Interval-Based Hybrid Dynamical System for Modeling Dynamic Events and Structures

Hiroaki Kawashima

Interval-Based Hybrid Dynamical System for Modeling Dynamic Events and Structures

Hiroaki Kawashima

Abstract

This thesis explores the problem of modeling dynamic events and structures based on a novel computational model, named an "interval-based hybrid dynamical system". The model integrates two types of systems that have different concepts of time: dynamical systems, which are suitable for describing physical phenomena (consider time as physical metric entity), and discrete-event systems, which are suitable for describing human subjective or intellectual activities (consider time as ordinal state transition).

Firstly, we assume that a complex dynamic event such as human behavior consists of dynamic primitives (such as "open", "close", and "remain closed" in lip motions). Once the set of dynamic primitives is determined, a complex behavior can be partitioned into "temporal intervals" based on the primitives. Secondly, we assume that not only temporal orders but the duration lengths or temporal differences among beginning and ending time points of the temporal intervals, which we refer to as "timing structures", have crucial information to understand dynamic events appear in human communication.

Based on the assumptions above, we propose an interval-based hybrid dynamical system, which has a two-layer architecture that consists of a finite state automaton and a set of linear dynamical systems. In this architecture, each dynamical system represents a dynamic primitive that corresponds to a discrete state of the automaton; meanwhile the automaton controls the activation timing of the dynamical systems. Thus, the overall system can generate, analyze, and describe complex dynamic events based on the structures of temporal intervals.

In spite of the flexibility of the systems, the learning process has a difficulty due to its paradoxical nature; that is, it requires us to solve temporal segmentation and system identification problems simultaneously. We therefore propose a two-step learning method. The first step of the method estimates the number of linear dynamical systems and its parameters based on the hierarchical clustering of dynamical systems, and the second step refines overall system parameters.

Experiments on simulated and real image data show that the proposed method successfully solves segmentation and system identification problems from input time-varying signals.

Applying the proposed model to describe structured dynamic events that consists of multipart primitives, we can extract and analyze dynamic features based on the timing structures extracted from temporal intervals. We examined the effectiveness of using the timing structures to analyze and discriminate fine-grained facial expression categories such as intentional and spontaneous smiles of which existing methods had difficulty to represent the difference.

Finally, we propose a "timing structure model" that directly represents timing structures in multimedia signals, such as synchronization and mutual dependency with organized temporal differences among temporal patterns of media signals. Experiments on simultaneously captured audio and video data show that time-varying signals of one media signal can be generated from another related media signal by using the trained timing structure model.

Acknowledgement

I first wish to express my deepest gratitude to my adviser, Professor Takashi Matsuyama. He provided insightful advice about the concepts of states and time, and guided me conscientiously throughout this study. Although the contents of this thesis include the results of the most recent three years, I could never complete the thesis without thoughtful suggestions given by him since I was a bachelor's student.

I also wish to express sincere appreciation to my thesis committee, Professor Toyoaki Nishida and Professor Tatsuya Kawahara for taking time to review my thesis and giving helpful comments.

Faculty members in Matsuyama laboratory, Professor Atsuto Maki, Professor Kazuhiko Sumi, and Professor Hitoshi Habe, have always encouraged and supported me to complete the thesis. I also like to thank Professor Akihiro Sugimoto at National Institute of Informatics, Professor Toshikazu Wada at Wakayama University, Professor Shinsaku Hiura at Osaka University, and Professor Shogo Tokai at Fukui University. They supported my master's and bachelor's research when they were in Kyoto University.

I deeply indebted to all the members of Matsuyama laboratory, especially my colleagues Mr. Takeshi Takai, Mr. Shohei Nobuhara, Mr. Takatsugu Hirayama, and Mr. Hiromasa Yoshimoto. I would like to thank secretaries in Matsuyama Laboratory, Ms. Yoshie Takahara, Ms. Yoshiko Nobuhara, Ms. Yui Sato, Ms. Satomi Taku, Ms. Tomoko Hayama, Ms. Yuko Okayama, Ms. Yuki Onishi, and Ms. Hiromi Monobe, for their assistance.

Finally, I wish to express my thanks to my family who has always supported me. My special thanks go to my wife Mina, my son Yuki, and my baby who will join us this spring.

Contents

Chapter 1

Introduction

1.1 Modeling Dynamic Events

Understanding dynamic situations and performing appropriate behaviors in the situations is indispensable mechanisms for biological systems to survive in the real world. The mechanisms are also crucial for artificial information systems to realize intellectual functions, such as recognizing dynamic scenes, predicting events in the scene, controlling themselves to be desirable states, and providing useful information to users.

To understand dynamic situations in the real world, systems measure multiple time-varying signals from multiple sensors. The sensors can be a set of cameras, microphones, and tactile sensors. From the acquired multimodal signals, the system first recognizes "where" and "what objects" exist in the scene (object recognition), and then recognizes "when" and "what kind of / how" dynamic events have occurred or are occurring (event recognition).

In general case, these two processes of object and event recognition can be done in parallel. Biological motion in human perception analyzed by moving light displays [Joh73], or its interpretation systems [Ras80, CS94] are typical examples where motion itself plays an important role to determine objects (e.g., arms and legs). However, to concentrate on temporal aspect of event recognition, we here assume that object recognition is done beforehand.

Similar to event recognition, systems determine "when" and "what kind of / how" dynamic events should be generated in response to recognized events from dynamic situations to perform appropriate behaviors. For instance, one of the most important issues in robotics is the method to determine the activation patterns of actuators in the situations in which the robot body contacts with envi-

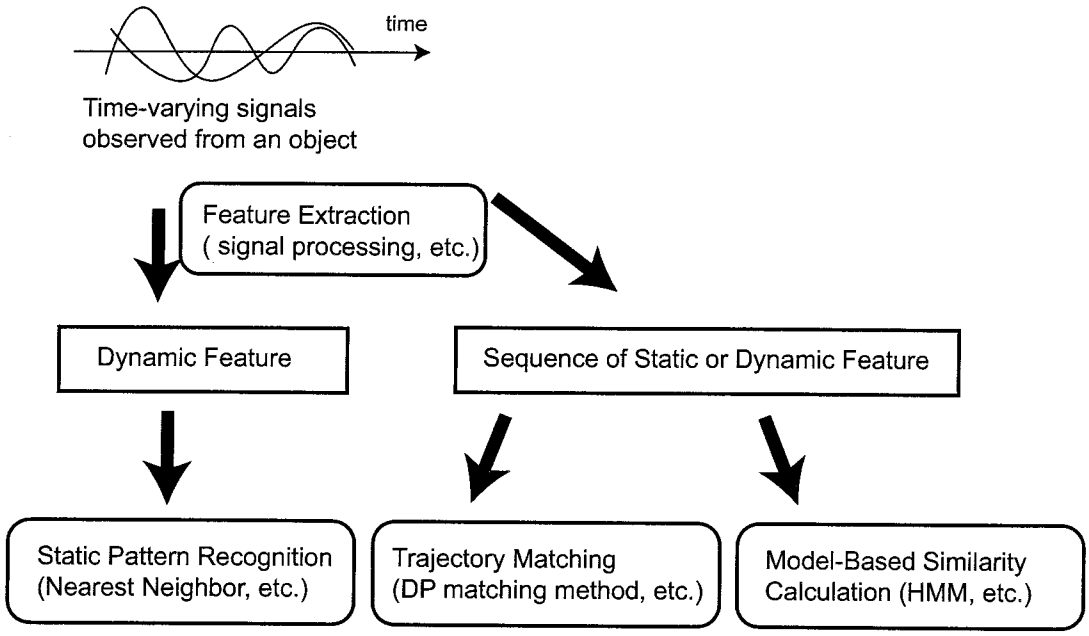


Figure 1.1: Flow of Event Recognition.

ronment and interacts with human.

As for the concrete methods of event recognition and generation, there exist many studies in wide variety of areas that cope with real-world problems; for example, computer vision, speech, and robotics [AC99, Gav99, Nak00]. Especially, the methods of event recognition, which we mainly handle in this thesis, can be categorized into several cases based on the flow of information processing shown in Figure 1.1:

- The use of static pattern recognition methods with dynamic features.
- The use of trajectory-based temporal pattern matching.
- The use of state-space models for similarity calculation.

In the following, we briefly describe each of the cases above. As we will introduce there, the model proposed in this thesis can be categorized as one of the state models (or state-transition models). Since we will discuss state models in details as it becomes relevant (in Section 1.2), in this section, we give only a concise comparison among the event recognition techniques.

Static Patter Recognition Methods with Dynamic Features

The event recognition methods that utilize static patterns of dynamic features have been introduced since the dawn of gesture and activity recognition studies. Once a set of time-varying signals is given as an observation of the object (e.g., a set of pixels from each of captured images), these methods extract dynamic features based on signal processing; for example, spectrum of signals, optical flow fields of visual features, and temporal integral of signals (e.g., motion-energy or motion-history images) can be dynamic features. The methods then exploit extracted dynamic features with classical static pattern recognition techniques, such as nearest neighbor methods and template matching, and classify the category of the input signals [PN94, NA94, EP97, BD97, ZMI01].

Since these approaches utilize dynamic features that become interfaces between time-varying signals and static pattern recognition techniques, the user can select variety of pattern recognition methods once the dynamic features are extracted. However, the approaches are sometimes sensitive to spatio-temporal fluctuation of input signals because most of the methods are difficult to represent the variance of signals (e.g., time warping). In addition, these methods often require problem-specific signals, such as periodic signals in gait motion, and restricted to use in a narrow domain.

Trajectory-Based Temporal Pattern Matching

Straightforward methods to recognize events from the input time-varying signals or temporal sequences of features, which we refer to as observations or observed sequences, are the use of dynamic programming (DP) matching methods. In these methods, a prototypical (or reference) temporal pattern (trajectory) is selected for each of the event categories in advance of recognition. In the recognition phase, similarity between an observed sequence and each of the prototypical patterns in every category is calculated based on DP matching. The DP matching methods have been used in speech recognition and gesture recognition systems [DP93, TSKO94, NMN97, KP98].

Although these methods enable us to use sequences of static features on behalf of dynamic features, and to search simple temporal patterns with inexpensive computational cost, we often fail to model the inner-class variation of patterns. This is because the prototypical pattern does not have expressive power to represent distribution of patterns. The expression of inner-class variation is often

essential to realize speaker independent speech or gesture recognition systems. As a result, state-space models described in successive paragraphs are widely used in the current recognition systems that require expressing inner-class variation. In fact, some of DP matching methods can be deduced as a special case of the state-space models.

Model-Based Dynamic Event Recognition and Generation

State models represent temporal structure of events based on the change of the states apart from observation space (i.e., feature space or original signal space). Hidden Markov models (HMMs) and differential equation systems are well-known models widely used in speech and visual motion recognition fields. Thanks to the states, the models can represent variation of patterns in the observation space by defining mapping functions between states and observations. In the recognition phase, one of the states is activated by the combination of observed data and the previous state. Since state-space models maintain memory as the activation of the states, they can successfully track an observation sequence, and classify the sequence into one of the event categories. We can also use the model to generate time-varying signals if the model is a generative model.

The model we propose in this thesis is categorized as one of the state models. The state and its transition are however designed by the integration of two different system concepts: *dynamical systems*, which is suitable for describing physical phenomena (consider time as physical metric entity), and *discrete-event systems*, which is suitable for describing human subjective or intellectual activities (consider time as ordinal state transition).

In the remaining of this chapter, we first describe two different "concepts of time" in Section 1.2, and show the two kinds of systems, each of which represents "events" based on different concept of time. In Section 1.3, we describe an intuitive concept of integrating the two systems, and introduce some existing studies about the methods of the integration. The main idea of our proposed system is described in Section 1.4, where we introduce the use of "temporal intervals" to integrate the two different concepts of time, and show the target situation of the proposed system where we focus in this thesis. Finally, we show the overview of this thesis in Section 1.5.

1.2 Subjective Time and Physical Time

1.2.1 Definition of Events and Time

Before describing two different system concepts in state models, we first define the terms and notions of event and time, which are significant to differentiate between the two system concepts.

Discrete Events and Dynamic Events

The term *event*, which we used in the previous section, has two different notions, each of which has been used in different ways. One is considered to occur at an instantaneous point in time (e.g., "switch the coffee maker on") [TKT+00, KCB95], and the other is considered to occur in a temporal region and have a duration length (e.g., "travel from A to B") [All83]. To be precise, the former event does not necessarily occur at a single instant in time, however, the duration of the occurrence is negligible or nonessential. On the other hand, the continually-changing patterns of objects are essential in the latter event.

In this thesis, we therefore use the following terms to distinguish the notions of event:

Discrete events. Discrete events are the occurrence of something that are independent of temporal metric; the discrete events take discrete values (i.e., elements in a finite set). In this thesis, we assume that the discrete events occur instantaneously in time similar to delta functions.

Dynamic events. Dynamic events are the occurrence of something that is described by time-varying signals and has physical energy in the real world; thus, dynamic events have duration lengths. The sensors can convert dynamic events as trajectories of observable signals in the space that have spatio-temporal metric based on the exchange of the energy (e.g., optical-electrical conversion).

Comparing the above definitions with the existing studies in artificial intelligence, discrete events correspond to the actions used in *situation calculus* [MMM69], which represents a temporal (causal) relation between an action of an agent at one situation and the result of the action at the next situation in time. Whereas the situation calculus works well when there exists a single agent

performing instantaneous or discrete actions, it fails to represent duration of actions or temporal structures among actions of multiple agents, which can be overlapped with each other.

Alternative formalisms to represent these temporal structures are known as *event calculus* [KS86], Allen's *interval-based temporal logic* [All83, All84], and so on. These approaches assume that events have duration lengths in time; therefore, those events used in the approaches correspond to dynamic events defined here.

Subjective Time and Physical Time

In ancient Greek, there were two different concepts of time described by two words: *kairos* and *chronos*, which are the names of Greek gods originally. Kairos is the moment or occasion of making meaning; kairotic time is measured by discrete events. For example, a marriage and childbirth is a kairotic time (moment). Chronos, on the other hand, is the time that flows linearly; chronological time is a concept of time that describes the continual change of dynamic events, which are measurable by clocks (e.g., Newton's laws of motion).

Generalizing the notions of kairos and chronos, we define the following two kinds of time.

Subjective Time (Kairos): temporal order among recognized discrete events. Let $\mathbf{T}_s$ be a countable set that has no mathematical structures, such as distances and relations, among the elements. The set $(\mathbf{T}_s, \leq)$ becomes a subjective time axis, where $\leq$ denotes an ordered relation between elements in the underlying set $\mathbf{T}_s$. We often use a set of linearly ordered natural numbers $\mathbf{N}$ for $\mathbf{T}_s$.

Physical or Objective Time (Chronos): metric entity that linearly progresses. Let $\mathbf{T}_p$ be a set that has no mathematical structures. The set $(\mathbf{T}_p, \leq, d)$ becomes a physical time axis, where d denotes a distance function between two elements in the underlying set $\mathbf{T}_p$ (e.g., $d(t_1, t_2) = |t_1 - t_2|$, where $t_1, t_2 \in \mathbf{T}_p$). We often use set $\mathbf{R}^+ = \{t | t \geq 0, t \in \mathbf{R}\}$ for $\mathbf{T}_p$, where $\mathbf{R}$ is a set of real numbers.

Note that the most significant difference of physical time from subjective time lies in the existence of metric property.

1.2.2 Discrete-Event Systems and Dynamical Systems

Based on the two concepts of time described in the previous subsection, we can categorize existing state models, which are used for event recognition and generation, into following systems:

- Discrete-event systems
- Dynamical systems

As we will describe in this subsection, each of discrete-event systems and dynamical systems defines its state transition based on subjective time and physical time, respectively.

The integrated system of these two systems is referred to as *hybrid dynamical systems*, as we will describe in Section 1.3.

Discrete-Event Systems

A discrete-event system has a set of *discrete states*, which is represented by a finite set, and it does not change the discrete state before an input discrete event occurred. Therefore, the state transition is described based on the subjective time. The simple case of the state transition becomes the following function, which is used in finite state automata:

$$M(\text{state}_{\text{now}}, \text{event}_{\text{input}}) = \text{state}_{\text{next}} \qquad (M : Q \times A \to Q), \tag{1.1}$$

where A and Q is a set of discrete events and discrete states, respectively.

The representative model of discrete-event systems is the Turing machine, which initially models "a man in the process of computing a real number" based on the finite number of discrete states (configurations) [Tur36, Tur50]. Finite state automata [KCB95, BW97, WBC97, MM98b, WM00], HMMs [Rab89, HAJ90, Nak00, YOI92, SP95, BOP97], and Petri nets [DAJ95] are examples of discrete-event systems that are widely used for modeling structure of discrete events.

Discrete-event systems have the advantage of being able to model long-term contexts, relations of temporal order, overlaps, and inclusion among events [PMB97], and other discrete structures such as coupling between action and perception [KCB95]. However, discrete-event systems have signal-to-symbol problems; that is, it requires us to define an event set in advance. Moreover, human-designed states depend on our recognizable event scales. Therefore, the

continual changes produced by smooth dynamics are difficult for discrete-event systems to describe.

Dynamical Systems

In contrast to discrete-event systems, dynamical systems define the state transition based on the physical time using the formulation of differential or difference equations. The simple case of the differential equation becomes:

$$\frac{\mathrm{d}\,\mathrm{state}(t)}{\mathrm{d}t} = F(\mathrm{state}(t)) \qquad (F : \mathbf{R}^n \to \mathbf{R}^n), \tag{1.2}$$

where $t \in \mathbf{R}^+$ and n is the dimensionality of the internal (continuous) state space, where *internal states* are defined. Note that the system changes the state even if there are no input signals.

Cybernetics, which had been advanced by Norbert Wiener since 1940s, is the study of "teleological mechanisms" involving regulatory feedback in animals (living organisms) and machines [Wie61]. Cybernetics influenced wide area of automatic systems including dynamical systems in control theory [AM79]. Gaussian and non-Gaussian linear dynamical systems [Rao97, IB98, RB99, BCMS01, DCWS03, DD05] are often used for modeling dynamic events that have physical dynamics. As for nonlinear dynamics, recurrent neural networks [FH88, Rob94, Dor96, Mor96, UT00, HWK01] and other nonlinear dynamical systems [dFNG98, GR99, OTN02] are used for modeling complex events such as robot motion.

Dynamical systems have the advantage of modeling continually-changing dynamic events such as human motion and utterance. However, the systems are not suitable to represent complex structures of signals, such as duration lengths of dynamic events, patterns of the duration lengths, and other temporal relations exist in multiple dynamic events that occur concurrently.

1.3 Hybrid Dynamical Systems

Hybrid dynamical systems (hybrid systems) that integrate dynamical systems and discrete-event systems are introduced to overcome the disadvantages of the two systems in a complementary manner.

In the following subsections, we first give a basic idea of hybrid dynamical systems to see how the disadvantages can be solved by the interaction between

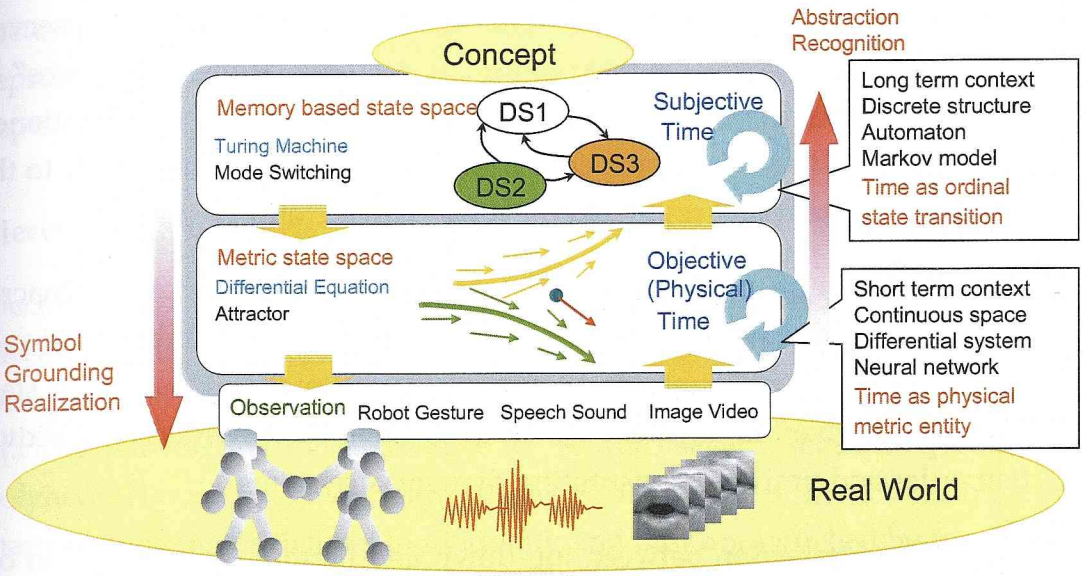


Figure 1.2: The concept of hybrid dynamical systems.

discrete-event systems and dynamical systems that constitute the overall system (Subsection 1.3.1). We then introduce some existing approaches that integrate discrete-event systems and dynamical systems, and discuss the idea of the integration in most of the approaches are different from integrating the two different concepts of time (Subsection 1.3.2).

1.3.1 Interaction in a Hybrid Dynamical System

Figure 1.2 shows the concept of hybrid dynamical systems. In a hybrid system, a discrete-event system decides the activation timing of multiple dynamical systems. The discrete-event system provides a solution to represent a discrete structure of primitives; meanwhile, the dynamical systems represent detailed dynamics in each primitive and also provide metric properties among the primitives. A typical interaction among the discrete-event system and the dynamical systems in a hybrid dynamical system is as follows:

1. Dynamic events in the real world, such as speech utterance, lip motion, and human (robot) motion, are measured as multimedia signals.

2. At the boundary of the hybrid dynamical systems and the real world, dynamic primitives are represented by various attractors in the internal state space based on differential equations. Each dynamical system, therefore,

changes the internal state in the physical-time domain based on observed time-varying signals. This process can be regarded as the "resonance" of dynamical systems with observed signals, and the signals are partitioned into temporal intervals, where each partitioned interval corresponds to the dynamical system that resonated the most with the observed signal.

3. The segmentation result of the observed signal determines the macro-transition of the discrete states, which can be considered as the flow of subjective time in the discrete-event system. If the probability of each state transition is given in advance, the discrete-event system affects the activation order and timing of constituting dynamical systems.

4. Hence, the interaction between top-down and bottom-up information occurs simultaneously at the two boundaries; that is, the boundary between the real world and the constituent dynamical systems (see 2), and the boundary between the dynamical systems and the discrete-event system (see 3).

Due to the interaction above, each of disadvantages in discrete-event systems and dynamical systems can be solved as follows:

- The dynamical systems provide interfaces between signals in the real world and discrete states (symbolic entities) in the discrete-event system; this architecture resolves the signal-to-symbol problem of discrete-event systems.

- The discrete-event system represents the structures of discrete events that are produced by the constituent dynamical systems; thus, complex temporal relation among dynamics are described in this architecture.

1.3.2 Existing Approaches

Because of the high capability of modeling nonlinear and complicated events, hybrid dynamical systems are currently attracting great attention in various fields including controls, robotics, computer vision, graphics, neural networks, and other computer science fields. In the following paragraphs, we introduce some of the existing hybrid dynamical systems in these fields. Mathematical analysis of hybrid dynamical systems can be found in [GV89, MMdB+91, ACH+95, ZM95].

As we will see in the remaining of this subsection, the notion of "hybrid" differs among the studies. Note that most of the existing hybrid dynamical system

focus on integrating discrete states, represented by symbols, and internal states, represented by continuous values, rather than integrating the two different concepts of time (i.e., subjective and physical time) described in Subsection 1.2.1.

Piecewise ARX Models

Piecewise AutoRegressive eXogenous (ARX) models are the ARX models that use piecewise linear (PWL) or piecewise affine (PWA) maps as the regression function [FMLM03, RBL04, KHS+04]. A PWL map constructs a nonlinear function $f(x)$ by partitioning the domain $\mathcal{X} \subset \mathbf{R}^n$ into several regions $\mathcal{X}_1, ..., \mathcal{X}_N$ with polyhedral boundaries, which are referred to as *guardlines*. In each region, a linear mapping function is defined individually, and they are switched by the condition of $x \in \mathcal{X}$. As a result, the mapping function becomes nonlinear as a whole.

Piecewise ARX models are a class of hybrid systems for which the switching law between the affine submodels is specified by the shape of the guardlines; thus, the model represents nonlinear signals due to the switching. The conditions of discrete-state transition in the model, however, can be regarded as static because they are determined beforehand based on the design of guardlines.

Switching Dynamical Systems

Bregler et al. [Bre97] proposed a multilevel modeling method of human gate motion based on an architecture of a hybrid dynamical system. The model comprises multiple linear dynamical systems as its subsystems, and an HMM that switches the constituent subsystems. As a similar approach to the Bregler's model, a switching linear dynamical system (SLDS), which switches linear dynamical systems based on the state transition of the HMM, have become a common approach for modeling complex dynamics such as human motion [GH96, PRCM99, PRM00] (see [Mur98] for the survey of similar models). The stochastic linear hybrid systems [LWS02, BHJT04] are also the extension of Breglar's model.

In these approaches, the discrete and internal states are integrated. However, the macro-transition between constituent subsystems (i.e., linear dynamical systems) is modeled in the same time axis as the internal-state transition of each subsystem (i.e., physical time axis). Assuming that the system consists of a set of subsystems $\mathcal{Q} = \{q_1, ..., q_N\}$, then the SLDS models the transition from subsystem q_i to q_j as a conditional probability $P(s_t = q_j | s_{t-1} = q_i)$, where t is synchronized

to the internal-state transition in each subsystem. Some other method use particle filters on behalf of linear dynamical systems (Kalman filters) [BIR00], however the method also use the physical time to model the state transition in the HMM.

Segment Models

Segment models [ODK96] have been proposed in speech recognition fields as the unified model of segmental HMMs [Lev86, HAJ90] and other segment-based models [Mur02]. In contrast to SLDSs, segment models use *segments* as descriptors. Each of the segments represents a temporal region in which one of the discrete states is activated. Since a discrete state corresponds to a dynamic event such as phonemes and subwords, each of which is represented by a subsystem, the discrete-state transition of the segment model represents temporal order of dynamic events apart from the physical-time domain. Thus, the conditional probability of the state transition becomes $P(s_k = q_j | s_{k-1} = q_i)$, where k represents the temporal order of the subsystem activation. Motion texture [LWS02], which is proposed for motion generation purpose, can be also categorized as one of the segment models.

Since the transition between the subsystems is modeled independently from the physical-time domain, the model handles one aspect of integrating the concepts of physical time and subjective time. However, because this model is proposed as a unified framework of segmental HMMs, it focuses on modeling only state duration rather than complex temporal structures among discrete events. We will discuss the details of this point in the next section.

