied from 40 to 400 points. For SST, g and m were selected to equal zero and n respectively (as in the RSST). The value of l was selected to be 3 after experimenting with 300 random samples of the time series and selecting the value that achieved best performance.

For evaluation purposes, we treated the problem as discrete event detection and used specificity and sensitivity to measure the effectiveness of SST and RSST as follows:

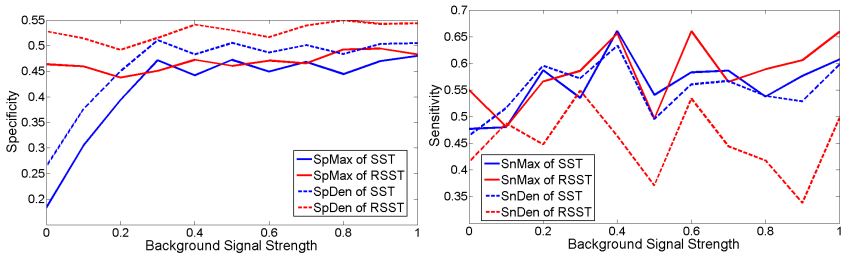
Definition 5.1 (Sensitivity of Maximum Response *SnMax*)
The maximum response of the transform in a window of width $\tau/100$ around the true points of change, averaged over all true points of change. τ is length of the *embedded pattern*.

Definition 5.2 (Sensitivity of Response Density *SnDen*)
The average response of the transform in a window of width $\tau/10$ around the true points of change, averaged over all true points of change.

Definition 5.3 (Specificity of Maximum Response *SpMax*)
The maximum response of the transform in all $\tau/10$ windows of the time series excluding the $\tau/10$ windows around the true change points.

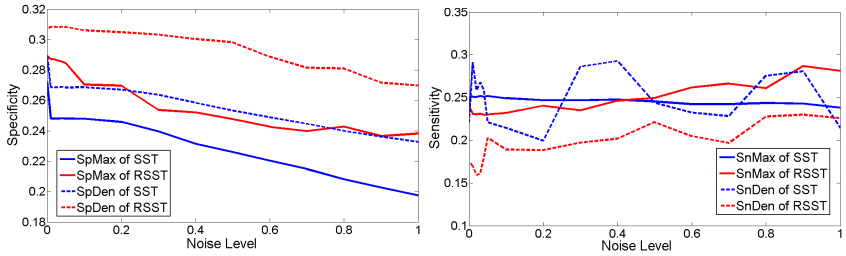
Definition 5.4 (Specificity of Response Density *SpDen*)
The average response of the transform in all the points of the time series excluding all $\tau/10$ windows around the true change points.

Fig. 3-a shows the effect of background signal strength on the specificity of both SST and RSST. As the Figure shows the specificity of RSST is always higher than SST in terms of Specificity of Response Density *SpDen*. This means that RSST is more suitable for applications in which the response around the change points not only the response at the change points is important including our application. In terms of Specificity of Maximum Response (SpMax), RSST is also superior for SST both in low and high background signal strengths with a period in which both transforms perform



(a) Effect of Background Signal Strength on the specificity

(b) Effect of Background Signal Strength on the sensitivity



(c) Effect of the noise level on the Specificity

(d) Effect of the noise level on the Sensitivity

Fig. 3 comparison between RSST and SST using synthetic data

equally between 40% and 70% in our data sets.

Fig. 3-b shows the effect of background signal strength on the sensitivity of both SST and RSST. As the figure shows there is no clear superiority of one of the transforms over the other in terms of Sensitivity of the Maximum Responses but SST shows slight superiority in the Sensitivity of the Response Density. This can be attributed to the reduced specificity of SST which results in many high responses, some of them comes naturally very near to the change points increasing the apparent sensitivity of SST. This effect did not happen in the SenMax matrix as the maximum response will not be affected with these noisy false responses.

