Distributed Signal and Image Processing: Particle Filters, Context Grammars, and Dynamic Games

BY

JING HUANG
B.S. (Huazhong University of Science and Technology) 2009
M.S. (University of Illinois at Chicago) 2012

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Chicago, 2013

Chicago, Illinois

Defense Committee:

Dan Schonfeld, Chair and Advisor
Milos Zefran
Natasha Devroye
Daniela Tuninetti
Shmuel Friedland, Math, Stats, and Computer Sci

To my parents

To my husband

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Dan Schonfeld, for his guidance and support along the way of my Ph.D study. I am very grateful to my advisor for giving me this valuable opportunity to study and research in the video processing area. I want to express my sincere thanks to his inspiring instruction, his confidence in me finishing my study. I am very honored to be his student and a member of the multimedia communication lab (MCL).

I would also like to thank Professor Milos Zefran, Professor Natasha Devroye, Professor Daniela Tuninetti and Professor Shmuel Friedland for taking their precious time to serve on my committee.

In addition to my committee members, I would like to thank Dr. Christina Pavlopoulou for offering me the summer intern opportunity in Xerox Palo Alto Research Center, 2011. This intern experience had been very beneficial to me for connecting my research work with the real applications. I would like to thank for your guidance in my intern project on large-scale document classifications, where I first time utilized my knowledge in video for real products design. I would like to thank Dr. Faisal Ishitiaq for your insightful mentoring on my second summer internship in Motorola Mobility, 2012. By working on the project of text detection, my experience on video processing had been widely expanded. I would like to thank Dr. Vinayak Nagpal for your guidance in my third intern at Nokia research center also.

I would like to thank all my friends and colleagues in UIC. They have been a great support and encouragement to me during these years. I would like to thank Yao Li and Nanzhu Zhang, for being great friends to me. They have been my great support and make my life in UIC much happier. I would like to thank Liming Wang and Chong Chen, for their patience and valuable suggestions whenever I came with questions to ask. Their expertises in the area of video/image processing has benefited me a lot. Discussing with them was always very insightful. I want to express my sincere thanks to them for all their help. I would also like to thank my other lab mates for their great help during my study. They are Julia Yang, Yanying Chen and Carol Shi.

I cannot express enough of my gratitude to my parents and my husband. Their support and love have always been the source of my strength and the reason I have come this far. I would like to thank my dearest parent, who gave me a love of life. Their love have always been with me throughout these years. I also would like to thank my husband. His love and encouragement have supported me in everything throughout these years while I was frustrated. I look forward to our brand new life in California!

JH

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER PAGE

LIST OF TABLES

LIST OF TABLES (Continued)

TABLE **PAGE**

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

SUMMARY

A novel distributed video and image processing framework is presented in our work. Our work involves a serious of new algorithms in video processing and dynamical games. In the first part of our work, we present a distributed graph-based sequential particle filtering framework for visual tracking from single and multiple collaborative cameras in lossy networks. Many practical visual processing applications require a robust and efficient algorithm to handle occlusions for visual tracking from degraded visual data in camera networks that utilizes limited computational resources. Firstly, distributed graph-based particle filtering for visual tracking from one view is introduced. Specifically, two new distributed approaches: the graph-based sequential particle filtering framework and its hierarchical counterpart are proposed from one camera. We subsequently derive a distributed visual tracking solution from multiple cameras to handle object occlusions in the presence of frame loss by using collaborative particle filters. The proposed approach relies on Markov Properties and partial-order relations to derive a close-form sequential updating scheme on general graphs in lossy networks. The resulting distributed visual tracking technique is therefore robust to occlusion and sensor errors from specific camera views. Furthermore, the computational complexity of the proposed distributed approach from multiple cameras grows linearly with the number of cameras and objects in each camera. The resulting experiments further demonstrate the superiority of our approach to deal with severe occlusions in the presence of frame loss compared with existing methods. In the second part of our work, we propose a novel statistical estimation algorithm to stochastic context-sensitive grammars (SCSGs). First, we show that the SCSGs model can be solved by decomposing it into several causal stochastic context-free grammars (SCFGs) models and each of these SCFGs models can be solved simultaneously using a fully synchronous distributed computing framework. An alternate updating scheme based approximate solution to multiple SCFGs is also provided under the assumption of a realistic sequential computing framework. A series of statistical algorithms are expected to learn SCFGs subsequently. Specific to our case, a hybrid of general Forward-Backward Algorithm, Inside Algorithm and Expectation-Maximization Technique will be used to estimate the parameters for SCFGs. The SCSGs can be then used to represent multiple-trajectory. Experimental results demonstrate the improved performance of our method compared with existing methods for multiple-trajectory classification. In the third part of our work, we propose a Compressed-Sensing Game Theory (CSGT) framework to solve the Nash equilibria. We demonstrate that the proposed CSGT framework provides a polynomial complexity solution to the Nash Equilibria, thus allowing more general pay-off functions for certain classes of two-player dynamic games. We also provide numerical examples that demonstrate the efficiency of proposed CSGT framework in solving the Nash equilibria for two-player games in comparison to existing algorithms.

CHAPTER 1

INTRODUCTION

1.1 Literature Review

1.1.1 Visual Tracking

Recent technological trends have required the deployment of surveillance and visual tracking applications in distributed smart camera networks driven by its extensive availability and applications such as human-computer interaction, human activity analysis, etc. Multiple object tracking is a challenging task because of the exponentially increased computational complexity in terms of the degrees of freedom of the object and the severe image ambiguities incurred by frequent self-occlusions. Compared to relatively independent movements in multiple object tracking, the motion of articulated object has some inherent constraints. For example, the constraints between analyzed part and its neighboring part. Monocular video is capable of single object tracking and multiple isolated object tracking. However, multiple object tracking and articulated motion analysis usually have to deal with multi-object occlusions. Monocular video has intrinsic limitations in solving occlusions, due to the limited field of single view. There is a growing need to obtain more information by increasing the number of cameras (1). Tracking with multiple cameras not only expands coverage of the environment, but also helps to disambiguate in matching when subjects are occluded from a certain viewing angle. In many applications, bandwidth constraints, security concerns, and difficulty in storing and analyzing large amounts of data centrally at a single location necessitate the development of distributed camera networks. Thus, the development of distributed scene-analysis algorithms is highly required. Following this trend, the extensive availability of distributed camera networks calls for a robust and efficient algorithm for object tracking from degraded visual data that utilizes limited computational resources. For instance, video tracking from hand-held cameras or cameras mounted on mobile platforms is often required to track objects based on video sequences with missing frames due to acquisition and network errors. The unstable wireless connectivity and narrow bandwidth in mobile phones have made some frames unavailable and thus imposed tremendous constraints on our ability to deploy state-of-the-art tracking systems on mobile platforms. Therefore, robust and self-healing tracking algorithms which can deal with random frame loss are very desirable in such a lossy environment. Motivated by these practical applications, we seek to develop a robust and self-healing tracking algorithm to address missing frames on multi-camera platforms.

In our work, we propose a novel distributed framework of multiple object tracking from multiple collaborative cameras. The main contributions of this work can be summarized as follows.

- To the best of our knowledge, the distributed visual tracking framework in this work is the first distributed graph-based technique especially designed for smart camera networks. The resulting distributed visual tracking technique is therefore robust to occlusion and sensor errors from specific camera views. Furthermore, the computational complexity of the proposed distributed approach from multiple cameras grows linearly with the number of cameras and objects in each camera.

- Instead of using centralized joint data association for complex objects, we propose to adopt multiple collaborative trackers for multiple objects, which is a good choice for a smart camera. We first introduce an efficient decomposed inter-part interaction model and propose a novel graph-based particle filtering (GBPF) framework. To handle severe self-occlusion, high-level inter-unit interactions are further formulated; A novel hierarchical graph-based particle filtering (HGBPF) framework is proposed within one camera.
- We develop new distributed camera networks to avoid centralized approaches to visual tracking from multiple cameras. An additional likelihood density called a camera collaboration likelihood is introduced to characterize the collaboration between the same objects counterparts in different views. A novel camera collaboration model is proposed within collaborative particle filters framework, which is a good choice for smart camera networks.

As our distributed graph-based visual tracking framework from multiple camera views is derive from tracking technique with only one camera. We first provide a brief summary of previous work from one view. Since the visual information from a single fixed camera is quite limited, there is a growing interest in obtaining more information by increasing the number of cameras. Then a brief overview of visual tracking on multi-camera platform is also provided. Some related work of graphical models is also presented in this section.

Most early efforts for multiple object tracking use monocular video. A widely accepted approach that addresses many problems in this difficult task is based on a joint state-space representation and infers the joint data association (2) (3). Various improvements have been made, such as joint probabilistic data association filter for tracking complex visual objects (4). MacCormick and Blake (5) used a binary variable to identify foreground objects and proposed a probabilistic exclusion principle to penalize the hypothesis where two objects occlude. Isard and MacCormick combined a multi-blob likelihood function with the condensation filter and used a 3D object model providing depth ordering to solve the multiple-object occlusion problem (6). Tao et al. (7) proposed a sampling-based multiple-object tracking method using background subtraction. Khan et al. (8) proposed a MCMC-based particle filter which uses a Markov random field to model motion interaction. Although the above algorithms, which are based on a centralized process, can handle the problem of multiple-object occlusion in principle, they require a tremendous computational cost due to the complexity introduced by the high dimensionality of the joint-state representation which grows exponentially in terms of the number of objects tracked. Therefore, various decentralized solutions have been proposed for multiple object tracking. The decentralized approach was carried by Qu et al. (9) who proposed an interactively distributed multiple-object tracking framework using a magnetic-inertia potential model.

Compared to relatively independent movements in multiple object tracking, the motion of articulated object has some inherent constraints. Many approaches have been studied to avoid the problems inherent in articulated object tracking. Most early efforts of articulated motion analysis took advantages of 2D and 3D object models (10). A unified spatio-temporal articulated model was proposed by Lan and Huttenlocher (11). Kalman filters have been employed by many researchers to combat occlusions in articulated object tracking (12). Sequential Monte Carlo method or particle filter was demonstrated to be efficient for object tracking in clutter (10) and has also been introduced for articulated motion analysis. Deutsher et al. (13) modified the Condensation algorithm (10) by an annealed particle filter. Choo and Fleet (14) described a filter that used hybrid Monte Carlo to obtain efficient samples in high-dimensional spaces. Chang et al. (15) proposed an appearance-based particle filter for articulated hand tracking. The successful application of particle filtering was limited to situations where the dimension of the joint state is relatively small. For high-dimensional state spaces, many algorithms become computational inefficient.

Although various decentralized solutions have been proposed to deal with the problem of occlusion from only one camera view, monocular video has intrinsic limitations for multiple object tracking. Since the visual information from a single fixed camera is quite limited, there is a growing interest in obtaining more information by increasing the number of cameras (1) (16). Tracking with multiple cameras not only expands coverage of the environment, but also helps to disambiguate in matching when subjects are occluded from a certain viewing angle. However, using multiple cameras raises many additional challenges. The most critical difficulties presented by multi-camera tracking are to establish a consistent label correspondence of the same object among the different views and to integrate the information from different camera views for tracking that is robust to significant and persistent occlusion. Many existing approaches address the label correspondence problem by using different techniques such as feature matching (17), camera calibration and 3D environment model (18). Integration of information from multiple cameras to solve the multiple object occlusion problem has been developed. Static and active cameras are used together in (19). Iwase et al. (20) integrated the tracking data of soccer players from multiple cameras by using homography and a virtual ground image. A particle filter-based approach is presented by Gatica-Perez et al. (21) for tracking multiple interacting people in meeting rooms. Nummiaro et al. (22) proposed a color-based object tracking approach with a particle filter implementation in multi-camera environments. Recently, Du and Piater (23) presented a very efficient algorithm using sequential belief propagation to integrate multi-view information for a single object in order to solve the problem of occlusion with clutter.

Recently, probabilistic graphical models have been used to facilitate the analysis of high dimensionality signal processing problems. It provides a more simple and distinct way to visualize the structure of the probability model. For example, variational analysis methods (24) (25) are generally used to obtain approximate inference for loopy Markov networks. They provide lower bounds of the approximation as a theoretical benefit. In contrast to variational analysis methods, loopy belief propagation (26) often converges and when it do, it gives a better approximation. Furthermore, Bayesian filtering framework has become very popular for object tracking. It provides a recursive formulation of the posterior probability density function in dynamical systems. Analytical solutions for the optimal Bayesian filtering problem are known only for special cases including the linear and gaussian case (Kalman filter (27)). Particle filters (10) (28) (29) (30) provide a general framework for estimating the probability density function of general non-linear and non-Gaussian systems. They are based on a Monte Carlo approach, where the density is represented by a set of random samples. Samples can be drawn from any distribution called the proposal density or the importance function, but sample weights should be properly adjusted so that the sample set fairly approximates the posterior density. Particle filtering on general graphs is first proposed by (31). They split the general graph into multiple cycle-free subgraphs and apply the filtering algorithm on cycle-free graphs in a distributed way. However, graph-based particle filtering does not incorporate stochastic missing frames behavior into the estimation. Therefore, robust particle filtering algorithm to address random frame loss due to acquisition and network errors on multi-camera platform is very desirable.

1.1.2 Video Classification

The application of the grammars to syntactic pattern recognition has received tremendous attention in recent years due to its wide applicability in diverse areas such as gesture recognition, biological sequence analysis, speech signal processing, etc. Finite-state grammars (FSGs), also known as hidden Markov models, have achieved a great success in the speech community (32). They were used in modern tracking systems (33) and in machine vision (17). On the other hand, context-free grammars (CFGs) are studied in (34) for gesture recognition and implementation of an online parking lot monitoring task. In (35) they were used in modeling the dynamics of a bursty wireless communications channel. Reference (36) applies these syntactic models to the study of biological sequence analysis and RNA. Finally, application of syntactic modeling in pattern recognition is covered in depth in (37). Among the many grammars and languages that have been investigated for practical applications, FSGs and CFGs, as well as their stochastic counterparts, stochastic finite-state grammars (SFSGs) and stochastic context-free grammars (SCFGs), are currently the most widely used classes of grammars. In general, Bayesian estimation of SFSGs and SCFGs are polynomial complexity, whereas this is not true for stochastic context-sensitive grammars (SCSGs). In general, estimation of context-sensitive grammars (CSGs) can not be solved in close form. In particular, the learning algorithms for finite-state grammars (FSGs) and context-free grammars (CFGs) rely on the causality of the model. However, CSGs can not be solved in exact form since such a model is inherently non-causal. Previous works have focused on formulating other classes of grammars lie between the CFGs and CSGs (38). In our preliminary work, we propose learning algorithms on SCSGs directly.

Object trajectory-based analysis and recognition has gained significant interest in scientific research recently. This trend can be attributed mainly to several different reasons. First, object trajectory data is becoming more easily available due to advances in sensor technology and computing techniques (15). On the hardware side, advancements in sensor technology are resulting in low-cost versatile sensors. On the software side, advancements in computer vision have led to the design of robust object trackers that can handle occlusions, shape deformations and intensity changes in single- and multi- camera settings. Second, moving object trajectories are often used to describe and analyze an object's behavior. Motion trajectories provide rich spatio-temporal information about an object's activity. Abnormal behaviors can be detected by comparing newly detected moving object trajectories with normal trajectories. Doing so requires a parsimonious representation of object trajectories. Such a representation could be provided by stochastic grammars (39).

Furthermore, developing high-accuracy activity classification and recognition algorithms using motion trajectories is still an extremely challenging task particularly when the number of activities to be recognized is relatively large. Multi-class classification problem is inherently challenging. Chance classification of n classes is 1/n, as n goes to infinity, chance accuracy goes to 0. The object trajectory is typically modeled as a sequence of consecutive locations of the object on a coordinate system resulting in a vector in 2-D or 3-D Euclidean space. The measurement parameters, at each point in time, needed for object localization can be arbitrarily high-dimensional vectors including x and y- projections, distance, silhouette of the object shape, and other data corresponding to object appearance and environment. Developing scalable activity recognition algorithms based on this high dimensionality cue is an extremely challenging task.

Our work is focused on stochastic context-sensitive grammars (SCSGs) learning and presents a series of statistical estimation approaches to estimate the parameters for stochastic grammars. To the best of our knowledge, this is the first work to provide learning algorithms on SCSGs. There are two constraints in learning SCSGs: one is how to split complex grammars into simpler ones that can be analyzed; the other one is to determine a discrete structure (topology) of the grammar and estimate probabilistic parameters in a simpler grammar. In this work, SCSGs models are first decomposed into several causal SCFGs, and a approximate simultaneous solution to each of these distributed causal SCFGs is provided by adopting an alternate updating scheme. Our proposed model is then successfully applied to multiple-trajectory classification.

A series of algorithms are expected to learn stochastic grammars for trajectories and predict their categories based on learned stochastic grammars.

1.1.3 Compressed sensing game theory

Game theory is a mathematical model describes and analyzes scenarios with interactive decisions. In recent years, there has been a growing interest in adopting cooperative and non-cooperative game theoretic approaches to model many communications and networking problems, such as cognitive radio systems, sensor networks, defense networks and gene regulatory networks (40). Many of these applications employ solution concepts such as correlated equilibria and Nash equilibria. Nash equilibria can capture decision balance among all players at the expense of computation. Correlated equilibria extends the Nash equilibria and are benign to solve. Although widely used and computationally less expensive, the correlated equilibrium could be too "broad". Furthermore, the true correlations among the players could be neglected for the solutions. Compared with correlated equilibria, the Nash equilibria assume that agents act independently and have received great attention in the signal processing and communication communities (41). The existence of Nash equilibrium requires essential use of Brouwer fixed point theorems (42) and in general, any algorithm solving the fixed point problem would unconditionally require an exponential number of function evaluations. The particular path following algorithm developed by Lemke and Howson (43) was recently proven to require, even in the best case for some instances, an exponential number of steps (44). The general problem for solving Nash equilibria has been shown PPAD-complete (45), which is currently lack of general efficient algorithm. Instead, research attentions for efficient algorithm have been put on various special classes of the problem. For example, the problem of computing a Nash equilibrium in a two-player zero-sum game is solvable in polynomial time by Khachiyan's ellipsoid algorithm (46). The problem with convex payoff functions can be solved by convex optimization tools such as interior point method (47). However, it is still unknown whether we can have efficient algorithm if we have situations other than those special cases.

Compressed sensing is a signal processing technique for efficiently acquiring and reconstructing a signal, by finding solutions to under-determined linear systems. This takes advantage of the signal's sparseness in some domain, allowing the entire signal to be determined from relatively few measurements. Donoho showed that the number of linear equations can be small and still contain nearly all the information to reconstruct the signal (48).

CHAPTER 2

DISTRIBUTED GRAPH-BASED PARTICLE FILTERING FRAMEWORK FOR SINGLE AND MULTIPLE COLLABORATIVE CAMERAS IN LOSSY NETWORKS

2.1 Theoretical Foundations

This section describes theoretical framework of our graph-based algorithm for distributed smart camera networks, including the stochastic dynamical models, particle filtering and graphical models.

2.1.1 Stochastic Dynamical Models

To define the problem of tracking, the propagation process must be set out in terms of discrete time t. The state of the object in the model is denoted x_t, and its history is $x_{0:t}$. Similarly, the set of measurements (e.g., color histogram and edge) at time t is z_t with history $z_{1:t}$.

A somewhat general assumption is made for the probabilistic framework that the object dynamics formulate a temporal first-order Markov chain. The new state x_t depends only on the immediately preceding state, independent of the earlier history. Therefore, we have

$$p(x_t|x_{t-1}, x_{t-2}, \ldots, x_0) = p(x_t|x_{t-1}) \tag{2.1}$$

In this case, the dynamics are entirely determined by the form of the conditional density $p(x_t|x_{t-1})$. However, the first-order Markov model cannot accurately characterize the dynamics of moving objects. Therefore, this still allows more general and accurate dynamics. To address this problem, a more general assumption is adopted for the probabilistic framework that the object dynamics form a high-order Markov chain (49) so that

$$p(x_t|x_{t-1}, x_{t-2}, \ldots, x_0) = p(x_t|x_{t-1}, \ldots, x_{t-m}) \tag{2.2}$$

—the new state is conditioned directly on the immediately m preceding state. Both first- and high-order hidden Markov models are mainly used for single object tracking. Recently, graph-based sequential particle filtering for conditional density propagation is performed by a sequential updating scheme in a predetermined order in (31). Compared with first- and high-order particle filtering, graph-based sequential particle filtering can be used to many emerging multiple object tracking applications.

In our work, we establish a distributed graph-based framework of particle filtering in dealing with frame loss for these stochastic dynamical models on multi-camera platform. This distributed framework has the striking property that, of course, it is more useful for realistic problems. For example, in a wireless lossy network formed by several camera-equipped mobile phones, it is quite easy to lose some visual data during the video transmission process, thus make it ineffective to use existing methods (10)(49)(31).

2.1.2 Particle Filtering

From a Bayesian perspective (28), the tracking problem is to recursively calculate some degree of belief in the state x_t at time t, taking different values, given the data $z_{1:t}$ up to time t. Then, the conditional state density $p(x_t|z_{1:t})$ may be obtained, recursively, in two stages: prediction and update. In this section, a first-order markovian discrete-time state space model is assumed. The rule for propagation of state density over time is

$$p(x_t|z_{1:t}) = k_t p(z_t|x_t)p(x_t|z_{1:t-1}) \tag{2.3}$$

where

$$p(x_t|z_{1:t-1}) = \int_{x_{t-1}} p(x_t|x_{t-1})p(x_{t-1}|z_{1:t-1}) \tag{2.4}$$

Particle filters (10) have been proposed as a nonlinear and non-Gaussian method for Bayesian estimation, without incurring excessive computational load. We denote $\{x_{0:t}^i, w_t^i\}_{i=1}^N$ as a random measure that characterizes the posterior pdf $p(x_{0:t}|z_{1:t})$, where $\{x_{0:t}^i\}$ is a set of N particles with associated weights $\{w_t^i, i = 1, \ldots, N\}$. Therefore,

$$p(x_{0:t}|z_{1:t}) \approx \sum_{i=1}^{N} w_t^i \delta(x_{0:t} - x_{0:t}^i) \tag{2.5}$$

where

$$w_t^i \propto w_{t-1}^i \frac{p(z_t|x_t^i)p(x_t^i|x_{t-1}^i)}{q(x_t^i|x_{t-1}^i, z_t)} \tag{2.6}$$

In (2.6), $p(z_t|x_t^i)$ is the likelihood and $p(x_t^i|x_{t-1}^i)$ is the transition probability and $q(x_t^i|x_{t-1}^i, z_t)$ is the proposal density. Therefore, the posterior filtered density can be approximated as

$$p(x_t|z_{1:t}) \approx \sum_{i=1}^{N} w_t^i \delta(x_t - x_t^i) \tag{2.7}$$

The normalized weights π_t^i are given by

$$\pi_t^i = \frac{w_t^i}{\sum_{j=1}^{N} w_t^j} \tag{2.8}$$

The state estimate $\hat{x_t}$ is given by the sample mean

$$\hat{x_t} = \sum_{i=1}^{N} \pi_t^i x_t^i \tag{2.9}$$

Parallel with the derivation of particle filtering for the first-order markovian discrete-time state space model, particle filters for the high-order Markov chain and general graphical models could be easily derived.