1.4 Interval-Based Hybrid Dynamical System

In this thesis, we propose a novel hybrid dynamical system that integrates the concepts of subjective and physical time by exploiting *temporal intervals* (intervals, in short) defined in this section. We refer to the system as an *interval-based hybrid dynamical system* (interval system, in short). Interval systems are similar to segment models in respect that the both models are able to describe the temporal order of dynamic events, each of which is represented by a subsystem, apart from the physical-time domain. However, the concept of interval systems are different from the segment models because we concentrate on modeling temporal structure among multiple discrete events extracted by constituent subsystems (i.e., tempo-

rally dividing points of complex dynamic events) rather than only modeling the duration lengths of dynamic events (i.e., temporally divided parts of complex dynamic events).

For the above reason, we use the term "intervals" instead of "segments". In other words, our motivation is bringing Allen's interval-based temporal logic [All83, All84], which exploits 13 topological relations between two intervals (e.g., meets, during, starts with, etc.), into the class of hybrid systems. Once the intervals are explicitly defined, we can fabricate flexible models to represent complex structures among multiple types of dynamics, which can be appeared concurrently in human behavior and interaction (e.g., tempo and rhythms of utterance, synchronization/delay mechanism of speech and lip motion, and action timing generation in response to input events in interactive systems).

In the following subsections, we first define the notion of an "interval" (Subsection 1.4.1), and show how the temporal structures that have vital information for human can be described by the intervals (Subsection 1.4.2). We then give a concept of an interval-based hybrid dynamical systems (Subsection 1.4.3), and finally we discuss the expressive power and the limitations (Subsection 1.4.4).

1.4.1 Definition of Intervals

The definition of "dynamic events" in Subsection 1.2.1 is independent of cognitive processes, however, significant dynamic events are perceptible units for some "cognitive subjects". We therefore define an "interval" as a temporal difference between the beginning and ending points of the dynamic event that is perceived by some cognitive processes of humans or artificial systems. The length of the interval corresponds to the duration of the perceived dynamic event in the physical-time domain (see Figure 1.3).

Regarding human cognition, those perceptible units are not restricted to the dynamic events recognized consciously. While humans are not able to be aware of some events, the unconsciously perceived events are often processed without awareness, and exploited to provide appropriate decision and action. For instance, as we learn techniques in football, the learner can be aware of primitive motions (dynamic events) constituting an overall kicking action, such as pulling the leg back and moving it forward. However, once the learner has acquired the skill of the action, he or she can provide the action without awareness of each primitive motions.

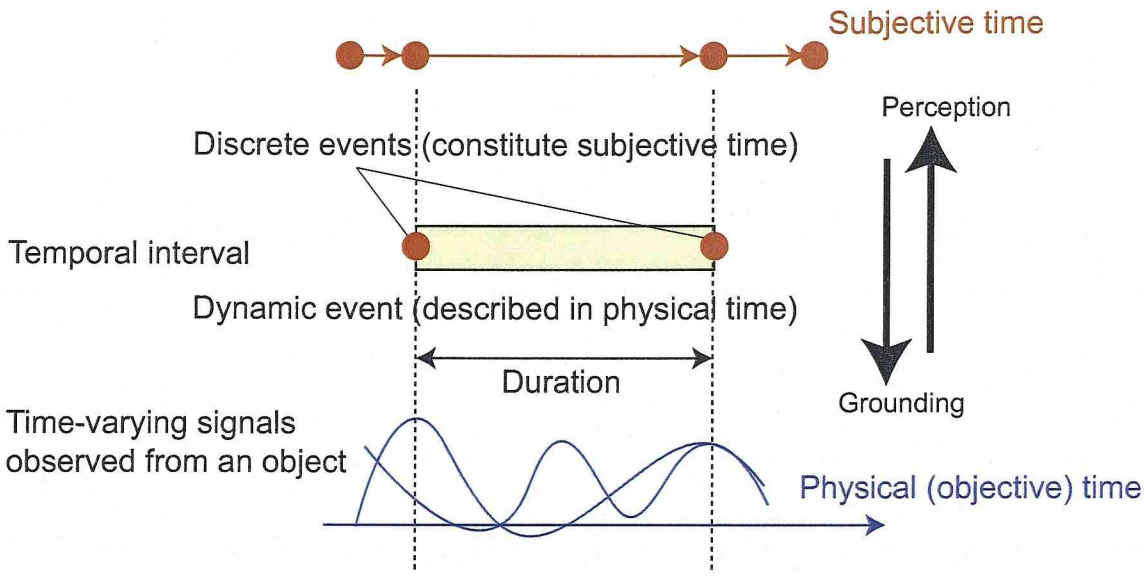


Figure 1.3: Definition of a temporal interval.

Whether the humans are aware of the units or not, the beginning and ending points of perceptible dynamic events are essential to understand the temporal property of situations. We consider these instantaneous points in time as discrete events as shown in Figure 1.3; thus, a set of the discrete events constitutes subjective time, and temporal ordering relations among the discrete events become important for the situations incorporated by discrete-event systems as we described in Subsection 1.2.1. In the next subsection, we see how the significant temporal structures are described by the intervals.

1.4.2 Dynamic Structures Exploited by Humans

Temporal relations among discrete events (i.e., beginning and ending time points of perceptible dynamic events) have significant information for humans and artificial information systems to describe the situations of environments, to understand the meaning of object behaviors, and to generate actions in appropriate occasion.

Allen proposed an interval-based temporal logic to describe temporal relation among multiple actions that occur simultaneously and have many interact with each other [All83, All84]. The logic represents the relationships between temporal intervals based on temporal ordering relations of discrete events (beginning

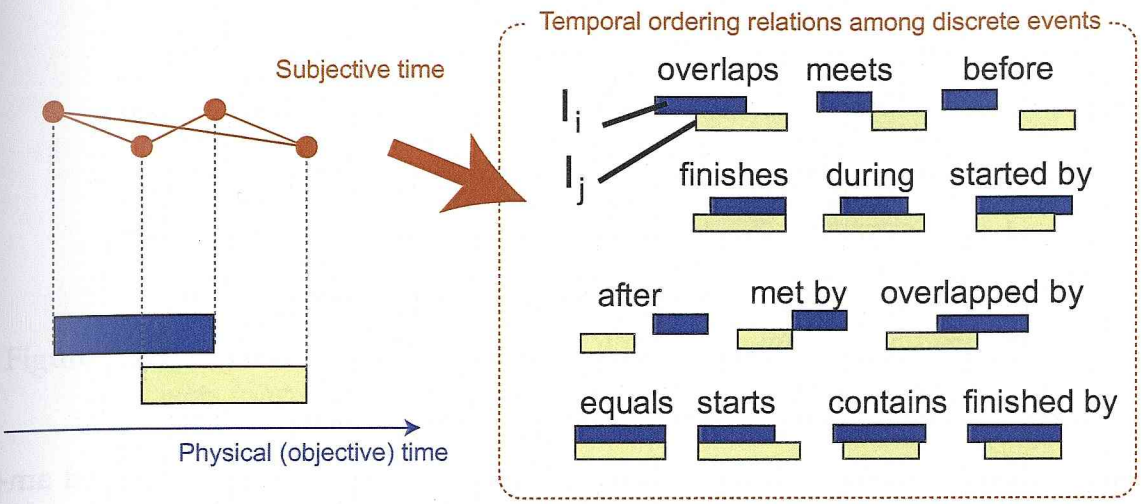


Figure 1.4: 13 temporal relations between two intervals I_j and I_j used in Allen's interval-based temporal logic, which exploits temporal ordering relations among discrete events (beginning and ending points of intervals) in the subjective time.

and ending time points) obtained from two intervals (Figure 1.4). As a result, it successfully represents temporal relations between multiple intervals in a hierarchical manner using constraint propagation techniques.

While temporal ordering relations among discrete events are indispensable to realize intelligent functions, we humans exploit not only temporal orders but also metric properties such as temporal differences among discrete events. In particular, these metric properties have crucial information for understanding temporal features appeared in the real world such as in verbal and nonverbal human communication, and for performing appropriate behaviors in complex environment.

In subsequent paragraphs, we see some examples of metric properties that we humans exploit.

Duration lengths of dynamic events. As we described in the previous subsection, each interval has a duration length as its metric property. Some psychological experiments suggest that duration lengths of facial actions play important roles for human judgments of basic facial-expression categories [KBM+01, KK05]. In addition, the duration lengths of stationary gaze often used to estimate his or her interest to the objects [WYH05].

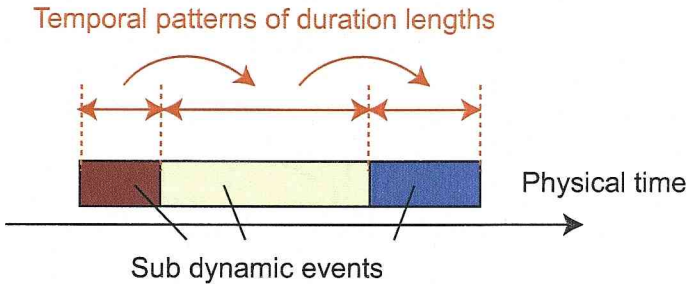


Figure 1.5: Rhythm of dynamic events observed in a single signal.

Rhythms of dynamic events. Because the term "rhythm" is often used ambiguously, we need a definition of the term. Once an action is partitioned into subactions, we obtain a sequence of sub dynamic events. For a simple definition, we here refer to patterns of duration lengths appeared in the sequence of subactions as rhythms (Figure 1.5). We humans are sensitive to the rhythms of not only music performances but also general events in various situations; for example, human gait motion (easy to detect an injury of others), swinging arm motion in communication (phasic gestures [WBC97]), and sports (feint motion in ball games to foil the others rhythmic prediction).

Synchronization among dynamic events. Intervals can be obtained from not only a single media signal but multiple media signals captured from multipart motion, multiple sensor modalities, and other situations; the intervals of concurrent dynamic events therefore can be overlapped each other. In these case, temporal differences between beginning points or between ending points among dynamic events often become significant in some situations (e.g,. synchronization/delay mechanisms among dynamic events) (Figure 1.6). For example, it is well-known fact that the simultaneity between auditory and visual patterns influences human perception (e.g., the McGurk effect [MM76]). Synchronized motion or sound generation among performers are also indispensable for music and dance performances [Mat96].

Action timing generation in response to perceived events. Timing generation mechanisms of actions exploits the metric properties as well. One can control the beginning timing of utterances based on the other's speech signals (Figure 1.7). This pause and overlap lengths often convey rich information of one's intention or affective states [OKYI96, NDKN02]. Timing generation is also essen-

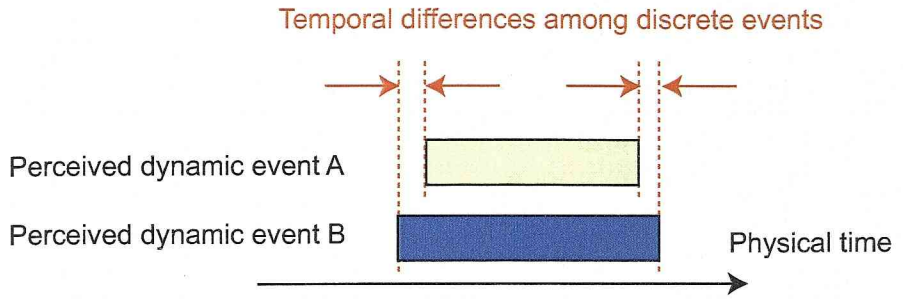


Figure 1.6: Temporal relation between dynamic events appeared in multiple objects (parts) or different media signals (e.g., synchronization).

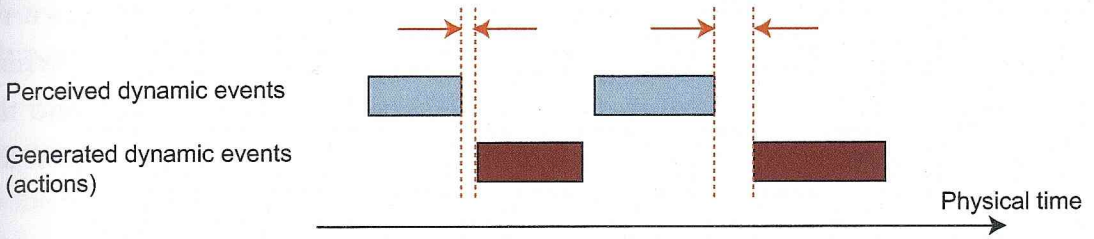


Figure 1.7: Generation of action timing in response to perceived events.

tial to perform articulated motions; humans and animals optimize control timing (e.g., insertion of torque power) of each different parts to realize effective body action [KOT+04, YKM06]. The timing between body parts are also essential; for example, multi-part motion appeared in human body often described by timing of motions as we see in dance notation (e.g., Labanotation [Nak01]).

The objective of this thesis is to provide a computational model that represents dynamic structures described in the preceding paragraphs:

- Duration lengths and their patterns of dynamic events in a single signal
- Temporal metric relation of multiple dynamic events observed in multiple parts, objects, and different media signals, which can be overlapped each other
- Temporal metric relations of perceived and generating events

In this thesis, we use the term *timing structures* for these metric relations described by the distances among discrete events. In the next subsection, we exploit relations of temporal intervals and introduce a concrete system for modeling the timing structures.

1.4.3 Interval-Based Hybrid Dynamical System

An interval-based hybrid dynamical system (interval system) is the integration of a discrete-event system and multiple dynamical systems similar to existing hybrid dynamical systems described in Section 1.3. However, the rationale of the integration in an interval system is different from the existing studies. That is, we use a set of dynamical systems as cognitive processes, which we described in Subsection 1.4.1, for determining "temporal intervals".

Due to the intervals, the discrete-event system is able to describe a complex structure of dynamic events based on the relations of discrete events that are obtainable from the intervals as their beginning and ending time points. As a result, the intervals work as interfaces between the discrete-event systems, which represent the structure of discrete events in the subjective-time domain, and the dynamical systems, which represent continually-changing dynamic events in the physical-time domain.

Let us consider a human kicking motion as a dynamic event for example. If we observe the motion, we recognize that the leg moves based on several types of dynamics, such as two types of dynamics in bending backward and kicking forward if we assume bi-phasic motion, and we see that each of dynamics appears as temporal intervals in the physical time. Because the motions of other parts (e.g., arms) are essential for the kicking motion to take balance of the overall body, we also observe that several arm motions are closely related to the leg motions. Thus, primitive motions appear in multiple parts during a single motion execution can be represented by a structured multiple dynamics. This process can be generalized as the following conception; the observable signals (Figure 1.8 bottom) are produced by the *orchestration of dynamics* (Figure 1.8 top) that determines the activation timing of dynamics in the internal state spaces (Figure 1.8 middle).

What we want to do here is describing these structures that have significant temporal relations among multiple intervals each of which represented by a dynamical system. In addition, we demand the model to be learned from training data observed as input multivariate signals or extracted feature sequences from the signals. These objectives require two essential issues to be considered:

1. How to determine the concrete model of the dynamical system that represents each of dynamic events (primitives); the type of dynamic events should be considered because the model of dynamics affects the discrete events and their structures represented by the overall system.

2. How to model the temporal relations among discrete events (i.e., beginning and ending points of intervals) with their metric properties; the complexity of the model is too high to be trained if we take all the relations among discrete events into account, some simplification is therefore required.

1. Types of the Dynamics for Modeling Dynamic Events

In this thesis, we focus on modeling human behaviors observed in communication. Therefore, it is plausible to use linear dynamical systems as a type of dynamics for modeling dynamic events, such as visual motion, because most of dynamics produced by humans is the effect of muscular action.

Another option for modeling dynamics is the use of nonlinear dynamical systems; for example, recurrent neural networks [MM98a], polynomial systems [OTN02], and other systems that use nonlinear mapping functions in their state transition or observation (e.g., extended Kalman filtering [SHST00]). Nonlinear dynamical systems might be important for modeling such as consonant sounds in speech; we however assume that most types of dynamics are represented by linear dynamical systems because of the following reasons.

- Nonlinearity in the signal can be reduced to some degree (1) if we assume enough order for the Markov process, and (2) if we select appropriate static or dynamic features in the feature extraction phase.

- Nonlinear signals or nonlinear feature sequences can be represented by piecewise-linear systems if we choose appropriate units of primitives.

In particular, we exploit the second points; that is, we assume a complex dynamic event comprising a set of sub dynamic events. Those sub dynamic events are often referred to as motion primitives [NNYI04], movemes [Bre97], visemes [NLP+02], motion textons [LWS02], modes [NKHM05], and so on.

Then, we assume the observed signals or feature sequences that are describing each of the temporal regions of sub dynamic events is represented by a linear dynamical system. For example, a cyclic lip motion can be described by a set of simple lip motions such as "open", "close", and "remain closed" (Figure 1.9). Once the set of sub dynamic events is determined, a complex action can be partitioned by temporal intervals that have labels of the linear dynamical systems and their duration lengths.

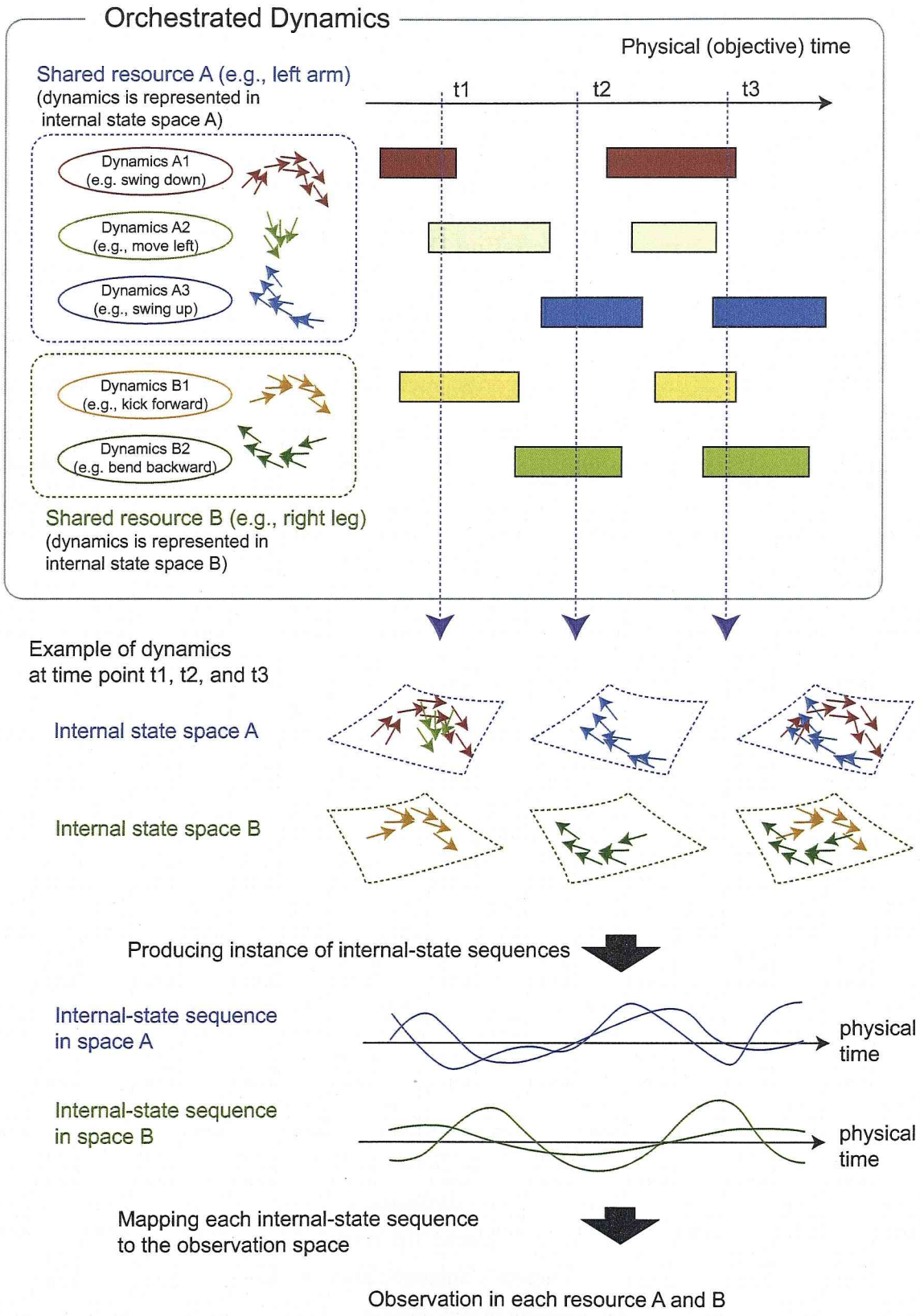


Figure 1.8: Orchestration of dynamics.

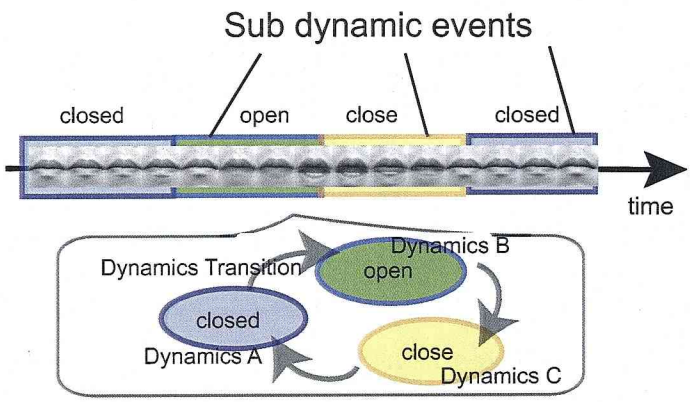


Figure 1.9: An example of dynamic events that consist of sub dynamic events.

2. Simplification for Modeling Dynamic Structures

As we see in Figure 1.8, the orchestrated dynamics have complex structures among multiple intervals. Whereas our final goal is to describe this kind of general structures, the complexity of the model describing the structure is not negligible because we require the model to be learned from training data; the stability and computational cost of the learning depend on the complexity of the model.

To reduce the complexity as simple as the model is trainable from real data, we set some assumptions for constraints of the model to describe a subset of general structures:

- A set of features or parameters that represent configurations of a single resource (e.g., a body part, an object, and a type of media) form a single multivariate signal.
- A signals of a single resource can be partitioned into intervals by multiple linear dynamical systems that share a single internal state space; each of the intervals is represented by a linear dynamical system.
- The intervals represented by linear dynamical systems of a single resource have no gaps or overlaps each other in the physical time; thus, the dynamics in a single resource switches from one to another, and the beginning points of one interval corresponds to the ending points of the next interval.
- The metric relation between intervals in different resources can be described by the temporal differences of beginning and ending points of the intervals (another assumption is introduced in Chapter 5 to specify the interval pairs).

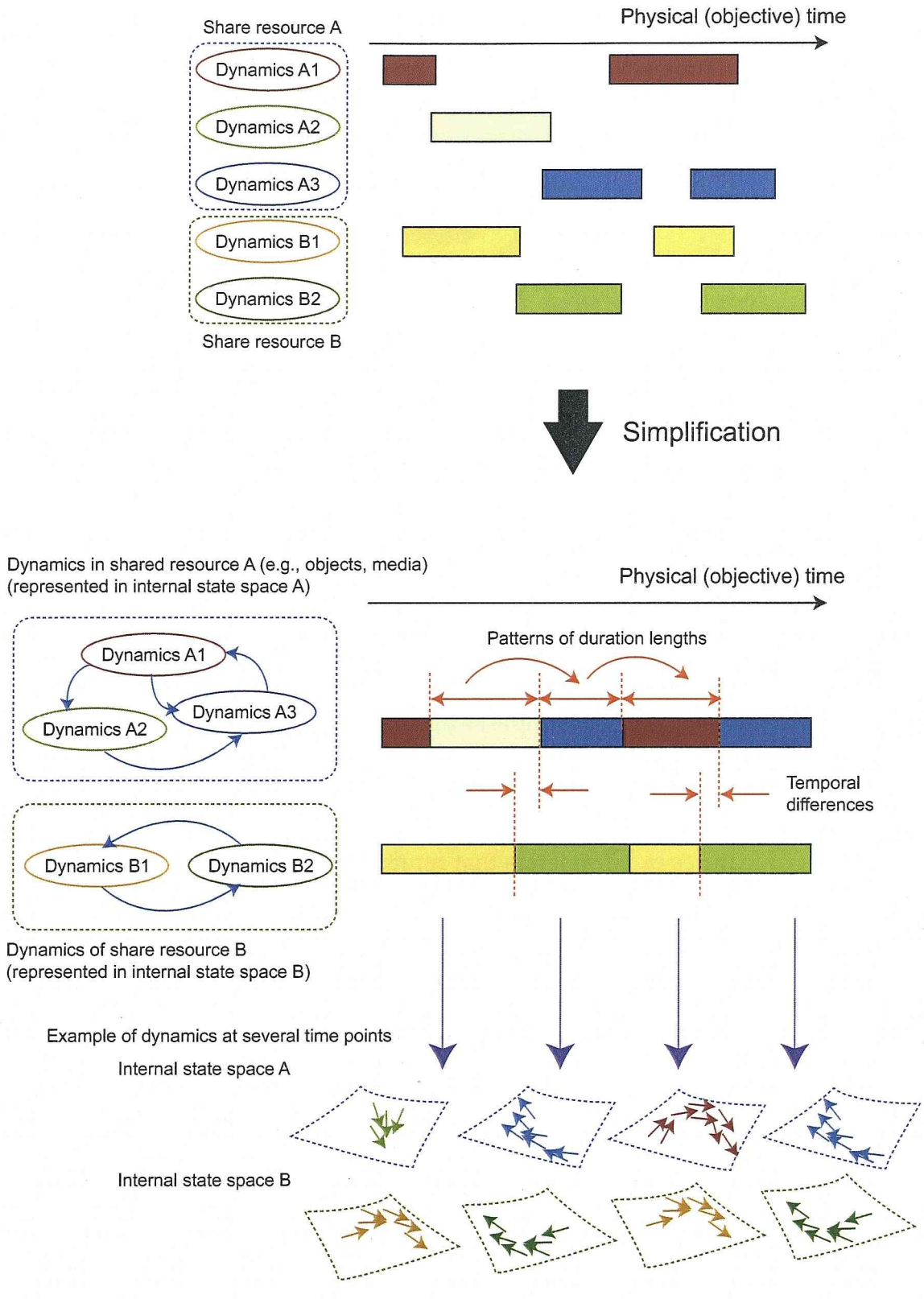


Figure 1.10: Integrataion of two interval-based hybrid dynamical systems.

Based on the third assumption, the discrete events of one resource are linearly (totally) ordered. Note that each discrete event has a type of dynamics that is finished by the event. Therefore, we use simple automaton to model the order of the types of dynamics finished by each of discrete events in a single resource. Especially, we consider one-to-one correspondence between discrete states of the automaton and the type of linear dynamics (see Chapter 2 for details). In this thesis, we use the term "an interval-based hybrid dynamical system" (or an interval system) to refer the system that represents a single resource.