Fig. 3-c shows the effect of noise level on the specificity of the two transforms. RSST is clearly superior to SST in terms of specificity at all noise levels and the rate of degradation of its specificity is less than SST. This superiority is especially clear in the SpDen metric which is again more important than SpMax for our application. Fig. 3-d shows the effect of noise level on the sensitivity of the two transforms. Here again SST is slightly superior to RSST due the effect of noisy high responses near the change points explained earlier.

5.4 Application to Physiological Data Mining

To test the accuracy of RSST on real world data we analyzed the respiration response of 22 participants with ages ranging from 18 to 45 while they were explaining an assembly/disassembly to a listener who either listened carefully (natural session) or did not attend to the explanation (unnatural session). This led to 44 explanation sessions. For more details on this experiment refer to Mohammad et al. [16]. The respiration signal of the speaker was sampled with rate 100Hz using a Polymate (TEAC Company) device. Every instructor ranked the behavior of the listener after every session in a scale from 1 to 7. Scores over 4 were considered natural sessions and scores under two were considered unnatural sessions.

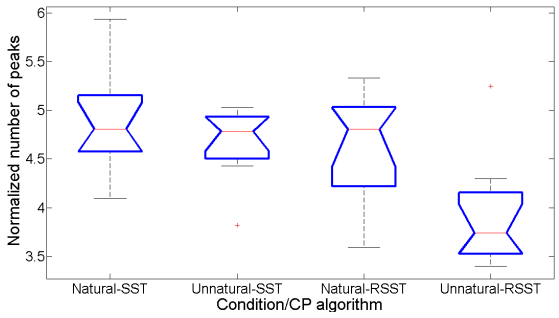


Fig. 4 Number of Peaks in the Change Points detected by SST and RSST

Fig. 4 shows a box chart of the number of peaks (change score > 0.6) using SST and RSST in the two experiment conditions using SST and RSST. For visual clarity, the number of peaks in all the sessions was normalized separately for SST and RSST signals. It can be seen easily that RSST distinguishes between the two conditions while SST fails to do so. Applying t-test no statistical significant difference between the SST response in natural and unnatural conditions was found while the difference in RSST response in these conditions was significant with $p < 0.001$.

5.5 Evaluation of RSST with Other Real world Data

This section reports the performance of the RSST algorithm on 6 different real-world data sets namely(sunspot, shuttle, powerplant, memory, phData, burst). For multidimensional time series, the transforms were applied to all dimensions independently. All the data sets used in this section are available from the UCR library [7].

To get a ground truth for evaluating the RSST algorithm on these data sets, three participants who have no prior knowledge of computer science or this research were asked to select the points of change in each signal and the results of their selection

were averaged and used as ground truth for the evaluation purposes.

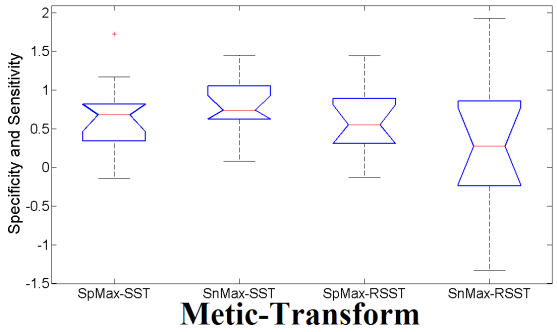


Fig. 5 Specificity/Sensitivity of RSST and SST on all the dimensions of six real world data sets

Fig. 5 shows the results of applying both RSST and SST to these small size real world data sets (between 200 and 2000 points). In these data sets, the performance of SST and RSST is very similar with no significant superiority in the complete set. Careful study of these data sets reveals that most of them have low noise levels, and strong background signals. Under these conditions the difference between SST and RSST vanishes. It is important though to notice that even in these conditions, SST did not outperform RSST.