2.1.3 Graphical Models

In our graph representation, each node represents a hidden state x_i and is linked to an observation z_i. We use V to denote the set of hidden states in graph G, i.e., $V = \{x_1, x_2, ...\}$.

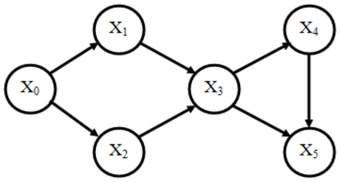


Figure 1. Example of a directed cycle-free graph (the observation layer is omitted).

Set V could be partitioned into disjoint sets, called layers, $\{V_l\}_{l=0}^{L}$. The number of elements in V_l is $K(l)$. Let us use $V_{0:l} = \{V_0, V_1, ...V_l\}$ to denote the collection of child sets up to order l, where we define the $0th - order$ child set V_0 as the set of nodes that have no parents. Furthermore, we use the notation $v_{m,l}$ to denote the mth node in V_l, $m = 1, 2, ..., K(l)$. The parent of $v_{m,l}$ is denoted as $Pa(v_{m,l})$. It could be easily seen that $Pa(v_{m,l}) \subseteq V_{0:l-1}$. The observation associated with $v_{m,l}$ is denoted as $o_{m,l}$, and $O_l = \{o_{m,l}, m = 1, 2, ..., K(l)\}$. We denote the order of node $v_{m,l}$ as $S_{(v_{m,l})}$. For the nodes within the same layer, the order could be arbitrarily assigned. Although therefore the order of the nodes in a graph will have many combinations, we only use one predetermined order for a graph. For example, one possible order of the graph in Fig. 1 is: $x_0, x_1, x_2, x_3, x_4, x_5$, where $S(x_0) = 1$, $S(x_3) = 4$, $S(x_5) = 6$.

For example, in Fig. 1, $V = \{x_0, x_1, x_2, x_3, x_4, x_5\}$, $V_0 = \{x_0\}$, $V_1 = \{x_1, x_2\}$, $V_2 = \{x_3\}$, $V_3 = \{x_4\}$ and $V_4 = \{x_5\}$; $v_{1,0} = x_0$, $v_{1,1} = x_1$, $v_{2,1} = x_2$, $v_{1,2} = x_3$, $v_{1,3} = x_4$, $v_{1,4} = x_5$; $Pa(v_{1,2}) = \{x_1, x_2\}$.

2.2 Distributed Graph-Based Particle Filtering Framework from Monocular View

2.2.1 Graph-Based Particle Filtering (GBPF) Framework

Multiple object tracking is a challenging task because of the exponentially increased computational complexity in terms of the degrees of freedom of the object and the severe image ambiguities incurred by frequent self-occlusions. Compared to relatively independent movements in multiple object tracking, the motion of articulated object has some inherent constraints. A good example of such constraints appears in the human body. It can be observed that there is a common relationship among arms and legs. Left arm moves forward while left leg moves backward. We start to derive a distributed graph-based framework from one view for articulated object tracking. When we apply this framework to multiple object tracking, we ignore the inherent constraints between complex parts and consider comparatively independent movements among multiple objects. We first introduce a decomposed inter-part interaction model in this section. Then a distributed graph-based particle filtering framework is formulated in this section. The proposed distributed graph-based framework is a good choice for a smart camera in camera networks.

2.2.1.1 Graphical Models for Articulated Object Representation

An articulated object can be represented by a graphical model in Fig. 2. The Fig. 2 can also be applied to multiple object tracking. It has two layers: the hidden state layer (circle nodes) and the observation layer (square nodes). Each circle node corresponds to a part of the articulated object. For instance, considering a human body, a part can be a torso, or a thigh,

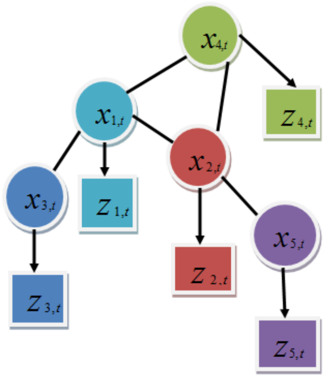


Figure 2. Graphical model for an articulated object at time t.

etc. The undirected links represent physical constraints among different articulated parts. Each individual part is associated with its observation. The directed link from a part's state to its associated observation represents the local observation likelihood. Instead of using the joint state representation for the whole articulated object, we denote the state of each part at time t by $x_{i,t}$, where $i = 1, 2, ..., M$ is the index of parts. In our implementation, the state $x_{i,t}$ is chosen as $x = (cx, cy, b, \Theta)$, where (cx, cy) is the center point; b is half of the length of the rectangle; Θ is the rotation angle of state around the center point with respect to the Y axis. The ratio of the length and width of the rectangle is held constant equal to its value obtained in the first frame. We denote the observation of $x_{i,t}$ by $z_{i,t}$.

The inter-part interaction density $p(x_{j,t} \mid x_{i,t})$ models the constraints between analyzed part i and its neighboring part j. Estimation of this density should adapt to different applications and is usually critical in practical implementation. To avoid high computational requirements for a joint state representation model, we develop an efficient decomposed inter-part interaction model based on (50). It can be observed that the relative locations and poses of two adjacent parts are independent. Therefore, by temporarily discarding the time index, we have

$$\begin{aligned} p(x_{j,t} \mid x_{i,t}) \quad &= p(cx^j, cy^j, b^j, \Theta^j | cx^i, cy^i, b^i, \Theta^i) \\ &= p(cx^j, cy^j \mid cx^i, cy^i) p(\Theta^j \mid \Theta^i) p(b^j \mid b^i) \end{aligned} \tag{2.10}$$

Where we assume that the size of an object part is not influenced by its neighboring parts. Without considering the size relation between two parts, $p(b^j \mid b^i)$ becomes uniformly distributed. Thus, we can further simply (2.10) to be

$$p(x_{j,t} \mid x_{i,t}) \propto p(\mathbf{c}^j \mid \mathbf{c}^i) p(\Theta^j \mid \Theta^i) \tag{2.11}$$

Where $\mathbf{c}^i = (cx^i, cx^i), \mathbf{c}^j = (cx^j, cx^j)$ are the coordinates of the center points. $p(\mathbf{c}^j \mid \mathbf{c}^i)$ models the location interaction of two adjacent parts. We adopt a "spring-joint" model similar to (50) for $p(\mathbf{c}^j \mid \mathbf{c}^i)$.

$$p(\mathbf{c}^j \mid \mathbf{c}^i) = \frac{1}{2\pi |\Sigma_{\mathbf{c}}|^{1/2}} \exp\{-1/2(\mathbf{c}^j - \mathbf{c}^i)^T \Sigma_{\mathbf{c}}^{-1} (\mathbf{c}^j - \mathbf{c}^i)\} \tag{2.12}$$

Where $\Sigma_{\mathbf{c}}$ is the covariance matrix of this bivariate normal distribution. In (2.10), $p(\Theta^j \mid \Theta^i)$ models the pose relation of two adjacent parts. It can be estimated either by some prior knowledge in particular applications, or by learning from training data.

2.2.1.2 Posterior Density Propagation And Sequential Importance Sampling

In our graph representation, we use V to denote the set of hidden states in graph G, i.e. $V = \{x_0, x_1, x_2, ...\}$. Set V could be partitioned into disjoint sets, called layers, $\{V_l\}_{l=0}^{L}$. The number of elements in V_l is $K(l)$. Let us use set V_0 to denote the set of nodes that have no parents. Furthermore, we use the notation $v_{m,l}$ to denote the mth node in V_l, $m = 1, 2, ..., K(l)$. The parents of $v_{m,l}$ can be denoted as $Pa(v_{m,l})$. It could be easily seen that $Pa(v_{m,l}) \subseteq V_{0:l-1}$. The observation associated with $v_{m,l}$ is denoted as $o_{m,l}$, and $O_l = \{o_{m,l}, m = 1, 2, ..., K(l)\}$. We denote the order of node $v_{m,l}$ as $S(v_{m,l})$. Although therefore the order of the nodes in a cycle-free graph will have many combinations, we only use one predetermined order.

The nodes in set V_0 do not have any parents, and the conditional probability is given as prior $p(v_{m,0})$. For the node $v_{m,l}$ in V_l, we formulate the conditional density $p(v_{m,l}, V_{0:l-1}|o_{m,l}, O_{1:l-1})$ as

$$p(v_{m,l}, V_{0:l-1}|o_{m,l}, O_{1:l-1}) \propto p(o_{m,l}|v_{m,l})p(v_{m,l}|Pa(v_{m,l}))p(V_{0:l-1}|O_{1:l-1}) \qquad (2.13)$$

We further use sequential importance sampling technique (10) again as the paradigm. We denote $\{\{v_{m,l}, V_{0:l-1}\}^i, w^i_{S(v_{m,l})}\}_{i=1}^{N}$ as a random measurement that characterizes the posterior density in (2.13), where $\{v_{m,l}, V_{0:l-1}\}^i$ is a set of N particles with associated weights $w^i_{S(v_{m,l})}$, $i = 1, 2, \ldots, N$. The weights are normalized such that $\sum_{i=1}^N w^i_{S(v_{m,l})} = 1$.

Therefore,

$$p(v_{m,l}, V_{0:l-1}|o_{m,l}, O_{1:l-1}) \approx \sum_{i=1}^{N} w^i_{S(v_{m,l})} \delta(\{v_{m,l}, V_{0:l-1}\} - \{v_{m,l}, V_{0:l-1}\}^i) \tag{2.14}$$

where the weights are given by

$$w^i_{S(v_{m,l})} \propto \frac{p(\{v_{m,l}, V_{0:l-1}\}^i|o_{m,l}, O_{1:l-1})}{q(\{v_{m,l}, V_{0:l-1}\}^i|o_{m,l}, O_{1:l-1})} \tag{2.15}$$

By using the conditional independence properties (51), we obtain $q(v_{m,l}, V_{0:l-1}|o_{m,l}, O_{1:l-1}) = q(v_{m,l}|o_{m,l}, Pa(v_{m,l}))q(V_{0:l-1}|O_{1:l-1})$. The weight update equation can be further obtained as follows:

$$w^i_{S(v_{m,l})} \propto w^i_{S(v_{m,l})-1} \frac{p(o_{m,l}|v^i_{m,l})p(v^i_{m,l}|\{Pa(v_{m,l})\}^i)}{q(v^i_{m,l}|\{Pa(v_{m,l})\}^i, o_{m,l})} \tag{2.16}$$

In (2.16), $p(o_{m,l}|v^i_{m,l})$ is the likelihood, and $p(v^i_{m,l}|\{Pa(v_{m,l})\}^i)$ captures the relationship between the particles of the current node and its parents, and $q(v^i_{m,l}|\{Pa(v_{m,l})\}^i, o_{m,l})$ is the proposal density.

In applications, we still consider one filtered estimate at each time step. Therefore, we have

$$p(v_{m,l}|o_{m,l}, O_{1:l-1}) \approx \sum_{i=1}^{N} w^i_{S(v_{m,l})} \delta(v_{m,l} - v^i_{m,l}) \tag{2.17}$$

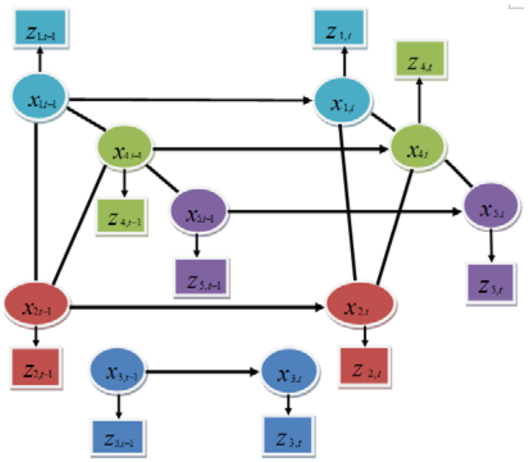


Figure 3. Graphical model of articulated object tracking.

The graphical model of articulated object tracking is given in Fig. 3. The graphical model in Fig. 3 can also be applied to multiple object tracking if we ignore the inherent constraints between complex parts and consider comparatively independent movements among multiple objects. The notation $x_{i,t}$ represents the hidden state of part i at time t, and $z_{i,t}$ is the observation associated to it. The undirected link between circle nodes represents the interaction among parts, e.g. self-occlusion. When a graph has cycles, e.g. parts 1, 2 and 4, we could split the graph into multiple cycle-free subgraphs by following the approach presented in (31) and apply the particle filtering algorithm on cycle-free graphs in a distributed way.

From (2.16), (2.17) and the graphical model of articulated object tracking, we get

$$p(x_{i,t}|z_{i,t}, z_{1:\{S(x_{i,t})-1\}}) \approx \sum_{n=1}^{N} w_{i,t}^{n} \delta(x_{i,t} - x_{i,t}^{n}) \tag{2.18}$$

The weights are normalized such that $\sum_{n=1}^{N} w_{i,t}^{n} = 1$. The weight update equation can be further obtained as follows:

$$w_{i,t}^{n} \propto w_{S(x_{i,t})-1}^{n} \frac{p(z_{i,t}|x_{i,t}^{n})p(x_{i,t}^{n}|\{Pa(x_{i,t})\}^{n})}{q(x_{i,t}^{n}|\{Pa(x_{i,t})\}^{n}, z_{i,t})} \tag{2.19}$$

In (2.19), $p(x_{i,t}^{n}|\{Pa(x_{i,t})\}^{n})$ captures the relationship between the particles of the current node and its parents. Depending on whether there are interactions among different parts of articulated objects at the same time, $p(x_{i,t}^{n}|\{Pa(x_{i,t})\}^{n})$ could be divided into two categories as follows:

1. If there are no interactions (e.g., occlusions) among different parts at the same time (e.g., part 3), $p(x_{i,t}^{n}|\{Pa(x_{i,t})\}^{n}) = p(x_{i,t}^{n}|x_{i,t-1}^{n})$ is determined by the dynamics of the isolated part, which is usually considered as a random walk or learned from training data. Therefore, (2.19) becomes

$$w_{i,t}^{n} \propto w_{S(x_{i,t})-1}^{n} \frac{p(z_{i,t}|x_{i,t}^{n})p(x_{i,t}^{n}|x_{i,t-1}^{n})}{q(x_{i,t}^{n}|\{Pa(x_{i,t})\}^{n}, z_{i,t})} \tag{2.20}$$

2. If different parts of articulated object interact at time t (e.g., parts 1, 2, 4 and 5), we have $p(x_{i,t}^{n}|\{Pa(x_{i,t})\}^{n}) = p(x_{i,t}^{n}|x_{i,t-1}^{n}, \{ln(x_{i,t})\}^{n})$, where $ln(x_{i,t})$ represents the interacting parts of x_i at time t. Because interacting parts must compete for limited observations, we therefore model this probability as $p(x_{i,t}^{n}|x_{i,t-1}^{n}, \{ln(x_{i,t})\}^{n}) \propto p(x_{i,t}^{n}|x_{i,t-1}^{n})\phi(x_{i,t}^{n}, \{ln(x_{i,t})\}^{n})$. We define $\phi(x_{i,t}^{n}, \{ln(x_{i,t})\}^{n})$ as $\phi(x_{i,t}^{n}, \{ln(x_{i,t})\}^{n}) = \prod_{x_{j,t} \in ln\{x_{i,t}\}} \{\sum_{m=1}^{N_j} w(x_{j,t}^{m})p(x_{j,t}^{m}|x_{i,t}^{n})\}$,

where N_j is the total number of samples of part j. The density $p(x_{j,t}^m|x_{i,t}^n)$ models the interaction between two neighboring parts' samples $x_{j,t}^m$ and $x_{i,t}^n$. $w(x_{j,t}^m)$ acts as a weight to the associated interaction. They work together to constrain the neighboring parts and prevent different parts of the articulated object from separating over time. Therefore, (2.19) becomes

$$w_{i,t}^n \propto w_{S(x_{i,t})-1}^n \frac{p(z_{i,t}|x_{i,t}^n)p(x_{i,t}^n|x_{i,t-1}^n)}{q(x_{i,t}^n|\{Pa(x_{i,t})\}^n, z_{i,t})} \times \prod_{x_{j,t} \in ln\{x_{i,t}\}} \{\sum_{m=1}^{N_j} w(x_{j,t}^m)p(x_{j,t}^m|x_{i,t}^n)\} \tag{2.21}$$

2.2.2 Hierarchical Graph-Based Particle Filtering (HGBPF) Framework

The interaction inside an articulated object lies not only in the adjacent parts but also some "high-level" nonadjacent "part groups". For clarity, we define a group of parts as a unit, which is denoted by $X_{I,t}$, where $I = 1, ..., M'$; M' is the total number of units. For instance, each limb of a human body contains two parts and can, thus, be regarded as a unit. Similar to the model in Fig. 2, but considering the "high-level" unit interaction as well, we represent the same articulated object in Fig. 2 by a hierarchical graphical model as illustrated in Fig. 4. Compared with the model in Fig. 2, the difference of this hierarchical model is that it introduces a high-level layer containing big blue ellipse nodes and red curve links. Each big ellipse node corresponds to a unit of the articulated object. The undirected curve links between units represent "high-level" interaction. We denote the related neighboring units of $X_{I,t}$ by $ln(I)$, the joint state of all these related neighboring units by $X_{ln(I),t} = \{X_{K,t}, K \in ln(I)\}$, and the corresponding observations by $Z_{ln(I),t} = \{Z_{K,t}, K \in ln(I)\}$. We assume that $p(X_{ln(I),t}|x_{i,t}) = p(X_{ln(I),t}|X_{I,t})$.

This assumption assumes all the parts $x_{i,t} \in X_{I,t}$ share the same "relation" with the neighboring units $X_{ln(I),t}$, which is the interaction between high-level units $X_{ln(I),t}$ and $X_{I,t}$.

Compared with the graph-based particle filtering framework in the previous subsection, hierarchical framework in this subsection introduces an additional high-level inter-unit weighting factor. We can also use the sequential Monte Carlo method (10) to approximate the conditional density propagation rule derived in the previous section. Therefore, (2.19) becomes

$$\begin{aligned} & w_{i,t}^{n} \propto w_{S(x_{i,t})-1}^{n} \frac{p(z_{i,t}|x_{i,t}^{n})p(x_{i,t}^{n}|x_{i,t-1}^{n})}{q(x_{i,t}^{n}|\{Pa(x_{i,t})\}^{n}, z_{i,t})} \\ & \times \prod_{x_{j,t} \in ln\{x_{i,t}\}} \{\sum_{m=1}^{N_j} w(x_{j,t}^{m}) p(x_{j,t}^{m}|x_{i,t}^{n})\} \\ & \times \prod_{K \in ln(I)} \{\sum_{l=1}^{N_K} w(X_{K,t}^{l}) p(X_{K,t}^{l}|X_{I,t}^{n})\} \end{aligned} \tag{2.22}$$

Where n is the sample index of part i (unit I), m is the sample index of part j, l is the sample index of unit K, N_j is the total sample number of part j, and N_K is the total sample number of unit K. The density $p(X_{K,t}^{l}|X_{I,t}^{n})$ models the interaction between two neighboring units' samples $X_{K,t}^{l}$ and $X_{I,t}^{n}$. $w(X_{K,t}^{l})$ acts as a weight to the associated interaction. By using Markov Properties (51), the $w(X_{K,t}^{l})$ can be further approximated by a normalized product of all parts' weights in unit K. Proposed distributed hierarchical graph-based framework from one view is given in Algorithm 1. For the one without hierarchy, we integrate interaction weights from neighboring parts as in (2.21) if neighboring parts exist.

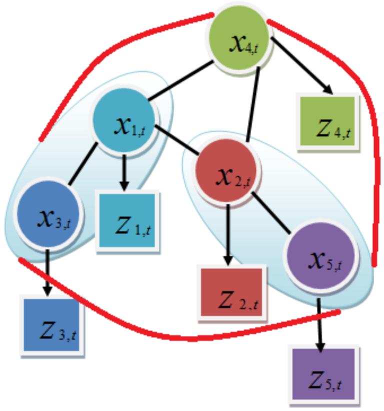


Figure 4. Hierarchical graphical model of an articulated object.

Algorithm 1 *Proposed distributed HGBPF from monocular view*

```
— for $l = 1, 2, ...$ \\ Layers
    — for $m = 1 : k(l)$
        ∗ If neighboring units or parts of $v^j_{m,l}$ exist
        ⋄ integrate interaction weights from neighboring
        units and neighboring parts as in (2.22).
        ⋄ estimate the conditional density for $v^j_{m,l}$ as in (2.17).
        ∗ Else \\ No interaction
        ⋄ calculate weights as in (2.20) and estimate the
        conditional density for $v^j_{m,l}$ as in (2.17).
    — end for $m$
— end for $l$
```

2.3 Distributed Graph-Based Particle Filtering Framework from Multiple Views in Lossy Networks

The extensive availability of distributed camera networks calls for a robust and efficient algorithm for object tracking from degraded visual data due to acquisition and network errors. We first present dynamic modeling of visual tracking from multiple collaborative views. We subsequently provide a distributed visual tracking solution from multiple cameras to handle occlusion problem. Then an exact solution for frame loss problem within one camera is presented for distributed smart camera networks. The resulting distributed visual tracking technique from multiple collaborative cameras is therefore robust to occlusion and sensor errors from specific camera views.