Consequently, a general structure of orchestrated dynamics, such as shown in the top of Figure 1.10, is simplified to the architecture comprises two interval systems as shown in the bottom of the figure. We see that only one type of dynamics can be activated at a single time point in one resource, and each automaton models the activation order of the multiple dynamics of the resource. The transition of dynamics provides complex dynamics as a whole, and determines the behavior of a produced instance of an internal-state sequence.

We view the interval system proposed in this thesis as an initial step toward understanding human-human interaction and realizing human-machine interaction systems. Therefore, as evaluating the interval system, we concentrate on verifying how the interval systems are suitable for modeling dynamic events produced by humans, such as facial motion, body motion, utterances, and behaviors, based on the assumptions above.

1.4.4 Expressive Power and Limitations

Expressive Power

Modeling duration length of a dynamic events. It is well known fact that the HMMs represent only the exponential distributions of state duration if we use the model with observations that occur in fix-length intervals [ODK96]. This situation is quite common in speech and gesture recognition systems that use sampled signals as input data. Let us consider a HMM that has more than two discrete states, and assume that the observations occur based on a fixed-length sampling rate. Let a_{ii} be a transition probability that the HMM preserves state q_i at an occurrence of an observation. Then the probability that the HMM sustains state q_i during time length t — and changes the state after the duration — becomes $a_{ii}^t(1 - a_{ii})$. Thus, the HMMs are restricted to represent the exponential distributions of discrete-state duration. On the other hand, the interval system is able

to utilize general functions for the interval length distribution similar to segment models. Moreover, the interval system explicitly models the relation among interval lengths of multiple dynamic events, and provides the expressive power that we describe in the next paragraph.

Modeling patterns of duration lengths of dynamic events in a single signal. As for the expressive power of the interval-based hybrid dynamical system for modeling patterns of duration lengths, we compare the system with several existing models in the language theory. Let us consider the situation of modeling a dynamic event E that comprises two types of dynamics a and b, each of which represents a sub dynamic event constitutes E. Let us assume that each of dynamics a and b appears only once in this order, and that each dynamics continues the same length. To describe this situation, we can use a sequence $\{a^t b^t \mid t \in \mathbf{N}\}$ to denote the change of dynamics in the physical time, where we assume t represents the length of the dynamics described in a discrete time, which sampled by a fixed-length rate. Whereas an automaton can not represent these patterns because it is in the class of context-free grammar (CFG), the interval system can describe these relations; for example, we can explicitly set the adjacent interval to be the same duration lengths.

This result can be extended to the relation among three types of dynamics. Let us consider the situation that a dynamic event comprises three types of dynamics a, b, and c, and assume they appear in this order with the same length. We can use a sequence $\{a^t b^t c^t \mid t \in \mathbf{N}\}$ similar to the previous example. Although the sequence is represented by a context-sensitive grammar rather than a CFG in this case, we are able to describe this situation if we explicitly model the relation of the duration between dynamics pairs (a, b) and (b, c). Consequently, the interval system is more expressive than CFG in some aspect of modeling the patterns of duration lengths.

Modeling temporal differences among discrete events in multiple signals. Early integration [CR98] is one of common methods to model the relation between multimodal signals. This method combines two feature vectors observed from different modalities at a single frame, and forms a single vector. In other words, the early integration utilizes a frame-based integration. The method however have disadvantage of modeling metric structures of discrete events, such as lengths of temporal gaps (pauses) and overlaps in two intervals. To describe these

structures, the method can not avoid increasing the Markov order of the model. For instance, if the maximum length of the temporal gaps between discrete events is $l_{\max}$, the Markov order in the model is required to be greater or equal to $l_{\max}$ because the relation between one discrete event at time t and the other event at $t + l_{\max}$ have to be preserved in the model. The size of $l_{\max}$ is however not small in general case. As a result, the computational cost and memory size of the frame-based models easily increases. In the proposed framework in this thesis, on the other hand, models these temporal gaps explicitly based on temporal differences among discrete events. Thus, the number of the model parameters becomes small enough to train and apply in real problems.

Limitations

Chaotic dynamics. Nonlinear systems often have important temporal or geometric features, and able to describe complex behavior without modeling stochastic processes [AIYK00]. For example, some dynamics have positive Lyapunov exponents, which determine how fast the system becomes unpredictable in time. These dynamical systems, which are referred to as *chaotic systems*, can represent a wide variety of signals in spite of using only small degree of freedom.

On the other hand, some dynamical systems have non-integer fractal dimension, and they generate strange attractors that have recursive structures in their internal state space. The systems therefore generate complex signals that have layered dynamics even if we use only a single dynamical system.

Despite of the capabilities of those nonlinear dynamics described above, in this thesis, we use only linear dynamical systems because nonlinear systems are sometimes hard to identify from real data, and difficult to predict their macro behaviors. The limitation that we use only linear dynamics becomes significant if the behavior of signals is inherently chaotic or have recursive strange attractors. We however anticipate that most of signals observed in human behaviors, such as motion and utterance, can be represented by a combination of linear dynamics, as we described in the previous subsection.

Layered structures among discrete events. There exist layered structures of dynamic events in space and time; for example, a running motion can be decomposed into several body motions such as arm motions, and the arm motion sometimes comprises different types of dynamics. However, as shown in the previous subsection, we focus on modeling two-layer structure; that is, relation between

a dynamic events and its sub dynamic events. This limitation becomes significant if the dynamic events have grammatical structures (e.g., sign languages). In Chapter 6, we provide detailed discussion of extending the framework proposed in this thesis to deal with discrete events that have complex layered structures.

1.5 Overview of the Thesis

In this section, we present the organization of the subsequent chapters in this thesis. Figure 1.11 depicts the overview of this thesis.

Modeling Structures of a Single Signal (Chapter 2)

In this thesis, we first concentrate on modeling a single signal from a single source for the simplest case, and describe the relation of adjacent intervals based on the correlation of their duration lengths. Duration-length relation of the adjacent intervals corresponds directly to our cognitive sense of time such as tempo and rhythms, which are crucial information to represent features of the dynamic events.

Another advantage of explicitly modeling interval relations is that it enhances robustness against outliers during temporal partitioning process; in other words, the top-down knowledge works as a constraint to the lower-level process. For instance, if the duration distributions of the subsystems are biased toward a long length, the system will not change the subsystem before the activation of the subsystem sustains enough length. As a result, the system improves the robustness of representing temporal structures that can be partitioned into temporal intervals.

Chapter 2 describes a detailed model structure and an inference algorithm that searches the optimal interval sequence that provides the highest probability for the given observation. Then, we verify the inference algorithm using simulated data.

Learning Method of an Interval-Based Hybrid Dynamical System (Chapter 3)

In spite of the flexibility of hybrid dynamical systems, especially for modeling human behaviors such as gestures and facial expressions, few applications have exploited the system to handle real-world signals. The reason is largely due to the paradoxical nature of the learning process: temporal segmentation and system identification problems need to be solved simultaneously.

Chapter 3 proposes a two-step learning method to identify the interval system. In particular, we propose a novel clustering algorithm as the first step of the learning method; the algorithm extracts a set of dynamical systems from observed sequences, and is applicable to general hybrid dynamical systems. We evaluate the effectiveness of the proposed learning method using simulated and real data.

Analysis of Timing Structures in Multiple Signals (Chapter 4)

As we described in Subsection 1.4.2, temporal metric relations among multiple objects or multimodal signals often have significant structures to identify dynamic events. Applying the interval-based hybrid dynamical systems to describe structured dynamic events, we can analyze dynamic features based on the timing structures extracted from temporal intervals.

Chapter 4 shows how the interval system can be applied to describe and analyze temporal relation between multiple objects. We apply the system to represent complex motion appeared in each facial part independently, and examine the effectiveness of using the timing structures to analyze and discriminate fine-grained facial expression categories such as intentional and spontaneous smiles of which existing methods had difficulty to represent the difference.

Modeling Timing Structures in Multiple Signals for Timing Generation (Chapter 5)

Timing structures are also essential to provide appropriate behaviors at appropriate timing in response to the perception of dynamic events occurred in the situations. To realize the function of timing generation, we model temporal structures among different kind of media signals from multiple sensors by extending the analysis in Chapter 4.

Chapter 5 shows a general framework for modeling and utilizing mutual dependency among media signals based on the temporal relations among intervals. In this chapter, we provide a novel algorithm that generates timing of dynamic events in one media signals (e.g., lip motions in a visual signal) from another related input signal (e.g., an audio signal).

Finally, Chapter 6 summarizes the investigation in this thesis, and concludes with a discussion of open issues.

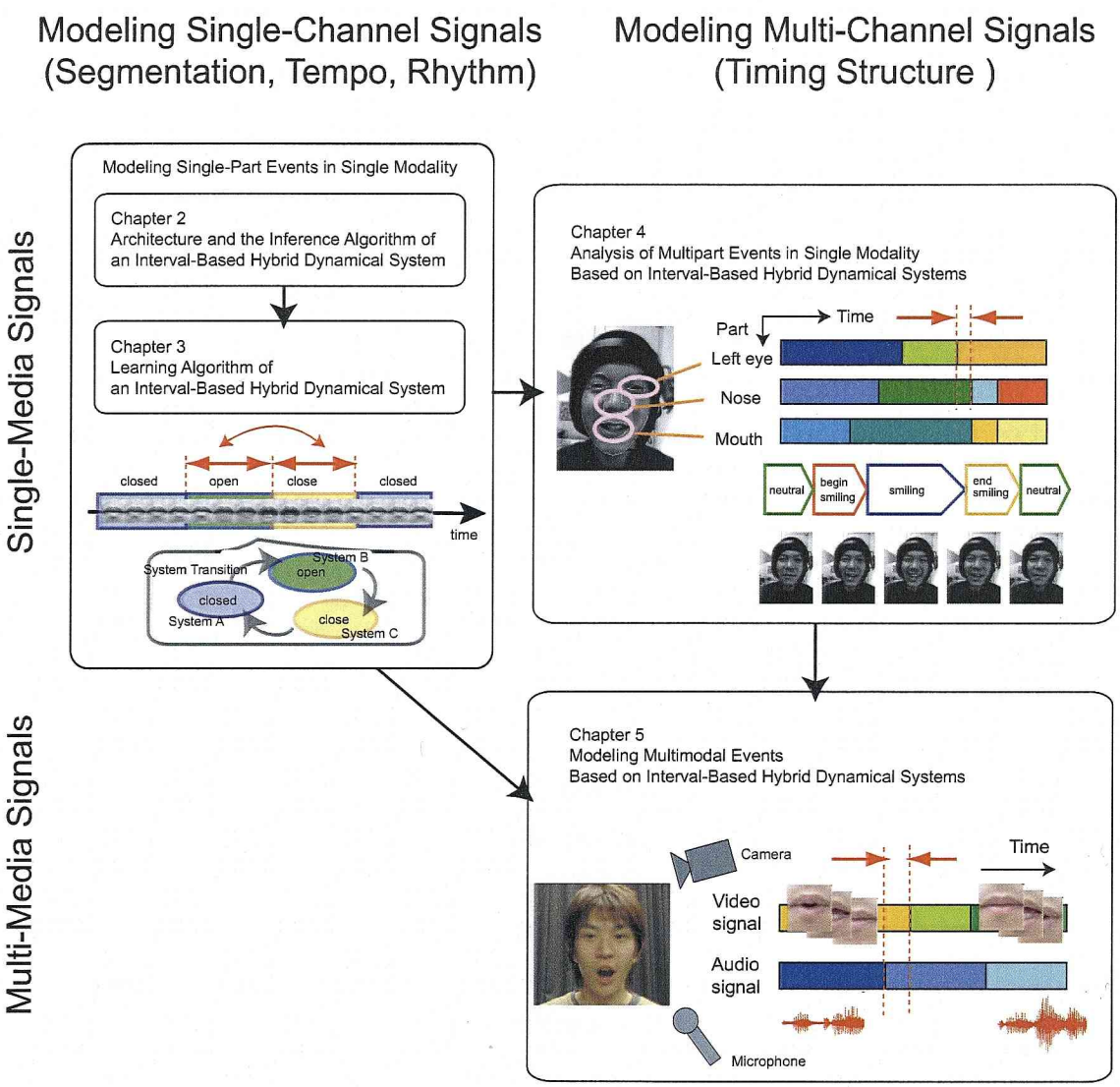


Figure 1.11: Overview of the thesis.

Chapter 2

Interval-Based Hybrid Dynamical System

In this chapter, we introduce an interval-based hybrid dynamical system (interval system). The system consists of a finite state automaton and a set of multiple linear dynamical systems as we described in the previous chapter. Each linear dynamical system represents a dynamic primitive that corresponds to a discrete state of the automaton; meanwhile the automaton controls the activation timing of the dynamical systems. Thus, the interval system can generate and analyze complex multivariate sequences that consist of temporal regimes of dynamic primitives (see Figure 1.9 for the example).

2.1 System Architecture

An interval system has a two-layer architecture (Figure 2.1). The first layer (the top dashed box in Figure 2.1) has a finite state automaton as a discrete-event system that models stochastic transitions between discrete events. The second layer (the second-top dashed box in Figure 2.1) consists of a set of linear dynamical systems $\mathcal{D} = \{D_1, ..., D_N\}$. To integrate these two layers, we introduce *intervals* (the middle of Figure 2.1); each interval is described by $< q_i, \tau >$, where q_i denotes a discrete state in the automaton and τ denotes the physical temporal duration length of the interval.

As we described in Subsection 1.4.3, the ending points of the intervals can be considered to be the discrete events that the automaton models. While the automaton models only the order of discrete events without physical-time metric,

the intervals provides physical-time grounding for the automaton due to the duration length τ.

We assume that a dynamical system D_i characterizes the type of dynamics in the interval $< q_i, \tau >$. Therefore, each state in the automaton corresponds to a unique linear dynamical system in the second layer; that is, q_i denotes the label of the corresponding linear dynamical system as well as a state in the automaton. Note that multiple different intervals can correspond to the same state in the automaton (i.e., their dynamics are described by the same linear dynamical system).

Signal Generation and Segmentation

An interval system is a stochastic generative model. Once the interval system has been constructed by learning as will be described in Chapter 3, it can generate a multivariate signal sequence by activating the automaton. The activated automaton first generates a sequence of intervals (the middle of Figure 2.1), each of which then generates a signal sequence based on its corresponding linear dynamical system (the second bottom of Figure 2.1). Note that the activation timing and period of the linear dynamical system are controlled by the duration length of the interval.

When a temporal sequence of observed signal data (multivariate sequence) is given, the system finds the activation timing and period of the linear dynamical systems based on the likelihood calculation. That is, the observed sequence is partitioned into a group of sub-sequences so that the dynamic signal variation in each sub-sequence can be described by a linear dynamical system, which is denoted by the discrete-state label of the interval covering that sub-sequence (see Section 2.4 for details). As a result, the observation sequence is transformed into a sequence of internal states that is partitioned by an interval sequence.

Notations

We define some terms and notations for later discussions. Firstly, we simply use the term "dynamical systems" to denote linear dynamical systems.

Internal state. All the constituent dynamical systems are assumed to share an n-dimensional internal state space. Each activated dynamical system can generate sequences of real valued internal state vector $x \in \mathbf{R}^n$, which can be

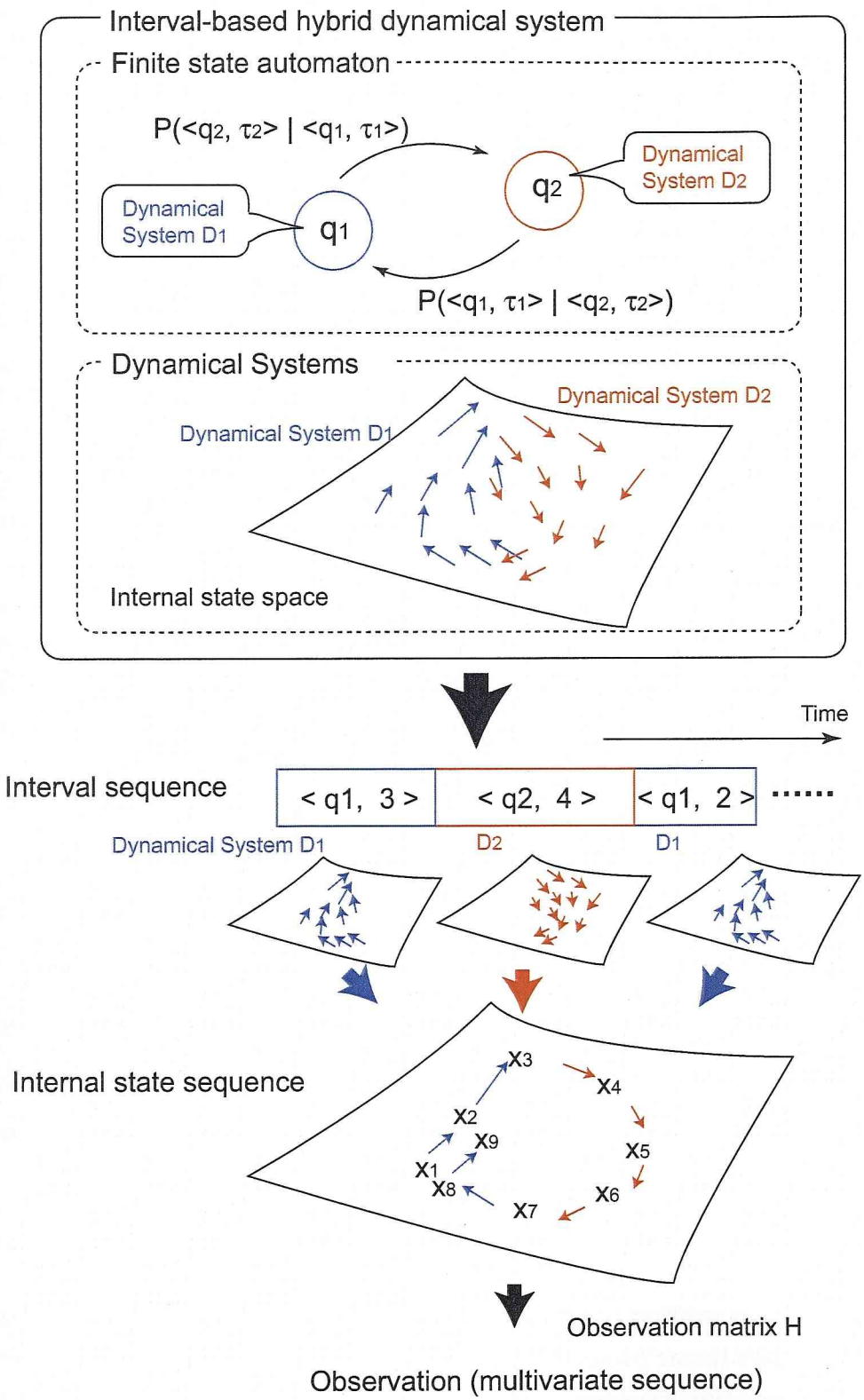


Figure 2.1: Interval-based hybrid dynamical system and the generation of a multivariate sequence.

mapped onto the observation space by a linear function. We assume such linear transformation function is also shared by all the dynamical systems.

Observation. An observation sequence is described by a multivariate vector $y \in \mathbf{R}^m$ sequence in a m-dimensional observation space.

Discrete state. The finite state automaton has a discrete state set $Q = \{q_1, ..., q_N\}$. Each state $q_i \in \mathcal{Q}$ corresponds to the dynamical system D_i, respectively.

Duration lengths of intervals. The duration length that an interval continues described by a positive integer because we assume the interval system as a discrete time model. To reduce parameter size, we set a minimum duration length $l_{\min}$ and a maximum duration length $l_{\max}$; we define a duration length as $\tau \in \mathcal{T} \triangleq \{l_{\min}, ..., l_{\max}\}$.

Interval. An interval generated by the automaton is defined as a combination of a discrete state and a duration length. We use notation $< q_i, \tau > \in \mathcal{Q} \times \mathcal{T}$ to represent the interval that has state q_i and duration τ.

2.2 Linear Dynamical Systems

2.2.1 Formulation

The state transition of dynamical system D_i in the internal state space, and the mapping from the internal state space to the observation space is modeled as the following linear equations:

$$\begin{aligned} x_t &= F^{(i)} x_{t-1} + g^{(i)} + \omega_t^{(i)} \\ y_t &= H x_t + v_t, \end{aligned} \tag{2.1}$$

where $F^{(i)}$ is a transition matrix and $g^{(i)}$ is a bias vector. H is an observation matrix that defines linear projection from the internal state space to the observation space. $\omega^{(i)}$ and v is the process noise and the observation noise. Note that each dynamical system has $F^{(i)}$, $g^{(i)}$, and $\omega_t^{(i)}$ individually. We assume each of noise term $\omega^{(i)}$ and v has Gaussian distribution $\mathcal{N}_{x_t}(0, Q^{(i)})$ and $\mathcal{N}_{y_t}(0, R)$, respectively. Here, we use the notation $\mathcal{N}_x(a, B)$ to denote a Gaussian distribution that has

average vector a and covariance matrix B in the space of variable x:

$$\mathcal{N}_x(a, B) = (2\pi)^{-n/2}|B|^{-1/2}\exp\left\{-\frac{1}{2}(x-a)^\top B^{-1}(x-a)\right\}, \tag{2.2}$$

where n is a dimension of vector x.

We assumed that all the dynamical systems share a single internal state space. The main reason is that we want to reduce parameters in the interval system; it is, however, possible to design the system with an individual internal state space for each dynamical system. In such cases, observation parameters $H^{(i)}$ and $R^{(i)}$ are required for each dynamical system. Although they provide more flexibility in models, a large parameter space causes problems such as over-fitting and high computational costs.

Probability Density Distributions

Using the formulation and notation mentioned above, we can consider probability density distribution as follows:

$$\begin{aligned} p(x_t|x_{t-1}, s_t = q_i) &= \mathcal{N}_{x_t}(F^{(i)}x_{t-1}, Q^{(i)}) \\ p(y_t|x_t, s_t = q_i) &= \mathcal{N}_{y_t}(Hx_t, R), \end{aligned} \tag{2.3}$$

where the probability variable s_t is an activated discrete state at time t (i.e., dynamical system D_i is activated). The second equation is independent of the probability variable s_t because of the assumption in the previous paragraph. In this thesis, we use p to denote probability density function and P for probability.

Since the state distribution is recursively calculated by the density distributions above, we define the initial state distribution as follows:

$$p(x_1|, s_1 = q_i) = \mathcal{N}_{x_1}(x^{(i)}_{\text{init}}, V^{(i)}_{\text{init}}). \tag{2.4}$$

2.2.2 Class of Linear Dynamical Systems

The class of linear dynamical systems can be categorized by the eigenvalues of the transition matrix, which determine the zero-input response of the system. In other word, these eigenvalues determine the behavior of generable time-varying patterns (trajectories) in the state space.

Without Bias Term

To concentrate on the temporal evolution of the state in the dynamical system, let us assume the bias and the process noise term is zero in Equation (2.1). Using the eigenvalue decomposition of the transition matrix:

$$F = E\Lambda E^{-1} = [e_1, ..., e_n]\text{diag}(\lambda_1, ..., \lambda_n)[e_1, ..., e_n]^{-1},$$

we can solve the state at time t with initial condition x_0:

$$x_t = F^t x_0 = (E\Lambda E^{-1})^t x_0 = E\Lambda^t E^{-1} x_0 = \sum_{p=1}^{n} \alpha_p e_p \lambda_p^t, \tag{2.5}$$

where e_p and λ_p is a corresponding eigenvalue and eigenvector pair. We omit the indices i for simplification. A weight value α_p is determined from the initial state x_0 by calculating $[\alpha_1, ..., \alpha_n]^\top = E^{-1}x_0$.

Hence, the generable patterns from the system can be categorized by the position of the eigenvalues (poles) $\lambda_1, ..., \lambda_n$ on the complex plane. Especially, the arguments (angle) of eigenvalues in a complex plain determine the state will oscillate or not:

- At least one negative or complex eigenvalue exists → oscillating.
- All the eigenvalues have real number → non-oscillating.

On the other hand, the absolute values of eigenvalues determine the state will converge or not:

- At least one absolute value of eigenvalue exceeds one → diverging.
- All the absolute values of eigenvalues are smaller than one → converging.

Figure 2.2 shows examples of state trajectories when the dimensionality of the state is two.

For instance, the system can generate time-varying patterns that converge to zero if and only if $|\lambda_p| < 1$ for all $1 \leq p \leq n$ (using the term in control theory, we can say that the system is stable); meanwhile, the system can generate non-monotonic or cyclic patterns if the imaginary parts of eigenvalues have nonzero values.

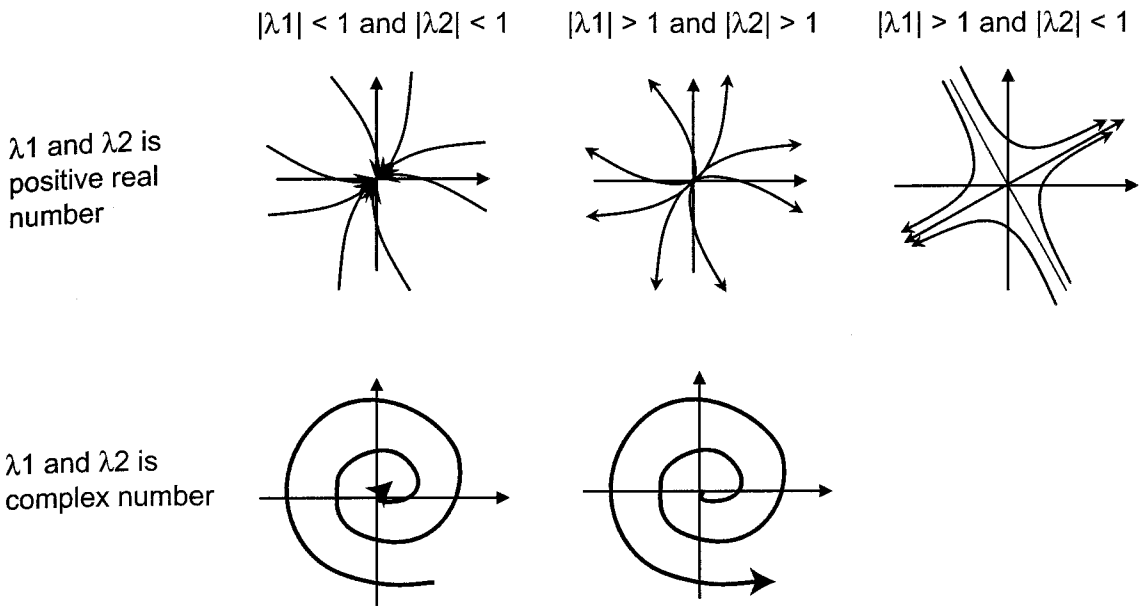


Figure 2.2: Examples of dynamics class when the dimensionality of the state is two.

With Bias Term

We first consider the dynamical system has converging behavior. In case that the system equation has bias vector g as shown in Equation (2.1), the state converges to a certain position x_{conv} in the state space. We can calculate state x_{conv} using a similar method to linear type recurrence equations.