§6 Synthetic Data Evaluation of MC Algorithms

To evaluate the effectiveness of the proposed modifications of Catalano algorithm in solving the constrained motif discovery problem we will compare the performance of the following algorithms on a wide range of synthetic data with embedded motifs:

1. Projections [Pro]: A general motif discovery algorithm introduced by Buhler and Tompa [17] and is the basis of many other state-of-the-art motif discovery algorithms [17, 1]. This algorithm requires the specification of the exact motif length.
2. Catalano et al.'s algorithm: The original Catalano et al.'s algorithm [8] (see section 3). This algorithm is abbreviated as *CAT* in the rest of this section.
3. RSST(proposed in section 5.1) +MCFull (proposed in section 3.1).
4. RSST(proposed in section 5.1) +MCInc (proposed in section 3.2).

50400 different synthetic time series with controlled embedded motifs were produced by changing various features of the time series. Every time series was com-

posed of a recurring pattern called the *background signal* with *embedded motifs* embedded at random locations with controllable numbers. Uniform noise was added to the time series. The patterns used were sinusoidal, linear, square, sawtooth, and two random patterns. Time series length was changed from 1×10^3 to 1×10^6. The relative number of embedded motifs was changed from 1% to 10% of the length of the time series. The length of the embedded motif was changed from 50 to 100 points. The noise level, defined as the peak value of the uniform noise added divided by the peak-to-peak value of the *motifs*, was changed from 0% to 20%. The background signal strength, defined as the peak-to-peak value of the *background signal* divided by the peak-to-peak value of the *embedded pattern*, was changed between 0% to 40%. The generating process of both the *background signal* and the *motif* was also randomly selected from five different generating processes.

For CAT, MCFull and MCInc, the parameters l_{min}, l_{max} were fixed at 40 and 100 respectively. For the Projections algorithm the correct motif size was given to the algorithm and the SAX [1] parameters were selected using the heuristic described by Minnen et al. [9].

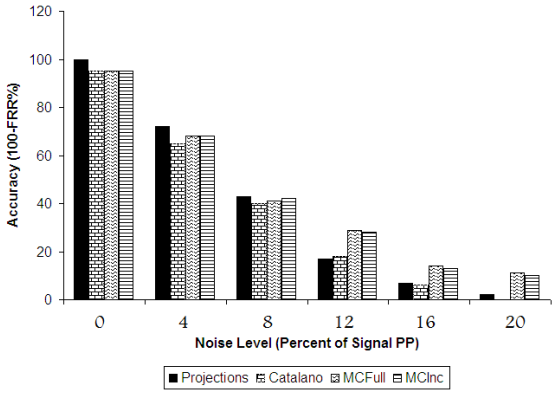
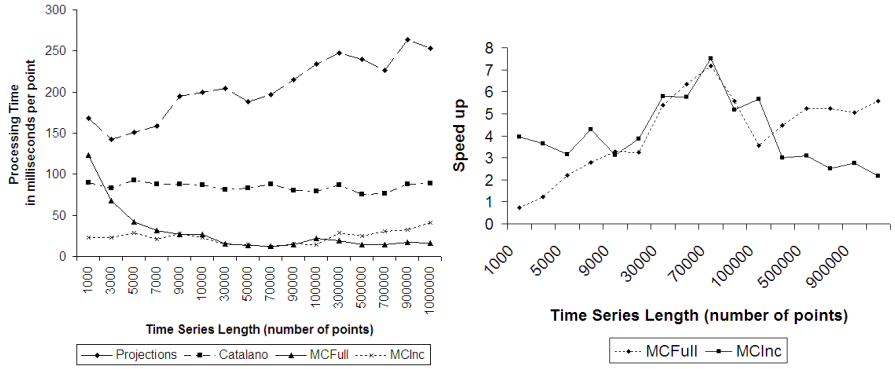


Fig. 6 Effect of noise on the four algorithms

Fig. 6 compares the accuracy of the four algorithms as a function of noise level when allocated the same time to run. The accuracy is defined as $100 - FRR$ where FRR is the percentage False Rejection Rate. The Projections algorithm has the best noise robustness for low noise levels due to its exhaustive nature while with higher noise levels MCInc and MCFull shows slower degradation in performance.