2.3.1 Dynamic Graphical Modeling of Multiple Object Tracking from Multiple Views

The graphical model is an intuitive and convenient tool to model and analyze complex dynamic systems. We illustrate the dynamic graphical model of two consecutive frames for multiple objects in two collaborative cameras in Fig. 5. The graphical model of two collaborative cameras in Fig. 5 can also be applied to articulated object tracking on multi-camera platform if we consider the inherent constraints between complex parts of the object. The notation $x_{1,t}^{A}$ represents the hidden state of object 1 at time t from camera A, and $z_{1,t}^{A}$ is the observation associated to it. The undirected link in each camera between states nodes represents the interaction among objects, e.g. occlusion. The directed link between consecutive states of the same object in each camera represents the state dynamics. The directed curve link between the counterpart states of the same object in two cameras represents the camera collaboration. This collaboration is activated between any possible collection of cameras only for objects which need help to improve their tracking performances. For instance, in Fig. 5, all objects in camera B at time t do not need to activate the camera collaboration because they do not interact with the other objects at all. In this case, each objects can be robustly tracked independently. On the other hand, objects 1 and 2 in camera A at time t activate camera collaboration since they interact and may undergo occlusion problem. Therefore, external information from other cameras may be helpful to make the tracking of these two objects more stable. As a result, we can incorporate the estimations of other cameras into the camera we focus on. Instead of storing and analyzing large amounts of data centrally at a single camera, our solution to the problem of visual tracking on multi-camera platform adopt a distributed camera networks, in which each camera can receive the projections from other cameras. Furthermore, our graphical model of two collaborative cameras in Fig. 5 can be directly applied to the camera networks with N cameras.

We then decompose the graphical model in Fig. 5 for every object in each camera by performing four steps: (1) each sub-model aims at one object in one camera; (2) for analysis of objects of a specific camera, only neighboring objects which have direct links to the analyzed object are kept. All the nodes of both nonadjacent objects and their observations are removed; (3) each undirected "interaction" link is decomposed into two different directed links from different objects. The direction of the link is from the analyzed object's state to the other object's observation; (4) since the camera collaboration link from a object's state in the analyzed

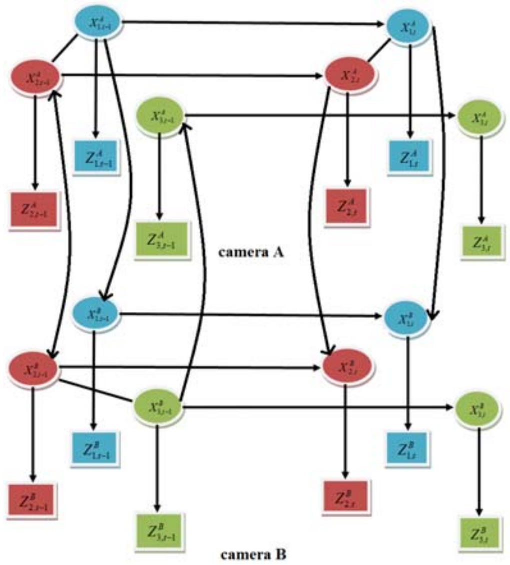


Figure 5. The general dynamic graphical model for multiple object tracking using multiple collaborative cameras.

camera view to its counterpart state in another view and the link from this counterpart state to its associated observation have the same direction, this causality can be simplified by a direct link from the grandparent node to its grandson as illustrated in Fig. 6.

2.3.2 Distributed Graph-Based Particle Filtering Framework from Multiple Views

2.3.2.1 Distributed Bayesian Formulation and Multi-Camera Correspondence

In this section, we present a generic statistical framework to model the interaction among cameras for multi-camera tracking. For multi-camera multi-object tracking, we will dynamically estimate the posterior based on observations from both the object and its neighbors in the

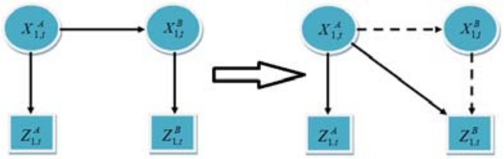


Figure 6. Equivalent simplification of camera collaboration link. The link causality from grandparent to parent then to grandson node is replaced by a direct link from grandparent to grandson node.

current camera view as well as its counterpart in other camera views. Without loss of generality, we illustrate our framework by using L cameras. We use the notation $v^j_{m,l}$ to denote the mth node in layer V_l in camera j, and $o^j_{m,l}$ is the observation associated to it, where $m = 1, 2, ..., K(l)$, $j = 1, 2, ...L$. We use the notation O_l and V_l to denote $O_l = \{o_{m,l}, m = 1, 2, ..., K(l)\}$ and $V_l = \{v_{m,l}, m = 1, 2, ..., K(l)\}$, respectively. Taking camera L as an example, the posterior density we estimate is $p(v^L_{m,l}, V^L_{0:l-1}|\{o^j_{m,l}, O^j_{1:l-1}\}^L_{j=1})$. Two assumptions are made in derivation: (i) we assume observations at different time are independent, both mutually and with respect to the dynamic process (10); (ii) given an object's state, the associated observations in different cameras are conditionally independent.

$$\begin{aligned}
& p(v^L_{m,l}, V^L_{0:l-1}|\{o^j_{m,l}, O^j_{1:l-1}\}^L_{j=1}) \\
= \quad & \frac{1}{k} p(o^L_{m,l}|v^L_{m,l}) p(v^L_{m,l}|Pa(v^L_{m,l})) \prod_{j=1}^{L-1} p(o^j_{m,l}|v^L_{m,l}) \cdot p(V^L_{0:l-1}|\{O^j_{1:l-1}\}^L_{j=1}) \qquad (2.23)
\end{aligned}$$

A novel likelihood density $\prod_{j=1}^{L-1} p(o^j_{m,l}|v^L_{m,l})$ called a "collaboration likelihood" is used to characterize the collaboration between the same object's counterparts in different views. When not activating the camera collaboration and regarding its projections in different views as independent, the proposed framework can be identical to the distributed approach presented in the previous section with one camera.

2.3.2.2 Collaborative Particle Filters

The collaborative particle filters framework is first proposed in (52). A novel camera collaboration model is proposed within collaborative particle filters framework in this section. Details are given below.

Taking camera L as an example, a particle set $\{\{v_{m,l}^L, V_{0:l-1}^L\}^i, w_{S(v_{m,l})}^{L,i}\}_{i=1}^N$ is employed to represent the posterior $p(v_{m,l}^L, V_{0:l-1}^L|\{o_{m,l}^j, O_{1:l-1}^j\}_{j=1}^L)$, where $\{v_{m,l}^L, V_{0:l-1}^L\}^i$ is a set of N particles with associated weights $w_{S(v_{m,l})}^{L,i}, i = 1, 2, \dots, N$. Therefore,

$$p(v_{m,l}^L, V_{0:l-1}^L|\{o_{m,l}^j, O_{1:l-1}^j\}_{j=1}^L) \approx \sum_{i=1}^N w_{S(v_{m,l})}^{L,i} \delta(\{v_{m,l}^L, V_{0:l-1}^L\} - \{v_{m,l}^L, V_{0:l-1}^L\}^i) \quad (2.24)$$

If $p(v_{m,l}^{L,i}|\{Pa(v_{m,l}^L)\}^i)$ is the proposal density how we generate the particles, the corresponding weights are given by

$$w_{S(v_{m,l})}^{L,i} \propto w_{S(v_{m,l})-1}^{L,i} p(o_{m,l}^L|v_{m,l}^{L,i}) \prod_{j=1}^{L-1} p(o_{m,l}^j|v_{m,l}^{L,i}) \quad (2.25)$$

In (2.25), the weights of local likelihood $p(o_{m,l}^L|v_{m,l}^{L,i})$ can be calculated by objects' color histogram (10), edge likelihood (10) and interaction between objects (31); The collaboration weights of camera collaboration likelihood $\prod_{j=1}^{L-1} p(o_{m,l}^j|v_{m,l}^{L,i})$ is evaluated by the model as follows:

In our distributed camera networks, the problem of visual tracking from multiple views can be solved with the independent visual tracking from each individual view. Taking camera L as an example, the reference camera L receives camera collaboration message from all other related cameras. Specifically, we rely on collaboration among the multiple cameras to provide an iterative refinement of the independent tracking performance obtained for each camera. Without loss of generality, a set of N particles is considered for each object in each camera. Particles $\{v_{m,l}^{j,k}\}_{k=1}^{N}$ in camera j ($j \in \{1, 2, ...L-1\}$) are roughly weighted as $\{\pi_{S(m,l)}^{j,k}\}_{k=1}^{N}$ by doing tracking according to local likelihood (31) in view j firstly. In this paper, we use $\pi_{S(m,l)}^{j,k}$ to denote normalized weight, while $w_{S(m,l)}^{j,k}$ is its raw counterpart. Then they are mapped to view L, producing $\Phi(v_{m,l}^{j,k})$, where $\Phi(\cdot)$ is a function of $v_{m,l}^{j,k}$ characterizing the epipolar geometry transformation. According to epipolar geometry theory (53), a point in one view can find an epipolar line in another view. Therefore, each particle $v_{m,l}^{j,k}$ in camera j can find an epipolar "band", i.e. $\Phi(v_{m,l}^{j,k})$ in camera L. After that, the collaboration likelihood can be calculated based on $\Phi(v_{m,l}^{j,k})$. Fig. 7 shows the procedure to calculate the collaboration weight for each particles based on $\Phi(v_{m,l}^{j,k})$ in camera L. The particles $\{v_{m,l}^{L,1}, \ldots, v_{m,l}^{L,N}\}$ are represented by circles. Only middle lines of the bands, i.e. $\{\Phi(v_{m,l}^{j,1}), \Phi(v_{m,l}^{j,2}), \ldots, \Phi(v_{m,l}^{j,N})\}$ are shown in the Fig. 7. The collaboration weight for particle $v_{m,l}^{L,i}$ can be computed as

$$\Psi_{S(m,l)}^{j,L}(v_{m,l}^{j,k}, v_{m,l}^{L,i}) \propto \exp(-dist^2(v_{m,l}^{L,i}, \Phi(v_{m,l}^{j,k}))) \tag{2.26}$$

$$\pi_{S(m,l)}^{L,i} \propto \prod_{j=1, j\neq L}^{L} \sum_{k=1}^{N} \pi_{S(m,l)}^{j,k} \Psi_{S(m,l)}^{j,L}(v_{m,l}^{j,k}, v_{m,l}^{L,i}) \tag{2.27}$$

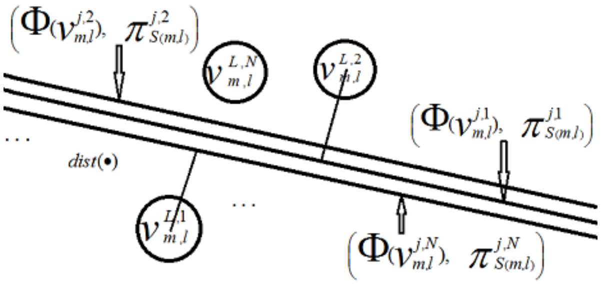

$\left(\Phi(v_{m,l}^{j,2}), \pi_{S(m,l)}^{j,2}\right)$
$v_{m,l}^{L,N}$
$v_{m,l}^{L,2}$
$\left(\Phi(v_{m,l}^{j,1}), \pi_{S(m,l)}^{j1}\right)$
...
$dist(\bullet)$
$v_{m,l}^{L,1}$
...
$\left(\Phi(v_{m,l}^{j,N}), \pi_{S(m,l)}^{j,N}\right)$

Figure 7. Calculating camera collaboration weights in camera L.

$dist(\cdot)$ denotes a *Point-Line distance* between $v_{m,l}^{L,i}$ and $\Phi(v_{m,l}^{j,k})$. The collaboration weights are then passed from camera L to all other related cameras in a similar way. In this case, we do iteration several times between different views to get more stable estimation.

2.3.3 Graph-Based Particle Filtering Framework with Missing Frames

The development of distributed camera networks calls for a robust and efficient algorithm for object tracking from degraded visual data that utilizes limited computational resources. For instance, video tracking from hand held cameras or cameras mounted on mobile platforms is often required to track objects based on video sequences with missing frames due to acquisition and network errors. Since our distributed particle filtering framework use the estimations of other views to refine the tracking performance of the view we focus on, the frame loss problem will be considered during the video transmission process. An exact solution for frame loss problem within one camera is presented for distributed smart camera networks in this section.

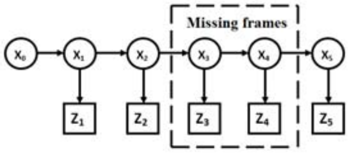


Figure 8. First-order hidden Markov chain model with missing frames.

2.3.3.1 Isolated Objects Tracking With Missing Frames

When objects are isolated from each other, an independent first-order or high-order Markov chain model is adopted for each object (10). The assumption of a first-order Markov model leads to a simple expression of the posterior density propagation and efficient implementation. Thus, we introduce it first.

Figure. 8 illustrates an example of a first-order hidden Markov chain model with several consecutive frames for single object. The circle nodes represent the states of the objects (e.g., x_4). The square nodes denote the observations associated with the hidden states (e.g., z_4). The directed line between two consecutive states represents the conditional density and forms a temporal first-order Markov chain (e.g., $p(x_3|x_2)$). The directed link from object x_i to its observation z_i can be characterized by the local likelihood $p(z_i|x_i)$. We use a dashed box to represent missing frames (e.g., x_4, x_3).

The first-order hidden Markov chain model is an acyclic directed graph (51). The *Moral graph* associated with a directed graph is the undirected graph on the same vertex set and with an edge set obtained by including all edges in the directed graph together with all edges necessary to eliminate forbidden Wermuth configurations. Note that the first-order hidden

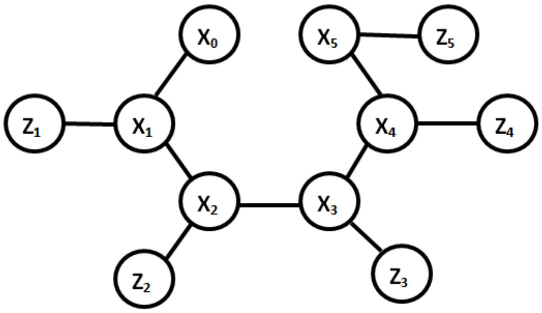


Figure 9. Moral graph of first-order hidden Markov chain model.

Markov chain model possesses the same independence interpretations as its associated *Moral graph* (51). Fig. 9 illustrates the *Moral graph* of the first-order hidden Markov model.

Proposition 1: Given missing states, i.e. $x_{t-1:t-n}$ in the first-order hidden Markov model (HMM), the current state x_t only depends on x_{t-n-1}. Let us use $x_{t-1:t-n}$ and $\mathbf{K}\backslash\{\mathbf{p}\}$ to denote $x_{t-1}, x_{t-2}, \ldots, x_{t-n}$ and the set of all the elements in $\mathbf{K}$ except $\mathbf{p}$, respectively.

$$P(x_t|x_{t-1:0}\backslash\{x_{t-1:t-n}\}, z_{t-1:1}\backslash\{z_{t-1:t-n}\}) = P(x_t|x_{t-n-1}) \tag{2.28}$$

Proof: We use $\mathbf{a} \perp \mathbf{b} \mid \mathbf{c}$ to denote random variables $\mathbf{a}$, $\mathbf{b}$ are conditionally independent on $\mathbf{c}$. By using the *Separation Theorem* (51) on the *moral graph* of the first-order HMM, we obtain conditional independence as follows:

$$x_t \perp x_{t-1:0}\backslash\{x_{t-1:t-n-1}\} \mid x_{t-n-1} \tag{2.29}$$

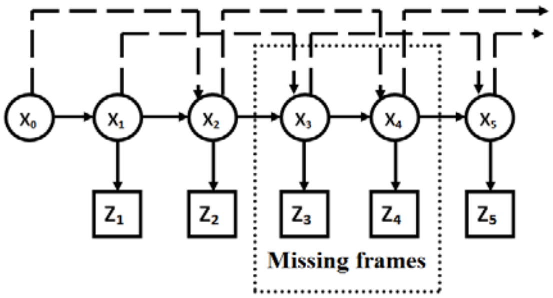


Figure 10. Second-order hidden markov chain model with missing frames.

$$x_t \perp z_{t-1:1} \backslash \{z_{t-1:t-n}\} \mid \{x_{t-1:0} \backslash \{x_{t-1:t-n}\}\} \tag{2.30}$$

This completes the proof.

Remark: From the first-order hidden Markov model (HMM), it is clear that the conditional probability distribution of the hidden variable x_t at time t, given the values of the hidden variable x at all times except the missing states $x_{t-1}, x_{t-2}, \ldots, x_{t-n}$, depends only on the value of the hidden variable x_{t-n-1}: the values at time $t-n-2$ and before have no influence. Actually, this property indicates that the value of the hidden variable x_{t-n-1} contains all the remaining information to predict the hidden variable x_t at time t.

Compared with traditional first-order particle filtering derived based on first-order hidden Markov model, if the state-space model is a mth-order Markov chain (49), the current state x_t depends on the past m states (i.e., $x_{t-1}, x_{t-2}, \ldots, x_{t-m}$). Figure. 10 illustrates an example of a second-order hidden Markov chain model (i.e., $m = 2$). The circle nodes represent the states of the objects (e.g., x_4). The square nodes denote the observations associated with the hidden

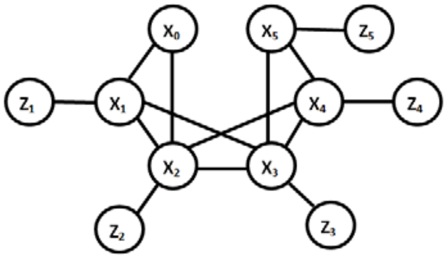


Figure 11. Moral graph of second-order hidden markov chain model.

states (e.g., z_4). Solid lines represent the first-order hidden Markov model and the dashed lines denote the second-order dependencies. The directed link from object x_i to its observation z_i can be characterized by the local likelihood $p(z_i|x_i)$. We use dashed box to represent missing frames (e.g., x_4, x_3). Figure. 11 illustrates the *Moral Graph* of the second-order hidden Markov chain model.

Proposition 2: Let us use $x_{t-a_n:t-a_1}$ to denote $\{x_{t-a_1}, x_{t-a_2}, \ldots, x_{t-a_n}\}$, where $a_1 < a_2 < a_3 \ldots < a_n$. Given missing states $x_{t-a_n:t-a_1}$ in the mth-order hidden Markov model (HMM), we have

$$p(x_t|x_{0:t-1}, z_{1:t-1}\backslash\{x_{t-a_n:t-a_1}, z_{t-a_n:t-a_1}\}) = p(x_t|x_{t-a_x-m:t-1}\backslash\{x_{t-a_x:t-a_1}\}) \tag{2.31}$$

where assuming that there are m complete items before x_{t-a_x}, while there are less than m complete items before $x_{t-a_1}, \ldots, x_{t-a_{x-1}}$, where $a_1 < m$. Therefore, we have $x_{t-a_x-m} > x_{t-a_{x+1}}$.

Proof: By utilizing the *Separating Theorem* (51) on the *Moral graph* of the high-order hidden Markov model, we can obtain the following conditional independence:

$$x_t \perp z_{1:t-1} \backslash \{z_{t-a_n:t-a_1}\} | x_{0:t-1} \backslash \{x_{t-a_n:t-a_1}\} \tag{2.32}$$

$$x_t \perp x_{0:t-a_x-m-1} \backslash \{x_{t-a_n:t-a_{x+1}}\} | x_{t-a_x-m:t-1} \backslash \{x_{t-a_x:t-a_1}\} \tag{2.33}$$

This completes the proof.

Remark: This property indicates that the values of the hidden states $x_{t-a_x-m:t-1} \backslash \{x_{t-a_x:t-a_1}\}$ contain all the remaining information to predict the hidden variable x_t at time t. Both first-order and mth-order hidden Markov model (HMM) can be used for visual tracking of single object. The assumption of the first-order Markov model can simplify the expression of the posterior density and the implementation of visual tracking, however, firstly, it can not accurately characterize the dynamics of moving objects; secondly, particle filtering based on a first-order Markov model is extremely sensitive to loss of particle information from the previous time instant, when particles are lost or delayed. The dynamics of the object is given as $x_k = Ax_{k-m} + ... + Fx_{k-2} + Gx_{k-1} + Hv_k$. The coefficients could be learned from a set of pre-labeled training sequences (54). We adopt different order hidden Markov models for the dynamics of the object based on the training set. We use sequential importance sampling method for estimation. For brevity, we omit the formulation of posterior density and particle filtering implementation for first-order and high-order hidden Markov model.

2.3.3.2 Multiple Occluded Objects Tracking With Missing Frames

As mentioned in details in previous section, We use V to denote the set of hidden states in graph G; i.e., $V = \{x_1, x_2, ...\}$. Set V could be partitioned into disjoint sets, called layers, $\{V_l\}_{l=0}^{L}$. The number of element in V_l is $K(l)$. We use the notation $v_{m,l}$ to denote the mth node in V_l, $m = 1, 2, ..., K(l)$. The parents of $v_{m,l}$ is denoted as $Pa(v_{m,l})$. It could be easily seen that $Pa(v_{m,l}) \subseteq V_{0:l-1}$. The observation associated with $v_{m,l}$ is denoted as $o_{m,l}$, and $O_l = \{o_{m,l}, m = 1, 2, ..., K(l)\}$. We denote the order of node $v_{m,l}$ as $S_{(v_{m,l})}$. For the nodes within the same layer, the order could be arbitrarily assigned.