Let us assume that the process is not stochastic but deterministic (i.e., noise term is zero) same as the previous paragraph. Substituting x_{conv} for x_t and x_{t-1} in Equation (2.1), we get the following equation:

$$x_{\text{conv}} = Fx_{\text{conv}} + g. \tag{2.6}$$

From the equation above, the convergence point becomes:

$$x_{\text{conv}} = (I - F)^{-1}g. \tag{2.7}$$

Calculating subtraction of each term between original Equation (2.1) and Equa-

tion (2.6), we get the following equation:

$$\begin{aligned} x_t - x_{\text{conv}} &= F(x_{t-1} - x_{\text{conv}}) \\ &= F^t(x_0 - x_{\text{conv}}) \end{aligned} \tag{2.8}$$

Here, the Equation (2.8) determine the temporal evolution of the state when the state converges to x_{conv}. From Equation (2.7) and Equation (2.8), we get

$$\begin{aligned} x_t &= F^t x_0 + (I - F^t)(I - F)^{-1} g \\ &= E\Lambda^t E^{-1} x_0 + E(I - \Lambda)^{-1}(I - \Lambda^t)E^{-1} g. \end{aligned} \tag{2.9}$$

In general case (i.e., the state might diverge), we can recursively apply the Equation (2.1) and get the following equation:

$$\begin{aligned} x_t &= Fx_{t-1} + g = F(Fx_{t-2} + g) + g \\ &= F^t x_0 + (\sum_{u=1}^{t-1} F^u) g \end{aligned} \tag{2.10}$$

Substituting $\sum_{u=1}^{t-1} F^u = (I - F^t)(I - F)^{-1}$ for the second term of the equation above, we get the same equation as Equation (2.9). Thus, the Equation (2.9) is a general (i.e., independent of eigenvalues) equation for the temporal evolution of the state. We can easily deduce Equation (2.7) from Equation (2.9) as a special case when $\lim_{t\to\infty} F^t = O$ (i.e., all the eigenvalues are smaller than one).

2.2.3 Probabilistic State Inference

In this section, we show the probabilistic inference of the internal state in linear dynamical systems. Let us assume that the internal state has a Gaussian distribution at each time points. Then, the transition of the internal state becomes a Gauss-Markov process, which is inferable in the same manner as Kalman filtering [AM79].

The inference consists of the following two steps:

1. Prediction step

2. Observation (Collection) step

In the next two paragraphs, we describe each of the steps.

Prediction Step

Because we assumed that the probability density of the internal state is a Gaussian distribution, the state distribution of time $t-1$ under the condition of observation from 1 to $t-1$ is represented by the following equation:

$$p(x_{t-1}|y_1^{t-1} = \hat{y}_1^{t-1}, s_{t-1} = q_i) = \mathcal{N}_{x_{t-1}}(x_{t-1|t-1}^{(i)}, V_{t-1|t-1}^{(i)}), \tag{2.11}$$

where $x_{t-1|t-1}^{(i)}$ is a mean vector and $V_{t-1|t-1}^{(i)}$ is a covariance matrix, and $\hat{y}_1^{t-1} = \hat{y}_1, ..., \hat{y}_{t-1}$ is an observation sequence from 1 to $t-1$.

Using Equation (2.3) and (2.11), we can calculate the predicted state distribution under the condition of observations from 1 to $t-1$ as follows:

$$\begin{aligned} p(x_t|y_1^{t-1} = \hat{y}_1^{t-1}, s_t = q_i) &= \int_{x_{t-1}} p(x_t|x_{t-1}, s_t = q_i) p(x_{t-1}|y_1^{t-1} = \hat{y}_1^{t-1}, s_{t-1} = q_i) \\ &= \int_{x_{t-1}} \mathcal{N}_{x_t}(F^{(i)}x_{t-1}, Q^{(i)}) \mathcal{N}_{x_{t-1}}(x_{t-1|t-1}^{(i)}, V_{t-1|t-1}^{(i)}) \\ &= \mathcal{N}_{x_t}(x_{t|t-1}^{(i)}, V_{t|t-1}^{(i)}), \end{aligned} \tag{2.12}$$

$$\text{where} \quad \begin{cases} x_{t|t-1}^{(i)} = F^{(i)} x_{t-1|t-1}^{(i)} \\ V_{t|t-1}^{(i)} = F^{(i)} V_{t-1|t-1}^{(i)} {F^{(i)}}^\top + Q^{(i)} \end{cases}$$

We can also calculate the predicted observation distribution using the predicted state distribution and Equation (2.3):

$$\begin{aligned} p(y_t|y_1^{t-1}, s_t = q_i) &= \int_{x_t} p(y_t|x_t, s_t = q_i) p(x_t|y_1^{t-1}, s_t = q_i) \\ &= \int_{x_t} \mathcal{N}_{y_t}(Hx_t, R) \mathcal{N}_{x_t}(x_{t|t-1}^{(i)}, V_{t|t-1}^{(i)}) \\ &= \mathcal{N}_{y_t}(y_{t|t-1}^{(i)}, M_{t|t-1}^{(i)}), \end{aligned} \tag{2.13}$$

$$\text{where} \quad \begin{cases} y_{t|t-1}^{(i)} = H x_{t|t-1}^{(i)} \\ M_{t|t-1}^{(i)} = H V_{t|t-1}^{(i)} H^\top + R \end{cases}$$

Observation Step

After the prediction step in the previous paragraph, the state distribution at time t can be calculated once the observation data y_t becomes available. Using Bayesian rule with Equation (2.3), (2.12), and (2.13), we can update the state distribution at

time t under the condition of observations from time 1 to t as follows:

$$\begin{aligned} p(x_t|y_1^t = \hat{y}_1^t, s_t = q_i) &= \frac{p(y_t|x_t)p(x_t|y_1^{t-1} = \hat{y}_1^{t-1}, s_t = q_i)}{p(y_t|y_1^{t-1} = \hat{y}_1^{t-1}, s_t = q_i)} \\ &= \frac{\mathcal{N}_{y_t}(Hx_t, R)|_{y_t=\hat{y}_t}\mathcal{N}_x(x_{t|t-1}^{(i)}, V_{t|t-1}^{(i)})}{\mathcal{N}_{y_t}(y_{t|t-1}^{(i)}, M_{t|t-1}^{(i)})|_{y_t=\hat{y}_t}} \\ &= \mathcal{N}_{x_t}(x_{t|t}^{(i)}, V_{t|t}^{(i)}) \end{aligned} \tag{2.14}$$

$$\text{where} \quad \begin{cases} x_{t|t}^{(i)} &= x_{t|t-1}^{(i)} + K_t^{(i)}(\hat{y}_t - y_{t|t-1}^{(i)}) \\ V_{t|t}^{(i)} &= (V_{t|t-1}^{(i)\ -1} + H^\top R^{-1} H)^{-1} \\ &= (I - K_t^{(i)} H) V_{t|t-1}^{(i)} \\ K_t^{(i)} &= V_{t|t}^{(i)} H^\top R^{-1} \end{cases}$$

Hence, the mean vectors $x_{t|t-1}^{(i)}, x_{t|t}^{(i)}, y$ and covariance matrices $V_{t|t-1}^{(i)}, V_{t|t}^{(i)}$ are updated every sampled time t using the prediction and observation steps by turns.

2.2.4 Likelihood Calculation of the Linear Dynamical System

Now, we show how to calculate the likelihood of a linear dynamical system with respect to the observation sequence in an interval.

Suppose that the dynamical system D_i represents an observation sequence $\hat{y}_{t-\tau+1}^t \triangleq \hat{y}_{t-\tau+1}, ..., \hat{y}_t$ in the interval $< q_i, \tau >$, which has a duration length τ. Then, the likelihood score of the system D_i with respect to the observation sequence $\hat{y}_{t-\tau+1}^t$ is calculated by the following equation:

$$\begin{aligned} d_{[t-\tau+1,t]}^{(i)} &\triangleq P(y_{t-\tau+1}^t = \hat{y}_{t-\tau+1}^t | < q_j, \tau >) \\ &= \prod_{t'=t-\tau+1}^{t} \gamma^m \, p(y_{t'} = \hat{y}^{t'} | y_{t-\tau+1}^{t'-1} = \hat{y}_{t-\tau+1}^{t'-1}, s_{t'} = q_j), \end{aligned} \tag{2.15}$$

where we assume Gaussian distribution $N(x_{\text{init}}^{(i)}, V_{\text{init}}^{(i)})$ for the initial state distribution in the interval as we described in Subsection 2.2.1 (see Equation (2.4)); that is, we substitute $p(y_{t'} = \hat{y}^{t'} \,|\, y_{t-\tau+1}^{t'-1} = \hat{y}_{t-\tau+1}^{t'-1}, s_{t'} = q_j)$ with $\mathcal{N}_{y_t'}(Hx_{\text{init}}^{(i)}, \ HV_{\text{init}}^{(i)} H^\top + R)$ when $t' = t - \tau + 1$. On the other hand, γ^m is a volume size of observations in the observation space to convert probability density values to probabilities. m is the dimensionality of observation vectors. γ is

assumed to be provided manually based on the size of the observation space (the range of each element in observation vectors). This likelihood score is used in Section 2.4 to evaluate the fitting of linear dynamical system to the given multivariate sequences.

Substituting Equation (2.13) into Equation (2.15), we finally get the likelihood of linear dynamical system D_i under the assumption of the Gauss-Markov process:

$$\begin{aligned} d^{(i)}_{[t-\tau+1,t]} &= \prod_{t'=t-\tau+1}^{t} \gamma^m \mathcal{N}_{y_{t'}}(y^{(i)}_{t'|t'-1}, M^{(i)}_{t'|t'-1})|_{y_{t'}=\hat{y}_{t'}} \\ &= \frac{\gamma^m}{(2\pi)^{m/2}} \prod_{t'=t-\tau+1}^{t} |M^{(i)}_{t'|t'-1}|^{-1/2} \exp\left\{-\frac{1}{2}(\hat{y}_{t'} - y^{(i)}_{t'|t'-1})^\top M^{(i)\,-1}_{t'|t'-1}(\hat{y}_{t'} - y^{(i)}_{t'|t'-1})\right\} \\ &= \exp\left\{\tau m \log\frac{\gamma}{\sqrt{2\pi}} - \frac{1}{2}\sum_{t'=t-\tau+1}^{t}\left(\log|M^{(i)}_{t'|t'-1}| + e_{t'}^\top M^{(i)\,-1}_{t'|t'-1} e_{t'}\right)\right\}, \\ &\quad \text{where} \quad e_{t'} = \hat{y}_{t'} - y^{(i)}_{t'|t'-1}. \end{aligned} \tag{2.16}$$

Note that we use $y^{(i)}_{t'|t'-1} = Hx^{(i)}_{\text{init}}$ and $M^{(i)}_{t'|t'-1} = HV^{(i)}_{\text{init}}H^\top + R$ for the initial distribution parameters at $t' = t - \tau + 1$.

2.3 Interval-Based State Transitions of the Automaton

2.3.1 Interval-Based State Transition

In this section, we define transition of discrete states in the automaton that generate interval sequences. Here, we assume the first-order Markov property for the generated intervals. The difference from conventional state transition models, such as hidden Markov models, is that the automaton models not only the transition of discrete states but also the correlation between the adjacent interval duration lengths.

Let $\mathcal{I} = I_1, ..., I_K$ be an interval sequence generated by the automaton. To simply the model, we assume that the adjacent intervals have no temporal gaps or overlaps. Here, the interval I_k depends on only the previous interval I_{k-1} because of the Markov property assumption. Then, the Markov process of intervals can

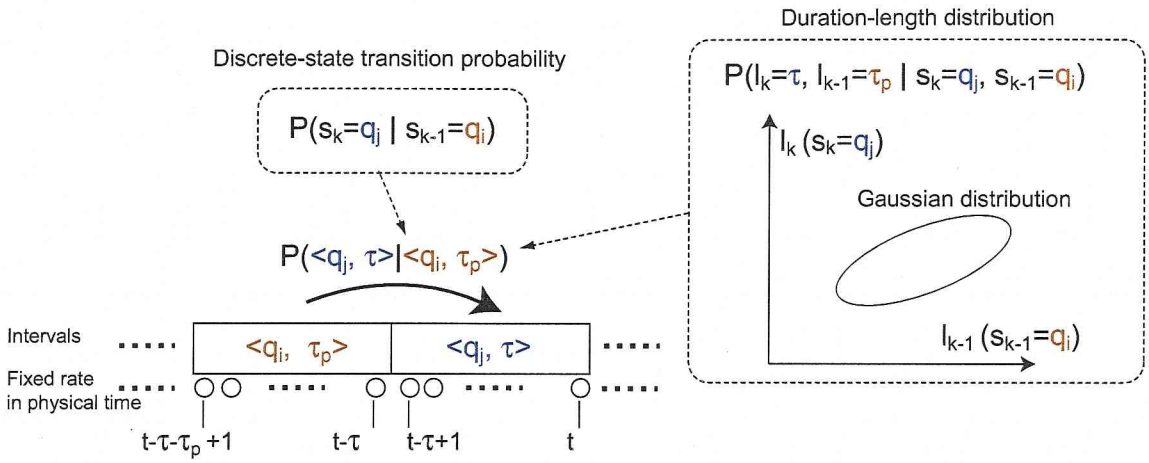


Figure 2.3: First-order Markov property is assumed for a sequence of intervals.

be modeled by the following conditional probability:

$$P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >),$$

where it denotes that the interval $< q_j, \tau >$ occurs after the interval $< q_i, \tau_p >$ (see Figure 2.3).

The probability $P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >)$ requires a large parameter set, which cause not only computational costs but also the problem of over-fitting during a training phase. We therefore use a parametric model for the *duration-length distribution*:

$$h^{(ij)}(l_k, l_{k-1}) \triangleq P(l_k, l_{k-1} | s_k = q_j, s_{k-1} = q_i), \tag{2.17}$$

where the two-dimensional distribution models a joint probability density function of duration lengths in adjacent interval pairs that has state q_i and q_j in this order. s_k and l_k is a probability variable of the discrete state and the duration length in the interval I_k, respectively.

We can assume an arbitrary density function as $h^{(ij)}(l_k, l_{k-1})$. For convenience, we use a two-dimensional Gaussian distribution normalized in the range of $[l_{\min}, l_{\max}]$, as shown in the top right in Figure 2.3; thus, the parameter set of the function $h^{(ij)}(l_k, l_{k-1})$ becomes $\{h_m^{(i)}, h_v^{(i)}, h_c^{(ij)}\}$, where $h_m^{(i)}$ and $h_v^{(i)}$ denotes mean and variance of duration lengths in discrete state q_i, and $h_c^{(ij)}$ denotes covariance between the adjacent duration lengths in discrete-state sequences $q_i q_j$ $(i \neq j)$.

Using the above notations and assuming that the current discrete state is independent on the duration of the previous interval, we can calculate the interval transition probability as follows:

$$\begin{aligned} P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >) &= P(l_k = \tau | s_k = q_j, s_{k-1} = q_i, l_{k-1} = \tau_p) \\ &\quad \times P(s_k = q_j | s_{k-1} = q_i) \\ &= \hat{h}^{(ij)}(\tau, \tau_p) A_{ij}, \end{aligned} \tag{2.18}$$

where $\hat{h}^{(ij)}(l_k, l_{k-1})$ is a one-dimensional Gaussian distribution:

$$\begin{aligned} \hat{h}^{(ij)}(l_k, l_{k-1}) &\triangleq P(l_k | s_k = q_j, s_{k-1} = q_i, l_{k-1}) \\ &= \frac{h^{(ij)}(l_k, l_{k-1})}{\sum_{l_{k-1}} h^{(ij)}(l_k, l_{k-1})}, \end{aligned}$$

and A_{ij} is a *discrete-state transition probability*:

$$A_{ij} \triangleq P(s_k = q_j | s_{k-1} = q_i), \tag{2.19}$$

where $i \neq j$.

Note that, in the conventional discrete state models such as HMMs and SLDSs, the diagonal elements of the matrix $[A_{ij}]$ define the probabilities of the self loops. In the interval system, on the other hand, the diagonal elements are separated from the matrix and defined as duration-length distributions. As a result, the balance between diagonal and non-diagonal elements varies due to the current state duration.

2.3.2 Probabilistic Inference of the Intervals

Now, we describe how to inference the intervals (i.e., states and duration lengths) based on the interval-based state transition. Unlike conventional (i.e., frame-wise) discrete state inference, we have to consider two different temporal representation: the order of intervals k and time t at the same time. As shown in the following paragraphs, the probabilistic inference becomes recursive calculation based on time t.

Let us consider how to calculate all the probabilities of every possible interval when the parameters of the automaton are given. Because we assumed the first-order Markov property, the probability of the interval $< q_j, \tau >$ can be cal-

culated using all the possible intervals that have occurred just before the interval $< q_j, \tau >$:

$$\begin{aligned} & P(I_k =< q_j, \tau >) \\ = & \sum_{<q_i,\tau_p> \in Q \times T} P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >) P(I_{k-1} =< q_i, \tau_p >), \end{aligned} \tag{2.20}$$

where the summation for $< q_i, \tau_p >$ does not include q_j (i.e., there are no self loops such as $q_j \rightarrow q_j$).

Although this equation gives us general idea of the inference algorithm, this recursion is based on the temporal order of intervals k, and we need to map all the intervals to the physical time line. Here, we introduce a variable f_t that takes one of the binary values {0, 1}. If $f_t = 1$, it denotes the interval "finishes" at time t, which follows Murphy's notation that is used in a research note about segment models [Mur02]. Using this notation, we can rewrite Equation (2.20) as the following time-based equation:

$$\begin{aligned} & P(s_t = q_j, l_t = \tau, f_t = 1) \\ = & \sum_{i(i \neq j)} \sum_{\tau_p} \{ P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1) \\ & \times P(s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1) \}, \end{aligned} \tag{2.21}$$

where

$$\begin{aligned} & P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1) \\ = & P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >). \end{aligned} \tag{2.22}$$

If we assume a finite length for the generated sequence, let the length be T, the probability $\sum_j \sum_\tau P(s_t = q_j, l_t = \tau, f_t = 1) = P(f_t = 1)$ becomes 1 at the final time point $t = T$.

Approximation (assuming the independence of the previous duration length)

If we can approximate that the interval probability is independent of the duration length of the previous interval, we can use the following equation as substitute

for Equation (2.22):

$$P(s_t = q_j, f_t = 1) = \sum_{\tau} P(s_t = q_j, l_t = \tau, f_t = 1)$$
$$= \sum_{\tau} \sum_{i(i \neq j)} P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, f_{t-\tau} = 1) P(s_{t-\tau} = q_i, f_{t-\tau} = 1). \tag{2.23}$$

Example (Interval lattice)

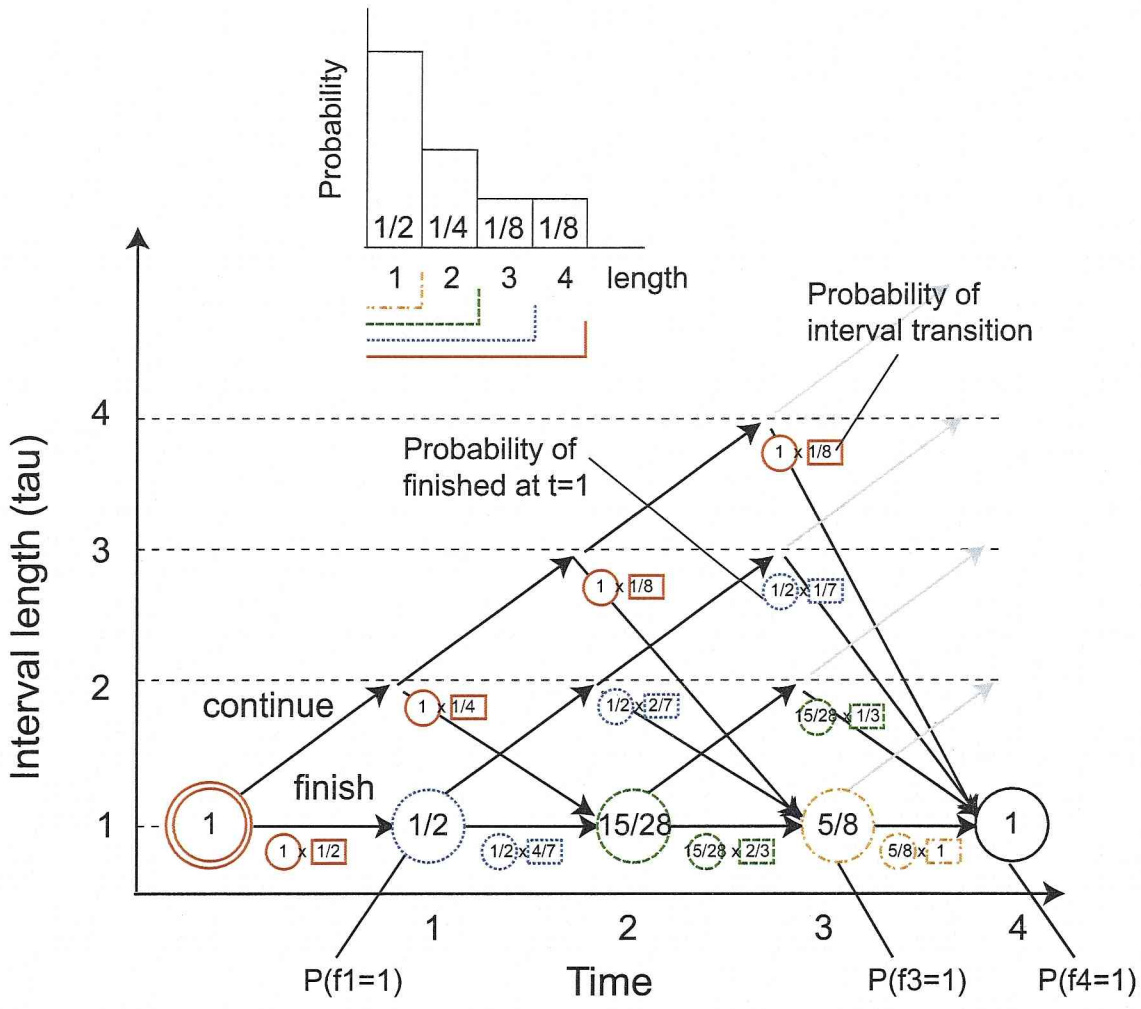
Figure 2.4 shows an intuitive example of this recursive calculation when the total sequence length is four. In this example, the number of states is two (e.g., q_1 and q_2), we can therefore omit state transition probability $A_{12} = A_{21} = 1$. The initial state is q_1 or q_2, and the succeeding intervals are labeled by these two states by turns. To simplify the example, we assume the following duration-length distribution (not a Gaussian):

$$\hat{h}^{(ij)}(l_k = 1, l_{k-1}) = 1/2, \quad \hat{h}^{(ij)}(l_k = 2, l_{k-1}) = 1/4,$$
$$\hat{h}^{(ij)}(l_k = 3, l_{k-1}) = 1/8, \quad \hat{h}^{(ij)}(l_k = 4, l_{k-1}) = 1/8,$$

where the distribution is independent of l_{k-1}. Therefore, this is an example of Equation (2.23).

The arrow that has beginning point at (time t, length τ) represents the state q_1 (or q_2) continues or finishes at time t with duration length τ. Four circle nodes at the bottom (except the leftmost node) represent $P(f_t = 1) = \sum_{j=1}^{2} \sum_{\tau=1}^{4} P(s_t = q_j, l_t = \tau, f_t = 1)(t = 1,2,3,4)$, and one of the path from the leftmost circle to the rightmost circle determines an interval sequence (partitioned sequence). For example, the bottom path represents that all the comprising intervals have length 1, and the indices of the intervals therefore correspond to time points (i.e., $k = t$).

Each node has multiple input from other nodes, which corresponds to the summation over duration length τ in Equation (2.23). We see that $P(f_4 = 1)$ takes 1 because all the possible interval sequences finish at the final time point $t = 4$.



Examples

q1	q2	q1	q2

Interval sequence that corresponds to the bottom path

q1	q2

Interval sequence that corresponds to the top path

Figure 2.4: Interval lattice and examples of the generated interval sequences

2.4 Inference of the Interval-Based Hybrid Dynamical System

This section describes a probabilistic inference method that searches the optimal interval sequence to represents an input multivariate sequence. The method assumes that the interval system have been trained beforehand.

As we will see in the following paragraphs, the inference method recursively finds the intervals that provide the highest likelihood score with respect to the input. This is done by generating all the possible intervals and by selecting the optimal interval sets at every time t based on a dynamic programming technique. As a result, the input sequence is partitioned and labeled by discrete states that determine the most likely dynamical system to represent a multivariate sequence in each interval. In other words, the inference is a model fitting process that fits intervals to the given multivariate sequences.

The likelihood of the trained model with respect to the input sequence is obtained simultaneously as the score of the fitting precision. This inference process is required in the EM algorithm of the interval system identification as we will see in Section 3.4.

2.4.1 Forward Algorithm

The most naive method for the interval-sequence search is that first calculates the likelihood scores of the model from all the possible interval sequences independently, and then finds the best interval sequence that provides the largest likelihood. However, the computational cost becomes order of $O(N^T)$ in this case. To avoid unnecessary calculation, we exploit a recursive calculation similar to HMMs.

Let us first consider the forward algorithm of the interval system. Suppose that input multivariate data y have been observed from time 1 to t, and the interval $I_k =< q_j, \tau >$ ends at time t. Considering all the possible intervals that have occurred just before the interval I_k, we can decompose the joint probability

$P(I_k =< q_j, \tau >, y_1^t)$ as the following recursive equation:

$$\begin{aligned}
&P(I_k =< q_j, \tau >, y_1^{e_k}) \\
&= P(y_{e_{k-1}+1}^{e_k} | I_k =< q_j, \tau >) \\
&\times \sum_{<q_i, \tau_p> \in \mathcal{Q} \times \mathcal{T}} \left\{ P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >) P(I_{k-1} =< q_i, \tau_p >, y_1^{e_{k-1}}) \right\}
\end{aligned}$$

where e_k and e_{k-1} are the ending points of interval I_k and I_{k-1}, respectively, and $e_k = t$.