Fig. 7-a shows the processing time per point for every one of the tested algorithms. It is clear that the Projections algorithm is subquadratic although not linear



(a) Processing Time per point of the input time series for the four tested motif discovery algorithms.

(b) The Speedup obtained over Catalano et al.'s original algorithm for the two proposed algorithms

Fig. 7 Effect of signal complexity on SST and RSST.

while the CAT, MCFull, and MCInc are all linear especially for long time series. The average speedup obtained by using MCFull was 4.19 times over the Catalano algorithm and 10.76 times over the Projections algorithm. The average speedup obtained by MCInc was 4.09 over the Catalano algorithm and 9.84 times over the Projections algorithm.

To further study the effect of time series length on the speedup obtained by using the MCFull and MCInc algorithms, Fig. 7-b plots the speedup over the CAT algorithm as a function of the time series length. As the figure shows, MCFull can even be slower than CAT algorithm for short time series while MCInc shows four folds increase in speed at the same conditions. When the time series length increases, the advantage of MCInc over MCFull decreases until for long time series, MCFull achieves faster processing than MCInc.

§7 Application to Human-Robot Interaction

Gesture interfaces are becoming more common in controlling robots due to their ease of use and expressibility. In most cases a set of predefined gestures have to be learned by the operator before (s)he can effectively operate the robot [18, 19]. Our goal in this section is to utilize MCFull and RSST algorithms detailed above to develop an unsupervised learning system that allows the robot to learn the meaning of free hand gestures, the actions related to some task and their associations by just watching other human/robot actors being guided to do the task by different operators. This kind of learning by watching experienced actors is very common in human learning. The main challenge in this case is that the learning robot has no a-priori knowledge of the actions

done by the actor, the commands given by the operator, or their association and needs to learn the three in an unsupervised fashion from a continuous input stream of actor movements and operator's free hand gestures.

The steps we use in this work can be summarized in the following: First, synchronized hand motion $g(t)$ and actor behavior signals $a(t)$ are collected. The behavior of the actor is encoded as a set of features that represent the state of the actor in relation to the operator (e.g., angle between their heads, distance between them, difference in body orientation etc).Second, the RSST algorithm (section 5.1) is applied to both $g(t)$ and $a(t)$ to find the points of change in the two signals and the results are named $p(t)$ and $b(t)$ respectively. The time series $p(t)$ and $b(t)$ are then combined to form the constraint used to discover gesture types and primitive actions using:

$$g_{const}(t) = \frac{p(t) + b(t)}{\max\limits_{\tau=1:|g|}(p(\tau) + b(\tau))} \tag{14}$$

The fourth step is to apply the MCFull algorithm (section 3.1) to the time series $g(t)$ using the constraint $g_{const}(t)$ (section 3.1) to find the recurrent motifs G and their occurrences $\{o_G\}$ in the hand motion of the operator. These motifs represent the gesture types discovered and their occurrences are the discovered occurrences of the gestures. The same procedure is then used to discover the basic actions of the actor (A) and their occurrences (o_A). Finally the discovered gestures are associated with corresponding actions as described in the following paragraph to build a Bayesian Network that can be used to control the learning robot once associated with the appropriate controllers.

Once both gestures (G) and basic actions (A) are discovered, it is possible to associate each gesture with one or more actions as follows:

- For every discovered gesture motif occurrence(o_G), associate it with a list of all action occurrences (o_A) at time t to $t + T$, where T is a domain specific maximum delay between the gesture (command) and the operator's action. Call this associated map $Act(o_G)$.
- For every gesture motif G find the consistency of the occurrence of every action just after gesture occurrences as:

$$c(G, A) = \frac{count(A \in Act(o_G))}{|\{o_G\}|} \tag{15}$$

- Build a Bayesian Network of gestures and their associated actions by assigning every gesture motif (G) and action motif (A) a node and connecting nodes G and A iff $c(G, A)$ is greater than a threshold (Th_c). The threshold can be automatically determined by the *knee-in-the-curve* method used in section 5.1 to