Proposition 3: Given missing objects, $v_{b_1,l-a_1}, v_{b_2,l-a_2}, \ldots, v_{b_n,l-a_n}$, where $v_{b_1,l-a_1} \in Pa(v_{m,l})$, $a_1 < a_2 < \ldots < a_n$, we have

$$\begin{aligned}
&p(v_{m,l}|V_{0:l-1}\backslash\{v_{b_1,l-a_1}, \ldots, v_{b_n,l-a_n}\}, O_{1:l-1}\backslash\{o_{b_1,l-a_1}, \ldots, o_{b_n,l-a_n}\}) \\
&= p(v_{m,l}|Pa(v_{m,l}), Pa(v_{b_1,l-a_1}), \ldots, Pa(v_{b_x,l-a_x})\backslash\{v_{b_1,l-a_1}, \ldots, v_{b_x,l-a_x}\}) \qquad (2.34)
\end{aligned}$$

Where we assume that none of elements in $Pa(v_{b_x,l-a_x})$ is missing, while each of $Pa(v_{b_{x-1},l-a_{x-1}})$, $\ldots, Pa(v_{b_1,l-a_1})$ is not complete.

Proof: By utilizing the *Separating Theorem* (51) and the *Markov properties* (51) on the *Moral graph* of an arbitrary cycle-free graph, we can obtain the following conditional independence:

$$v_{m,l} \perp O_{1:l-1}\backslash\{o_{b_1,l-a_1}, \ldots, o_{b_n,l-a_n}\}|V_{0:l-1}\backslash\{v_{b_1,l-a_1}, \ldots, v_{b_n,l-a_n}\} \qquad (2.35)$$

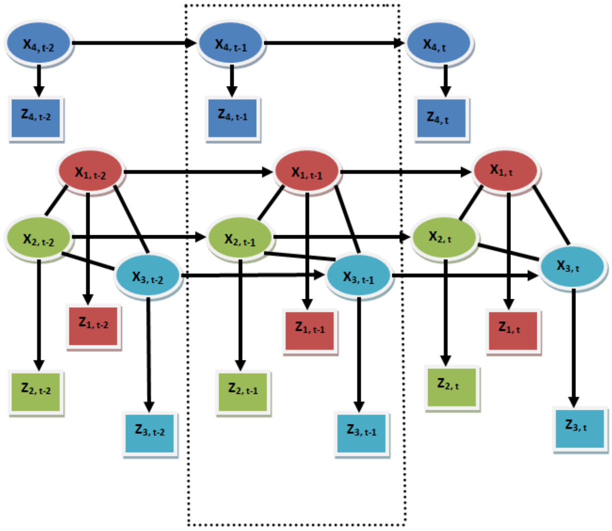


Figure 12. Graphical model of multiple (articulated) object tracking in the presence of frame loss.

$$v_{m,l} \perp \{V_{0:l-1} \backslash \{v_{b_1,l-a_1}, \ldots, v_{b_n,l-a_n}\} - A\} | A \tag{2.36}$$

where A denotes $Pa(v_{m,l}), Pa(v_{b_1,l-a_1}), \ldots, Pa(v_{b_x,l-a_x}) \backslash \{v_{b_1,l-a_1}, \ldots, v_{b_x,l-a_x}\}$.

For brevity, we omit the formulation of posterior density estimation and particle filtering implementation on cycle-free graphs with missing frames. When a general graph has cycles, we could split the graph into multiple cycle-free subgraphs by following the approach presented in (31) and apply the tracking algorithm based on cycle-free graphs in a distributed way to address missing frames. The proposed method is utilized for multiple object tracking with missing frames. The graphical model of multiple object tracking in the presence of frame loss is given in Fig. 12. The graphical model of multiple object tracking with missing frames in

Fig. 12 can also be applied to articulated object tracking with missing frames. The ellipse nodes represent the states of the objects (e.g., $x_{1,t}$). The square nodes denote the observations associated with the hidden states (e.g., $z_{1,t}$). The notation $x_{1,t}$ represents the state of object 1 at time t, and $z_{1,t}$ is the observation associated to it. The undirected link between states nodes represents the interaction among objects (e.g., occlusion). We use dashed box to represent missing frames (e.g., frame at time $t-1$). Different colors of the ellipse nodes represent different objects (for example, the green ellipse node represents object 2). Depending on whether there are interactions among different objects at the same time, $p(x^i_{m,t}|\{Pa(x_{m,t})\}^i)$ with missing frames could be divided into two categories as follows:

1. If there are no interactions among different objects at the same time, $p(x^i_{m,t}|\{Pa(x_{m,t})\}^i) = p(x^i_{m,t}|x^i_{m,t-2})$ is determined by the dynamics of the object, which is usually considered as a random walk; e.g., see object 4 in Fig. 12.

2. If the objects interact at time t (e.g., objects 1, 2 and 3), we first split the graph with cycles (interaction) by splitting an undirected edge into two directed edges with two directions (31). Then we have $p(x^i_{m,t}|\{Pa(x_{m,t})\}^i) = p(x^i_{m,t}|x^i_{m,t-2}, \{ln(x_{m,t})\}^i)$, where $ln(x_{m,t})$ represents the interacting objects of x_m at time t; e.g., $p(x^i_{2,t}|x^i_{2,t-2}, x^i_{1,t}, x^i_{3,t})$ described in Fig. 12.

For objects with interaction, we consider not only the dynamics of the same object at adjacent time, but also the occlusion among different objects at the same time. We could model this probability as $p(x^i_{m,t}|x^i_{m,t-2}, \{ln(x_{m,t})\}^i) = p(x^i_{m,t}|x^i_{m,t-2}) \prod_j (1 - \frac{1}{\alpha} \exp\{-\|x^i_{m,t} - x^i_{j,t}\|^2/\alpha^2\})$, where $\|\cdot\|$ represents the Euclidean distance between object j and m.

In the area of multiple object tracking, similar to Fig. 12, arbitrary video sequence could be modeled as a specific graph. We could first split the general graph with cycles into multiple directed cycle-free subgraphs. Then we could apply our proposed method to deal with missing frames distributively in each cycle-free subgraph. Although the particle filtering solution in dealing with missing frames on each cycle-free subgraph leads to an exact estimation, the distributed solution to particle filtering on general graphs provides an approximate solution because there is no guarantee that an update scheme can be used to represent the joint density on general graphs with complex cycles (55). The complexity of the approximated solution of particle filtering on general graphs grows exponentially along with the number of nodes belongs to cycles. Heuristics could be considered to further reduce the computation complexity, e.g., only splitting the local cycles. The resulting distributed visual tracking technique is therefore robust to sensor errors from specific camera views. Furthermore, the computational complexity of the proposed distributed approach grows linearly with the number of objects in each camera if proper heuristic approach is considered. The illustrious advantage of our proposed algorithm is that we could resolve loss of object with any number and location in an arbitrary graph derived from video sequence.

We then give the general framework for distributed graph-based particle filtering approach using multiple collaborative cameras in the presence of frame loss in Algorithm 2.

Algorithm 2 *Proposed GBPF using multiple collaborative cameras*

```
• for $j = 1, 2, 3, ...L$ \\ $L$ cameras
    — for $l = 1, 2, ...$ \\ Layers
        ∗ for $m = 1 : k(l)$
            ◇ if frames (objects) are missing, find exact ones
            by proposition 1, 2 and 3
            ◇ integrate collaboration weights from all other
            cameras with local likelihood in camera $j$
            ◇ estimate the conditional density for $v^j_{m,l}$
        ∗ end for $m$
    — end for $l$
• end for $j$
```

2.4 Experimental Results

With two or more cameras, we have more observations about the object's location, the tracking result will be more robust consequently. However, more cameras will definitely result in more computational load. Therefore, our distributed camera collaboration system will only be activated when occlusions exist. We can still incorporate the observations from other cameras to improve the reference camera's tracking performance even when the information from one other camera is missing. In this section, the tracking performances of distributed framework for video sequences from one camera are presented first. Then the results for visual tracking in distributed smart camera networks are illustrated both qualitatively and quantitatively in this section.

2.4.1 Video Sequences from One Camera

In this section, we report some of the experimental results. Our proposed distributed framework from one view is applied to both articulated object tracking and multiple object tracking in this section. The tracking performance of the proposed two graph-based methods from one view are compared both qualitatively and quantitatively with the multiple independent particle filtering (MIPF) (10), mean field Monte Carlo (MFMC) (56), respectively. For all methods, we use 50 particles for each part. In our simulations, the tracked parts are identified by the user. The number of tracked parts in articulated object is therefore pre-determined, and the parts are assumed to have a uniform prior distribution. Different colors of the rectangular are used to label the parts. The dynamics of the object is considered as a random walk, with the noise variance the same for all comparative methods.

The **Boy** sequence is from a video taken by a hand-held camcorder, which is common for a lot of activities nowadays. The video **Boy** contains a boy moving his arms. We apply graph-based particle filtering (GBPF) framework to the boy in the video using 50 particles per part. For comparison, we implemented MIPF (10) as baseline method. As shown in Fig. 13, the figures in the first row report the tracking results of GBPF framework. By exploiting the physical adjacent constraints of the human body, the GBPF improves the tracking performance in that the connections among parts are preserved well. The figures in the second row reports the tracking performance for MIPF (10). Note that our proposed GBPF can track each part of

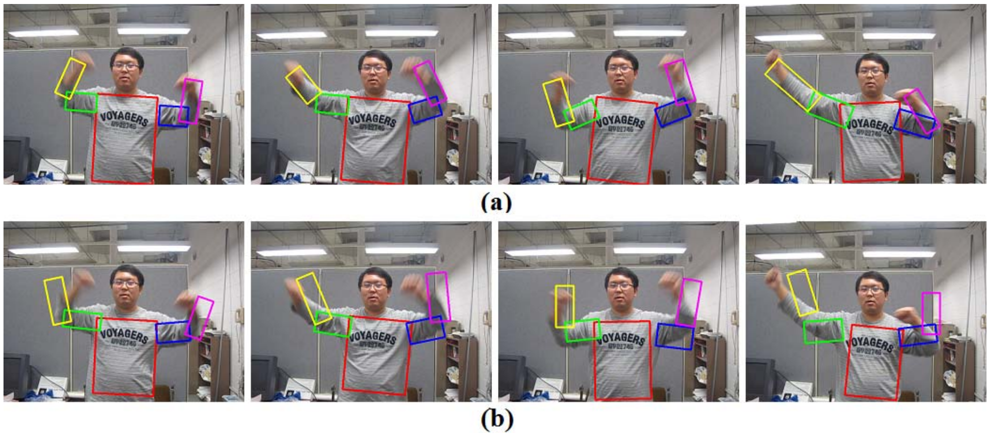


Figure 13. Tracking results of the **Boy** sequence for frames 63, 74, 100, 149: (a) The proposed graph-based particle filtering (GBPF) method; and (b) multiple independent particle filters (MIPF) (10). 50 particles are used for each part.

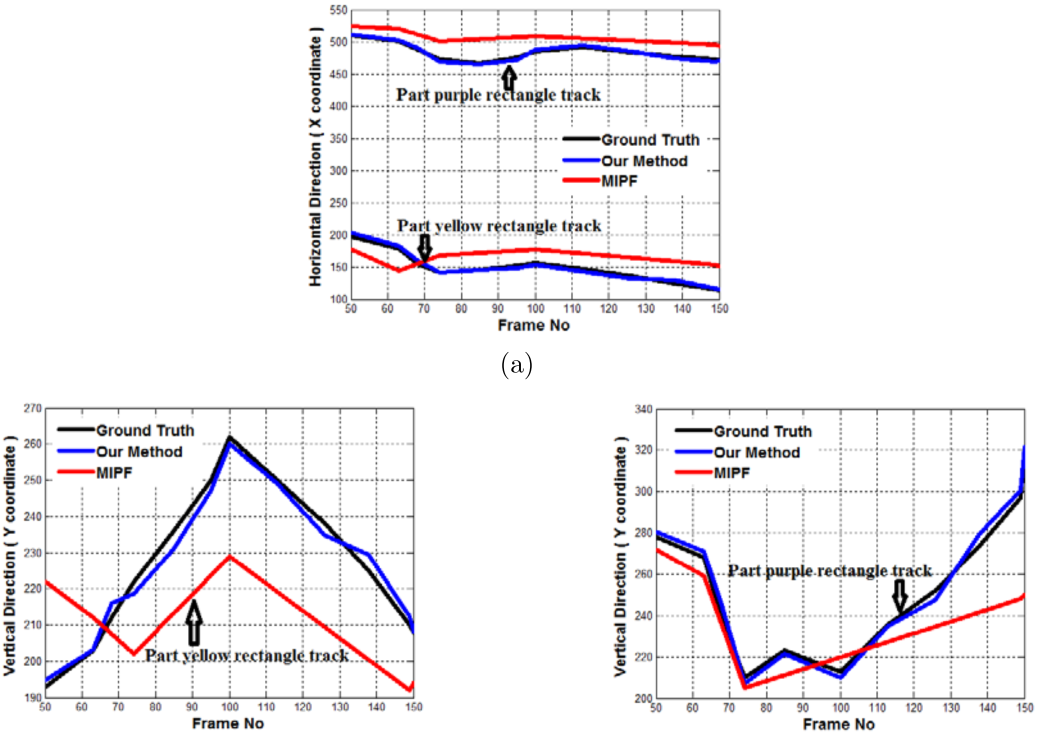
the object in most frames for irregular motions, while MIPF (10) loses track after self-occlusion. Table I compares the average MSE and computational time of the two methods in Fig. 13. From Table I, we indeed observe that the computation time required for the proposed GBPF is higher than the MIPF (10), simply because we introduce interaction weighting factor among different parts of the object. Furthermore, we compare the horizontal and vertical coordinates of two selected parts' trajectories (Yellow and Purple rectangles) from the **Boy** sequence for frame 50 to frame 150. Fig. 14 illustrates that the selected part loses track after fast motion and self-occlusions in MIPF (10). Note that the proposed GBPF is more accurate than MIPF (10).

TABLE I

STATISTICAL TRACKING PERFORMANCE FOR THE **BOY** SEQUENCE (PER FRAME).

Method	GBPF	MIPF
Average MSE	2.08	9.67
CPU time	42.6	19.7

The **Walking** sequence contains a person walking forward inside a lab. The similar color of the torso and arms, and the frequent severe self-occlusions among limbs make it difficult for articulated motion analysis. We illustrate the sample result frames of our proposed two graph-based framework, MIPF (10) and MFMC (56) in Fig. 15. MFMC (56) outperforms MIPF (10) since it keeps the connection among the parts. However, it could not produce satisfactory results when self-occlusions present. Compared with MFMC (56), our proposed GBPF improves the performance in that the connections among parts are preserved well. This is because our proposed framework uses separated interaction models for the location and rotation. By handling the high-level interaction among arms and legs and using a learned model of limb poses, HGBPF framework gives the best results. Table II compares the average MSE and computational time of the four methods in Fig. 15. From Table II, we indeed observe that MFMC (56) improves the tracking performance compared with MIPF (10). Moreover, the proposed two methods model the interaction constraints of an articulated object more efficiently compared with MFMC (56) and therefore achieve more accurate tracking results.



(a)

(b)

(c)

Figure 14. The horizontal and vertical coordinates of the trajectories of selected objects of the **Boy** sequence.

When we apply this distributed graph-based particle filtering (GBPF) framework to multiple object tracking, we ignore the inherent constraints between complex parts and consider comparatively independent movements among multiple objects. A typical sequence of multiple object tracking is **Shopping Center** sequence which is shown in Fig. 16. We apply graph-based particle filtering (GBPF) framework to the **Shopping Center** sequence. For comparison, we implemented MIPF (10) as baseline method. In this case, we have less confidence on the color

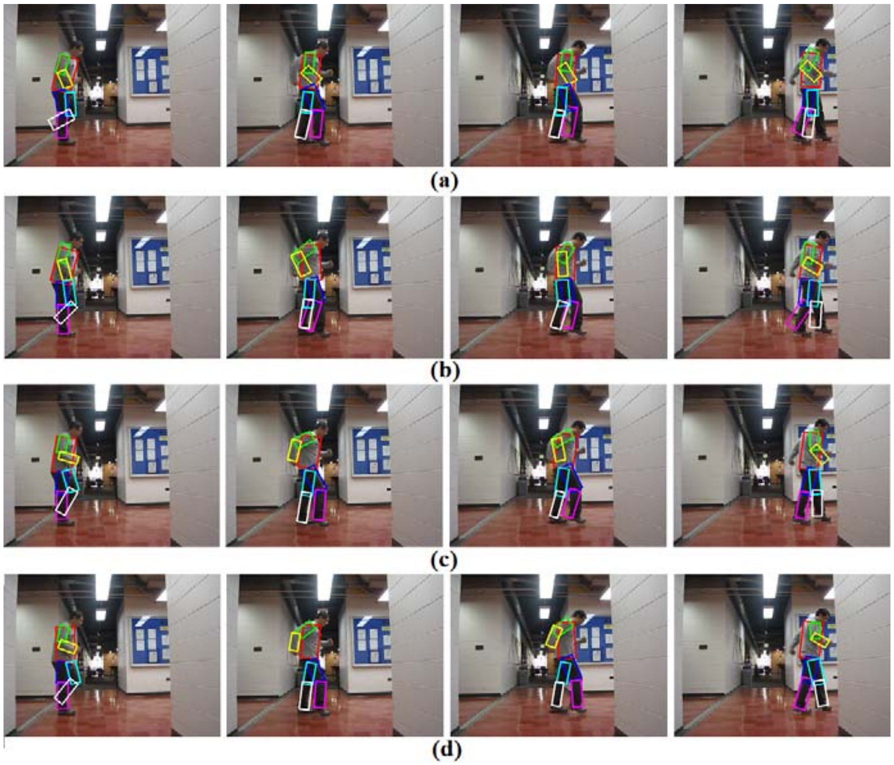


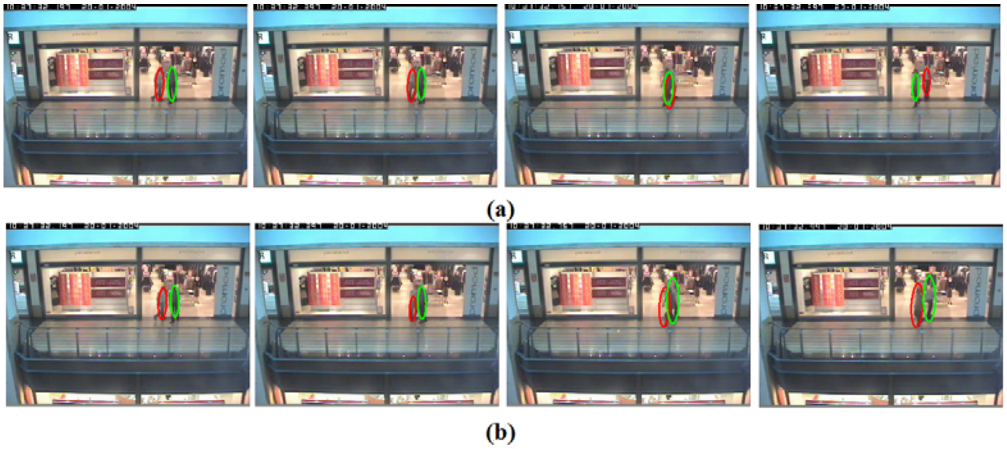
Figure 15. Tracking results of the **Walking** sequence for frames 16, 32, 53 and 65: (a) multiple independent particle filters (MIPF) (10); (b) mean-field Monte Carlo (MFMC)(48); the proposed GBPF method; and (d) proposed HGBPF method. 50 particles are used for each part.

histogram due to the clutter environment, but we want to test the tracking performance of two methods in the extreme case. Nonetheless, our proposed GBPF can easily trace the person in every frame, while MIPF (10) loses track in the first half.

TABLE II

STATISTICAL TRACKING PERFORMANCE FOR THE **WALKING** SEQUENCE (PER FRAME).

Method	MIPF	MFMC	GBPF	HGBPF
Average MSE	6.08	3.67	2.12	1.55
CPU time	28.6	36.1	45.7	49.6



(a)

(b)

Figure 16. Tracking results of the **Shopping center** sequence for frames 824, 836, 847, 873: (a) The proposed graph-based particle filtering (GBPF) method; and (b) multiple independent particle filters (MIPF) (10). 50 particles are used for each object.

2.4.2 Video Sequences on Multi-camera Platform

Visual tracking on multi-camera platform is often required to track objects based on video sequences with missing frames due to acquisition and network errors. In this section, we carry out a series of experiments to test the proposed distributed graph-based particle filtering (GBPF) framework from multiple views in the presence of frame loss. The proposed distributed framework from multiple collaborative cameras is compared both qualitatively and quantitatively with the multiple independent particle filtering (MIPF) (10), distributed graph-based framework from multiple views independently in section 4, respectively.

We denote p as the loss rate of each frame in the experiments. Without loss of generality, two cameras are used in our framework first. The *Tennis* video clip contains five tennis balls moving across each other. Fig. 17 shows the tracking results of MIPF (10), distributed framework based on two cameras independently and distributed framework from two collaborative cameras. Our framework in distributed camera networks could handle occlusion well in the presence of frame loss compared to MIPF (10) and distributed framework from one view. Compared with MIPF (10), our distributed framework from one view improves the performance in that the connections among parts are preserved well. This is because our proposed framework uses separated interaction models for each object. By handling the occlusion problem in distributed smart camera networks, our framework on multi-camera platform gives the best results.