As we described in Subsection 2.3.2, this recursion is based on the interval order, and is difficult to cope with observation data that comes every time point t. We therefore rewrite this equation as the following time-based recursion, which is similar to Equation (2.22) rewritten from Equation (2.22):

$$\begin{aligned}
& P(s_t = q_j, l_t = \tau, f_t = 1, y_1^t) \\
= \; & P(y_{t-\tau+1}^t | s_t = q_j, l_t = \tau, f_t = 1) \\
& \times \sum_{i(i \neq j)} \sum_{\tau_p} \{ P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1) \\
& \quad \times P(s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1, y_1^{t-\tau}) \}
\end{aligned} \tag{2.24}$$

To initialize the forward algorithm, we have to calculate the probability of the interval that appears at the first time in each interval sequence.

$$\begin{aligned}
& P(s_t = q_j, l_t = \tau, f_t = 1, f_1^{t-1} = 0, y_1^t) \\
= \; & P(y_1^t | s_t = q_j, l_t = t) P(s_t = q_j) P(l_t = t | s_t = q_j)
\end{aligned} \tag{2.25}$$

The duration length probability $P(l_t | s_t = q_j)$ can be calculated by the following equation:

$$\begin{aligned}
\hat{h}^{(j)}(l_t) \triangleq P(l_t = \tau | s_t = q_j) &= \sum_i \sum_{l_{k-1}} P(l_t, l_{k-1} | s_k = q_j, s_{k-1} = q_i) \\
&= \sum_i \sum_{l_{k-1}} h^{(ij)}(l_k, l_{k-1}).
\end{aligned} \tag{2.26}$$

Here, we show the overall forward algorithm in Algorithm 1, where we use the following notation:

$$\alpha_t(i, \tau) \triangleq P(s_t = q_i, l_t = \tau, f_t = 1, y_1^t).$$

We also use the notation $d^{(i)}_{[t-\tau+1,t]}$ for $P(y^t_{t-\tau+1}|s_t = q_i, l_t = \tau, f_t = 1)$, which we defined in Equation (2.15). As we have shown in Section 2.2, $d^{(i)}_{[t-\tau+1,t]}$ denotes the likelihood of the linear dynamical system D_i with respect to the observation from $t-\tau+1$ to t, and can be calculated by using parameters of dynamical system D_i. On the other hand, the interval transition probability $P(s_t = q_j, l_t = \tau, f_t = 1|s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1)$ can be calculated by Equation (2.18) in Section 2.3.

Algorithm 1 Forward Algorithm

for $t \leftarrow 1$ to $l_{min} - 1$ **do**
  Fill $\alpha_t(i,\tau)$ by 0
end for
for $t \leftarrow l_{min}$ to T **do**
  $\tau_{\max} \leftarrow \min(l_{\max}, t - l_{\min})$
  for $j \leftarrow 1$ to N **do**
    for $\tau \leftarrow l_{\min}$ to $\tau_{\max}$ **do**
      $\tau_{p\max} \leftarrow \min(l_{\max}, t-\tau)$
      $\alpha_t(j,\tau) \leftarrow d^{(j)}_{[t-\tau+1,t]} \sum_i \sum_{\tau_p=l_{\min}}^{\tau_{p\max}} A_{ij}\hat{h}^{(ij)}(\tau,\tau_p)\alpha_{t-\tau}(i,\tau_p)$ # Eq. (2.24)
    end for
    if $t \leq l_{max}$ **then**
      $\alpha_t(j,t) \leftarrow d^{(j)}_{[1,t]}\pi(j)\hat{h}^{(j)}(t)$ # Initialization (Eq. (2.25))
    end if
  end for
end for

Approximated Forward Algorithm

Assuming that the duration-length distribution is independent of the duration length of the previous interval, we can use the following recursive equation, which is deduced from Equation (2.23), on instead of Equation (2.24):

$$\begin{aligned} & P(s_t = q_j, f_t = 1, y_1^t) \\ = & \sum_\tau P(y^t_{t-\tau+1}|s_t = q_j, l_t = \tau, f_t = 1) \\ \times & \sum_{i(i\neq j)} \{P(s_t = q_j, l_t = \tau, f_t = 1|s_{t-\tau} = q_i, f_{t-\tau} = 1, y_1^{t-\tau}) \\ & \times P(s_{t-\tau} = q_i, f_{t-\tau} = 1, y_1^{t-\tau})\} \end{aligned} \tag{2.27}$$

In precise, the above equation calculates $P(s_t = q_j, f_t = 1, y_1^t)$, where one of f_1 to f_{t-1} is 1. Therefore, we also require the probability of the interval that appears

at the first time in each interval sequence: $P(s_t = q_j, f_t = 1, f_1^{t-1} = 0, y_1^t)$. Then, we have to add to this probability to Equation (2.27), when t is not grater than the maximum interval length $l_{\max}$.

Here, we can use the following relation:

$$P(s_t = q_j, f_t = 1, f_1^{t-1} = 0, y_1^t) = P(s_t = q_j, f_t = 1, l_t = t, y_1^t),$$

because $l_t = t$ implicitly denotes that the interval does not finish from time 1 to $t-1$. Therefore, the initialization becomes exactly the same equation as Equation (2.25).

The overall forward algorithm becomes Algorithm 2, where we use the following notation:

$$\alpha_t(i) \triangleq P(s_t = q_i, f_t = 1, y_1^t).$$

Algorithm 2 Forward Algorithm (assuming the independence of previous dur.)

for $t \leftarrow 1$ to $l_{min} - 1$ **do**
  Fill $\alpha_t(i)$ by 0
end for
for $t \leftarrow l_{min}$ to T **do**
  $\tau_{\max} \leftarrow \min(l_{\max}, t - l_{\min})$
  for $j \leftarrow 1$ to N **do**
    $\alpha_t(j) \leftarrow \sum_{\tau=l_{\min}}^{\tau_{\max}} d^{(j)}_{[t-\tau+1,t]} \sum_i A_{ij} \hat{h}^{(j)}(\tau) \alpha_{t-\tau}(i)$ # Equation (2.27)
    if $t \leq l_{max}$ **then**
      $\alpha_t(j) \leftarrow \alpha_t(j) + d^{(j)}_{[1,t]} \pi(j) \hat{h}^{(j)}(t)$
    end if
  end for
end for

2.4.2 Viterbi Algorithm

The forward algorithm often causes numerical underflow when the length of the input becomes longer. That is, at each step of the recursion in Equation (2.24), the probability such as $\alpha_t(j, \tau)$ becomes smaller than the previous probability, and finally the probabilities get below the machine epsilon.

In the following paragraphs, we describe the Viterbi algorithm in which we can take logarithm of the formulation to avoid the numerical underflow problem. Although this algorithm returns only the most likely interval sequence, it

is enough information for some applications. For instance, we use this Viterbi algorithm to train the interval system, as we see in Chapter 3.

This algorithm is based on a dynamic programming method similar to the Viterbi algorithm of HMMs without that it requires the consideration of duration lengths. Suppose that the algorithm have found the optimum interval sequence from time 1 to $t-\tau-\tau_p$ that maximize probability $P(s_{t-\tau}=q_i, l_{t-\tau}=\tau_p, f_{t-\tau}=1, y_1^{t-\tau})$. Let use the following notation for this maximized probability:

$$\delta_{t-\tau}(i,\tau_p) \triangleq \max_{s_1^{t-\tau-\tau_p}} P(s_1^{t-\tau-\tau_p}, s_{t-\tau}=q_i, l_{t-\tau}=\tau_p, f_{t-\tau}=1, y_1^{t-\tau}). \tag{2.28}$$

Then, we can calculate probability $\delta_t(j,\tau)$ based on the following equation, which can be deduced from Equation (2.24).

$$\begin{aligned}
&\delta_t(j,\tau) = \max_{s_1^{t-\tau}} P(s_1^{t-\tau}, s_t=q_j, l_t=\tau, f_t=1, y_1^t) \\
=\ & P(y_{t-\tau+1}^t | s_t=q_j, l_t=\tau, f_t=1) \\
& \times \max_{i(i\neq j),\tau_p} \{P(s_t=q_j, l_t=\tau, f_t=1 | s_{t-\tau}=q_i, l_{t-\tau}=\tau_p, f_{t-\tau}=1)\delta_{t-\tau}(i,\tau_p)\}.
\end{aligned} \tag{2.29}$$

Since this recursive calculation gives us the maximum probabilities of all the time points with possible states and duration lengths, we can get the most likely interval sequence using traceback of arguments (i.e., states and duration lengths) that gives the maximized probabilities at each recursion step. We therefore need to record the following arguments together with $\delta_t(j,\tau)$ at the maximization of Equation (2.29):

$$\begin{aligned}
&(s_t^*(j,\tau), l_t^*(j,\tau)) \\
=\ & \arg\max_{i(i\neq j),\tau_p} \{P(s_t=q_j, l_t=\tau, f_t=1 | s_{t-\tau}=q_i, l_{t-\tau}=\tau_p, f_{t-\tau}=1)\delta_{t-\tau}(i,\tau_p)\}.
\end{aligned}$$

Traceback

Now, we describe a traceback algorithm for searching the most likely interval sequence. Using Equation (2.29) recursively, we get the maximized probability at final time point ($t=T$)

$$\max_{s_1^{T-\tau}} P(s_1^{T-\tau}, s_T=q_j, l_T=\tau, f_T=1, y_1^t) = \delta_T(j,\tau). \tag{2.30}$$

Here, we can find the state and duration length of the most likely interval sequence by finding the maximum probability of $\delta_T(j,\tau)$:

$$(cs_K, cl_K) = \arg\max_{j,\tau} \delta_T(j,\tau). \tag{2.31}$$

Then, we the most likely interval sequence by calculating the following recursion:

$$\begin{aligned} cs_{k-1} &= s^*_{ce_k}(cs_k, cl_k), \quad cl_{k-1} = l^*_{ce_k}(cs_k, cl_k) \\ ce_{k-1} &= ce_k - cl_k, \end{aligned}$$

where ce_k is the ending point of interval k, which is initialized by $ce_K = T$, cs_k and cl_k is the state and duration length of interval k. Finally, we get intervals $< cs_k, cl_k >$ $(k \leq K)$ that comprise the most likelihood interval sequence. Note that the total number of intervals (K) is known. In the actual algorithm, we therefore use very large integer for K, or use increment of the indices on behalf decrementing index k.

Taking logarithm of all the equations, we get the overall algorithm shown in Algorithm 3.

Approximated Viterbi Algorithm

Assuming that the duration-length distribution is independent of the duration length of the previous interval, we can use the following recursive equation, which is deduced from Equation (2.27), instead of Equation (2.29):

$$\begin{aligned} \delta_t(j) &\triangleq \max_{s_1^{t-1}} P(s_1^{t-1}, s_t = q_j, f_t = 1, y_1^t) \\ &= \max_{\tau} [P(y_{t-\tau+1}^t | s_t = q_j, l_t = \tau, f_t = 1) \\ &\quad \times \max_{i(i \neq j)} \{P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, f_{t-\tau} = 1) \max_{u_1^{t-\tau-1}} \delta_{t-\tau}(i)\}] \end{aligned}$$

The overall algorithm is shown in Algorithm 4. As we described in the previous paragraph, the most likely interval sequence is given by the final traceback step. The difference from Algorithm 3 is that this approximated model does not require to recording maximized probabilities for the previous duration length.

Algorithm 3 Viterbi Algorithm

for $t \leftarrow 1$ to $l_{min} - 1$ **do**
  Fill $\log \delta_t(i,\tau)$ by $-\infty$
end for
for $t \leftarrow l_{min}$ to T **do**
  $\tau_{\max} \leftarrow \min(l_{\max}, t - l_{\min})$
  for $j \leftarrow 1$ to N **do**
    for $\tau \leftarrow l_{\min}$ to $\tau_{\max}$ **do**
      $\tau_{p\,\max} \leftarrow \min(l_{\max}, t - \tau)$
      $\log \delta_t(j,\tau) \leftarrow \log d^{(j)}_{[t-\tau+1,t]} + \max_{i,\tau_p} \left[\log A_{ij} + \log \hat{h}^{(ij)}(\tau,\tau_p) + \log \delta_{t-\tau}(i,\tau_p)\right]$
      $(s^*_t(j,\tau), l^*_t(j,\tau)) \leftarrow \arg\max_{i,\tau_p} \left[\log A_{ij} + \log \hat{h}^{(ij)}(\tau,\tau_p) + \log \delta_{t-\tau}(i,\tau_p)\right]$
      $\# i \in \{1,...,N\}, \tau_p \in \{l_{\min},...,\tau_{p\,\max}\}$
    end for
    if $t \leq l_{max}$ **then**
      $\log \delta_t(j,t) \leftarrow \log d^{(j)}_{[1,t]} + \log \pi(j) + \log \hat{h}^{(j)}(t)$
      $(s^*_t(j,t), l^*_t(j,t)) \leftarrow (0,0)$
    end if
  end for
end for
# Traceback
$ce_K \leftarrow T$
$(cs_K, cl_K) \leftarrow \arg\max_{j,\tau} \log \delta_T(j,\tau)$
while $cs_k > 0$ **do**
  $cs_{k-1} \leftarrow s^*_{ce_k}(cs_k, cl_k), \quad cl_{k-1} \leftarrow l^*_{ce_k}(cs_k, cl_k)$
  $ce_{k-1} \leftarrow ce_k - cl_k$
  $k \leftarrow k - 1$
end while

Algorithm 4 Viterbi Algorithm (assuming the independence of previous dur.)

for $t \leftarrow 1$ to $l_{min} - 1$ **do**
  Fill $\log \delta_t(i)$ by $-\infty$
end for
for $t \leftarrow l_{min}$ to T **do**
  $\tau_{\max} \leftarrow \min(l_{\max}, t - l_{\min})$
  for $j \leftarrow 1$ to N **do**
    $\delta_t(j) \leftarrow \max_\tau \left[\log d^{(j)}_{[t-\tau+1,t]} + \max_i \left\{\log A_{ij} + \log \hat{h}^{(j)}(\tau) + \delta_{t-\tau}(i)\right\}\right]$
    $l^*_t(j) \leftarrow \arg\max_i \left\{\log A_{ij} + \log \hat{h}^{(j)}(\tau) + \delta_{t-\tau}(i)\right\}$
    $s^*_t(j) \leftarrow \arg\max_\tau \left[\log d^{(j)}_{[t-\tau+1,t]} + \max_i \left\{\log A_{ij} + \log \hat{h}^{(j)}(\tau) + \delta_{t-\tau}(i)\right\}\right]$
    $\# i = (1, ..., N), \tau = (l_{\min}, ..., \tau_{\max})$
    if $t \leq l_{max}$ **then**
      $\delta'_t(j) \leftarrow \log d^{(j)}_{[1,t]} + \log \pi(j) + \log \hat{h}^{(j)}(t)$
      if $\delta_t(j) < \delta'_t(j)$ **then**
        $\delta_t(j) \leftarrow \delta'_t(j), \quad s^{(*)}_t(j) \leftarrow 0, \quad l^{(*)}_t(j) \leftarrow t$
      end if
    end if
  end for
end for
# Traceback
$ce_K \leftarrow T$
$cs_K \leftarrow \arg\max_j \log \delta_T(j)$
$k \leftarrow K$
while $cs_k > 0$ **do**
  $cl_k \leftarrow l^*_{ce_k}(cs_k), \quad cs_{k-1} \leftarrow s^*_{ce_k}(cs_k)$
  $ce_{k-1} \leftarrow ce_k - cl_k$
  $k \leftarrow k - 1$
end while

2.4.3 Calculation Cost

The forward and Viterbi algorithms require searching all the possible intervals for all the current intervals at every sampled time t. Therefore, the computational cost becomes $O((NL)^2T)$ $(L = l_{\max} - l_{\min} + 1)$, which requires a greater cost compared to HMMs, which requires only $O(N^2T)$. However, the cost can be reduced drastically in the case that the range L is small.

Approximated forward and Viterbi algorithms, which assume the independence of the previous interval duration length, requires computational cost $O(N^2LT)$.

In addition, if we calculate likelihood of dynamical systems (i.e., Equation (2.15)) before the recursive calculation, the actual cost can be also reduced.

2.5 Verification of the Inference Algorithms

In this section, we verify the capability of the interval system and the validity of the proposed inference algorithms (i.e., the forward and Viterbi algorithms).

First, the parameters of an interval system were given manually, and interval sequences were generated by the system for test data. Then, the data was used as input of the forward and Viterbi algorithms.

To concentrate on the interval-based state transition in the interval system, we set the observation matrix $H = I$ (unit matrix) and the observation noise covariance $R = O$ (zero matrix). The number of discrete states was $N = 3$ and the dimensionality of the internal state space was $n = 3$. The range of the interval duration was $[l_{\min} = 1, l_{\max} = 30]$. We set the probability matrix of the discrete-state transition as $A_{12} = A_{23} = A_{31} = 1$ and 0 for all the other elements to generate loops such as $q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow q_1$. The initial distribution of the discrete state was 1 for q_1 and 0 for the remaining states q_2 and q_3.

The two dimensional distribution of the duration length $h^{(ij)}(l_k, l_{k-1})$ had {mean $(h_m^{(i)})$, variance $(h_v^{(i)})$} of {6, 5}, {12, 30} and {16, 50} for q_1, q_2, and q_3, respectively. The covariance $(h_c^{(ij)})$ between the pairs of $\{q_1, q_2\}, \{q_2, q_3\}$ and $\{q_3, q_1\}$ was 12, 35 and 15, respectively. These covariances were designed to generate interval sequences that the duration lengths of the intervals were monotonically increased in the sequence of q_1, q_2, and q_3.

The transition matrices, bias vectors, and initial mean vectors of the dynamical

systems were as follows:

$$F^{(1)} = \begin{bmatrix} 0.6 & -0.1 \\ -0.1 & 0.2 \end{bmatrix}, g^{(1)} = \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix}, x_{\text{init}}^{(1)} = \begin{bmatrix} 1.0 \\ -1.0 \end{bmatrix}$$

$$F^{(2)} = \begin{bmatrix} 0.3 & 0.0 \\ 0.0 & 0.6 \end{bmatrix}, g^{(2)} = \begin{bmatrix} -0.7 \\ 0.0 \end{bmatrix}, x_{\text{init}}^{(2)} = \begin{bmatrix} 0.3 \\ 1.0 \end{bmatrix}$$

$$F^{(3)} = \begin{bmatrix} 0.5 & 0.1 \\ -0.1 & 0.3 \end{bmatrix}, g^{(3)} = \begin{bmatrix} 0.6 \\ -0.6 \end{bmatrix}, x_{\text{init}}^{(3)} = \begin{bmatrix} -1.0 \\ 0.0 \end{bmatrix}$$

The process noise covariance matrices $Q^{(i)}$ $(i = 1,2,3)$ and the covariance matrices $V_{\text{init}}^{(i)}$ $(i = 1,2,3)$ of the initial state distribution were set to zero in the generation step, and was set to $0.001I$ (where I is a unit matrix) in the fitting step of intervals.

Figure 2.5(a) shows a generated interval sequence from the finite state automaton. The length of the sequence was $T = 100$. We see that the duration of the intervals increases monotonically in the sequence of q_1, q_2, and q_3 because we set positive correlation between the adjacent intervals (i.e., q_1 to q_2 and q_2 to q_3).

In parallel of this discrete-state transition, each dynamical system was activated by the discrete state, and generates a sequence of signal. Figure 2.5(b) shows a generated observation sequence. In this experiment, this sequence corresponds to the generated internal state sequence as a result of observation parameters $H = I$ and $R = O$. We see that the time-varying pattern of the observation changes based on the transition of the intervals.

To verify the algorithms of forward and Viterbi inference, we input a generated sequence shown in Figure 2.5(b) to the original interval system (i.e., the system that generated the input sequence). Figure 2.5 (c) shows the result of the forward inference. Each line denotes the following probabilities of the discrete states under the condition of observations from time 1 to t:

$$P(s_t = q_j | y_1^t) = \frac{\sum_\tau P(I_t =< q_j, \tau >, y_1^t)}{P(y_1^t)}$$

Figure 2.5 (d) shows the result of backtracked intervals after the Viterbi algorithm. We see that both of the forward and Viterbi inferences have found optimal interval sequences that are consistent with the original interval sequence in Figure 2.5 (a).

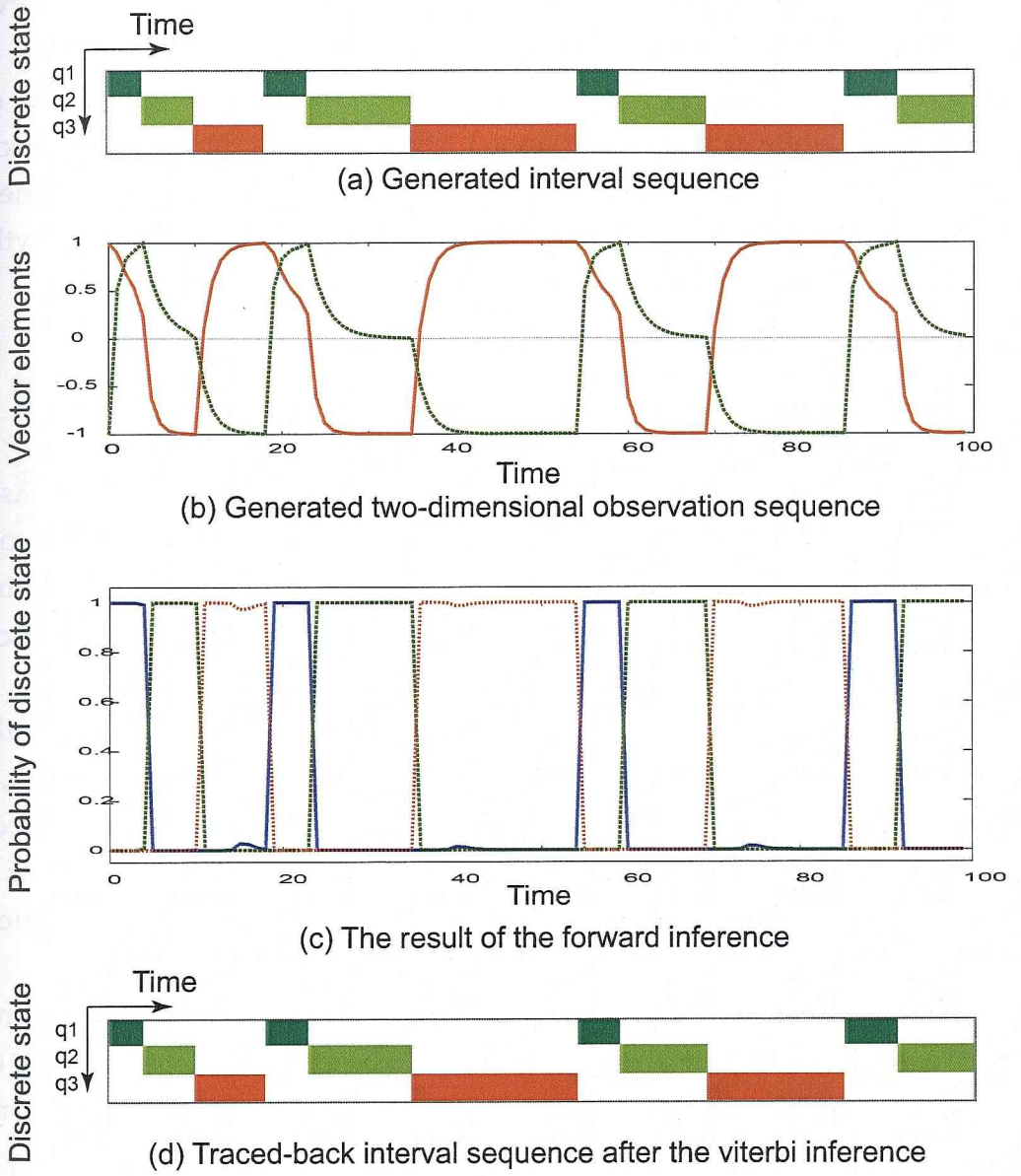


Figure 2.5: Verification of the forward and Viterbi algorithms. (a) A generated interval sequence by a finite state automaton. (b) A generated observation sequence by three dynamical systems using interval sequence of (a). The solid line denotes the first element; the dashed line denotes the second element. (c) The result of the forward inference using (b) as input. Each line represents probability $P(s_t = q_j|y_1^t)$ $(j = 1,2,3)$. (d) The result of the backtracked interval sequence after the Viterbi inference using (b) as input. We can see that the original interval sequence is obtained correctly.

To see the influence of parameters in the duration-length distributions, we generated interval sequences with zero covariance between adjacent intervals. Figure 2.6 shows an example of the generated sequence. We see that the first state sequence of q_1, q_2, and q_3 has non-monotonic changes of duration (i.e., q_3 is shorter than q_2, while q_2 is longer than q_1), which implies that the each discrete state duration is decided independently. Consequently, the correlation modeling with covariances between the adjacent intervals is necessary to represent rhythms and tempo as patterns of duration lengths.

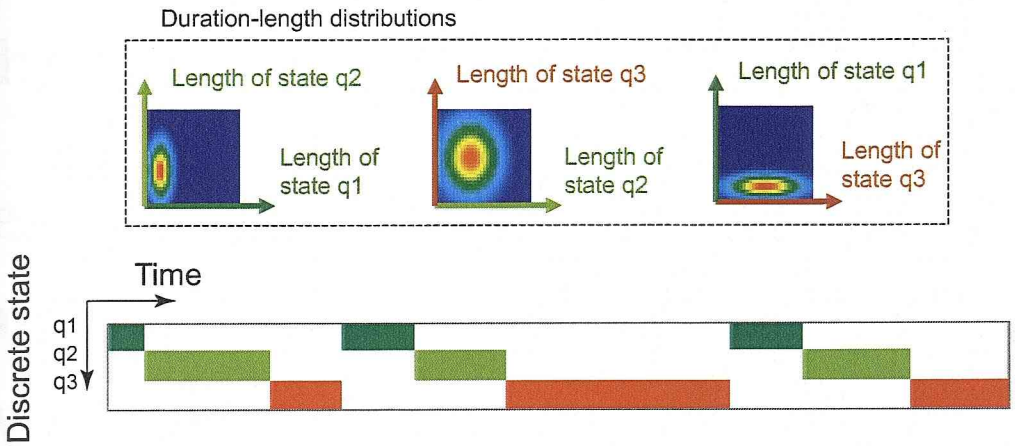
2.6 Discussion

In this chapter, we proposed a computational model, which we refer to as the interval-based hybrid dynamical system, that comprises a finite state automaton (discrete-event system) and multiple linear dynamical systems. Each linear dynamical system represents a dynamic primitive that corresponds to a discrete state of the automaton.

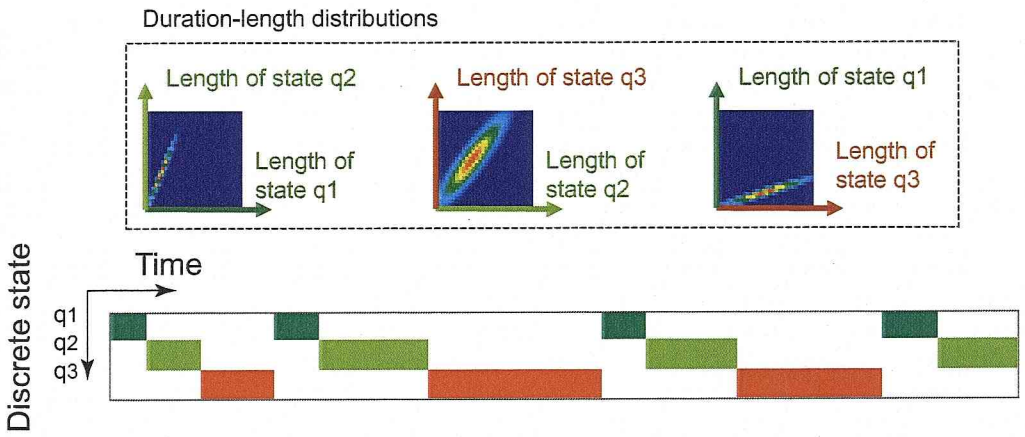
The key idea of integrating two systems is the use of temporal intervals in which each constituting linear dynamical system is activated. As a result, the temporal order of discrete state is mapped to the physical-time domain based on the duration lengths of intervals. Due to the duration-length modeling of discrete states, we successfully represent temporal patterns of discrete event such as rhythms based on the correlation of adjacent interval lengths.