find $l(t)$. The probability assigned to the edge from G to A ($E(G,A)$) is set to $c(G,A)$. A special edge $E(G,NULL)$ is added between every gesture node G and the node representing doing nothing ($NULL$ action node) and associate it with the probability $1-\sum_{a\in\{A\}} c(G,a)$. The probability $P(G)$ is calculated as $|\{o_G\}| \Big/ \sum_{g\in\{G\}} |\{o_g\}|$. For action nodes, the conditional probability $P(A|G)$ is calculated as $c(G,A)$. Using these rules, it is straight forward to dive the conditional probability table at every node. Finally attach with each edge ($E(G,A)$) the average delay τ_{GA} and its variance ${\sigma^2}_{GA}$ between the occurrence of G and A to be used for action generation.

This induced BN along with the delay statistics (τ_{GA} and ${\sigma^2}_{GA}$) define a model of the interaction between the operator and the actor and can be used to activate the correct action when any of the learned gestures is detected in the input stream. The generation of actual behavior by the learning robot requires only associating a controller with each action node in the BN that is executed once this action node is active. The automatic development of such a controller is robot and domain dependent and is a future direction of research.

7.1 Evaluation Experiment

This section presents a feasibility study to assess the applicability of the technique presented in section 7. We first introduce the experimental procedure used for the evaluation and then provide the results and discussion of this experiment.

[1] Procedure

The evaluation experiment was designed as a Wizard of Ooz (WOZ) experiment in which an untrained novice human operator is asked to use hand gestures to guide the robot shown in Fig. 8-a along the two paths shown in Fig. 8-c in two consecutive sessions. The subject is told that the robot is autonomous and can understand any gesture (s)he will do. A hidden human operator was sitting behind a magic mirror and was translating the gestures of the operator into the basic primitive actions of the WOZ robot that were decided based on an earlier study of the gestures used during navigation guidance (for details of this earlier work refer to Mohammad and Nishida [20]). At the same time the software was used for online annotation of the beginnings and endings of every gesture of the participant as well as every action of the robot. These annotations was then used as ground truth for evaluating the accuracy of the gesture discovery phase

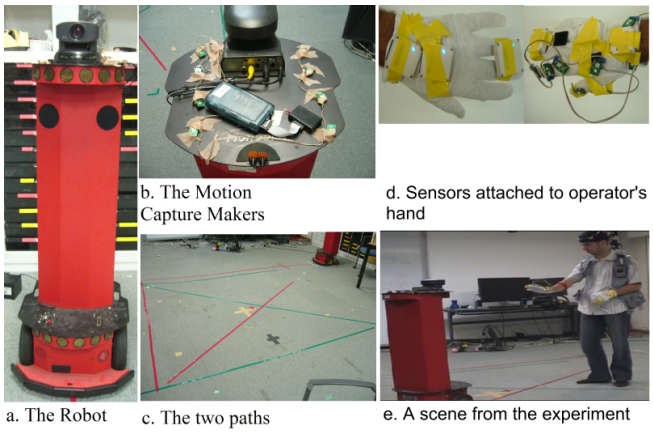


Fig. 8 The robot used in the experiment (a), the motion capture markers attached to it (b), the paths that were used (c), the sensors attached to the operator's hands (d), and a scene from the experiment (e)

of the algorithm. The decision of using a WOZ operated robot rather than a human actor was based on the assumption that subjects may not use the same kinds of gestures when dealing with human and robotic partners.

Eight participants of ages 21 to 34 (all males) that have no prior knowledge of robots and do not study in the field of engineering were selected for this experiment. The total number of sessions conducted was 16 sessions with durations ranging from 5:34 minutes to 16:53 minutes. The motion of the participant's hands was measured by six B-Pack sensors [21] attached to both hands as shown in Fig. 8-d generating 18 channels of data. The PhaseSpace motion capture system (PhaseSpace Inc.) was also used to capture the location and direction of the robot using eight infrared markers as shown in Fig. 8-c. The location and direction of the subject was also captured by the phaseSpace system using six markers attached to the head of the subject (three on the forehead and three on the back). Eight more motion capture markers were attached to the thumb and index of the right hand of the subjects to allow reconstruction of the pose of these two fingers for further behavioral analysis but these sensors were not used in the analysis given in this paper.