To further evaluate the proposed algorithm, typical surveillance video *Lab* sequence is shown in Fig. 18. This data set was recorder from two views with missing frames, i.e. $p = 0.1$. The fundamental matrix between two views is computed from 8 matching points in a pair of im-

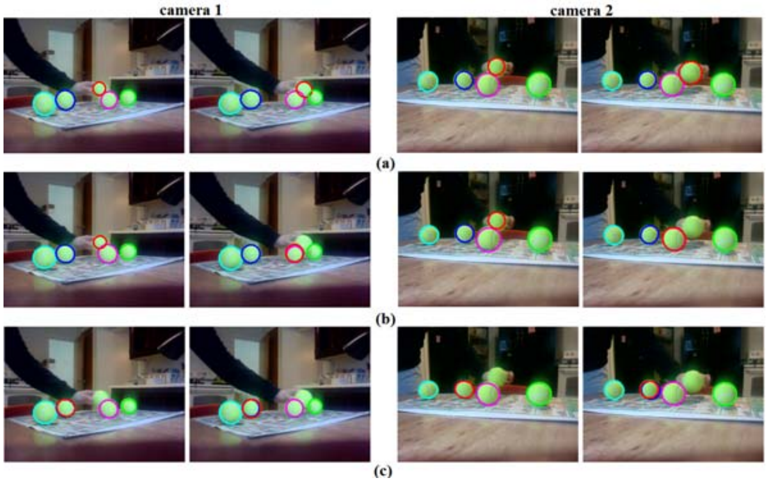


Figure 17. Tracking results of the **Tennis** sequence for frame 170, 181 with 50 particles per object and missing frames, i.e. $p = 0.1$: (a) distributed graph-based framework using two collaborative cameras; (b) distributed graph-based framework from multiple views independently; and (c) multiple independent particle filtering (MIPF) (10).

ages (53). In Fig. 18, we compare the tracking results of (a) multiple independent particle filtering (MIPF) (10), (b) distributed graph-based framework from multiple views independently and (c) distributed graph-based framework using two collaborative cameras. Note that distributed framework from two cameras can trace objects in every frame, while distributed framework from one view loses track after heavy occlusion. Both distributed framework outperform MIPF (10) in dealing with missing frames, which validates our theory. Furthermore, we compare the horizontal and vertical coordinates of two people's trajectory from the second view of *Lab* sequence for frame 250 to frame 325. Fig. 19 illustrates that the people coming from the right side loses track completely after occlusion in MIPF (10). Note that the proposed method

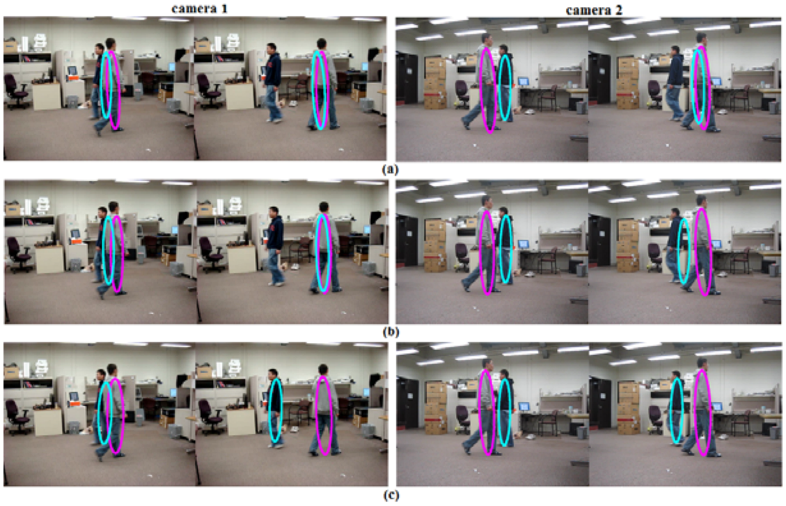


Figure 18. Tracking results of the **Lab** sequence for frame 280, 295 with missing frames, i.e. $p = 0.1$: (a) multiple independent particle filtering (MIPF) (10); (b) distributed graph-based framework from multiple views independently; and (c) distributed graph-based framework using two collaborative cameras. We use 80 particles per object.

based on two collaborative cameras is more accurate than MIPF (10). Table III compares the average MSE in second view and computational time of the three methods in Fig. 18.

Our distributed graph-based framework on multi-camera platform is also tested with three cameras. The **People** sequence (57) was recorded with three cameras. In Fig. 20, we illustrate the tracking performance of our distributed system from three views. As shown in Fig. 20, the figures in the first row report the tracking results for frame 1407 from three views. We report the tracking results for frame 1449 from three views on the second row of Fig. 20 also. Our distributed graph-based framework from three cameras can handle the occlusion problem pretty well. Table IV provides the specific time for each part of likelihood estimation. As can

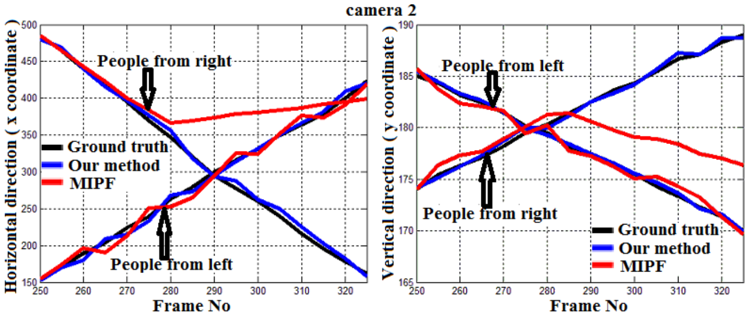


Figure 19. The horizontal and vertical coordinates of two people's trajectory on the **Lab** sequence. The illustration of ground truth, proposed algorithm using two collaborative cameras, MIPF (10) with 80 particles per object and missing frames, i.e. $p = 0.1$.

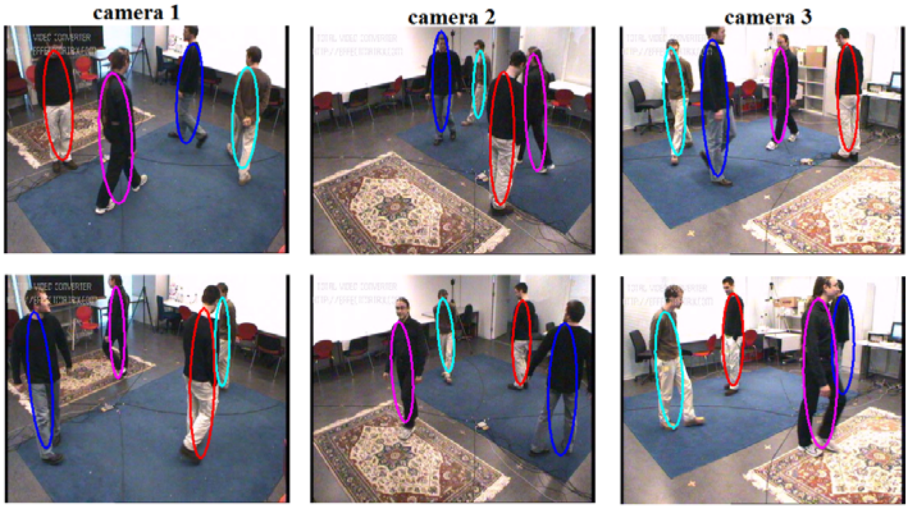


Figure 20. Tracking results of the **People** sequence (51) for frame 1407 and 1449 with 50 particles per object using distributed graph-based framework from three collaborative cameras.

TABLE III

STATISTICAL TRACKING RESULTS ON *LAB* SEQUENCE PER FRAME.

Method	MIPF	GBPF from one view	GBPF from multiple cameras
Average MSE (camera 2)	8.55	4.83	1.67
CPU time	6.84	6.93	8.1

TABLE IV

AVERAGE COMPUTATION TIME COMPARISON OF DIFFERENT PARTS IN PARTICLE FILTERING ESTIMATION ON *PEOPLE* SEQUENCE.

Color/edge likelihood	Interaction likelihood	Camera collaboration likelihood
25	5.1	4.67

be seen in Table IV, compared with the most time-consuming components which relate to color and edge likelihood estimation, the computational time for camera collaboration likelihood is negligible. It is because the camera collaboration likelihood estimation only involves some efficient numerical calculations. Moreover, in our distributed framework we impose a preset limit on the number of iterations (usually less 5 iterations), which results in negligible overhead for the computation time of the iterative algorithms. The data presented in all Tables have been averaged over ten trials.

CHAPTER 3

DISTRIBUTED CONTEXT-FREE GRAMMARS FRAMEWORK AND ITS APPLICATION IN TRAJECTORY-BASED VIDEO CLASSIFICATION

3.1 Theoretical Foundations

3.1.1 Transformational Grammars

An overview of the formal language theory is provided in this section. A formal language can be broadly defined as any set of strings consisting of concatenations of symbols. The complete set of distinguishable symbols in the language is known as the alphabet and is denoted here by T. For example, an alphabet might be $T = \{a, b\}$, and one language over this alphabet might consist of all finite repetitions of the combinations *ab* followed by either *b* or *aa*; in this language, the strings *b*, *aa* and *ababaa* are valid strings but *aba* is not. The general notion of a formal language is impractically broad. It is much more useful, and intuitive, to specify a language in terms of its structural patterns. This is often accomplished by defining a transformational grammar (37).

3.1.1.1 Chomsky Hierarchy of Grammars

In grammatical terminology, a deterministic grammar G_s is a four-tuple $G_s = (N, T, P, S)$. N is a finite set of nonterminal symbols, T is a finite set of terminal symbols, and $N \cap T = \emptyset$. P is a finite set of production rules, and $S \in N$ is the starting symbol. The grammars are divided into four different types according to the forms of their production rules. Specifically, only production rules of the form $A \rightarrow aA$ or $A \rightarrow a$ are allowed in regular grammars (RGs). This means that the left-hand side of the production must contain one nonterminal only, and the right-hand side could be either one terminal or one terminal followed by one nonterminal. Context-free grammars (CFGs) have production rules P of the form $A \rightarrow \eta$ where $A \in N$ and $\eta \in (N \cup T)^+$; the superscript Σ^+ indicates the set of all finite length strings of symbols in a finite set of symbols Σ, excluding the string of length 0. A context-free grammar is a special case of a context-sensitive grammar. Production rules of the form $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \eta \alpha_2$ are allowed in CSGs where α_1, α_2, $\eta \in (N \cup T)^+$. In other words, the allowed transformations of nonterminal A are dependent on its context α_1 and α_2. Also a context-sensitive grammar is a special (more restricted) form of a unrestricted grammar (UG). Any production rules of the form $\alpha_1 A \alpha_2 \rightarrow \gamma$ are allowed in unrestricted grammars (UGs) where α_1, α_2, $\gamma \in (N \cup T)^+$. Table V provides a condensed summary of the classes of grammars, their production rule structures, and classes of languages that they define. More detailed treatment of the Chomsky Hierarchy is given by (58).

A stochastic grammar G_s is a five-tuple $G_s = (N, T, P, P_s, S)$, P_s is the set of probability distributions over the set of production rules P. Stochastic grammars are classified and analyzed on the basis of their underlying characteristic grammars. A characteristic grammar is obtained from the stochastic grammar by removing the probability distribution P_s from the grammar definition.

TABLE V

DETERMINISTIC GRAMMARS, PRODUCTION RULES, AND LANGUAGES.

Grammar	Production Rule Structure	Language
FSGs	$A \rightarrow aA$, $A \rightarrow a$	Finite State (Regular) Language (RL)
CFGs	$A \rightarrow \beta$	Context-Free Language (CFL)
CSGs	$\alpha_1 A \alpha_2 \rightarrow \alpha_1 \eta \alpha_2$	Context-Sensitive Language (CSL)
UGs	$\alpha_1 A \alpha_2 \rightarrow \gamma$	Unrestricted (Type-0) Language (UL)

3.1.1.2 Existing Methods for Learning Grammars

The problem of learning stochastic grammars from training sequences has two aspects: determining a discrete structure (topology) of the grammar and estimating probabilistic parameters in the grammar. Based on maximum likelihood criterion, efficient estimation algorithms for probabilistic parameters have been proposed: a Forward-Backward Algorithm for hidden Markov models (59) and an Inside-Outside Algorithm for stochastic context-free grammars (SCFGs) (60). Both algorithms are iterative algorithms which are based on the Expectation-Maximization (EM) Technique that increases the likelihood of the training sample in each step until a local maximum is reached.

The standard method for estimating the parameters of SCFGs (i.e., the probabilities of the productions) from a set of training examples is known as the Inside-Outside Algorithm (60). However, it requires the grammar to be in Chomsky normal form, which is inconvenient to handle in many practical problems (and requires more nonterminals). Further, the computational time for this algorithm is quite long compared to the Forward-Backward Algorithm. To avoid such problems, Sakakibara et al. (61) have developed a new method for training SCFGs that is a generalization of the Forward-Backward Algorithm to tree grammars and which is more efficient than the Inside-Outside Algorithm. The new algorithm, called Tree-Grammar EM, requires structured strings as training examples. This algorithm uses a similar idea to the identification of context-free grammars from structured strings shown in (62). Since information on the grammatical structure is given explicitly in training examples, Tree-Grammar EM Algorithm does not have to consider all possible derivations of the training examples when estimating the grammar's parameters, as the Inside-Outside Algorithm must do. Also a similar idea of estimation algorithm has been developed by Eddy and Durbin (36) by the name of covariance model.

Learning grammatical structures of stochastic grammars is challenging. The difficult arises from two specific aspects of the problem of learning grammars from examples: determining the grammatical structure (topology) of the unknown grammar and identifying the set of nonterminals in the grammar. The first problem is especially hard because the number of all possible grammatical structures for a given positive example becomes exponential in the length of the positive example. To overcome the hardness of learning CFGs from examples without structural information available, Sakakibara and Kondo (63) have proposed a new hypothesis representation method, called tabular representation, which consists of a table-like data structure similar to the parse table used in the Cocke-Younger-Kasami Algorithm (64) for CFGs of Chomsky normal form. In particular, the learning algorithms for finite-state grammars (FSGs) and context-free grammars (CFGs) rely on the causality of the model. However, context-sensitive grammars (CSGs) can not be solved in exact form since such a model is inherently non-causal. Therefore, we still try to investigate a polynomial complexity solution for CSGs.

3.1.2 Trajectory Representation

This section provides a survey of the related work from recent literature in the area of trajectory representation. Studies into human psychology have shown the extra-ordinary ability of human beings to recognize object motion even from minimal information system such as Moving Light Displays (MLDs). In spite of the paucity of information, human observers easily perceive not only motion but also the kind of motion; e.g., walking, running, dancing, etc (65). Based on this understanding, object motion has been an important feature for the representation and discrimination of one object from another in video applications. Earlier approaches in motion-based methods focused on object tracking from raw and compressed domain videos (66) (67) (68) (69). Indexing and searching based on object motion as the dominant cue has attracted a lot of research activity in the past few years. Chen et. al. segment each trajectory into subtrajectories using fine-scale wavelet coefficients at high levels of decomposition (30). This approach suffers from the fact that the representation is based on ad hoc features which are not tolerant to affine transformations of the trajectories. Also the feature vectors lie in a non-uniform space, so the matching process has to compute the overall distance based on weighted average of individual features. F. Bashir et al. represent the subtrajectories using PCA coefficients (66). The work on trajectory indexing and retrieval segments the trajectories based on dominant sign changes in curvature data. In our trajectory classification system, the trajectory segmentation scheme looks for sharp changes in target's velocity and acceleration

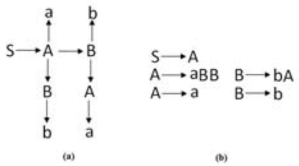


Figure 21. Illustration of a sequence generated by a simple context-free grammar.

through 1st and 2nd order derivatives. Consequently, trajectories are segmented at points of maximum change in curvature of the trajectory. We represent the subtrajectories using PCA because of its optimal energy compaction properties (70).

3.2 Learning Stochastic Context-Sensitive Grammars (SCSGs)

In this section, we first present limitations of current stochastic context-free grammars (SCFGs) model. Motivated by these practical applications, our noncausal stochastic context-sensitive grammars model (SCSGs) is provided then. A completely new distributed approach is proposed to learn context-sensitive grammars in exact form in this section also.

3.2.1 Problem Statement: Limitations of SCFGs

In our stochastic context-free grammars (SCFGs) representation, SCFGs are defined by a five-tuple $G_s = (N, T, P, P_s, S)$, where N is an alphabet of nonterminal symbols, T is an alphabet of terminal symbols such that $N \bigcap T = \varnothing$, P is a finite set of production rules of the form $A \rightarrow \eta$ where $A \in N$ and $\eta \in (N \cup T)^{+}$. We use P_s to denote the set of probability distributions over the set of production rules P. S is a special nonterminal called the start symbol.

For example, a sequence generated by a simple SCFG is illustrated in Fig. 21(a). Take Fig. 21 as our SCFG example, $T = \{a, b\}$, $N = \{A, B, S\}$, the start symbol $S = \{S\}$. The production rules of the simple SCFG example in Fig. 21 are illustrated as follows:

$$S \rightarrow A|B$$

$$A \rightarrow aBB|aAB|aAA|aBA|aA|aB|a$$

$$B \rightarrow bBB|bAB|bAA|bBA|bA|bB|b$$

We also illustrate the production rules we use in Fig. 21(b). For instance, with the grammars in Fig. 21(b), the string *abba* in Fig. 21(a) can be derived with the derivation: $S \rightarrow A \rightarrow aBB \rightarrow abB \rightarrow abbA \rightarrow abba$.

This example can be used to describe changes from horizontal and vertical directions. The main limitation of this grammar is that it allows only causal state dependencies. In particular, the learning and classification algorithms of this grammar rely on the causality of the model. However, a noncausal model is needed for some practical applications. For example, for multiple object motion trajectories analysis, we can not isolate the motion trajectories to individual objects and, thus, lose their interactions. Therefore, SCFGs can not characterize the intrinsic interactions between different trajectories at the same time, since such an application is inherently noncausal. Motivated by these noncausal practical applications, a new model with noncausal property is needed. In this paper, this goal can be achieved by our stochastic context-

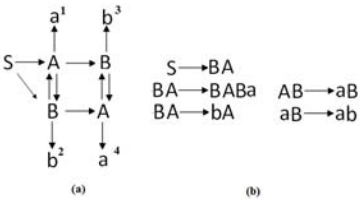


Figure 22. Illustration of a sequence generated by a context-sensitive grammar.

sensitive grammars model (SCSGs). The noncausal SCSGs can characterize the intrinsic state transition structure and behavior of complex systems involving multidimensional system states.

3.2.2 Stochastic Context-Sensitive Grammars

Take multiple object motion trajectories analysis as an example, a noncausal model is needed that can capture the representation of motion trajectories. The SCSGs model applied here has two advantages: i) The interactions (eg. occlusions between different objects) can be preserved well in our model. ii) The learning algorithm for our model avoids costly semantic analysis that would perform classification based on heuristics rather than the inherent probabilistic model used for multiple trajectories representation.

In our SCSGs representation, we denote the SCSGs as a five-tuple $G_s = (N, T, P, P_s, S)$, where N is an alphabet of nonterminal symbols, T is an alphabet of terminal symbols such that $N \bigcap T = \emptyset$, Production rules P of the form $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \eta \alpha_2$ are allowed in SCSGs where α_1, α_2, $\eta \in (N \cup T)^+$. In other words, the allowed transformations of nonterminal A are dependent on its context α_1 and α_2. We use P_s to denote the set of probability distributions over the set of production rules P. S is a special nonterminal called the start symbol.

For example, a sequence generated by a SCSG is illustrated in Fig. 22(a). Take Fig. 22 as a SCSG example, $T = \{a, b\}$, $N = \{A, B, S\}$, the start symbol $S = \{S\}$. The production rules of the simple SCSG example in Fig. 22 are illustrated as follows:

$$\begin{aligned} S &\rightarrow AB|BA|BB|AA \\ BA &\rightarrow BABa|BAAa|bA \\ bA &\rightarrow bB|bA|ba \\ AB &\rightarrow ABBb|ABAb|aB \\ aB &\rightarrow aA|aB|ab \\ AA &\rightarrow AABa|AAAa|aA \\ aA &\rightarrow aA|aB|aa \\ BB &\rightarrow BBBb|BBAb|bB \\ bB &\rightarrow bA|bB|bb \end{aligned}$$

We also illustrate the production rules we use in Fig. 22(b). For instance, with the grammars in Fig. 22(b), the string *baba* in Fig. 22(a) can be derived with the derivation: $S \rightarrow BA \rightarrow BABa \rightarrow bABa \rightarrow baBa \rightarrow baba$. Comparing to the terminals sequence generated by the SCSG in Fig. 22(a), we analyze the string *baba* from the rightmost a. This a serves as the first terminal generated by the SCSG. Then we consider the leftmost b as the second terminal generated by the grammar. The second rightmost b is considered as the third terminal, and so on. We generate the terminal sequence by using the generation rule above.

3.2.3 Learning SCSGs: A Distributed Approach

In this section, we are now in the position to deal with the stochastic context-sensitive grammars (SCSGs) model. We propose a novel solution to the arbitrary noncausal SCSGs by splitting the SCSGs into multiple causal SCFGs that are analytically solvable in a distributed computing framework, therefore referred to as distributed SCFGs. The learning and classification algorithms are processed on our distributed SCFGs sequentially using an alternate updating scheme to approximate solution of the noncausal SCSGs model. For simplicity, we first focus our discussion of distributed approach on the example in Fig. 22(a). The proposed scheme can be easily extended to more complex higher dimensional cases.

Fig. 23 demonstrates an example of splitting a noncausal SCSGs. The four types of causality are depicted in Fig. 24 first. In Fig. 24, gray state nodes represent neighboring state nodes that affect black state node. Black state node is the node we try to analyze. In a causal system, the neighboring state nodes that affect the analyzed node can only come from one of these four cases in Fig. 24. Our resulting subgraphs from splitting should preserve one type of causality as depicted in Fig. 24. The splitting rules are defined as follows. We first identify types of dependencies. Each noncausal (bi-directional or multidirectional) dependency between state nodes is decomposed into a couple of causal dependencies, by focusing on one-direction dependency while ignoring other dependencies once at a time. For the node involving noncausal dependencies, we connect it to its nearest neighboring state node by specifying one type of single-dependency to form a directional graph. All non-neighboring state nodes or nodes have different type of single-dependency and their observations are removed. Then we connect state

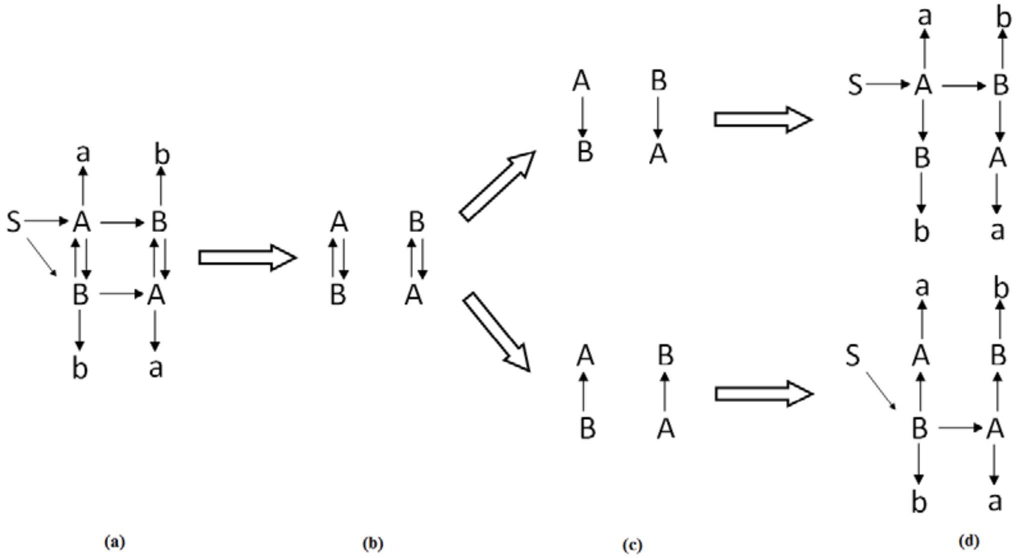


Figure 23. Example of splitting procedure: a) SCSGs; b) state nodes involving multidirectional dependencies; c) splitting dependencies into several causal dependencies; d) two distributed SCFGs.

nodes involving causal dependencies to the graph obtained in the previous step. All newly added nodes on the directional graph should preserve the causality. If the newly added node, together with the existing graph, makes any loop of dependency, that node will be removed. For each decomposed directional dependency, we repeat the steps above until all distributed dependency graphs are obtained. The state dependency information is still intact in the splitting procedure. Furthermore, since each distributed sub-model preserves the correlation between neighboring state nodes, the proposed framework is not a simple collection of uncorrelated causal models but an accurate representation of the original model.