In this chapter, we assumed that all the parameters of the interval system are given. We, however, require all the parameters to be estimated from training data in most of real problems. In Chapter 3, we introduce a learning method for the interval system.

Modeling relations among concurrent multiple streams (e.g., temporal relations of motions among facial parts) and relations among multimodal data (e.g., lip motion and speech data) is one of the important objectives for the interval-based representation. We try to analyze temporal structures among motion of facial parts in Chapter 4. A general modeling method of such a temporal structures, which we refer to as "timing structures", will be required to represent concurrent events. We will introduce a timing structure model in Chapter 5 using relation between intervals.



(a) Generated interval sequence using zero covariance distributions



(b) Generated interval sequence using non-zero covariance distributions

Figure 2.6: A generated interval sequence using zero covariance duration-length distributions (shown in (a)) for comparison to the interval sequence generated by using non-zero covariance distributions (shown in (b), which corresponds to Figure 2.5 (a)).

Chapter 3

Learning Method for the Interval-Based Hybrid Dynamical System

In this chapter, we propose a two-step learning method for the interval-based hybrid dynamical system (interval system). The method consists of (1) an agglomerative clustering process of dynamics, which estimates approximate parameters of each dynamical system, and (2) a refinement process of overall system parameters, which include parameters of the constituting finite state automaton and dynamical systems, based on the approximated expectation-maximization (EM) algorithm.

3.1 Difficulties in Identification of Hybrid Dynamical Systems

Let us assume that only a large amount of multivariate sequences is given as training data. Then, in the most of hybrid systems, the system identification process that estimates all the system parameters becomes difficult because of its paradoxical nature. That is, the identification of the subsystems requires partitioned and labeled training data; meanwhile, the segmentation and labeling processes of training data require an identified set of subsystems. Moreover, the number of the subsystems is also unknown in general. Therefore, the parameter estimation problem requires us to simultaneously estimate temporal partitioning of training data (i.e., segmentation and labeling) and the set of subsystems (i.e., the number

of the subsystems and their parameters). This problem also lies in the learning process of the interval system proposed in the previous chapter.

In the following paragraphs, we introduce relevant work as for the identification methods for various hybrid systems. We distinguish two cases when the number of subsystems is known or unknown.

3.1.1 In Case the Number of Subsystems is Known

The EM algorithm [DLR77] is one of the most common approaches to solve this kind of paradoxical problems when the number of subsystems is given. The algorithm estimates parameters based on the iterative calculation. In each step, the algorithm conducts model fitting to the training data using the model parameters that were updated in the previous step. Then, the parameters are updated based on the result of the current model fitting process.

Many of the hybrid system identification methods exploit the EM algorithm; for example, segmental models [ODK96], motion textures [LWS02], stochastic linear hybrid systems [BHJT04] (which follows the method in [LWS02]), and SLDSs [GH96, PRCM99, PRM00] applied this algorithm. A well-known identification technique for piecewise linear (PWL) models is introduced as k-mean like clustering [FMLM03]. This clustering technique can be regarded as an approximation (i.e., hard clustering version) of the EM algorithm, which is often referred to as soft clustering [DHS00].

Although the convergence of the EM algorithm is guaranteed, it strongly depends on the selection of initial parameters and often converges to a locally optimal solution, especially if the model has a large parameter space to search. As the alternative approach, an identification problem of PWL models can be recasted as a mixed integer programming (MIP), which find the globally optimal solution [RBL04, KHS$^+$04]. We can apply the method when the logical switching conditions between subsystems can be transformed into a set of inequalities; however, it is difficult to transform the dynamic switching conditions, which are modeled by an automaton. Another disadvantage of MIP lies in its computational complexity. The problem is well known to be NP-hard in the worst case [FMLM03, RBL04]. For these reasons, we exploit the EM algorithm rather than the MIP-based methods to identify the interval system.

3.1.2 In Case the Number of Subsystems is Unknown

Most of previous works in hybrid system identification assumed that the number of subsystems is given because the number often corresponds to that of manually defined operative conditions in the controlled object. In contrast, a set of dynamic primitives in human motion (e.g., primitive motion appeared in facial expressions) is undefined in most of cases. Whereas some heuristic sets of dynamic primitives are defined by human observation, they do not guarantee that the set is appropriate for man-machine interaction systems (e.g., automatic recognition of facial motion patterns). Hence, we should estimate the number of the subsystems (primitives) from a given training data set. The problem is often referred to as the cluster validation problem [DHS00] (see Subsection 3.3.5 for details).

In stochastic linear hybrid systems [LWS02, BHJT04], an online clustering based on a greedy algorithm is applied to determine the number of subsystems (i.e., linear dynamical systems) and initialize the EM algorithm [LWS02, BHJT04]. The drawback is that it depends on the order of data presentation, and is also sensitive to outliers in training data. Moreover, the algorithm requires deciding appropriate thresholds beforehand. As a result, the algorithm tends to return too many subsystems.

3.2 Two-Step Learning Method for the Hybrid Dynamical Systems

3.2.1 Parameters

The goal of the interval system identification is to estimate the parameters the following parameters from a large number of multivariate sequences:

- N : the number of dynamical systems (which corresponds to the number of discrete states)
- H, R : an observation matrix and an observation noise covariance matrix
- $F^{(i)}, g^{(i)}, Q^{(i)}$ $(i = 1, ..., N)$: transition matrices, bias vectors, and process noise covariance matrices for each linear dynamical system
- $x_{\text{init}}^{(i)}, V_{\text{init}}^{(i)}$ $(i = 1, ..., N)$: initial mean state vectors and initial state covariance matrices for each linear dynamical system

- $A_{ij}\,(i,j=1,...,N,i\neq j)$: a discrete-state transition matrix
- $\pi_i\,(i=1,...,N)$: an initial state distribution
- $h_m^{(i)}, h_v^{(i)}, h_c^{(i|j)}\,(i,j=1,...,N,i\neq j)$: parameters for two-dimensional duration distribution $h^{(i|j)}(l_k, l_{k-1})$ in adjacent intervals

We use the following notation to denote the set of parameters for convenience:

The parameter set of the overall system:

$$\Theta = \{\theta_i, A_{ij}, \pi_i, h_m^{(i)}, h_v^{(i)}, h_c^{(ij)} | i,j=1,...,N, i\neq j\}$$

The parameter set of constituting dynamical system $D_i (i=1,...,N)$**:**

$$\theta_i = \{F^{(i)}, g^{(i)}, Q^{(i)}, x_{\text{init}}^{(i)}, V_{\text{init}}^{(i)}\}$$

We assume that the observation matrix H and noise covariance matrix R are given. In case that the matrices are unknown, the subspace system identification techniques such as [OM95] can be applied by assuming a single transition matrix. Although the subspace identification technique has large sensitivity to noise, some research results show that this technique successfully identifies an observation matrix especially from visual feature sequences [DCWS03].

3.2.2 Two-Step Learning Method

As we described in Section 3.1, the EM algorithm-based parameter estimation method requires to solving two problems:

1. Initialization of the EM algorithm
2. Estimation of the number of subsystems

To overcome the problems, we propose a two-step learning method (see also Figure 3.1), which we describe in the following sections, that estimates parameters Θ of the interval system.

The key idea of the learning method is that we divide the estimation process into two steps: a clustering process of dynamical systems to estimate a set of required dynamical systems and a parameter refinement of the estimated dynamical systems. The parameters of automaton are also estimated in the second step.

Step 1 Clustering of Dynamical Systems: The first step is a clustering process that finds a set of dynamical systems: the number of the systems and their parameters. This step employs a set of typical sequences (i.e., a subset of the given training data set), and the sequences have been already mapped to the internal state space. Then, we apply an agglomerative hierarchical clustering technique, which we propose in this thesis, to the training data. This clustering technique estimate a set of dynamical systems (see Subsection 3.3 for details). After this process, we get the estimated number of dynamical systems N and approximate parameters of dynamical systems $\theta_i (i = 1, ..., N)$.

Step 2 Refinement of the Parameters: The second step is a refinement process of the system parameters based on the EM algorithm. The process is applied to all the given training data, whereas the clustering process is applied to a selected typical training set. Although the EM algorithm strongly depends on its initial parameters, the previous clustering step provides an initial parameter set that is relatively close to the optimum compared to a randomly selected parameter set. This algorithm also estimates the parameters of automaton such as a discrete-state transition probabilities and duration-length distributions associated with the state transition (see Subsection 3.4 for details). Finally, we get the estimated parameter set Θ of the overall interval system.

3.3 Hierarchical Clustering of Dynamical Systems

3.3.1 Overview of the Clustering Algorithm

Let us assume that a multivariate sequence $y_1^T \triangleq y_1, ..., y_T$ is given as a typical training data. We can consider a single training data without loss of generality because we do not estimate the order of transition of dynamical systems (i.e., discrete-state transition probabilities) in this clustering step. Then we simultaneously estimate a set of linear dynamical systems $\mathcal{D}$ (i.e., the number of linear dynamical system N and their parameters $\theta_1, ..., \theta_N$) with an interval set $\mathcal{I}$ (i.e., segmentation and labeling of the sequence), from the training sample y_1^T. The number of intervals K is also unknown.

We formulate the problem as the search of the linear dynamical system set $\mathcal{D}$

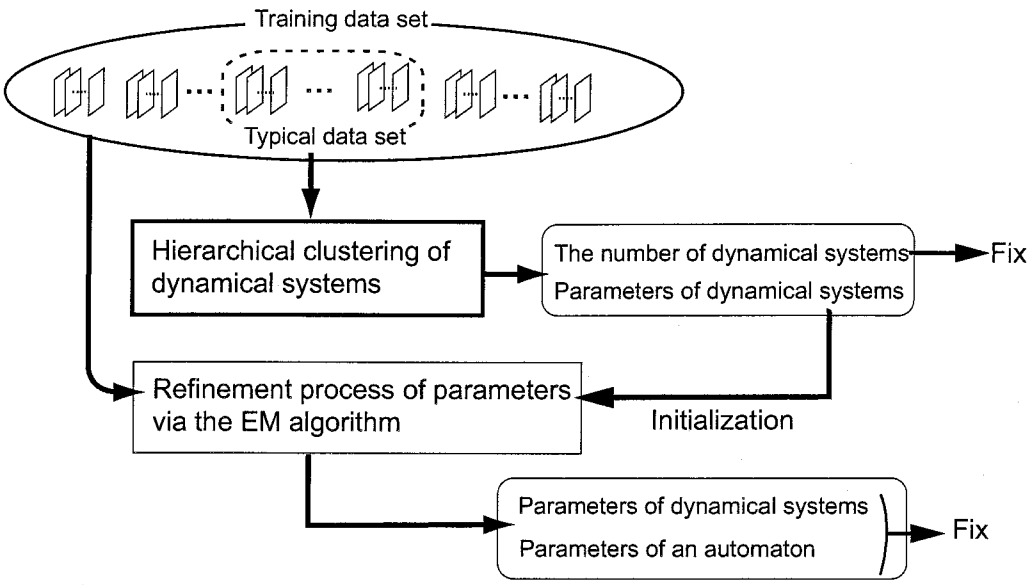


Figure 3.1: A two-step learning for the interval system.

and the interval set $\mathcal{I}$ that maximizes the overall likelihood with respect to the training data: $\mathcal{L} \triangleq P(y_1^T|\mathcal{I},\mathcal{D})$. Because the likelihood monotonically increases with an increase in the number of dynamical systems, we need to determine the right balance between the likelihood and the number N. A hierarchical clustering approach provides us an interface such as the history of model fitting errors in each merging steps to decide the number of dynamical systems (see Subsection 3.3.5 for details).

As we described in Subsection 3.2.1, we assume that the observation matrix H is given. That is, using the subspace system identification techniques such as [OM95] before the clustering step, we can simultaneously estimate observation matrix H and the corresponding internal state sequence $x_1^T \triangleq x_1, ..., x_T$ that can be mapped to the given observation sequence y_1^T by matrix H. In the following sections, we therefore focus on estimating parameters θ_i and the number of linear dynamical systems N from the given state sequence x_1^T.

The Hierarchical Clustering Algorithm of Dynamical Systems

The intuitive explanation of the hierarchical clustering algorithm is as follows:

1. Partition the training sequence into a group of very short sub-sequences and estimate a dynamical system that can model each sub-sequence respectively.

2. Compute the distance between each pair of estimated dynamical systems.

3. Merge the closest pair of dynamical systems; compute parameters of the merged dynamical system based on such sub-sequences that were modeled by the pair of dynamical systems to be merged.

4. Iterate the above agglomerating process until the closest distance between a pair of dynamical systems becomes greater than a pre-specified value.

After this process, we get the number of required dynamical systems N and approximate parameters of the dynamical systems. Note that the pre-specified value (i.e., threshold) required in step 4 determines the number of the estimated dynamical systems. It is however difficult to specify the value beforehand. Therefore, we first proceed with the algorithm until all the dynamical systems are agglomerated to a single dynamical system, and then estimate the number of the dynamical systems by evaluating the model fitting scores at each of the iteration steps.

Algorithm 5 shows the details of the clustering process. This algorithm requires initial partitioning of the training sequence, which we refer to as an initial interval set $\mathcal{I}_{\text{init}}$. We will discuss about the initialization in Subsection 3.3.2.

Algorithm 5 Agglomerative Hierarchical Clustering of Dynamical Systems.

for I_i in $\mathcal{I}_{\text{init}}$ **do**
  $D_i \leftarrow \mathsf{Identify}\,(I_i)$
  insert D_i to $\mathcal{D}$
end for
for all pair$(D_i,\ D_j)$ where $D_i,\ D_j \in \mathcal{D}$ **do**
  $\mathsf{Dist}\,(i,\ j) \leftarrow \mathsf{CalcDistance}\,(D_i,\ D_j)$
end for
while $N \geq 2$ **do**
  $(i^*,\ j^*) \leftarrow \arg\min_{(i,\ j)} \mathsf{Dist}\,(i,\ j)$
  $\mathcal{I}_{i^*} \leftarrow \mathsf{MergeIntervals}\,(\mathcal{I}_{i^*},\ \mathcal{I}_{j^*})$
  $D_{i^*} \leftarrow \mathsf{Identify}\,(\mathcal{I}_{i^*})$
  erase D_j^* from $\mathcal{D}$
  $N \leftarrow N - 1$
  for all pair$(D_i^*,\ D_j)$ where $D_j \in \mathcal{D}$ **do**
    $\mathsf{Dist}(i^*,\ j) \leftarrow \mathsf{CalcDistance}\,(D_{i^*},\ D_j)$
  end for
end while

In the first step of the algorithm, dynamical systems are identified (which is denoted by Identify in Algorithm 5) from each interval in the initial interval set $\mathcal{I}_{\text{init}}$ individually based on the identification method proposed in Subsection 3.3.3. $\mathcal{I}_i$ is the interval set that comprises intervals labeled by D_i.

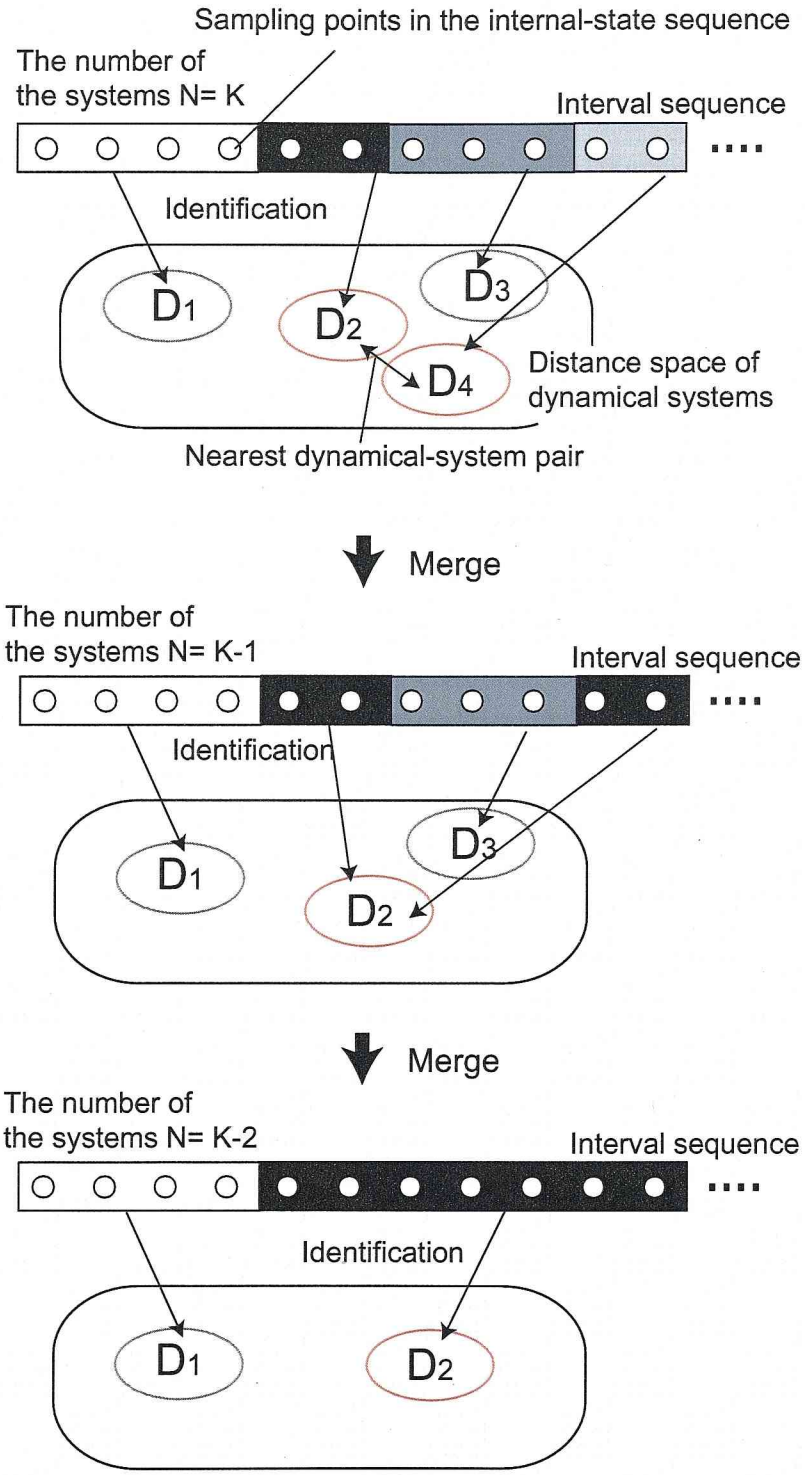


Figure 3.2: Hierarchical clustering of dynamical systems.

Then, we calculate the distances (denoted by CalcDistance) for all the dynamical system pairs based on the distance definition described in Subsection 3.3.4. After the distance calculation, the nearest dynamical systems are merged (which is denoted by MergeIntervals), and the parameter set of the merged system is estimated again. Here, two interval sets that belong to the nearest dynamical system pair are also merged in parallel. Figure 3.2 shows an example of the step in which the dynamical system D_2 and D_4 are merged.

Repeating the merging process in the previous paragraph, we get a single dynamical system in which all the dynamical systems are agglomerated.

3.3.2 Initialization of the Clustering Algorithm

The clustering algorithm proposed in the previous subsection iteratively merges intervals in the initial interval set $\mathcal{I}_{\text{init}}$. The final result of the clustering process is therefore affected by the estimation of set $\mathcal{I}_{\text{init}}$. In this section, we give some examples to determine $\mathcal{I}_{\text{init}}$.

Fixed-Length Segmentation

The simplest partitioning method of $\mathcal{I}_{\text{init}}$ is to divide the given training sequence into fixed length small intervals. However, this naive method is undesirable for real world problem such as signals from cameras and microphones because of its high computational cost. Let $|\mathcal{I}_{\text{init}}|$ be the number of initial intervals. Then, the actual computational cost of Algorithm 5 is proportional to $O(|\mathcal{I}_{\text{init}}|^2)$, which is due to the first distance calculation in the second block of Algorithm 5.

Zero-Velocity-Based Segmentation:

Now, we consider how to get the initial interval set that is appropriate for finding motion dynamics in human behavior. To reduce the computational cost, we first require $|\mathcal{I}_{\text{init}}|$ to be relatively smaller than the length of the training sequence. Second, we require that the initial intervals divide stationary pose from motion in the training sequence. This is useful for describing human motion based on simple primitives. Third, we require the intervals $I_i \in \mathcal{I}_{\text{init}}$ to be simple in the sense that the patterns in the initial intervals appear frequently in the training sequence.

To satisfy the conditions above, we exploit zero velocity of the signal. Let $\dot{x}_t$ be the velocity vector of given signal x_t $(t = 1, ..., T)$. If the Euclidean norm $||\dot{x}_t||$

becomes nearly zero (i.e., under the given threshold), we divide the signal at that temporal point. Then, if $||\dot{x}_t||^2$ becomes greater than the given threshold again, we also divide the signal at that point. Repeating this process, we get the initial intervals in which motion states and stationary states appear by turns.

Because x_t is a discrete-time signal, we use the following equation to calculate the norm of velocity $\dot{x}_t$:

$$||\dot{x}_t|| = \sqrt{\sum_{r=1}^{R} ||x_{t+r} - x_{t+r-1}||^2} \qquad (t = 1, ..., T - R), \tag{3.1}$$

where R is a smoothing parameter of velocity calculation.

3.3.3 Constrained System Identification Based on Eigenvalues

To identify the system parameters from only a small amount of training data, we have to use constraints for estimating desirable dynamics. In this chapter, we concentrate on extracting human motion primitives observed in such as facial motion, gaits, and gestures. Most of these motions are generated by the combination of intermittent muscle controls; therefore, constraints based on stability of dynamics might be suitable to find motion that converges to a certain state from an initial pose.

The key idea of estimating stable dynamics is to give constrains on the eigenvalues. As we see in Section 2.2.2, the dynamical system changes the state in a stable manner if all the eigenvalues are smaller than one. In the following paragraphs, we propose a novel system identification method that constrains all the eigenvalues of the system to be smaller than one. The method corresponds to Identify in Algorithm 5.

System Identification without Constraints

If the temporal range $[b, e]$ is represented by linear dynamical system D_i. Then, we can estimate the transition matrix $F^{(i)}$ and bias vector $g^{(i)}$ from the internal state sequence $x_b^{(i)}, .., x_e^{(i)}$ in the range $[b, e]$. This parameter estimation problem becomes a minimization problem of prediction errors.

If we have already estimated $F^{(i)}$ and $g^{(i)}$, we can predict a state vector at time

t from $x_{t-1}^{(i)}$ using Equation (2.1). We then get the prediction error vector:

$$\epsilon_t = x_t^{(i)} - (F^{(i)} x_{t-1}^{(i)} + g^{(i)}). \tag{3.2}$$

Thus, the summation for the squared norms of all the error vectors in the range $[b, e]$ becomes

$$\sum_{t=b+1}^{e} ||\epsilon_t||^2 = \sum_{t=b+1}^{e} ||x_t^{(i)} - (F^{(i)} x_{t-1}^{(i)} + g^{(i)})||^2. \tag{3.3}$$

Finally, we can estimate optimal $F^{(i)}$ and $g^{(i)}$ by solving the following least squares problem:

$$F^{(*i)}, g^{(*i)} = \arg\min_{F^{(i)}, g^{(i)}} \sum_{t=b+1}^{e} ||\epsilon_t||^2. \tag{3.4}$$

In the next paragraph, we show only the result of this estimation. As for details, refer to Appendix B.

Using notations

$$\begin{aligned} \hat{X}_0^{(i)} &= [x_b^{(i)} - m_0^{(i)}, ..., x_{e-1}^{(i)} - m_0^{(i)}], \\ \hat{X}_1^{(i)} &= [x_{b+1}^{(i)} - m_1^{(i)}, ..., x_e^{(i)} - m_1^{(i)}], \end{aligned}$$

we can calculate the estimated transition matrix and bias vector as follows:

$$F^{(i)*} = \hat{X}_1^{(i)} \hat{X}_0^{(i)\dagger}, \tag{3.5}$$

$$g^{(i)*} = m_1 - F^{(i)*} m_0, \tag{3.6}$$

where, m_0 and m_1 are mean vectors of columns in $X_0^{(i)}$ and $X_1^{(i)}$, respectively:

$$m_0^{(i)} \triangleq \frac{1}{l-1} \sum_{t=b}^{e-1} x_t^{(i)}, \quad m_1^{(i)} \triangleq \frac{1}{l-1} \sum_{t=b+1}^{e} x_t^{(i)}. \tag{3.7}$$

$X_0^{(i)\dagger}$ is a Moore-Penrose generalized inverse (pseudo inverse) of $X_0^{(i)}$. Inverse matrix $X^\dagger$ can be defined as:

$$X^\dagger = \lim_{\delta^2 \to 0} X^\top (X X^\top + \delta^2 I)^{-1} = \lim_{\delta^2 \to 0} (X^\top X + \delta^2 I)^{-1} X^\top. \tag{3.8}$$

See also [Alb72, Koh89] for details.

System Identification with an Eigenvalue Constraint

We now show the method to constrain the eigenvalues of transition matrix $F^{(i)}$. Using Equation (3.5) and Equation (3.8), we can get the following equation:

$$\begin{aligned} F^{(i)*} &= \lim_{\delta^2 \to 0} \hat{X}_1^{(i)} \hat{X}_0^{(i)\top} (\hat{X}_0^{(i)} \hat{X}_0^{(i)\top} + \delta^2 I)^{-1} \\ &= \lim_{\delta^2 \to 0} X_1^{(i)} \hat{X}_0^{(i)\top} (\hat{X}_0^{(i)} \hat{X}_0^{(i)\top} + \delta^2 I)^{-1}, \end{aligned} \tag{3.9}$$

where I is the unit matrix and δ is a nonzero real value. Here, we used $\hat{X}_1^{(i)} \hat{X}_0^{(i)\top} = X_1^{(i)} \hat{X}_0^{(i)\top}$ (see Appendix B).

For the constraint on the eigenvalues, we stop the limit in the Equation (3.9) before $\hat{X}_0^{(i)\top} (\hat{X}_0^{(i)} \hat{X}_0^{(i)\top} + \delta^2 I)^{-1}$ converges to its generalized inverse matrix. In other word, we use

$$F_{\delta^2}^{(i)} = X_1^{(i)} \hat{X}_0^{(i)\top} (\hat{X}_0^{(i)} \hat{X}_0^{(i)\top} + \delta^2 I)^{-1} \tag{3.10}$$

for the constrained transition matrix.