The following four feature channels were used to represent actor's behavior in the action discovery phase:

- The directional speed of the robot in the XZ (horizontal) plane in the direction the robot is facing (by its cameras)(1 channel).
- The direction of the robot in the XZ plane as measured by the angle it makes

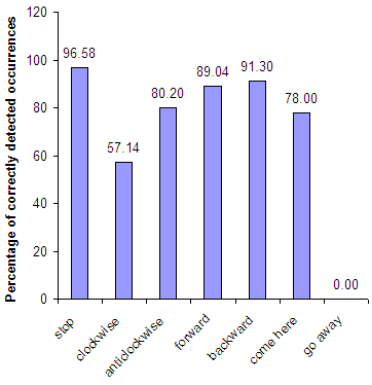


Fig. 9 The detection accuracy for the gestures used in the experiment.

with the X axis (1 channel).

- The relative angle between the robot and the actor (1 channel).
- The distance between the operator and the actor (1 channel).

[2] Results and Discussion

The goal of the gesture discovery phase is to find the gesture types and their occurrences from the time series generated by the operator.

Table 1 shows the accuracy of the algorithm in discovering gestures and gesture occurrences. As shown in the table, the algorithm discovered all the types of gestures used by the subjects except the *go away* gesture which was mostly confused with the *move backward* gesture. The algorithm also made four false positive errors. The algorithm can then detect 85.7% of the gesture types and in average it finds 82% of the occurrences of discovered gestures. Fig. 9 shows the percentage of accurately discovered occurrences to the total number of occurrences of every gesture type.

The goal of the action discovery is to find the basic actions of the actor from the actor state features as well as their occurrences. Four features where used in this experiment as explained in the previous subsection.

- The directional speed of the robot. The algorithm discovered the actions corresponding to *Stop Advancing*, *Forward*, and *Backward* commands.
- The direction of the robot. The algorithm discovered the actions corresponding to *rotate clockwise*, *rotate anticlockwise* and *stop rotating* commands.
- The relative angle between the robot and the actor. The algorithm detected the action corresponding to the first stage of *Come Here* command in which the

Table 1 Confusion matrix showing the accuracy of detecting the occurrences of discovered gestures

Discovered Gesture	Meaning	Gesture Occurr.	St. 322	Cl. 70	ACl. 101	For. 73	Back. 69	Come 50	Go 18
G1	St.	319	**311**	3	4	0	0	1	0
G2	Cl.	56	2	**40**	12	1	1	0	0
G3	ACl.	102	0	17	**81**	0	0	2	2
G4	For.	73	0	1	1	**65**	4	1	1
G5	Back.	80	1	0	0	3	**63**	0	**13**
G6	Come	54	0	3	0	11	0	**39**	1
G7	Unk.	9	1	0	1	1	3	3	0
G8	Unk.	4	1	1	0	1	1	0	0
G9	Unk.	5	0	0	1	0	0	2	2
G10	Unk.	14	0	3	2	3	2	1	3

St.: Stop Cl.: Rotate Clockwise ACl.: Rotate Anticlockwise
For.: Move Forward Back.: Move Backward Come: Come here
Go: Go Away Unk.: Unknown Gesture

robot faces the human. An extra false positive motif was discovered in this dimension that does not correspond to any commands of the operator and was a result of the noise in the input data.

- The distance between the operator and the actor. The algorithm discovered the actions corresponding to the second phase of the *Come Here* command in which the actor approaches the operator. Even though some operators used the command *Go Away* it was not discovered by our algorithm (a single false negative).

The total number of basic actions discovered by the algorithm is eight actions. 87.5% of these actions correspond to authentic actions that were implemented in the interface of the WOZ software. The accuracy of detecting occurrences of the discovered actions was 88.3%.