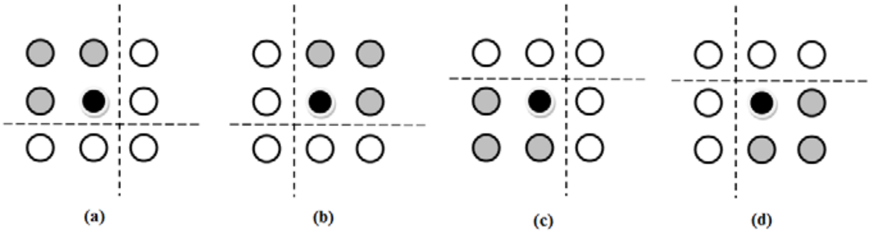


Figure 24. Four types of causality: a) Type I (Up and Left); b) Type II (Up and Right); c) Type III (Down and Left); d) Type IV (Down and Right).

Our proposed distributed framework can be applied to more complex SCSGs. In our framework, we decompose the SCSGs, by focusing on one causal direction of state transition at a time, while ignoring other directions of state transition probabilities. Fig. 25(a) depicts one more complex example of a SCSG. We decompose the noncausal model in Fig. 25(a) using the proposed distributed approach. We get two distributed SCFGs, shown in Fig. 25(b) and (c). The solution of splitting SCSGs into multiple SCFGs and alternate updating parameters of the SCFGs is essentially equivalent to the Loopy Believe Propagation (LBP) (25) on graphical model. Our decomposition of SCSGs to SCFGs is equivalent to the removal of the restriction in a graph with loop to be loop-free. The convergence of LBP has been proved in (25), and a sufficient condition of convergence is given in (71). Moreover, it is shown that the LBP algorithm will converge to a solution that is consistent with every cycle-free graph embedded in the graphical model. The estimation provided by the proposed distributed approach will converge to the identical solution obtained by the use of the LBP algorithm provided that the proper schedule is selected. We note, however, that although the final estimate will be identical, the

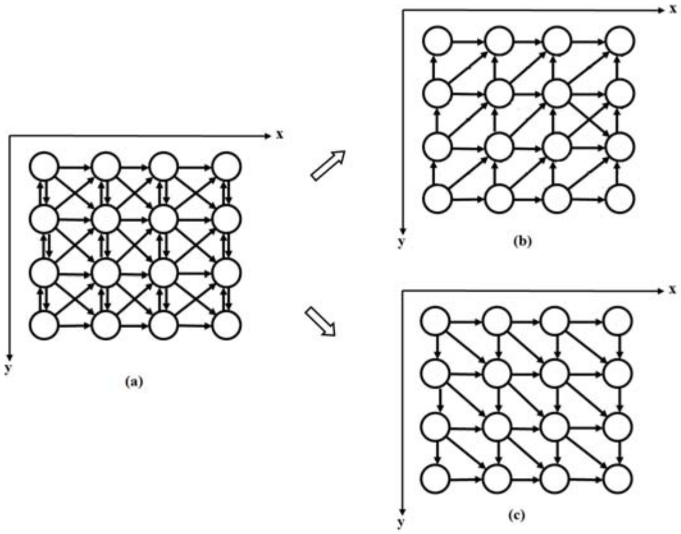


Figure 25. Decomposition of SCSGs to SCFGs: a) SCSGs; b) distributed SCFGs; c) distributed SCFGs.

proposed estimation presented in this paper is performed using a completely different algorithm. Specifically, in the case of the LBP algorithm, an iterative procedure is presented and implemented in a distributive manner on the graphical model and the convergence of the LBP must be ascertained. On the other hand, the proposed distributed SCFGs model provides a recursive solution to the learning and classification algorithms whose convergence is guaranteed.

We are now in the position to provide a simultaneous solution to the learning and classification algorithms of multiple distributed SCFGs. Without loss of generality, the SCSGs with applications to two trajectories as depicted in Fig. 26(a) are used as an example to explain our proposed distributed approach in details. The proposed scheme can be easily extended to more complex cases. In our SCSGs representation, we denote the SCSGs as a five-tuple $G_s = (N, T, P, P_s, S)$, where N is an alphabet of nonterminal symbols, T is an alphabet of terminal symbols such that $N \bigcap T = \emptyset$, P is a finite set of production rules. We use P_s to denote the set of probability distributions over the set of production rules P. S is a special nonterminal called the start symbol. Take Fig. 26(a) as our SCSGs model, $T = \{O_{1,0}, O_{1,1}, O_{1,2}, O_{1,3}, O_{2,0}, O_{2,1}, O_{2,2}, O_{2,3}\}$, $N = \{S, U_{1,0}, U_{1,1}, U_{1,2}, U_{1,3}, U_{2,0}, U_{2,1}, U_{2,2}, U_{2,3}\} \bigcup \{X_{1,0}, X_{1,1}, X_{1,2}, X_{1,3}, X_{2,0}, X_{2,1}, X_{2,2}, X_{2,3}\}$, $S = \{S\}$. The undirected edge between two states, for example, the edge between $X_{1,0}$ and $X_{2,0}$, makes the desired model non-causal. An undirected edge, by definition, could be split into two directed edges with two different directions. Therefore, we propose a novel solution to it, where we "decompose" SCSGs model in Fig. 26(a) into several causal SCFGs. We approximate the simultaneous solution of each of the distributed causal SCFGs by an alternate updating scheme. For example, we distribute the SCSGs model in Fig. 26(a) to two causal SCFGs, shown in Figs. 26(b)-(c). We approximate the simultaneous solution of multiple distributed SCFGs by a sequential alternating updating scheme. One example updating scheme is depicted in Fig. 26(b)-(c). The numbers $\{1, 2, 3, 4, ...\}$ are the sequence orders of updating of model parameters.

The learning algorithms of the SCFGs we have provided in the following section are applied to each of the SCFGs in Figs. 26(b)-(c) sequentially. In this way, we provide an analytical learning procedure to the SCSGs model by decomposing it into several causal SCFGs models and each of these SCFGs models can be solved simultaneously using a fully synchronous distributed computing framework.

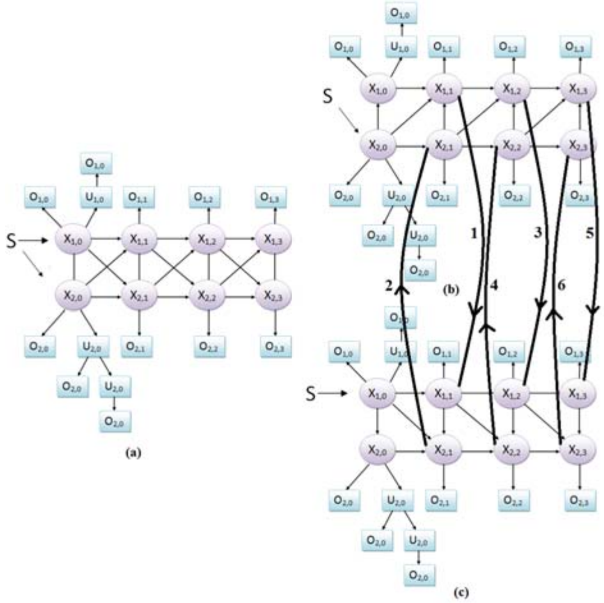


Figure 26. Illustration of a sequence generated by a simple context-sensitive grammar.

3.3 Learning Stochastic Context-Free Grammars (SCFGs)

In this section, we first decompose SCSGs model into multiple SCFGs. Then the statistical estimation algorithms for each SCFGs are applied sequentially. Specific to our case, a hybrid of general Forward-Backward Algorithm, Inside Algorithm and Expectation-Maximization Technique will be used to estimate the system parameters. Then using the forward recursion procedure in the Baum-Welch Algorithm (59) to compute the log-likelihood that each model gives to the test sample.

3.3.1 Stochastic Context-Free Grammars

In our stochastic context-free grammars (SCFGs) representation, SCFGs are defined by a five-tuple $G_s = (N, T, P, P_s, S)$, where N is an alphabet of nonterminal symbols, T is an alphabet of terminal symbols such that $N \bigcap T = \varnothing$, P is a finite set of production rules of the form $A \rightarrow \eta$ where $A \in N$ and $\eta \in (N \cup T)^+$. We use P_s to denote the set of probability distributions over the set of production rules P. S is a special nonterminal called the start symbol.

Take Fig. 27 as an example of a simple context-free grammar model, the start symbol $S = \{S\}$, $N = \{S, U_{1,0}, U_{1,1}, U_{1,2}, U_{1,3}, U_{2,0}, U_{2,1}, U_{2,2}, U_{2,3}, X_{1,0}, X_{1,1}, X_{1,2}, X_{1,3}, X_{2,0}, X_{2,1}, X_{2,2}, X_{2,3}\}$, $T = \{O_{1,0}, O_{1,1}, O_{1,2}, O_{1,3}, O_{2,0}, O_{2,1}, O_{2,2}, O_{2,3}\}$.

We are now in the position to deal with the SCFGs model in Fig. 27. Given a SCFG S, there are three basic problems for dealing with S. The first problem is to calculate the probability of the string w generated by S; the second one is to find the most probable path p of states to maximize $Pr(p|w, S)$; the last one is to estimate the parameters in S to maximize $Pr(w|S)$. The first two problems, calculating the probability $Pr(w|S)$ of a given string w assigned by a SCFG S and finding the most likely derivation tree of w by S, can be solved using dynamic programming methods (64). In the following section, we provide a series of learning algorithms to estimate the parameters in SCFGs.

3.3.2 Learning Stochastic Context-Free Grammars Model

In this section, we describe the statistical estimation algorithms of the SCFGs model. Let's take Fig. 27 as an example. The following notation will be used throughout this section.

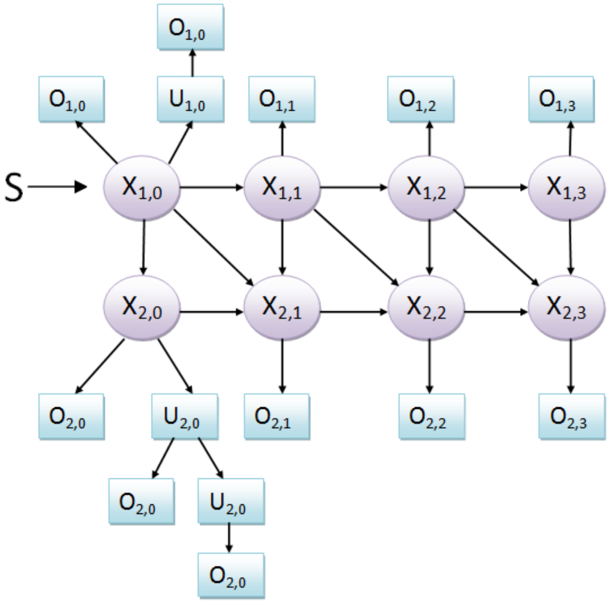


Figure 27. Illustration of a sequence generated by a simple context-free grammar.

The state variable $X_{i,t}$ represents the (unknown) state sequence at specific time t for row i, where $t = \{0, 1, ..., T\}$, $i = \{1, 2\}$; $r_{i,t}$ represents the observation associated with $X_{i,t}$. Each $r_{i,t} = (w_1, w_2, ..., w_{m_{i,t}})$ is a string of concatenated terminal symbols, and $m_{i,t}$ is the length of $r_{i,t}$. Take Fig. 7 as an example, $r_{1,0} = O_{1,0}O_{1,0}$, $r_{2,0} = O_{2,0}O_{2,0}O_{2,0}$, $r_{1,1} = O_{1,1}$.

It is convenient to introduce the following variables:

1. Inside variable: $\beta_j^{X_{i,t}}(i, t, p, q) = P(w_{pq}|A^j_{pq}, X_{i,t})$
2. Outside variable: $\alpha_j^{X_{i,t}}(i, t, p, q) = P(w_{1(p-1)}, A^j_{pq}, w_{(q+1)m}|X_{i,t})$

Where w_{pq} is the subsequence of terminals from pth position of $r_{i,t}$ to qth position, and A^j_{pq} is the nonterminal $A^j \in N_p$ that derives w_{pq}, or $A^j \Rightarrow w_{pq}$. In the following part, we first describe the application of general Forward-Backward Algorithm, Inside Algorithm and Expectation-Maximization Technique to estimate the system parameters (72). In order to classify a new test sample into one of the classes, we then use the forward recursion procedure in the Baum-Welch Algorithm (59) to compute the log-likelihood that each model gives to the test sample. If the i^{th} model is the most likely, then declare the class of the sequence to be class i.

-Inside Algorithm

The Inside Algorithm computes the probability, $O_{X_{i,t}}(r_{i,t})$, inductively as follows:

1. Initialization: $\beta^{X_{i,t}}_j(i,t,p,p) = P(A^j \rightarrow w_p|X_{i,t})$
2. Induction: $\beta^{X_{i,t}}_j(i,t,p,q) = \sum_{r,s} \sum_{d=p}^{q-1} P(A^j \rightarrow A^r A^s)\beta^{X_{i,t}}_r(i,t,p,d)\beta^{X_{i,t}}_s(i,t,d+1,q)$

   for $\forall j, 1 \leq p < q \leq m_{i,t}$.
3. Termination: $O_{X_{i,t}}(r_{i,t}) = \beta^{X_{i,t}}_1(i,t,1,m_{i,t})$

$O_{X_{i,t}}(r_{i,t})$ is the output probability of the string $r_{i,t}$ generated by the state $X_{i,t}$.

-General Forward-Backward (GFB) Algorithm

The computational time for Inside-Outside Algorithm is quite long compared to the Forward-Backward Algorithm. For structures generated by stochastic context-free grammars, the General Forward-Backward Algorithm is more efficient than the Inside-Outside Algorithm. Define

$F^{(p)}_{m,n,k,l}(i,t)$ $(t = \{0, 1, ..., T\}$, $i = \{1, 2\})$ as the probability of state corresponding to observation $r_{3-i,t}$ is state m, state corresponding to observation $r_{3-i,t-1}$ is state n, state corresponding to observation $r_{i,t-1}$ is state k and state corresponding to observation $r_{i,t}$ is state l, given the observations and model parameters. We use p to denote the p^{th} iterative procedure, then

$$F^{(p)}_{m,n,k,l}(i,t) = P(m = X_{3-i,t}, n = X_{3-i,t-1}, k = X_{i,t-1}, l = X_{i,t}|O, \Theta^{(p)}), \tag{3.1}$$

and define $G^{(p)}_m(i,t)$ as the probability of the state corresponding to observation $r_{i,t}$ is state m, then

$$G^{(p)}_m(i,t) = P(m = X_{i,t}|O, \Theta^{(p)}), \tag{3.2}$$

For any Markov process, if its state sequence satisfy the following property, then general Forward-Backward Algorithm (73) can be applied to it: the probability of all-state sequence S can be decomposed as products of probabilities of conditional-independent subset-state sequences $U_0, U_1, ...$, i.e., $P(S) = P(U_0)P(U_1/U_0)...P(U_i/U_{i-1})...$, where $U_0, U_1, ..., U_i, ...$ are subsets of all-state sequence in the proposed system (74). Define the observation sequence corresponding to each subset-state sequence U_i as OB_i. Subset-state sequences for our proposed model are shown in Fig. 28.

Define the forward probability $\alpha U_u(u), u = 1, 2, ...$ as the probability of observing the observation sequence $OB_v(v \leq u)$ corresponding to subset-state sequence $U_v(v \leq u)$ and having state sequence for u-th product component in the decomposing formula as U_u, given model

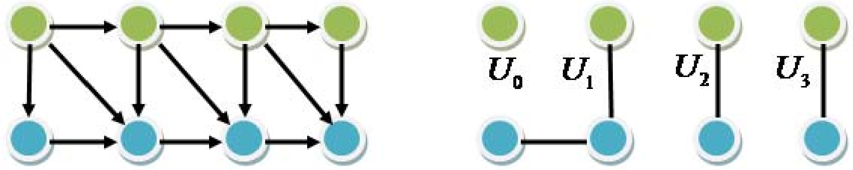


Figure 28. Proposed Markov-modulated stochastic context-free grammars (SCFGs) and its corresponding conditional-independent subset-state sequence decomposition structure for the GFB algorithm.

parameters Θ, i.e., $\alpha U_u(u) = P(S(u) = U_u, OB_v, v \leq u|\Theta)$, and the backward probability $\beta U_u(u), u = 1, 2, ...$ as the probability of observing the observation sequence $OB_v(v > u)$ corresponding to subset-state sequence $U_v(v > u)$, given state sequence for u-th product component as U_u and model parameters Θ, i.e., $\beta U_u(u) = P(OB_v, v \geq u|S(u) = U_u, \Theta)$. The recursive updating formula of forward and backward probabilities can be obtained as

$$\alpha U_u(u) = [\sum_{u-1} \alpha U_{u-1}(u-1) P(U_u|U_{u-1}, \Theta)] P(OB_u|U_u, \Theta) \tag{3.3}$$

$$\beta U_u(u) = \sum_{u+1} P(U_{u+1}|U_u, \Theta) P(OB_{u+1}|U_{u+1}, \Theta) \beta U_{u+1}(u+1) \tag{3.4}$$

The estimation formulas of $F^{(p)}_{m,n,k,l}(i,t)$, $G^{(p)}_m(i,t)$ are

$$G^{(p)}_m(i,t) = \frac{\alpha U_u(u) \beta U_u(u)}{\sum_{u:U_u(i,t)=m} \alpha U_u(u) \beta U_u(u)} \tag{3.5}$$

$$F^{(p)}_{m,n,k,l}(i,t) = \frac{\alpha U_{u-1}(u-1)P(U_u|U_{u-1},\Theta)P(OB_u|U_u,\Theta)\beta U_u(u)}{\sum_u \sum_{u-1} \alpha U_{u-1}(u-1)P(U_u|U_{u-1},\Theta)P(OB_u|U_u,\Theta)\beta U_u(u)} \tag{3.6}$$

-Expectation-Maximization Technique

In this subsection, EM algorithm is applied for parameter estimation. For a Markov chain with M states, $C_{i,t} = (C^1(A \rightarrow \eta; r_{i,t}), C^2(A \rightarrow \eta; r_{i,t}), ..., C^M(A \rightarrow \eta; r_{i,t}))$ and $C^m(A \rightarrow \eta; r_{i,t})$ is the number of counts the production rule $A \rightarrow \eta$ is applied in deriving $r_{i,t}$ based on state m. Let $\Phi = \{a_{m,n,k,l}, P^1(A \rightarrow \eta), ..., P^M(A \rightarrow \eta)\}$ be the model parameters, where $P^m(A \rightarrow \eta)$ is the set of production rules probabilities based on state m.

The Expectation step of the EM algorithm yields the following equation:

$$\begin{aligned} E_{\Phi^{(i)}}(log\mathfrak{Ln}(\Phi)) = \quad & \sum_{i=1}^{2}\sum_{t=0}^{T}\sum_{X_{i,t}}\sum_{A^{X_{i,t}}}\sum_{T^{X_{i,t}}} E_{\Phi^{(i)}}(C^{X_{i,t}}(A \rightarrow \eta; r_{i,t})) \times logP^{X_{i,t}}(A \rightarrow \eta)G_m(i,t) \\ & + \sum_{i=1}^{2}\sum_{t=0}^{T}\sum_{U}\sum_{U-1} log(a_{m,n,k,l})F_{m,n,k,l}(i,t) + \sum_{i=1}^{2}\sum_{t=1}^{T}\sum_{X_{1,0}} log(X_{1,0})G_m(1,0) \end{aligned} \tag{3.7}$$

where $E_{\Phi^{(i)}}(C^{X_{i,t}}(A \rightarrow \eta; r_{i,t}))$ can be computed using inside and outside variables, $a_{m,n,k,l}$ denote the probability of state m occurring immediately after the state n, k, l. The Maximization step of the EM algorithm could be computed by applying Lagrange Multiplier. Since the parameters we wish to optimize are independently separated into three terms in the sum, the three terms are the estimates of the prior distribution, the transition matrix, and the production rule probabilities, we can optimize the parameter term by term.