As we will see in the following paragraphs, we can determine the upper bound of eigenvalues in the given matrix from its elements based on Gershgorin's theorem (Appendix C). We can therefore make a constraint that all the eigenvalues to be smaller than one if we chose an appropriate nonzero value for δ, which controls the scale of elements in the matrix $F^{(i)*}$.

Moreover, this constraint method have the advantage of calculating the generalized inverse matrix in Equation (3.5) successfully even if the matrix $\hat{X}_0^{(i)} \hat{X}_0^{(i)\top}$ have zero eigenvalues (the matrix becomes singular), as we will see in the rest of this subsection.

Numerical Calculation for Searching δ

We now describe how to calculate δ that constrains the upper bound of eigenvalues. Here we omit the index i, which denotes the label of dynamical systems, to simplify the notation.

First, we decompose the matrix $\hat{X}_0$ based on the singular value decomposition (SVD) to facilitate the calculation of δ:

$$\hat{X}_0 = USV^\top, \tag{3.11}$$

where U is $n \times r$ and V is $(l-1) \times r$ matrix; here $r = \min(n, l-1)$. These matrices are column orthogonal; that is $U^\top U = I_r$ and $V^\top V = I_r$. The middle matrix $S = \mathrm{diag}[s_1, ..., s_r]$ is a $r \times r$ diagonal matrix; we assume that the diagonal elements (singular values) are sorted by descending order (i.e., $s_1 \geq s_2 \geq \cdots \geq s_r$). In addition, if $n \leq l-1$, then $r = n$ and U becomes an orthogonal square matrix that satisfies $U^\top U = UU^\top = I_n$; on the other hand, if $n \geq l-1$, then $r = l-1$ and V becomes an orthogonal square matrix that satisfies $V^\top V = VV^\top = I_{l-1}$. If the rank of matrix $\hat{X}_0$ becomes smaller than r, let the rank be r', the smallest $r - r'$ singular values become zero.

Using this decomposition, we can rewrite Equation (3.10) as

$$\begin{aligned} F_{\delta^2} &= X_1 V S U^\top \left((USV^\top)(VSU^\top) + \delta^2 I \right)^{-1} \\ &= (X_1 V) S U^\top U (S^2 + \delta^2 I)^{-1} U^\top \\ &= Z S (S^2 + \delta^2 I)^{-1} U^\top \\ &= \sum_{k=1}^{n} \frac{\sigma_k}{\sigma_k^2 + \delta^2} z_k u_k^\top, \end{aligned} \tag{3.12}$$

where we assumed all the singular values are nonzero, and replaced $X_1 V$ with $Z = [z_1, ..., z_n]$ $(z_i \in \mathbf{R}^r)$.

Suppose f_{rc} is an element in row r and column c of the transition matrix F. The upper bound of the eigenvalues $\mathcal{B}$ can be determined by the corollary of the Gershgorin's theorem (see Appendix C):

$$\mathcal{B} = \max_r \sum_{c=1}^{n} |f_{rc}|. \tag{3.13}$$

Here, we search the δ that satisfies $\mathcal{B} = 1$. Substituting Equation (3.12) into Equation (3.13) and set $\mathcal{B} = 1$, we obtain

$$\max_r \sum_{c=1}^{n} \left| \sum_{k=1}^{n} \frac{\sigma_k}{\sigma_k^2 + \delta^2} w_{rc}^{(k)} \right| = 1, \tag{3.14}$$

where $z_k u_k^\top = W^{(k)} = [w_{rc}^{(k)}]$. Solving this equation via an iterative numerical calculation, we can find the value of δ that constrains the eigenvalues to be smaller than one.

Since we need to take absolute values in the Equation (3.14), it is difficult to solve this equation using straight forward numerical analysis. We therefore use

the following relation to divide the optimization into two stages:

$$\sum_{c=1}^{n}\left|\sum_{k=1}^{n}\frac{\sigma_k}{\sigma_k^2+\delta^2}w_{rc}^{(k)}\right| \leq \sum_{c=1}^{n}\sum_{k=1}^{n}\frac{\sigma_k}{\sigma_k^2+\delta^2}|w_{rc}^{(k)}| = \sum_{k=1}^{n}\frac{\sigma_k p_{rk}}{\sigma_k^2+\delta^2}, \tag{3.15}$$

where $p_{rk} = \sum_{c=1}^{n}|w_{rc}^{(k)}|$.

Stage1. In the first stage, we solve the following Equation (3.16) with respect to δ^2 based on Newton's method. For the initial value of δ, the smallest singular value seems to work well in most cases.

$$\mathcal{B}_r'(\delta^2) \triangleq \sum_{k=1}^{n}\frac{\sigma_k p_{rk}}{\sigma_k^2+\delta^2} = 1 \quad (r=1,...,n). \tag{3.16}$$

Stage2. The result of δ^2 from the first stage becomes larger than the solution of the following equation:

$$\mathcal{B}_r(\delta^2) \triangleq \sum_{c=1}^{n}\left|\sum_{k=1}^{n}\frac{\sigma_k}{\sigma_k^2+\delta^2}w_{rc}^{(k)}\right| = 1 \quad (r=1,...,n). \tag{3.17}$$

In the second stage, we therefore search the solution of the above equation based on a naive search; that is, we make δ^2 smaller than the previous step at each iteration step until one of $\mathcal{B}_r(\delta^2)(r=1,...,n)$ exceeds 1. For the initialization, we use the maximum of δ^2, $\arg\max_{\delta^2}\mathcal{B}_r'(\delta^2)$, in the first stage.

Despite that we need to calculate $\mathcal{B}_r(\delta^2)$ for all the row of F_{δ^2}, in our experiments, we select only one row in the beginning of search algorithm (before the first stage) using a criterion such as $\arg\max_r p_{r0}$. This criterion work well empirically, and reduce the computational cost drastically. More appropriate criterion might be to use $\arg\max_r \mathcal{B}_r'$ based on the result of the first stage.

Advantage of Constrained Identification in Singular Case

From Equation (3.12), the decomposition of unconstrained transition matrix in Equation (3.5) becomes

$$F^* = \lim_{\delta^2\to 0} F_{\delta^2} = ZS^{-1}U^\top = \sum_{k=1}^{n}\frac{1}{\sigma_k}z_k u_k^\top. \tag{3.18}$$

We see that if one of the singular values becomes close to zero, every element of the estimated transition matrix get a large number (∞ if one of the singular value is zero). On the other hand, if we use the proposed constrained identification method, the function $1/\sigma_k$ is replaced by $\sigma_k/(\sigma_k^2 + \delta^2)$. Figure 3.3 shows the comparison of functions $1/\sigma_k$ and $\sigma_k/(\sigma_k^2 + \delta^2)$. We see that $\sigma_k/(\sigma_k^2 + \delta^2)$ is constrained to be in the desired range, which can be controlled by the value of δ^2; moreover, the value of $\sigma_k/(\sigma_k^2 + \delta^2)$ becomes zero when $\sigma_k = 0$. Thus, we can successfully estimate the transition matrix even if some singular values become zero.

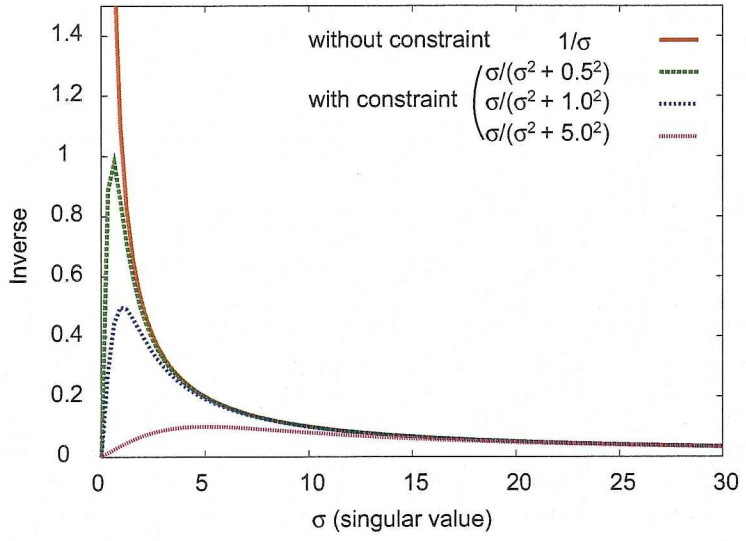


Figure 3.3: Functions $\frac{1}{s}$ and $\frac{s}{s^2+\delta^2}$ (when $\delta^2 = 0.5, 1.0, 5.0$).

3.3.4 Distance Definition between Dynamical Systems

In order to determine a pseudo distance between two linear dynamical systems, we first compare the following three approaches: (a) direct comparison of model parameters, (b) decreased likelihood of the overall system after the merging of two models [Bra95], and (c) distance measure of distributions (e.g., KL divergence [JR85]).

Approach (a) is not desirable particularly with regards to our bottom-up approach because a linear dynamical system has a large parameter set that often causes over-fitting. Hence, the distance in the parameter space does not always represent the dissimilarity among dynamical systems.

Approach (b), which is often exploited with stepwise-optimal clustering [DHS00], performs well in the ideal condition, but it requires the computational cost of likelihood scores for all the combination of the linear dynamical system pairs.

From our preliminary experiments, we observed that (c) provides stable dissimilarity measure for the hierarchical clustering algorithm with a realistic computational cost. Therefore, we define the distance between linear dynamical system D_i and D_j as an average of two asymmetric divergences:

$$Dist(D_i, D_j) = \frac{KL(D_i||D_j) + KL(D_j||D_i)}{2},$$

where each of the divergences is calculated as an approximation of KL divergence normalized by the interval length:

$$\begin{aligned} KL(D_i||D_j) &= \sum_{I_k \in \mathcal{I}_i} P(I_k|D_i) \log \left(\frac{P(y_{b_k}^{e_k}|D_i)}{P(y_{b_k}^{e_k}|D_j)} \right)^{\frac{1}{|I_k|}} \\ &\sim \frac{1}{|\mathcal{I}_i|} \sum_{I_k \in \mathcal{I}_i} \left\{ \log P(y_{b_k}^{e_k}|D_i) - \log P(y_{b_k}^{e_k}|D_j) \right\}, \end{aligned} \tag{3.19}$$

where $y_{b_k}, ..., y_{e_k}$ is an observation sequence partitioned by interval I_k. $|\mathcal{I}_i|$ is the summation of interval length in the interval set $\mathcal{I}_i$ that is labeled by a linear dynamical system D_i. Similarly, $|I_k| = e_k - b_k + 1$ is the length of interval I_k. We here approximated conditional probability of I_k to be $P(I_k|D_i) \sim |I_k|/|\mathcal{I}_i|$. We can use Equation (2.15) to calculate the likelihoods in Equation (3.19).

3.3.5 The Cluster Validation Problem

In real applications, it is an important problem to determine the appropriate number of dynamical systems (the number of clusters). The problem is often referred to as the cluster validation problem, which remains essentially unsolved. There are, however, several well-known criteria, which can be categorized into two types, to decide the number of clusters.

One of the types is defined based on the change of model fitting scores, such as log-likelihood scores and prediction errors (approximation of the log-likelihood scores), during the merging steps. If the score decreased rapidly, then the merging process is stopped [PH95]. In other words, it finds *knee* of the log-likelihood curve.

The other type is defined based on information theories, such as minimum description length and Akaike's information criterion. The information-theoretical criteria define the evaluation functions that consist of two terms: log-likelihood scores and the number of free parameters.

Although information-theoretical criteria work well in simple models, they tend to fail in evaluating right balance between the two terms, especially if the model becomes complex and has a large number of free parameters [LMZ98]. Because this problem also arises in our case, we use model fitting scores directly. First, we extract candidates for the numbers of the dynamical systems by finding peaks in the difference of model fitting errors between current and the previous steps. If the value exceeds a predefined threshold, then the number of dynamical systems in that step is added to the candidates. We consider that user should finally decide the appropriate number of the dynamical systems from the extracted candidates.

3.4 Parameter Refinement via the Expectation-Maximization Algorithm

This subsection describes a refinement process of the overall system parameter set Θ including the parameters of the automaton by exploiting the result of the clustering process. In the refinement process, the number of dynamical systems N is fixed (see Figure 3.1).

3.4.1 Overview of the Expectation-Maximization Algorithm

In order to refine the parameters, we apply an iterative estimation based on the EM algorithm. This algorithm starts from a given initial parameter set Θ_{init}, and repeats two steps: E (expectation) step and M (maximization) step. The E step fits the model to the given training samples based on the current model parameters, and calculates the expectation of log-likelihood with respect to all the given samples and to all the possible fitting instances. The M step updates the parameters to maximize the expectation of log-likelihood using the result of the statistics in the E step. After the iteration steps, the algorithm converges to the optimal parameters, and finds the result of the model fitting simultaneously.

Although the EM algorithm has been proved to converge, the obtained solutions are often trapped by local optima. This problem becomes significant espe-

cially if the size of the parameters increases. For a large parameter models, the algorithm therefore requires a parameter set that is relatively close to the optimum for the initialization.

The two-step learning method proposed in this chapter overcomes the initialization problem by exploiting the result of the clustering process applied for a typical training data set. From the viewpoint of the likelihood maximization, the clustering process estimates approximate parameters for the dynamical systems. To initialize remaining parameters such as the interval transition matrix $[A_{ij}]$ and interval duration distributions $h^{(ij)}(l_k, l_{k-1})$, we exploit Viterbi algorithm, which is introduced in Subsection 2.4.2, with some modification; that is, we set all the transition probabilities to be equal: $A_{ij} = 1/(N-1)$, and we also assume the interval duration $\hat{h}^{(ij)}(l_k, l_{k-1})$ to be uniform distributions in the range of $[l_{\min}, l_{\max}]$, where $i, j = 1, ..., N (i \neq j)$. As a result, Equation (2.29) of the Viterbi algorithm becomes

$$\delta_t(j, \tau) = P(y_{t-\tau+1}^t | s_t = q_j, l_t = \tau, f_t = 1) \max_{i(i \neq j), \tau_p} \{\delta_{t-\tau}(i, \tau_p)\}.$$

After the initialization above, parameter refinement iterations are applied to all the training data $\mathcal{Y} \triangleq \{y_1^{T_1}, ..., y_1^{T_M}\}$, where M is the number of the training sequences and $T_1, ..., T_M$ are the lengths of each sequence. Here, we approximate the EM algorithm because the original EM algorithm requires forward/backward inferences that often cause numerical underflow. We do not calculate the statistics for all the possible hidden variables (possible interval sequences), but use only a single interval sequence that gives the highest log-likelihood in each step. Thus, the total refinement process is described as follows:

1. Initialize $\Theta = \Theta_{\text{init}}$.

2. Repeat the following steps while the total likelihood changes greater than a given threshold ϵ:

   E step: search an optimal interval sequence, which is labeled by the discrete states, for all the training sequences based on the current parameter set Θ:

   $$\mathcal{I}^* = \arg\max_{\mathcal{I}} \log P(\mathcal{Y}, \mathcal{I} | \Theta),$$

   where $\mathcal{I}^*$ is the set of the searched interval sequences for all the training sequences.

M step: update the parameters based on the searched interval sequences in the E step:

$$\Theta = \arg\max_{\Theta'} \log P(\mathcal{Y}, \mathcal{I}^* | \Theta').$$

This algorithm is easily extended to use multiple interval sequences that give relatively high likelihood score rather than to use only an optimal interval sequence.

3.4.2 Parameter Estimation of the Automaton

The M step of the EM algorithm requires the estimation method of all the parameters in interval system Θ (see Subsection 3.2.1 for the details of the notation). Here, we show how to estimate the transition probabilities of discrete state, and the parameters of duration-length distributions. The parameter estimation of linear dynamical systems is similar to the method that we described in Subsection 3.3.3.

Discrete-State Transition Probabilities

Let $\mathcal{I}_1^*, ..., \mathcal{I}_M^*$ be the searched interval sequences for each training data $y_1^{T_1}, ..., y_1^{T_M}$ in the E step. Let s_k be the discrete state of interval $I_k^* \in \mathcal{I}_m^*$ $(m = 1, ..., M)$. The discrete-state transition probabilities A_{ij} $(i, j = 1, ..., N, i \neq j)$ are defined by Equation (2.19), we therefore estimate the probabilities by counting the frequency that the discrete state appears.

We first define some interval-label sets each of which specifies the subset in the interval set $\mathcal{I}_m^*$. Let $\mathcal{K}_m^{(i)}$ be the label set of intervals in which the discrete state is q_i; that is,

$$\mathcal{K}_m^{(i)} = \{k \,|\, s_k = q_i, I_k^* \in \mathcal{I}_m^*, k \geq 1\}. \tag{3.20}$$

Similarly, let $\mathcal{K}^{(i,j)}$ be the label set of intervals in which the discrete state is q_j and the discrete state of the previous interval is q_j; that is,

$$\mathcal{K}_m^{(i,j)} = \{k \,|\, s_k = q_j, s_{k-1} = q_i, I_k^* \in \mathcal{I}_m^*, k \geq 2\}. \tag{3.21}$$

Then, we can estimate the transition probability A_{ij} by the following equation:

$$P(s_k = q_j | s_{k-1} = q_i) = \frac{C(s_k = q_j, s_{k-1} = q_i)}{C(s_{k-1} = q_i)}, \tag{3.22}$$

where $C(s_k = q_j, s_{k-1} = q_i) = \sum_{m=1}^{M} |\mathcal{K}_m^{(i,j)}|$ is the frequency that states q_i and q_j

appears in the adjacent intervals in this order. $C(s_{k-1} = q_i) = \sum_{m=1}^{M} |\mathcal{K}_m^{(i)}|$ is the frequency that state q_i appeared in the interval sequence $\mathcal{I}^*$.

Initial Discrete-State Probabilities

The initial discrete-state probability π_i can be calculated by the similar method:

$$P(s_1 = q_i) = \frac{C(s_1 = q_i)}{M}, \tag{3.23}$$

where $C(s_1 = q_i)$ is the frequency that state q_i appears in the first interval of the searched interval sequences $\mathcal{I}_1^*, ..., \mathcal{I}_M^*$.

Parameters of Duration-Length Distributions

Let l_k be the duration length of interval $I_k^* \in \mathcal{I}_m^* (m = 1, ..., M)$. As we described in Subsection 2.3.1, we assume two-dimensional Gaussian functions for duration-length distributions (see also Equation (2.17)):

$$P(l_k, l_{k-1} | s_k = q_j, s_{k-1} = q_i) = \mathcal{N}\left(h_m^{(i)}, \begin{bmatrix} h_v^{(i)} & h_c^{(ij)} \\ h_c^{(ij)} & h_v^{(j)} \end{bmatrix}\right) \tag{3.24}$$

We can estimate the parameters of the Gaussian distribution in Equation (3.24) by the following equations:

$$\begin{aligned}
h_m^{(i)} &= \frac{1}{\sum_{m=1}^{M} |\mathcal{K}_m^{(i)}|} \sum_{m=1}^{M} \sum_{k \in \mathcal{K}_m^{(i)}} l_k \qquad (i = 1, ..., N) \\
h_c^{(i)} &= \frac{1}{\sum_{m=1}^{M} |\mathcal{K}_m^{(i)}|} \sum_{m=1}^{M} \sum_{k \in \mathcal{K}_m^{(i)}} (l_k - h_m^{(i)})^2 \qquad (i = 1, ..., N) \\
h_v^{(ij)} &= \frac{1}{\sum_{m=1}^{M} |\mathcal{K}_m^{(ij)}|} \sum_{m=1}^{M} \sum_{k \in \mathcal{K}_m^{(ij)}} (l_k - h_m^{(j)})(l_{k-1} - h_m^{(i)}) \qquad (i, j = 1, ..., N, i \neq j)
\end{aligned}$$

3.5 Experiments

To evaluate the proposed parameter estimation methods, we first used simulated sequences for training data because it provides the ground truth of the estimated parameters. The first experiments shown in the Subsection 3.5.1 is for the verification of the clustering algorithm including the determination criterion of the

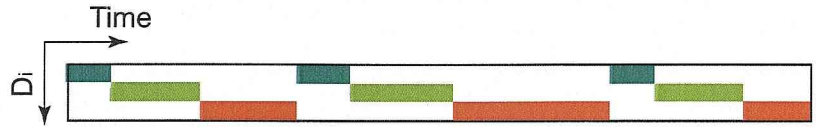
number of clusters (linear dynamical systems), and the second experiments in Subsection 3.5.2 is for the evaluation of the overall two-step learning method. In Subsection 3.5.3, we see how the proposed method is applicable to handle real data.

3.5.1 Evaluation of the Clustering Algorithm Using Simulated Data

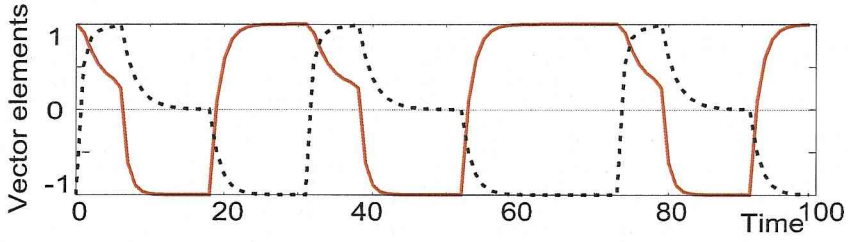
A multivariate sequence with length $T = 100$ was generated from the system that had the same parameters as the system in Section 2.5. In this sequence, three dynamic primitives were appeared in the temporal order of $D_1 \rightarrow D_2 \rightarrow D_3$, which repeats three cycles. The duration lengths of the discrete state are generated randomly based on the parameters of the duration-length distribution. Figure 3.4 (a) and (b) shows an example of the generated interval sequence and the two-dimensional observation sequence, respectively.

The hierarchical clustering algorithm proposed in Section 3.3 was applied to the sequence. Figure 3.4 (c) shows the overall model fitting errors between the original sequence Y and generated sequences $Y^{\mathrm{gen}}(N)$ by the extracted N dynamical systems. The error in each merge iteration step was calculated by the Euclidean norm: $Err(N) = ||Y - Y^{\mathrm{gen}}(N)|| = \sqrt{\sum_{t=1}^{T} ||y_t - y^{\mathrm{gen}}(N)_t||^2}$, which is the approximation of the overall log-likelihood score. The segmentation result of each merging step is shown in Figure 3.5. We see that the segmentation results from $N = 6$ to $N = 2$ have interval patterns that repeats three times, which correspond to the cycle of transition between the linear dynamical systems. Especially, the segmentation result of $N = 3$ exactly corresponds to the interval sequence in Figure 3.4 (a), which was used to generate the input sequence in Figure 3.4 (b).

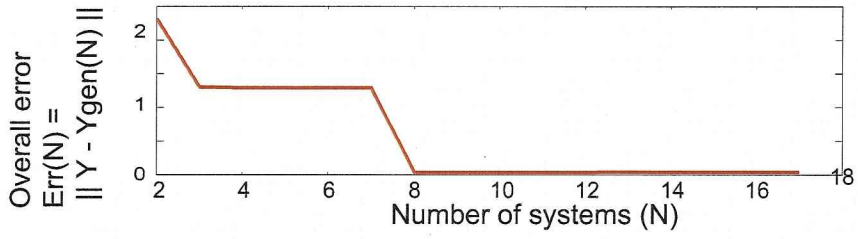
As we discussed in Subsection 3.3.5, we use the model fitting errors directly to extract candidates for the number of the dynamical systems. Figure 3.4 (d) shows the difference of the overall prediction error between current and previous steps $Err(N-1) - Err(N)$. We can find two peaks in $N = 6$ and $N = 3$ from this figure, where $N = 3$ is the ground truth and $N = 6$ is the largest number of dynamical systems in which the segmentation result has a cyclic pattern.



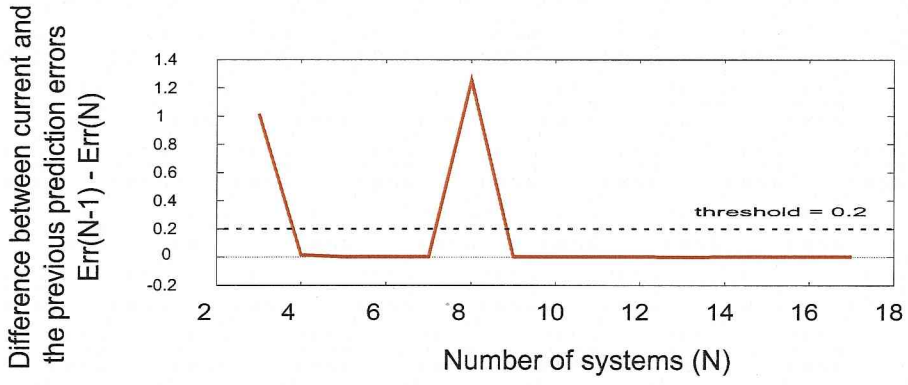
(a) Generated interval sequence of three dynamical systems



(b) Two-dimensional observation sequence generated from (a) (solid: the first element, dashed: the second element)



(c) Errors between the original and generated sequences



(d) Errors between the original and generated sequences

Figure 3.4: The Validation of the Clustering Algorithm (three-cycle input data).

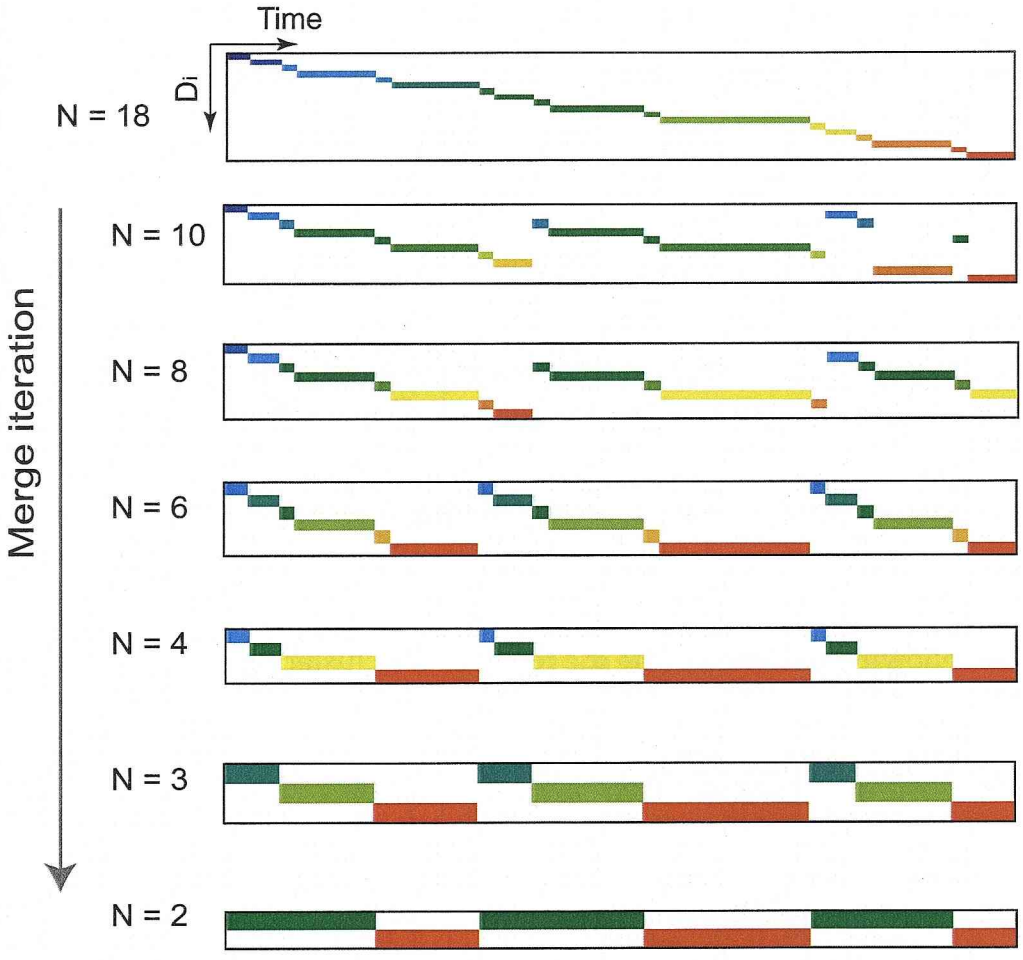


Figure 3.5: Segmentation result of each step in the clustering process.