To evaluate the final association phase of the proposed system; the data of the 16 sessions were divided into two sets (S_1 and S_2). One of the two sessions of every subject was assigned to every one of these sets but the order was shuffled so that every group contains the same number of first and second sessions. The proposed algorithm was applied to every set to generate the final Bayesian Network representing the association of the gestures and actions and then tested on the other set (2-fold cross validation). Fig. 10 shows the structure of the final BN learned in both sessions. It is interesting to notice that the algorithms successfully removed all of incorrectly discovered

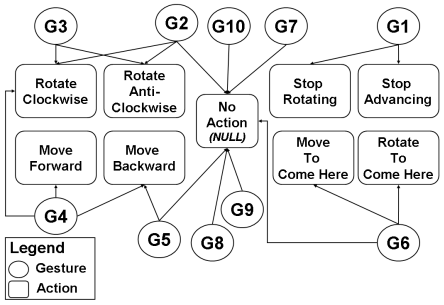


Fig. 10 The Bayesian Network learned by both training sessions

Table 2 Comparison between the four motif discovery algorithms with gesture and action streams.

Type	Total	Projections				CAT				MCFull			
		Type I Er.	Type II Er.	Accuracy	Time	Type I Er.	Type II Er.	Accuracy	Time	Type I Er.	Type II Er.	Accuracy	Time
Gestures	7	1	3	91.5	1	3	4	67.8	0.31	1	4	82	0.14
Actions	8	0	2	94.6	1	2	1	74.5	0.36	0	1	88.3	0.09

gestures, by associating them only with the NULL action, as they did not consistently correspond to any of the basic actions of the WOZ robot.

We compared the action with maximum probability from the model with the action activated by the WOZ operator during the test session. The two actions agree in 95.2% of the time which means that the learning algorithm could generate an accurate representation of the association between gestures and actions. Most of the errors were due to confusion between rotating clockwise and anticlockwise.

It is instructive to compare the accuracy of gesture and action motif discovery using the four algorithms compared in section 6. Table 2 shows the results of this comparison. In this table, Type I error means that a gesture (action) type is not discovered while Type II error means that random data are assumed to be a gesture (action). The accuracy is measured by the percentage of motif occurrences that were detected by the algorithm. For Projections we used exhaustive search after detection of motif stems to discover these occurrences but for Catalano's and MCFull algorithms we did not need to add this exhaustive search. Execution time was normalized so that the execution time of Projections is unity. In terms of detection of motif type the proposed algorithm achieved the same performance as Projections in general but in 12% of the time in general. For accuracy Projections was better than the proposed algorithm in finding motif

occurrences. This is expected given the exhaustive detection step used. In all cases the MCFull outperformed Catalano's algorithm which indicates that RSST was successful in detecting the change points in this dataset.

§8 Conclusion

This paper presented the Constrained Motif Discovery problem and the first two algorithms to solve it. The two algorithms (MCFull and MCInc) are modifications of the recently developed algorithm by Catalano et al. for solving unconstrained motif discovery problems. The paper also introduced the RSST algorithm for change point detection and compared it to traditional SST using both synthetic and real-world data sets, and showed how to use it to effectively convert unconstrained motif discovery problems into constrained motif discovery problems that can be solved in linear time.

The evaluation of combining RSST with MCInc and MCFull algorithms using synthetic data show four folds increase in speed over the original Catalano et al.'s algorithm in a large set of synthetic time series. Comparison with the PROJECTIONS algorithm also showed a ten folds increase in speed with minimal effect on accuracy.

RSST and MCFull were used to learn free hand gestures, actions, and their associations in a guided navigation task in which a human operator guided a robot to follow a predefined path using free hand gestures. The proposed system correctly discovered over 83% of the gestures and 87% of the actions involved and a Bayesian Network induced over the discovered gestures and actions achieved over 95% prediction accuracy.

References

1) Bill Chiu, Eamonn Keogh, and Stefano Lonardi. Probabilistic discovery of time series motifs. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 493–498, New York, NY, USA, 2003. ACM.