We can get the iterative updating formulas of parameters of the proposed model,

$$X_{1,0}^{(p+1)} = P(G_m(1,0)^{(p)}|O, \Theta^{(p)}) \tag{3.8}$$

$$P^{X_{i,t}}(A \to \eta)^{(p+1)} = \frac{\sum_{i=1}^{2} \sum_{t=1}^{T} E_{\Phi^{(i)}}^{(p)}(C^{X_{i,t}}(A \to \eta; r_{i,t}))G_m^{(p)}(i,t)}{\sum_{\eta} \sum_{i=1}^{2} \sum_{t=1}^{T} E_{\Phi^{(i)}}^{(p)}(C^{X_{i,t}}(A \to \eta; r_{i,t}))G_m^{(p)}(i,t)} \tag{3.9}$$

$$A = a_{m,n,k,l}^{(p+1)} = \frac{\sum_{i}^{2} \sum_{t}^{T} F_{m,n,k,l}^{(p)}(i,t)}{\sum_{l=1}^{M} \sum_{i}^{2} \sum_{t}^{T} F_{m,n,k,l}^{(p)}(i,t)}, \tag{3.10}$$

Once the stochastic context-free grammars (SCFGs) for all classes have been trained, the classification of new sample can be performed by computing the likelihood that SCFG i best describes the test sample. Given SCFGs for the L classes, and the observation sequence $O_1, O_2, ..., O_m$, we assign class label k as the SCFG that maximizes the likelihood given new sample. This computation is efficiently performed using the forward recursion procedure in the Baum-Welch Algorithm (59).

3.4 Application of Stochastic Grammars: Multiple-Trajectory video Classification

We propose to embody multiple-trajectory in a modified grammar representation, and implement statistical estimation algorithms. Section A presents modified grammar representation of multiple-trajectory.

3.4.1 Grammar Representation of Multiple-Trajectory

A trajectory in our work is a 2-D T-tuple corresponding to the x and y-axes projections of the object's centroid location at each instant of time, $\{(X_t, Y_t), t = 0, ..., T\}$. We classify the trajectories into separate classes. The word "class" refers to a type of activity for which we have a sufficient number of samples to train the system. The trajectories are segmented at points of maximum change in curvature of the trajectory. We represent the subtrajectories using PCA because of its optimal energy compaction properties (70). For M multiple trajectories, the PCA coefficient vectors of the input trajectories after segmentation are posted as an observation sequence $O_{i,t}$, where $t = \{0, 1, ..., T\}$, $i = \{1, 2, ..., M\}$; Let there be $T + 1$ subtrajectories for each trajectory. Then, the state variable $X_{i,t}$ represents one state at specific time t for trajectory i, where $t = \{0, 1, ..., T\}$, $i = \{1, 2, ..., M\}$; Each $X_{i,t}$ represents hidden state corresponding to observation $O_{i,t}$. We consider 4 possible directional observations $O_{i,t}$ emitted from each state variable $X_{i,t}$, which are up, down, left and right. We, therefore, propose to use continuous density stochastic context-sensitive grammars (SCSGs), where each state is modeled by a mixture of Gaussian. Without loss of generality, we illustrate our framework based on 2 trajectories. The undirected edge between 2 state (trajectories) denotes the "interaction" among the process (trajectories) at the same time. The directed edge between 2 state represents that the current state is influenced by the previous state. "Influences" or "interactions" among the process (trajectories) are modeled as dependencies of state variables among process (trajectories). We constrain the probabilistic dependencies of state in one trajectory at time t, on its own state at time $t-1$, as well as on the states of other trajectories that "interact" or "influence" on it.

Summary of stochastic context-sensitive grammars (SCSGs) model training and classification algorithms

-Training:

1. Decompose the stochastic context-sensitive grammars (SCSGs) model into multiple stochastic context-free grammars (SCFGs) model and apply the learning algorithm on each of it sequentially.
2. Assign initial values to $\{X_{1,0}, a_{m,n,k,l}, P^{X_{i,t}}(A \rightarrow \eta)\}$.
3. Update the forward and backward probabilities according to (3.3) and (3.4) and also update $F^{(p)}_{m,n,k,l}(i,t)$, $G^{(p)}_{m}(i,t)$ according to (3.5) and (3.6).
4. Update $\{X^{(p+1)}_{1,0}, a^{(p+1)}_{m,n,k,l}, P^{X_{i,t}}(A \rightarrow \eta)^{(p+1)}\}$ according to (3.8)-(3.10) using EM Algorithm.
5. Back to step 2, calculate the new log-likelihood function, stop if log-likelihood is below pre-set threshold.

-Classification:

1. To classify a trajectory into one of the classes, compute the log-likelihood that each model gives to the trajectory. If the i'th model is the most likely, then declare the class of the trajectory to be class i.

3.4.2 Simulation Results on Video Classification Application

In this section, we report experimental results of the proposed stochastic context-sensitive grammars (SCSGs) model applied to the problem of video classification that contains multiple motion trajectories. For simplicity, we only test our proposed algorithm on the video sequences including 2 trajectories. We first decompose the SCSGs model into multiple stochastic context-free grammars (SCFGs) model. Then a series of algorithms are expected to learn SCFGs sequentially and predict the categories of new sample based on learned stochastic grammars.

We test the classification performance of both stochastic context-sensitive grammars (SCSGs) model classifier, traditional coupled stochastic context-free grammars (CSCFGs) model classifier on 2 data sets: (A) Synthetic TWO-HANDS data set. (B) Subset of the Context Aware Vision using Image-based Active Recognition (CAVIAR) data set which contains video clips of multiple trajectories with interactions. The performances for the data set are reported in terms of the following criteria.

1) The average Receiver Operating Characteristics (ROC) curve.

The ROC curve captures the trade-off between false positive rate versus the true positive rate as the threshold on likelihood at the output of the classifier is varied. As a baseline case, the performance of a uniformly distributed random classifier is also presented in the ROC curve.

2) The Area Under Curve (AUC).

The AUC is a convenient way of comparing classifiers, which varies from 0.5 (random classfier) to 1.0 (ideal classifier).

3) The Optimal Operating Point (OOP)

The Optimal Operating Point based on equal error rate criterion yields an optimal trade-off between false positives and true positives under the equal cost assumption between false acceptance and correct acceptance.

4) Classification Accuracy.

The Classification Accuracy is defined as:

$$P_{Accuracy} = 1 - |F|/|S|, \tag{3.11}$$

where $|F|$ represents the cardinality of the false positives set, and $|S|$ represents the cardinality of the whole data set.

The synthetic TWO-HANDS data set is taken from the Australian Sign Language (ASL) data set. We select ten sign words, and combine each two different sign words, e.g., sign words "god" and "boy", as a class, to form a new class "god+boy" which contains two-hands 2D trajectory (x- and y-) samples. In this way, we construct 45 classes of two-hand trajectories, each of which contains 30 samples, totally 1350 two-hand trajectory samples to form our TWO-HANDS data set. We first use 50% samples as training data, and the rest as testing data. It should be noted here that the Receiver Operating Characteristics curve in Fig. 29 represents an average of 45 individual curves for a total of 675 queries for classification. As can be seen

TABLE VI

PERFORMANCE STATISTICS FOR TWO-HANDS DATA SET.

Classifier	proposed SCSGs	CSCFGs
OOP	0.81	0.77
AUC	0.86	0.83
Accuracy Rate	0.79	0.77

from the figure, our proposed SCSGs model classifier is more accurate than traditional CSCFGs model classifier. The Optimal Operating Point (OOP), Area Under Curve (AUC) and Average Classification Accuracy for two systems on the data set in Fig. 29 can be further illustrated from Table VI. Our proposed SCSGs model classifier achieves a 79% accurate rate of classification, and CSCFGs model classifier is 2% lower than proposed method. This experiment demonstrates the robustness and superior performance of our proposed classifier. Cross-validation method is used for estimating the performances of the two classifiers.

We further report the results on a wide range of data sizes from 10 to 45 classes. All classes are selected from the data set of 1350 trajectories with 45 classes. The results are reported in terms of the average probability of classification accuracy of the classifiers in Table VII. As the number of classes increases, the classification accuracy for both methods will decrease. However, our proposed method holds the superior performance compared to CSCFGs. To further compare our method with CSCFGs, the first 10 classes from original 45 classes are selected to provide further analysis of classification performances. Bar chart in Fig. 30 is used to demonstrate the

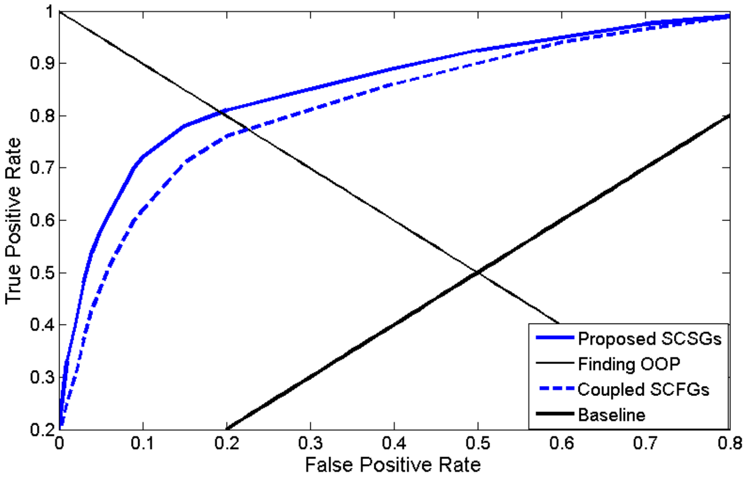


Figure 29. Illustration of the region of convergence (ROC) curve and Area Under Curve (AUC) for the proposed SCSGs and CSCFGs for TWO-HANDS data set.

accuracy rate for each class first. As can be seen from the figure, our SCSGs perform much better than CSCFGs, since our SCSGs can characterize the intrinsic state transition structure and behavior of complex systems involving states of multiple trajectories. The confusion matrix will summarize the results in more details. The confusion matrix for our proposed SCSGs and traditional CSCFGs are illustrated in Table VIII. For example, in the confusion matrix for CSCFGs, of the 15 actual samples for each class 4 and 5, the classifier predicts that one of the actual class 4 is class 5, and one of the actual class 5 is class 4; while our proposed SCSGs can distinguish class 4 and 5 with 100% accuracy. This example demonstrates that our proposed SCSGs perform much better than CSCFGs. Take class 2 as another example, in the confusion matrix for our SCSGs, of the 15 actual class 2, the proposed SCSGs predict that two are class

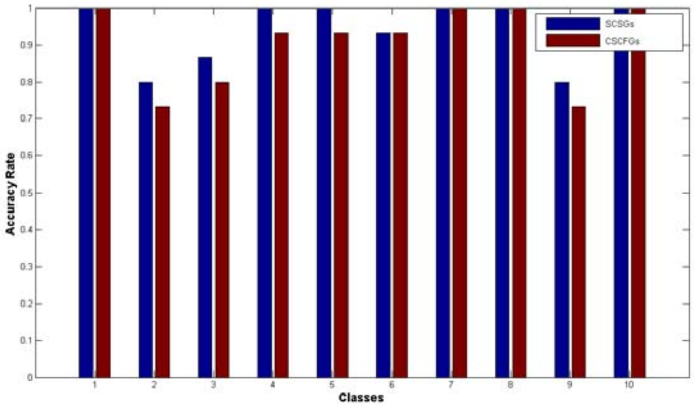


Figure 30. Illustration of the accuracy rate of each class for SCSGs and CSCFGs for TWO-HANDS data set.

1 and one is class 3, while in the confusion matrix for CSCFGs, of the 15 actual class 2, the CSCFGs predict that three are class 1 and one is class 3. We can see from this comparison that CSCFGs classifier has trouble distinguishing between class 1, 2 and 3. Our proposed method can distinguish one more class 2 from class 1. However, since the trajectories from these three classes are so similar to each other that it is indeed very difficult to distinguish them completely. The similar situations also happen to class 3 and 9. Table IX presents the classification accuracy of the data set of 1350 trajectories with 45 classes with different percentage of training data. As can be seen from the table, more training data can help us obtain more stable and accurate classification performance.

Then we test both SCSGs classifier, traditional CSCFGs classifier on CAVIAR data set, we select data classes that have 2 people interacting with each other, and use 50% samples as

TABLE VII

ACCURACY RATE VERSUS NUMBER OF CLASSES FOR TWO-HANDS DATA SET.

data size (classes)	10	20	30	45
proposed SCSGs	0.94	0.89	0.85	0.79
CSCFGs	0.91	0.88	0.82	0.77

TABLE VIII

CONFUSION MATRIX OF PROPOSED SCSGS AND CSCFGS FOR TWO-HANDS DATA SET.

			predicted class No.									
			1	2	3	4	5	6	7	8	9	10
actual class No.	1	SCSGs	15	0	0	0	0	0	0	0	0	0
		CSCFGs	15	0	0	0	0	0	0	0	0	0
	2	SCSGs	2	12	1	0	0	0	0	0	0	0
		CSCFGs	3	11	1	0	0	0	0	0	0	0
	3	SCSGs	1	1	13	0	0	0	0	0	0	0
		CSCFGs	1	2	12	0	0	0	0	0	0	0
	4	SCSGs	0	0	0	15	0	0	0	0	0	0
		CSCFGs	0	0	0	14	1	0	0	0	0	0
	5	SCSGs	0	0	0	0	15	0	0	0	0	0
		CSCFGs	0	0	0	1	14	0	0	0	0	0
	6	SCSGs	0	0	0	0	1	14	0	0	0	0
		CSCFGs	0	0	0	0	1	14	0	0	0	0
	7	SCSGs	0	0	0	0	0	0	15	0	0	0
		CSCFGs	0	0	0	0	0	0	15	0	0	0
	8	SCSGs	0	0	0	0	0	0	0	15	0	0
		CSCFGs	0	0	0	0	0	0	0	15	0	0
	9	SCSGs	0	0	0	0	0	0	0	3	12	0
		CSCFGs	0	0	0	0	0	0	0	4	11	0
	10	SCSGs	0	0	0	0	0	0	0	0	0	15
		CSCFGs	0	0	0	0	0	0	0	0	0	15

TABLE IX

ACCURACY RATE VERSUS PERCENTAGE OF TRAINING SET FOR TWO-HANDS DATA SET.

percentage of training data	30%	50%	80%
proposed SCSGs	0.60	0.79	0.86
CSCFGs	0.59	0.77	0.83

training data, and the rest as testing data. There are 9 classes of 180 two-people interacting trajectories. Fig. 31 lists samples of 2 trajectories from 2 classes of 9 classes in CAVIAR data set. The Receiver Operating Characteristics (ROC) curve for a total of 90 queries for classification is shown in Fig. 32. As can be seen from the figure, the performance of SCSGs classifier is better than traditional CSCFGs classifier. The Optimal Operating Point (OOP), Area Under Curve (AUC) and Average Classification Accuracy for two systems on the CAVIAR data set in Fig. 32 can be further illustrated from Table X. As shown in Table X, the average classification accuracy of our classifier reaches 85%, which is 3% higher than traditional CSCFGs classifier. The data presented in all Tables have been averaged over ten trials.

TABLE X

PERFORMANCE STATISTICS FOR CAVIAR DATA SET.

Classifier	proposed SCSGs	CSCFGs
OOP	0.84	0.79
AUC	0.90	0.86
Accuracy Rate	0.85	0.82

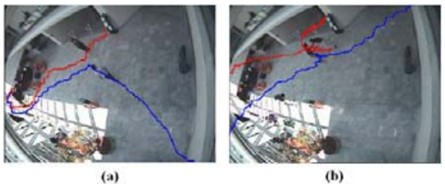


Figure 31. Illustration of the region of convergence (ROC) curve and Area Under Curve (AUC) for the proposed SCSGs and CSCFGs for CAVIAR data set.

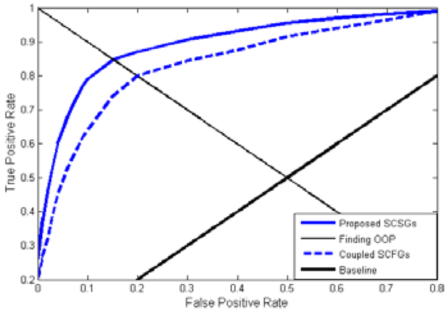


Figure 32. Multiple-trajectories samples of two classes in CAVIAR data set: (a) 2 trajectories sample from class 1: ”Two people meet and walk together”; (b) 2 trajectories sample from class 2: ”Two people meet, fight and run away”.

CHAPTER 4

COMPRESSED SENSING GAME THEORY (CSGT): A NOVEL POLYNOMIAL COMPLEXITY SOLUTION TO NASH EQUILIBRIA IN DYNAMICAL GAMES

4.1 Compressed sensing approach to solution of under-determined systems of linear equations

In this section, we provide a brief review of the idea behind the compressed sensing theory. We use the term signal to represent the solution data we are trying to acquire. Let $x \in \mathbb{R}^n$ represent a signal and $y \in \mathbb{R}^m$ a vector of linear measurements formed by taking inner products of x with a set of linearly independent vectors $a_i \in \mathbb{R}^n$, $i = 1, 2, ..., m$. In matrix format, the measurement vector is $y = Ax$, where $A \in \mathbb{R}^{m \times n}$ has rows a_i^T, $i = 1, 2, ..., m$. When the number of measurements m is equal to n, the process of recovering x from the measurement vector y simply entails solving a linear equations system. However, in many applications, one only has very fewer measurements compared to a much larger dimension of space the signal x resides in, i.e., $m \ll n$. In that case, the linear system $Ax = y$ is typically under-determined, permitting infinitely many solutions. In order to have a unique solution, one need to apply various regulatory conditions.

In compressed sensing, one adds the constraint of sparsity, allowing only solutions to have smallest number of nonzero coefficients. Specifically, we are trying to solve the follow optimization problem.

$$\min\{\|x\|_0 : Ax = y\}, \tag{4.1}$$

where the quantity $\|x\|_0$ denotes the number of non-zeros entries in x. (Equation 4.1) is a combinatorial optimization problem with a prohibitive complexity if solved by enumeration, and thus is not tractable. An alternative model is to replace (Equation 4.1) by (Equation 4.2) and solve a computationally tractable convex optimization problem:

$$\min\{\|x\|_1 : Ax = y\}. \tag{4.2}$$

Under favorable conditions the combinational problem (Equation 4.1) and convex programming (Equation 4.2) share a common solution (75). This equivalence result allows one to solve the L_1 problem, which is much easier than the original L_0 problem.

In this paper, we will employ the compressed sensing theory to solve the Nash equilibrium which can be formulated as solutions of a under-determined system of linear equations. More specifically, we use the basis pursuit model (Equation 4.2) for recovery represents a fundamental instance of compressed sensing. Certainly not the only one, many other recovery methods such as greedy-type algorithms are also available (76).

Theory of compressed sensing presently consists of two components: recoverability and stability. Recoverability addresses the central questions: what types of measurement matrices and recovery procedures ensure exact recovery of all k-sparse signals and how many measurements are sufficient to guarantee such a recovery? On the other hand, stability addresses the robustness issues in recovery when measurements are noisy and/or sparsity is inexact. We first review an important concept in compressed sensing.

Definition 1 *A measurement matrix A satisfies the* Restricted Isometry Property (RIP) *if the following inequality holds for all i-sparse vector x, $i \leq m$ and $\epsilon \in (0,1)$*

$$(1-\epsilon)\|x\|_2 \leq \|Ax\|_2 \leq (1+\epsilon)\|x\|_2 \tag{4.3}$$

Recoverability is ensured if the matrix $A \in \mathbb{R}^{m\times n}$ holds RIP for certain pairs of (i, ϵ) (75).

Choosing a measurement matrix A with $m < n$ that has proper RIP ensures exact recovery of signal. In practice, it is almost always the case that either measurements or the measurement matrix is inexact, or both. The compressed sensing stability studies the issues concerning how accurately a compressed sensing approach can recover signals under these circumstances (77). Stability results have been established for (Equation 4.2) and its extension

$$\min\{\|x\|_1 : \|Ax - y\|_2 \leq r\}. \tag{4.4}$$

Most compressed sensing methods have been shown to possess recoverability with known stability (76).

4.2 Compressed Sensing Framework and Nash Equilibrium

Let (S, u) be a game with n players, where S_i is the strategy set for player i, $S = S_1 \times S_2 ... \times S_n$ is the set of strategy profiles and u is the payoff function for $s \in S$. Let s_i be a strategy profile of player i and s_{-i} be a strategy profile of all players except for player i. When each player $i \in \{1, ..., n\}$ chooses their corresponding strategy s_i, which results in a strategy profile $s = (s_1, ..., s_n)$, then player i obtains his payoff $u_i(s)$. Note that the payoff of individual player depends on the strategy profile chosen by all players.

Definition 2 *A strategy vector $s \in S$ is said to be a* Nash equilibrium *if for all players i and each alternate strategy $s'_i \in S_i$, we have that*

$$u_i(s_i, s_{-i}) \geq u_i(s'_i, s_{-i}). \tag{4.5}$$

In other words, no player i can change his chosen strategy from s_i to s'_i and thereby improve his payoff, assuming that all other players stick to the strategies they have chosen in s. The Nash equilibrium we have considered so far are called pure strategy equilibrium. For the notion of mixed Nash equilibrium, let us enhance the choices of players so each one can pick a probability distribution over his set of possible strategies; such a choice is called a *mixed strategy.*

A *correlated equilibrium* is a probability distribution over strategy vector s (78). Let $p(s)$ denote the probability of strategy vector s, where we also use the notation $p(s) = p(s_i, s_{-i})$ when talking about a player i.

Definition 3 *The distribution is a* correlated equilibrium *if for all players i and all strategies $s_i, s'_i \in S_i$, we have the inequality*

$$\sum_{s_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i}} p(s_i, s_{-i}) u_i(s'_i, s_{-i}). \tag{4.6}$$

If player i receives a suggested strategy s_i, the expected profit of the player cannot be increased by switching to a different strategy $s'_i \in S_i$. Nash equilibria are special cases of correlated equilibria, where the distribution over S is the product of independent distributions for each player. Therefore Nash equilibrium is a special case of correlated equilibrium. More precisely, we have the following theorem revealing the relationship between correlated equilibria and Nash equilibria in a 2-player game (79).

Theorem 1 *In any non-degenerate 2-player game, the Nash equilibria reside in vertices of the polytope formed by correlated equilibria.*

Notice that the boundaries of the a polytope are determined by a system of linear equations. While the vertices are characterized as the solutions of various pairs of those linear equations, or equivalently the sparse solution of the differences between those pairs. The traditional compressed sensing is to seek sparse solution for under-determined system of equations. But in $Ax = y$ here, A is over-determined. One efficient way to solve this problem is to convert it to compressed sensing problem by letting $A^{-1}Ax = A^{-1}y$, finally we get $x = A^{-1}y$. A^{-1} here is the pseudo-inverse of A. We then can solve the optimization problem using the compressed sensing theory.

Given all the payoff functions of the 2-player mixing game, we can use the same idea to solve all the vertices of the polytope formed by solution sets of correlated equilibria. By Theorem 1, at least one of the vertices is the Nash equilibrium. We point out the major difference of our method to directly solving system of linear equations is that the latter case requires solving the equations for combinatorically many times since the vertices could be formed by combinatorically many boundaries. While our method only need to perform a joint convex optimization, although at the expense of combinatorically many storage requirement. For a 2-player mixing game, the following procedure is used to formulate the matrix A. For example, in a 2-play game, The pay off matrix of person a is $[a_1, a_2; a_3, a_4]$, the pay off matrix of person b is $[b_1, b_2; b_3, b_4]$; For the matrix A, we obtain the first row of matrix by doing substraction operations between second row and the first row of person a. we got $[a_3 - a_1, 0, a_4 - a_2, 0]$; The second row of the matrix is obtained by doing substraction operation between second row and the first row. We finally obtain $[0, a_1 - a_3, 0, a_2 - a_4]$. We perform the same operation for the person two. Finally, the 5*4 matrix of A we obtain is $[a_3 - a_1, 0, a_4 - a_2, 0; 0, a_1 - a_3, 0, a_2 - a_4; b_2 - b_1, b_4 - b_3, 0, 0; 0, 0, b_1 - b_2, b_3 - b_4; 1, 1, 1, 1]$;

We have the following theorem to characterize the usability of our method for solving the Nash equilibria.

Theorem 2 *Assume the payoff matrix $A^{-1} \in \mathbb{R}^{m \times n}$ satisfies the $(n-1, \sqrt{2}-1)-RIP$ condition, the compressed sensing based method will find the exact Nash equilibria.*

In practice, commonly one is not able to have a precise description of the payoff matrix A^{-1}. Instead, A^{-1} is a random matrix *per se.* We can model the uncertainty in the payoff as a Gaussian random matrix, which results A^{-1} as a Gaussian random matrix. We have the following theorem for the case where A^{-1} is a Gaussian matrix.

Theorem 3 *Given a Gaussian payoff matrix $A^{-1} \in \mathbb{R}^{m \times n}$, if with probability at least $1 - \delta$, the matrix $\frac{1}{\sqrt{m}}A^{-1}$ satisfies the $(i, \epsilon) - RIP$ property provided*

$$m \geq \frac{Ci}{\epsilon} \log\left(\frac{n}{\epsilon^2 i}\right), \tag{4.7}$$

where $i \geq 1$, $\epsilon \in (0, 1/2)$ and $\delta \in (0, 1)$. C is a constant which only depends on δ. Then with probability p where $p \sim O(e^{1-\delta})$, the compressed sensing based method will solve the exact Nash equilibria.

4.3 Experimental Results

Compressed-Sensing Game Theory (CSGT) framework serves as an efficient way to solve Nash equilibria for certain classed of 2-player games. Therefore, one of the advantages of CSGT is that it is computationally less expensive than is Lemke-Howson algorithm (80). To evaluate the performance of our CSGT, we ran several sets of experiments. In the first set of experiments, we compared the performance of CSGT to that of Lemke-Howson algorithm (80) on two 2-play games including battle of the sexes and prisoner's dilemma. The compressed sensing package (http://dsp.rice.edu/cs) is adopted in our experiment.

In battle of the sexes game, the husband would most of all like to go to the football game, while their wife would like to go to the opera. Both would prefer to go to the same place rather than different ones. The payoff matrix in Table XI shows an example of battle of the sexes game, where the wife chooses row and the husband chooses a column. In each cell, the first number represents the payoff to the wife and the second number represents the payoff to the husband. In our experiment, we formulate $5 * 4$ matrix by substraction operations. The A matrix in this example is $[-3, 0, 1, 0; 0, 3, 0, -1; -1, 3, 0, 0; 0, 0, 1, -3; 1, 1, 1, 1]$

TABLE XI

THE PAYOFF MATRIX OF BATTLE OF THE SEXES GAME

	Opera	Football
Opera	3, 1	0, 0
Football	0, 0	1, 3

Both CSGT and Lemke-Howson algorithm (80) were executed on this two-player game for 100 times. As can be seen from Fig. 1, the Nash equilibria (blue) are vertices of the correlated equilibria. It it also shown that our compressed sensing framework find the Nash equilibrium $(0.75, 0.75)$ first. Table XII compares the average computational time and the first

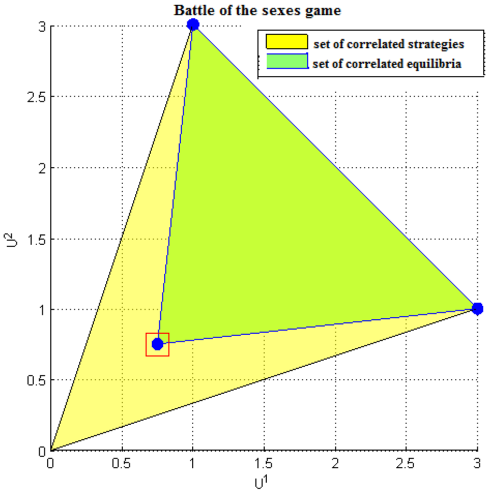


Figure 33. battle of the sexes game

TABLE XII

STATISTICAL PERFORMANCE FOR BATTLE OF THE SEXES GAME.

Method	NE	CPU time
CSGT	(0.75, 0.75)	0.031
Lemke-Howson	(0.75, 0.75)	1.16

Nash equilibrium solution found by these two methods. The table 2 demonstrates that CSGT solves the game far more quickly than Lemke-Howson.

In prisoner's dilemma game, each player chooses to either ”cooperate” or ”defect”. The payoff matrix in Table XIII shows an example of prisoner's dilemma game, where the player 1 chooses row and the player 2 chooses a column. In each cell, the first number represents the payoff to the player 1 and the second number represents the payoff to the player 2. The matrix implies that the "both cooperate" outcome is better than the "both defect" outcome.

TABLE XIII

THE PAYOFF MATRIX OF BATTLE OF THE SEXES GAME.

	Cooperate	Defect
Cooperate	4, 4	5, 1
Defect	1, 5	0, 0

Both CSGT and Lemke-Howson algorithm (80) were executed on this two-player game for 100 times. Table XIV compares the average computational time and the first Nash equilibrium solution found by these two methods. The table XIV demonstrates that CSGT solves the game far more quickly than Lemke-Howson.

TABLE XIV

STATISTICAL PERFORMANCE FOR PRISONER'S DILEMMA GAME.

Method	NE	CPU time
CSGT	(2.5, 2.5)	0.016
Lemke-Howson	(2.5, 2.5)	0.13

CITED LITERATURE

1. Khan, S. and Shah, M.: Consistent labeling of tracking objects in multiple cameras with overlapping fields of view. IEEE Trans. on Pattern Analysis and Machine Intelligence, 25(10):1355–1360, 2003.

2. Bar-Shalom, Y. and Jaffer, A.: Tracking and data association. Academic Press, San Diego, California, USA, 1998.

3. Hue, C., Cadre, J., and Perez, P.: Sequential monte carlo methods for multiple target tracking and data fusion. IEEE Transactions on Signal Processing, 50(2):309–325, 2002.

4. Rasmussen, C. and Hager, G.: Probabilistic data association methods for tracking complex visual objects. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(6):560–576, 2001.

5. MacCormick, J. and Blake, A.: A probabilistic exclusion principle for tracking multiple objects. International Journal of Computer Vision, 39(1):57–71, 2000.

6. Isard, M. and MacCormick, J.: Bramble: a bayesian multiple-blob tracker. In Proceedings of the 8th IEEE International Conference on Computer Vision (ICCV 01), volume 2, pages 34–41, 2001.

7. Tao, H., Sawhney, H., and Kumar, R.: A sampling algorithm for detection and tracking multiple objects. In Proceedings of the IEEE International Conference on Computer Vision Workshop on Vision Algorithm, 1999.

8. Khan, Z., Balch, T., and Dellaert, F.: An mcmc-based particle filter for tracking multiple interacting targets. In Proceedings of the 8th European Conference on Computer Vision, volume 4, pages 279–290, 2004.

9. Qu, W., Schonfeld, D., and Mohamed, M.: Real-time interactively distributed multi-object tracking using a magnetic-inertia potential model. In Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV 05), volume 1, pages 535–540, 2005.

CITED LITERATURE (Continued)

10. Isard, M. and Blake, A.: Condensation conditional density propagation for visual tracking. International Journal of Computer Vision, 29(1):5–28, 1998.

11. Lan, X. and Huttenlocher, D.: A unified spatio-temporal articulated model for tracking. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 04), volume 1, pages 722–729. IEEE, 2004.

12. Nickels, K. and Hutchinson, S.: Model-based tracking of complex articulated objects. IEEE Trans. Robot. Autom., 17(1):28–36, 2001.

13. J. Deutscher, A. B. and Reid, I. D.: Articulated body motion capture by annealed particle filtering. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 00), 2000.

14. Choo, K. and Fleet, D. J.: People tracking using hybrid monte carlo filtering. In Proceedings of the IEEE International Conference on Computer Vision (ICCV 01), 2001.

15. Chang, W. Y., Chen, C. S., and Hung, Y. P.: Appearance-guided particle filtering for articulated hand tracking. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 05), 2005.

16. Berclaz, F. F. J. and Fua, P.: Robust people tracking with global trajectory optimazation. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 06), pages 744–750, 2006.

17. Cai, Q. and Aggarwal, J. K.: Tracking human motion in structured environments using a distributed-camera system. IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(11):1241–1247, 1999.

18. Stauffer, C. and Tieu, K.: Automated multi-camera planar tracking correspondence modeling. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 03), volume 1, pages 259–266, 2003.

19. Micheloni, C., Foresti, G. L., and Snidaro, L.: A network of cooperative cameras for visual surveillance. In IEE Proceedings, Vision, Image & Signal Processing, volume 152, pages 205–212, 2005.

20. Iwase, S. and Saito, H.: Parallel tracking of all soccer players by integrating detected positions in multiple view images. In Proceedings of the 17th IEEE International Conference on Pattern Recognition (ICPR 04), volume 4, pages 751–754, 2004.

CITED LITERATURE (Continued)

21. Gatica-Perez, D., Odobez, J. M., Ba, S., Smith, K., and Lathoud, G.: Tracking people in meetings with particles. In Proceedings of the IEEE Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS'05), 2005.

22. Nummiaro, K., Koller-Meier, E. B., Svoboda, T., Roth, D., and Gool, L. V.: Color-based object tracking in multi-camera environments. In Proceedings of the 25th DAGM Symposium on Pattern Recognition, pages 591–599, 2003.

23. Du, W. and Piater, J.: Multi-view tracking using sequential belief propagation. In Proceedings of the Asian Conference on Computer Vision (ACCV 06), 2006.

24. Jaakkola, T. S.: Tutorial on variational approximation methods. Technical report, MIT AI Lab TR, 2000.

25. Jordan, M., Ghahramani, Z., T.Jaakkola, and L.Saul: An introduction to variational methods for graphical models. Machine Learning, 37:183–233, 2000.

26. Murphy, K. P., Weiss, Y., and Jordan, M.: Loopy belief propagation for approximate inference: An empirical study. In Proceedings of the conference on Uncertainty in Artificial Intelligence, pages 467–475, 1999.

27. Anderson, B. and Moore, J.: Optimal filtering. New York: Prentice Hall Inc., 1979.

28. Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T.: A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. IEEE Transactions on Signal Processing, 50(2):174–189, 2002.

29. Ristic, B., Arulampalam, S., and Gordon, N.: Beyond the Kalman Filter: Particle Filters For Tracking Applications. Norwood, MA: Artech House, 1st ed, 2004.

30. Liu, J. and Chen, R.: Sequential monte carlo methods for dynamic systems. J. Amer. Statist. Assoc., 93(443):1032–1044, 1998.

31. Pan, P. and Schonfeld, D.: Video tracking based on sequential particle filtering on graphs. IEEE Transactions on Image Processing, 20:1641–1651, 2011.

32. Deller, J. R., Hansen, J. H. L., and Proakis, J. G.: Discrete-time processing of speech signals. New York: Wiley-IEEE, 1999.

33. Blackman, S. and Popoli, R.: Design and analysis of modern tracking systems. Norwood, MA: Artech House, 1999.

34. Ivanov, Y. A. and Bobick, A. F.: Recognition of visual activities and interactions by stochastic parsing. IEEE Trans. Pattern Analysis and Machine Intelligence, 22:852–872, 2000.

35. Zhu, W. and Garcia-Frias, J.: Stochastic context-free grammars and hidden markov models for modeling of bursty channels. IEEE Trans. Veh. Technol., 53:666–676, 2004.

36. Durbin, R., Eddy, S., Krogh, A., and Mitchison, G.: Biological sequence analysis: Probabilistic models of proteins and nucleic acids. Cambridge, U.K.: Cambridge Univ. Press, 1998.

37. Fu, K. S.: Syntactic pattern recognition and applications. Englewood Cliffs, NJ: Prentice-Hall, 1982.

38. Joshi, A. and Rambow, O.: A formalism for dependency grammar based on tree adjoining grammar. Proceedings of the first international conferece on meaning text theory, 2003.

39. Wang, A. and Krishnamurthy, V.: Intent inference and syntactic tracking with gmti measurements. IEEE Transactions on Aerospace and Electronic Systems, 2011.

40. Filar, J., Vrieze, K., and Vrieze, O.: Competitive Markov Decision Processes. Springer Verlag, 1997.

41. MacKenzie, A. and Wicker, S.: Game theory and the design of self-configuring, adaptive wireless networks. IEEE Communication Magazine, 39:126–131, Nov. 2001.

42. Brouwer, L.: Uber abbildung von mannigfaltigkeiten. Mathematische Annalen, 71:97–115, 1910.

43. Lemke, C. and Howson, J. J.: Equilibrium points in bimatrix games. J. Soc. Indust. Appl. Math., 12, 1964.

44. Savani, R. and von Stengel, B.: Hard-to-solve bimatrix games. Econometrica, 74:397–429, 2006.

CITED LITERATURE (Continued)

45. Daskalakis, C., Goldberg, P., and Papadimitriou, C.: The complexity of computing a nash equilibrium. In Proceedings of the thirty-eighth annual ACM symposium on Theory of computing, pages 71–78. ACM, 2006.

46. Khachian, L.: A polynomial algorithm in linear programming. SSSR 244: English translation in Soviet Math. Dokl., 20, 1979.

47. Facchinei, F. and Pang, J.: Finite-dimensional Variational Inequalities and Complementarity Problems, volume 1. Springer, 2003.

48. Donoho, D. L.: Compressed sensing. IEEE Transactions on Information Theory, 52:1289–1306, 2006.

49. Pan, P. and Schonfeld, D.: Visual tracking using high-order monte carlo markov chain. In Proceedings of the 15th International Conference on Image Processing (ICIP 08), pages 2636–2639, 2008.

50. Qu, W. and Schonfeld, D.: Real-time decentralized articulated motion analysis and object tracking from videos. IEEE Transactions on Image Processing, 16(8), 2007.

51. Whittaker, J.: Graphical models in applied mathematical multivariate statistics. John wiley & Sons, 1990.

52. Du, W. and Piater, J.: Multi-camera people tracking by collaborative particle filters and principal axis-based integration. In Proceedings of the 7th Asian Conference on Computer Vision, pages 365–374, 2007.

53. Hartley, R. and Zisserman, A.: Multiple view geometry in computer vision. In Cambridge University Press: Cambridge, United Kingdom., 2000.

54. Blake, A. and Isard, M.: Active Contours. Springer-Verlag, Berlin/New York, 1998.

55. Jordan, M. I.: Learning in Graphical Models. Cambridge, MA: MIT Press, 1999.

56. Wu, Y., Hua, G., and Yu, T.: Tracking articulated body by dynamic markov network. In Proceedings of the 9th IEEE International Conference on Computer Vision(ICCV 03), volume 2, pages 1094–1101, 2003.

CITED LITERATURE (Continued)

57. Fleuret, F., Berclaz, J., Lengagne, R., and Fua, P.: Multi-camera people tracking with a probabilistic occupancy map. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008.

58. Durbin, R., Eddy, S., Krogh, A., and Mitchison, G.: Biological sequence analysis: Probabilistic models of proteins and nucleic acids. Cambridge, U.K.: Cambridge Univ. Press, 1998.

59. Rabiner, L. R.: A tutorial on hidden markov models and selected applications in speech recognition. Proc. IEEE, 77:257–286, 1989.

60. Baker, J. K.: Trainable grammars for speech recognition. Speech Comm. Papers for the 97th Meeting of the Acoustical Soc. of Am., pages 547–550, 1979.

61. Sakakibara, Y., Brown, M., Hughey, R., Mian, I. S., Sjolander, K., Underwood, R. C., and Haussler, D.: Stochastic contetx-free grammars to trna modeling. Nucleic Acids Research, 22:5112–5120, 1994.

62. Sakakibara, Y.: Efficient learning of context-free grammars from positive structural examples. Information and Computation, 97:23–60, 1992.

63. Sakakibara, Y. and Kondo, M.: Ga-based learning of context-free grammars using tabular representations. Proc. 16th Int'l Conf. Machine Learning, pages 354–360, 1999.

64. Aho, A. V. and Ullman, J. D.: The theory of parsing, translation and compiling. vol: Parsing, Prentice Hall, 1972.

65. Johansson, G.: Visual perception of biological motion and a model for its analysis. Perception and Psychophysics, 14(2):201–211, 1973.

66. Bashir, F., Khanvilkar, S., Schonfeld, D., and Khokhar, A.: Multimedia systems: Content-based indexing and retrieval. The Electrical Engineering Handbook, Sec. 4, Chapter 6, 2004.

67. Dimitrova, N. and Golshani, F.: Motion recovery for video content classification. ACM Transactions on Information Systems, 13(4):408–439.

68. Fablet, R. and Bouthemy, P.: Motion recognition using spatio-temporal random walks in sequence of 2d motion-related measurements. IEEE Int. Conf. on Image Processing, Oct. 2001:652–655.

CITED LITERATURE (Continued)

69. Sahouria, E. and Zakhor, A.: A trajectory based video indexing system for street surveillance. IEEE Int. Conf. on Image Processing, 1999.

70. Bashir, F. I., Khokhar, A. A., and Schonfeld, D.: Object trajectory-based activity classification and recognition using hidden markov models. IEEE Tansactions on Image Processing, 16:1912–1919, 2007.

71. Taga, N. and Mase, S.: On the convergence of loopy belief propagation algorithm for different update rules. IEICE Transactions on Fundamental Electronics Communication Computing Science, 2006.

72. Wang, A. and Krishnamurthy, V.: Signal interpretation of multifunction radars: Modeling and statistical signal processing with stochastic context free grammar. IEEE Transactions Signal Processing, 56:1106–1119, 2008.

73. Baum, L. E., Petrie, T., Soules, G., and Weiss, N.: A maximization technique occuring in the statistical analysis of probabilistic functions of markov chains. Ann. Math. Stat, I:164–171, 1970.

74. Schonfeld, D. and Bouaynaya, N.: A new method for multidimensional optimization and its application in image and video processing. IEEE Signal Processing Letters, 13:485–488, 2006.

75. Candès, E., Romberg, J., and Tao, T.: Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. Information Theory, IEEE Transactions on, 52(2):489–509, 2006.

76. Tropp, J. A. and Gilbert, A. C.: Signal recovery from random measurements via orthogonal matching pursuit. IEEE Transactions on Information Theory, 53(12):4655–4666, 2007.

77. Rudelson, M. and Vershynin, R.: On sparse reconstruction from fourier and gaussian measurements. Communications on Pure and Applied Mathematics, 61(8):1025–1045, 2008.

78. Aumann, R.: Correlated equilibrium as an expression of bayesian rationality. Econometrica: Journal of the Econometric Society, pages 1–18, 1987.

79. Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V. V.: Algorithmic Game Theory. Cambridge University Press, 2007.

CITED LITERATURE (Continued)

80. Lemke, C. E. and Howson, J. T.: Equilibrium points of bimatrix games. Journal of the Society for Industrial and Applied Mathematics, 1963.

81. Tam, W. J. and Zhang, L.: 3d-tv content generation: 2d-to-3d conversion. IEEE ICME, pages 1869–1872, 2006.

82. lin Chang, Y., Fang, C.-Y., fu Ding, L., yi Chen, S., and gee Chen, L.: depth map generation for 2d-to-3d conversion by short-term motion assisted color segmentation. IEEE ICME, 2007.

83. Rivas, E. and Eddy, S. R.: The language of rna: A formal grammar that includes pseudoknots. Bioinformatics, 16:334–340, 2000.

84. Dempster, A. P., Laird, N. M., and Rubin, D. B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society: Series B, 38:1–38, 1977.

85. Bashir, F. I., Khokhar, A. A., and Schonfeld, D.: Real-time motion trajectory-based indexing and retrieval of video sequences. IEEE Tansactions on Multimedia, 9:58–65, 2007.

86. Azarbayejani, A. and Pentland, A.: Recursive estimation of motion, structure, and focal length. IEEE Trans. Pattern Anal. and Mach. Intell., 17(6):562–575, 1995.

87. Wang, L. and Schonfeld, D.: Game theoretic model for control of gene regulatory networks. In Proceedings of the 2010 IEEE International Conference on Acoustics Speech and Signal Processing, pages 542–545. IEEE, 2010.

88. Hart, S. and Mas-Colell, A.: A simple adaptive proce- dure leading to correlated equilibrium. Econometrica, 68:1127–1150, 2000.

89. Wang, L., Krishnamurthy, V., and Schonfeld, D.: Factor graph-based structural equilibria in dynamical games. In Proceedings of the 2011 IEEE International Conference on Acoustics Speech and Signal Processing, pages 1964–1967. IEEE, 2011.

VITA

NAME:	Jing Huang
EDUCATION:	B.S., in Electronics Engineering, Huazhong University of Science and Technology, 2009 M.S., in Electrical and Computer Engineering, University of Illinois at Chicago, 2012 Ph.D., in Electrical and Computer Engineering, University of Illinois at Chicago, 2013