2) Tom Oates. Peruse: An unsupervised algorithm for finding recurring patterns in time series. In *International Conference on Data Mining*, pages 330–337, 2002.

3) Kyle L. Jensen, Mark P. Styczynxki, Isidore Rigoutsos, and Greogory N. Stephanopoulos. A generic motif discovery algorithm for sequenctial data. *BioInformatics*, 22(1):21–28, 2006.

4) Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Pranav Patel. Finding motifs in time series. In *In the 2nd workshop on temporal data mining, at the 8th ACM SIGKDD international*, pages 53–68, 2002.

5) D. Minnen, I. Essa, C.L. Isbell, and T. Starner. Detecting subdimensional motifs: An efficient algorithm for generalized multivariate pattern discovery. In *IEEE Int. Conf. on Data Mining (ICDM).*, 2007.

6) Heng Tang and Stephen Shaoyi Liao. Discovering original motifs with different lengths from time series. *Know.-Based Syst.*, 21(7):666–671, 2008.

7) E. Keogh, J. Lin, and A. Fu. Hot sax: efficiently finding the most unusual time series subsequence. *Data Mining, Fifth IEEE International Conference on*, pages 8 pp.–, Nov. 2005.

8) Joe Catalano, Tom Armstrong, and Tim Oates. Discovering patterns in real-valued time series. In *Knowledge Discovery in Databases: PKDD 2006*, pages 462–469, 2007.

9) D. Minnen, T. Starner, I. Essa, and C.L. Isbell. Improving activity discovery with automatic neighborhood estimation. In *Int. Joint Conf. on Artificial Intelligence*, 2007.

10) m. Basseville and I Kikiforov. *Detection of Abrupt Changes.* Printice Hall, Englewood Cliffs, New Jersy, 1993.

11) S. Kadambe and G.F. Boudreaux-Bartels. Application of the wavelet transform for pitch detection of speech signals. *Information Theory, IEEE Transactions on*, 38(2):917–924, Mar 1992.

12) Shoji Hirano and Shusaku Tsumoto. Mining similar temporal patterns in long time-series data and its application to medicine. In *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)*, page 219, Washington, DC, USA, 2002. IEEE Computer Society.

13) Edit Gombay. Change detection in autoregressive time series. *J. Multivar. Anal.*, 99(3):451–464, 2008.

14) Tsuyoshi Ide and Keisuke Inoue. Knowledge discovery from heterogeneous dynamic systems using change-point correlations. In *Proc. SIAM Intl. Conf. Data Mining*, 2005.

15) V. Moskvina and A. Zhigljavsky. An algorithm based on singular spectrum analysis for change-point detection. *Communications in Statistics.Simulation and Computation*, 32(4):319–352, 2003.

16) Yasser Mohammad, Yong Xu, Kenichi Matsumura, and Toyoaki Nishida. The h^3r explanation corpus:human-human and base human-robot interaction dataset. In *The fourth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP2008)*, December 2008.

17) J. Buhler and M. Tompa. Finding motifs using random projections. In *5th Internatinal Conference on Computational Biology*, pages 69–76, 2001.

18) Soshi Iba, Chris Paredis, and Pradeep Khosla. Interactive multimodal robot programming. *International Journal of Robotics Research*, 24(1):83–104, January 2005.

19) Matthieu Guillemot Yong Xu and Toyoaki Nishida. An experiment study of gesture-based human-robot interface. In *IEEE/ICME International Conference on Complex Medical Engineering-CME2007*, pages 458–464, 2007.

20) Yasser Mohammad and Toyoaki Nishida. Human adaptation to a miniature robot: Precursors of mutual adaptation. In *The 17th IEEE International Symposium on Robot and Human Interactive Communication, 2008. RO-MAN 2008*, pages 124–129, 2008.

21) R. Ohmura, F. Naya, H. Noma, and K. Kogure. B-pack: a bluetooth-based wearable sensing device for nursing activity recognition. *Wireless Pervasive Computing, 2006 1st International Symposium on*, pages 6 pp.–, Jan. 2006.