3.5.2 Evaluation of the Refinement Process Using Simulated Data

[Step1] Clustering

Ten multivariate sequences were generated from the system. The parameters were same as the system in the previous experiments, except that the state transition probability A_{31} was set to zero to generate non-cyclic sequences.

In the sequences, three dynamic primitives were appeared in the temporal order of $D_1 \rightarrow D_2 \rightarrow D_3$, and only the length of the duration varied. Figure 3.6 (a) and (b) shows an example of the generated interval sequence and the two-dimensional observation sequence, respectively.

The proposed clustering process was evaluated by the sequence in Figure 3.6 (b). First, the initial interval set was determined by zero-crossing points of the first-order difference as we described in Subsection 3.3.2. Then, the initial dynamical systems were identified from the interval set. The number of the initial dynamical systems was $N = 6$. Figure 3.6 (c) shows the obtained interval sequences during the clustering process. The number of the dynamical systems was reduced from $N = 6$ to $N = 2$ by the iterative merging process of nearest dynamical system pairs. In each iteration step, two interval sets that belong to the nearest two dynamical systems were also merged.

The following parameters are the result of the clustering algorithm:

$$F^{(1)} = \begin{bmatrix} 0.01 & -0.14 \\ -0.10 & 0.20 \end{bmatrix}, g^{(1)} = \begin{bmatrix} 0.75 \\ 0.74 \end{bmatrix}, x^{(1)}_{\text{init}} = \begin{bmatrix} 1.0 \\ -1.0 \end{bmatrix}$$

$$F^{(2)} = \begin{bmatrix} 0.86 & 0.30 \\ -0.21 & -0.06 \end{bmatrix}, g^{(2)} = \begin{bmatrix} -0.30 \\ 1.12 \end{bmatrix}, x^{(2)}_{\text{init}} = \begin{bmatrix} 0.53 \\ 0.92 \end{bmatrix}$$

$$F^{(3)} = \begin{bmatrix} 0.49 & 0.09 \\ -0.11 & 0.75 \end{bmatrix}, g^{(3)} = \begin{bmatrix} 0.15 \\ -0.16 \end{bmatrix}, x^{(3)}_{\text{init}} = \begin{bmatrix} -0.63 \\ 0.60 \end{bmatrix}$$

[Step2] EM Algorithm

For the evaluation of the refinement process, we used the extracted dynamical systems in the clustering process. We manually selected $N = 3$ for the number of dynamical systems, which corresponds to the number of the original system that generated the typical sequence. Additional nine sequences were generated for

the training data, and all the generated sequences including the typical sequence were used for the input of the EM algorithm.

The algorithm was initialized by the parameters found in the clustering process. Figure 3.7 (a) shows the searched interval sequences in the E step of each iteration step. We see that the partitioning of the intervals gradually converges to almost the same partitions as the original interval sequence shown in Figure 3.6 (a). The solid line in Figure 3.7 (b) shows the change of the overall log-likelihood of the system. We see that the algorithm almost converged at the ninth iteration. The dashed line in Figure 3.7 (b) shows the change of the overall log-likelihood when the duration distribution function $\hat{h}^{(ij)}$ was set to be uniform and the adjacent intervals was modeled to have no correlation. We see that the algorithm converges to a local optimum in this case.

The following parameters are the refined result via EM algorithms:

$$F^{(1)} = \begin{bmatrix} 0.60 & -0.10 \\ -0.10 & 0.20 \end{bmatrix}, g^{(1)} = \begin{bmatrix} 0.20 \\ 0.82 \end{bmatrix}, x_{\text{init}}^{(1)} = \begin{bmatrix} 1.00 \\ -1.00 \end{bmatrix}$$

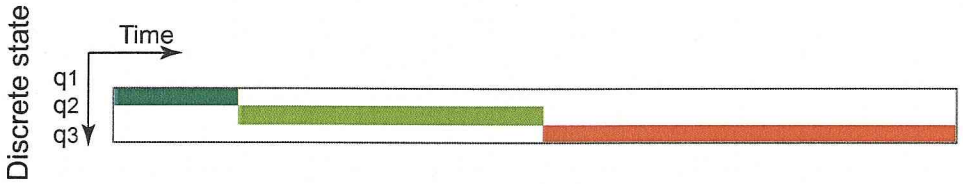
$$F^{(2)} = \begin{bmatrix} 0.32 & 0.02 \\ 0.06 & 0.52 \end{bmatrix}, g^{(2)} = \begin{bmatrix} -0.68 \\ 0.07 \end{bmatrix}, x_{\text{init}}^{(2)} = \begin{bmatrix} 0.25 \\ 1.00 \end{bmatrix}$$

$$F^{(3)} = \begin{bmatrix} 0.49 & 0.09 \\ -0.10 & 0.29 \end{bmatrix}, g^{(3)} = \begin{bmatrix} 0.60 \\ -0.60 \end{bmatrix}, x_{\text{init}}^{(3)} = \begin{bmatrix} 0.1 \\ -0.5 \end{bmatrix}.$$

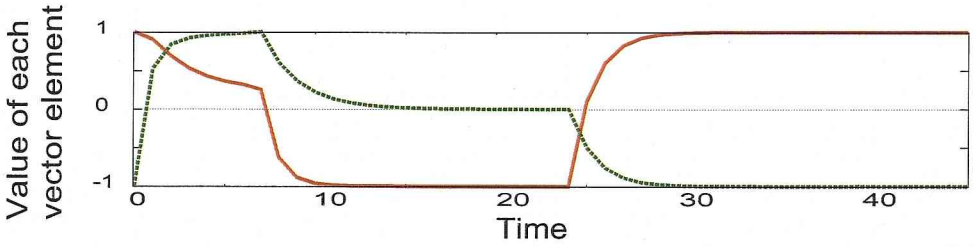
Consequently, the results of the clustering process become inaccurate if inappropriate sequences are selected for the typical training data. In spite of the inaccurate parameter estimation in the clustering process, the evaluation shows that the refinement process applied to all the training data recovers from the initial error. Especially, the meta-level features, such as modeling of duration lengths of intervals and relations between intervals, seem to work as constraints to the partitioning process in the EM algorithm.

3.5.3 Evaluation on Real Data

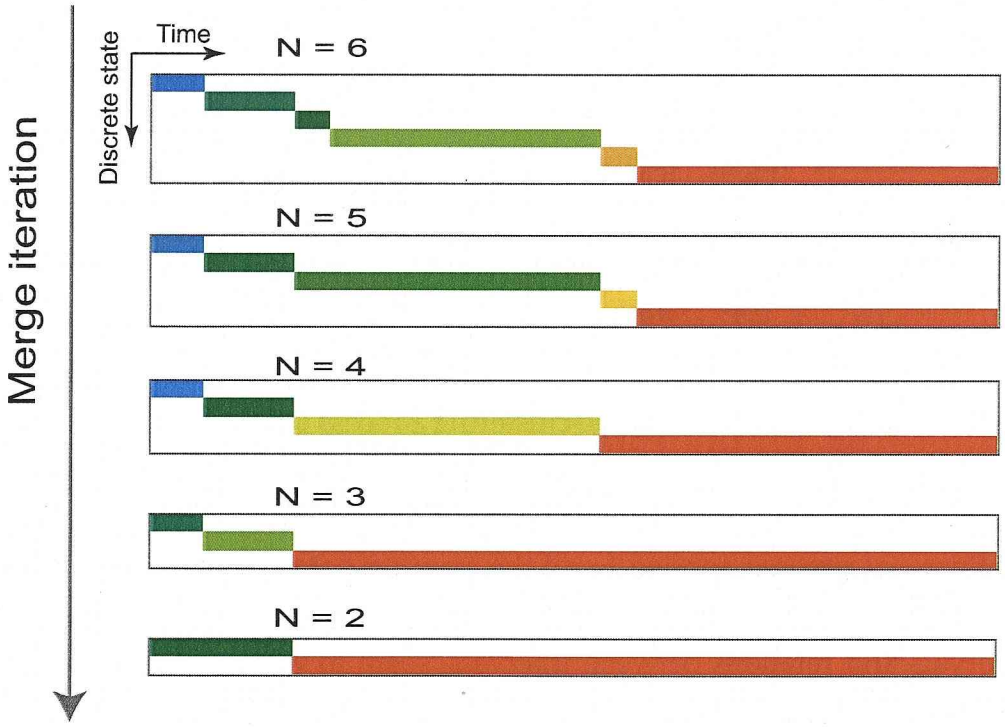
To evaluate the capability of the proposed method for real applications, we applied the clustering method to captured video data. A frontal facial image sequence was captured by 60fps camera during a subject was smiling four times. Facial feature points were tracked by the active appearance model [CET98, SEL03], and eight feature points around the right eye were extracted. The length



(a) Interval sequence generated for the typical training data

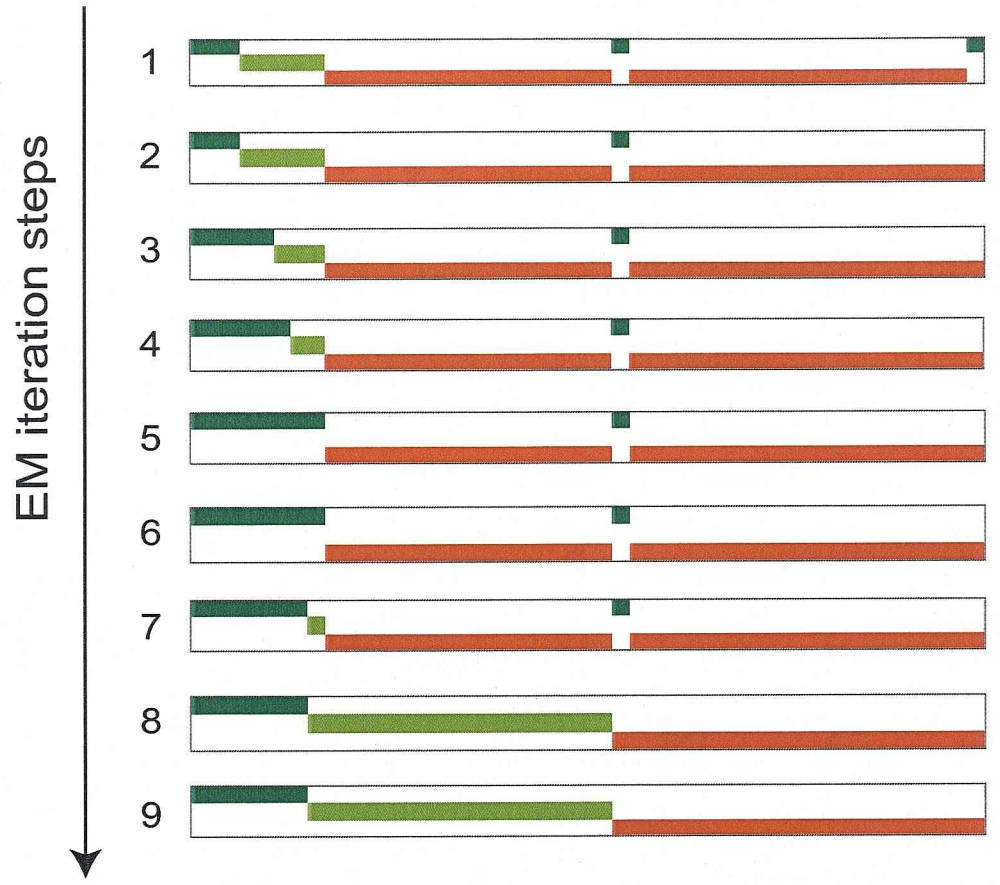


(b) Two-dimensional observation sequence selected for the typical data (solid: the first element, dashed: the second element)

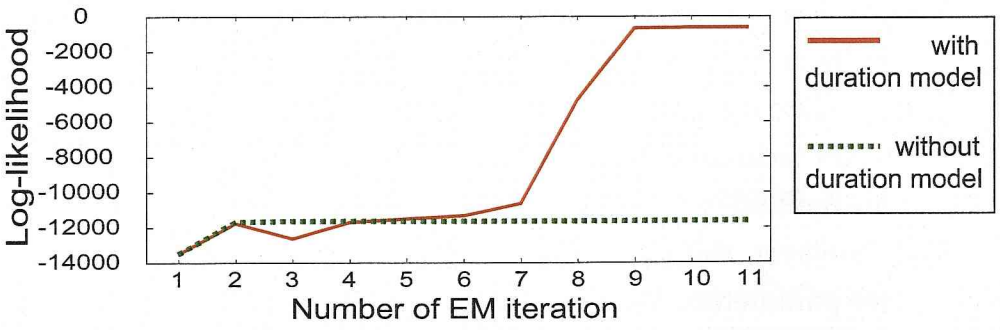


(c) Interval sequences partitioned by the clustered dynamical systems

Figure 3.6: The result of the clustering process applied to the typical training data shown in (b). The process was initialized by $N = 6$ and reduced the number of the dynamical systems (discrete states) by merging the nearest dynamical system pairs in each iteration steps.



(a) Searched interval sequence in each E step during the iteration



(b) The change of the overall log likelihood during the EM iteration

Figure 3.7: The result of the refinement process via the EM algorithm using all the training data.

of the sequence was $T = 1000$.

We applied the clustering method to the obtained 16-dimensional vector sequence that comprised x- and y-coordinates of the feature points (both coordinate coefficients were plotted together in Figure 3.8(a)). Figure 3.8(b) shows the overall model fitting errors between the original sequence Y and generated sequences $Y^{\mathrm{gen}}(N)$ by the extracted N dynamical systems. The candidates of the number of the dynamical systems were determined as $N = 3$ and $N = 6$ by extracting the steps in which the difference $Err(N-1) - Err(N)$ exceeded given threshold (we used 0.01 in this experiment). The difference $Err(N-1) - Err(N)$ is shown in Figure 3.8(c).

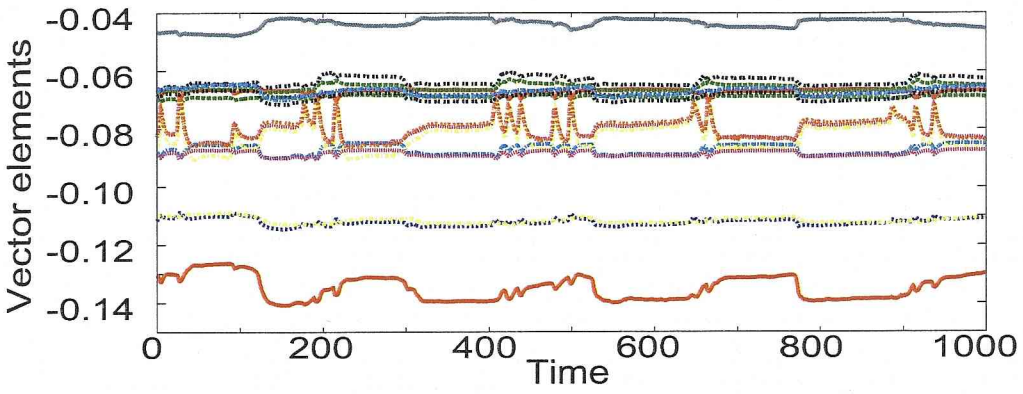
Figure 3.9(a) shows the intervals extracted from the clustering process when the number of dynamical systems was $N = 6$. Figure 3.9(b) shows the generated sequence from the extracted six dynamical systems, where each dynamical system was activated based on the partitioned intervals in Figure 3.9(a). The dominant dynamical systems D_3 and D_4 correspond to the intervals in which the eye had been remain closed and open, respectively. The other dynamical systems such as D_5 correspond to the eye blink motion.

Consequently, the history of model fitting errors during the clustering process helps us to decide the appropriate number of dynamical systems.

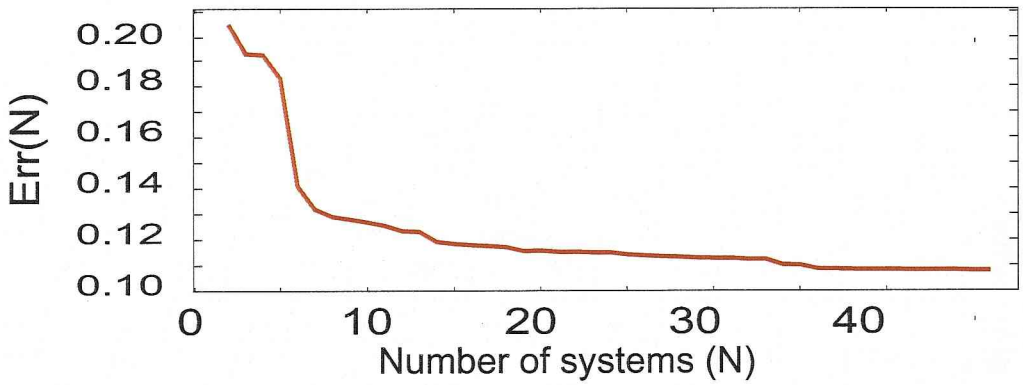
3.6 Discussion

In this chapter, we proposed a two-step learning algorithm to identify the interval-based hybrid dynamical system, which we introduced in the previous chapter. Especially, the hierarchical clustering of dynamical systems provides stable estimation technique of the number of dynamical systems and their parameters. This method can be regarded as one of model-based clustering methods [ZG03], however, the models are linear dynamical systems, which have a number of free parameters. We therefore proposed the constrained identification method of linear dynamical systems based on eigenvalues, which estimates stable dynamics from a small sample set of time-varying signals.

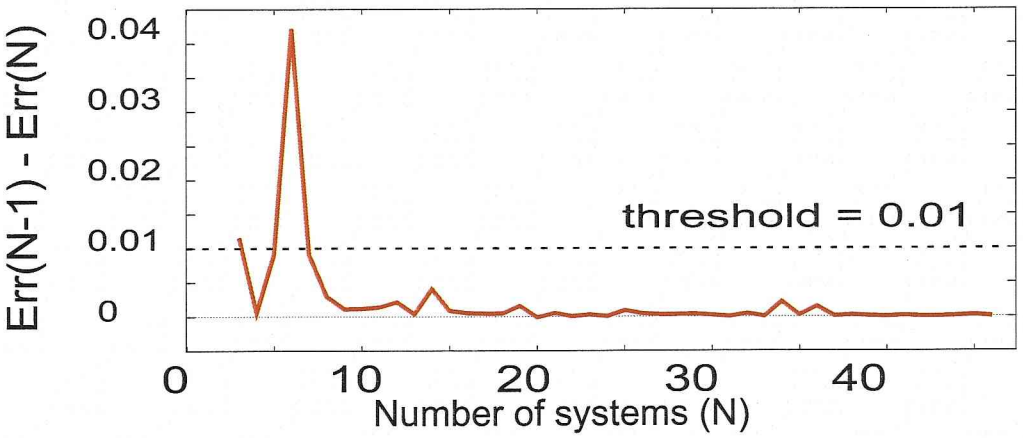
The experimental results on the simulated and real data show that the proposed parameter estimation method successfully finds a set of dynamical systems that is embedded in the training data and the transition probabilities between the dynamics with a modeling of adjacent interval duration lengths.



(a) Tracked feature points around the right eye

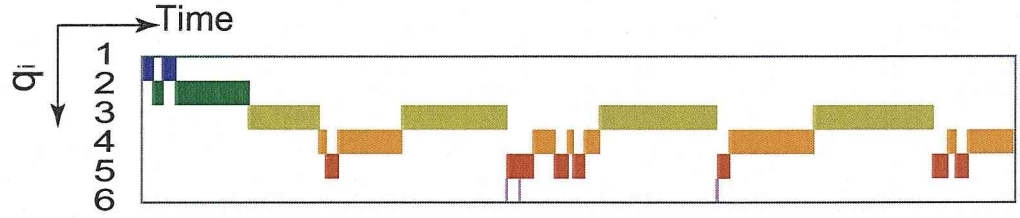


(b) Errors between original and generated sequences

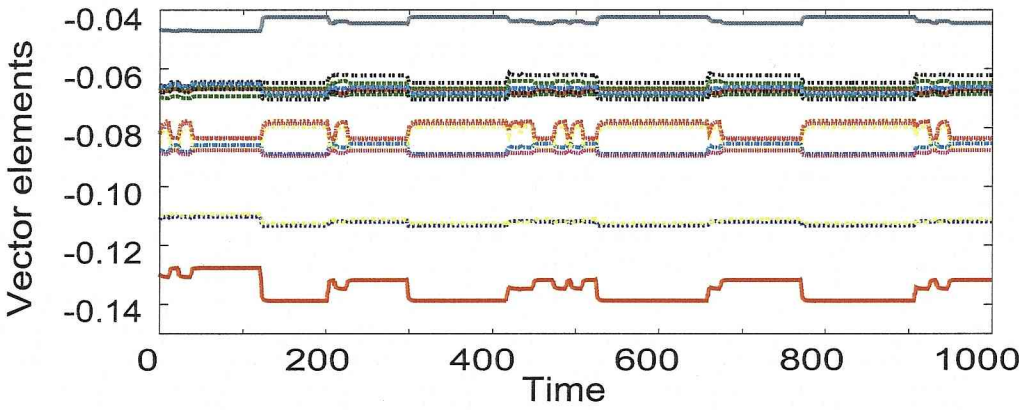


(c) Differences between current and the previous errors

Figure 3.8: Clustering results of the feature sequence around the right eye during a subject was smiling four times.



(a) Intervals partitioned by the dynamical systems



(b) Generated sequences from the clustered dynamical systems

Figure 3.9: Partitioned intervals during clustering and generated sequences from the clustered dynamical systems ($N = 6$).

Chapter 4

Analysis of Timing Structures in Multipart Motion of Facial Expression

In this chapter, we see how the interval-based hybrid dynamical system (interval system) can be applied to describe and analyze structured dynamic events. As we have shown in the preceding chapters, the interval system has the ability to describe dynamic events based on interval-based representation. We now apply the system to represent complex motion appeared in each facial part independently, and to analyze the dynamic structures of facial expression based on the temporal differences among beginning and ending time points of primitive motion.

4.1 Timing Structures in Facial Expression

4.1.1 Introduction

Facial expression plays an important role in our communication; for instance, it can nonverbally express emotions and intentions to others. Much progress has been made to build computer systems that recognize facial expression for human interfaces. However, these systems have problems that they don't use enough dynamic information in recognition, and the classification of facial expression relies on a fundamental category based on emotions (happiness, surprise, fear, anger, disgust, and sadness) [EP97].

Many systems developed so far describe facial expression based on "action units" (AUs) of the Facial Action Coding System (FACS) proposed by Ekman and

Friesen [EF75, TKC01]. An AU is defined as the smallest unit of facial movement that is anatomically independent and visually distinctive. FACS is a method for describing facial expression on the basis of the combination of AUs. FACS, however, has a major weakness: there is no time component of the description [EP97]. Moreover, there may be facial motion that AUs cannot express because AUs are heuristic motion patterns classified by human. It is also important to decide what categories of facial expression are appropriate as the outputs of facial recognition. Most previous systems categorize facial expression into one of six basic categories.

In human communication, however, facial expression is classified into one of the more fine-grained categories by subtle dynamic changes that are observed in facial components: the variety of changes and the timing of changes. This is because human facial expression is made of two mechanisms: (1) emotional expression produced by spontaneous muscular action, and (2) intentional display to convey some intention to others. To recognize the details of human emotion and intention, we believe that the analysis of the dynamic structure in facial expression is indispensable.

To realize such systems, we first assume the following points:

- Dynamic movement of each facial component (facial part) yields changes of facial expression.
- Movement of facial parts is expressed based on temporal intervals.

Based on the assumptions above, we define each interval as a temporal range that is expressed by a simple motion, where the intervals have beginning times, ending times, and labels of motion patterns, *modes*, as attributes.

We then provide a framework for recognizing facial expression in detail based on *timing structures*, which are defined as temporal relations among the beginning and ending times of multiple intervals. To extract the timing structures, we propose a novel facial expression representation, which we call a *facial score*. The score is similar to a musical score, which describes the timing of notes in music. Using the score, we can describe facial expressions as spatio-temporal combination of the intervals.

Whereas AUs are suitable motion units to distinguish emotional facial expression, they sometimes do not preserve sufficient dynamic information (e.g., time-varying patterns) of facial actions. Here, we take another approach; that is, we determine a set of modes from statistical analysis and describe facial actions based

on generative models. This approach extracts modes that have enough dynamic information from the viewpoint of pattern generation, and provides a unified framework that can be used not only for facial expression analysis but for facial expression generation.

As for the generative models, we utilize the interval systems proposed in Chapter 2. As for the mode determination, we exploit a bottom-up learning method that we proposed in Chapter 3. In this method, each mode is modeled by a linear dynamical system that has an ability to generate simple patterns, and the modes are extracted from clustering analysis that we described in Section 3.3 (see Subsection 4.2.3 for details).

In summary, the facial score is that it enables us to

- Describe timing structures in faces based on temporal intervals;
- Use motion primitives (i.e., modes) extracted from training data in a bottom-up manner.

Facial Expression Generation and Recognition Using the Facial Scores

Figure 4.1 depicts the overall flow of facial expression generation and recognition using the facial score (the top right of Figure 4.1).

Facial action generation. Once a facial score is obtained, we can activate interval systems to generate facial expression video just like as playing music according to a musical sore (down arrow at the right column in Figure 4.1).

Facial expression recognition. The flow of the facial expression recognition is as follows:

1. We first extract a series of feature vectors that characterize facial expression from a sequence of facial images (right arrow at the bottom in Figure 4.1).
2. We then partition the series of feature vectors and extract the modes simultaneously to obtain a facial score using interval systems and their learning method (up arrow at the right column in Figure 4.1).
3. Finaly, we extract timing structures, which contribute to recognition of the facial expression, from the facial score (left arrow at the top in Figure 4.1).

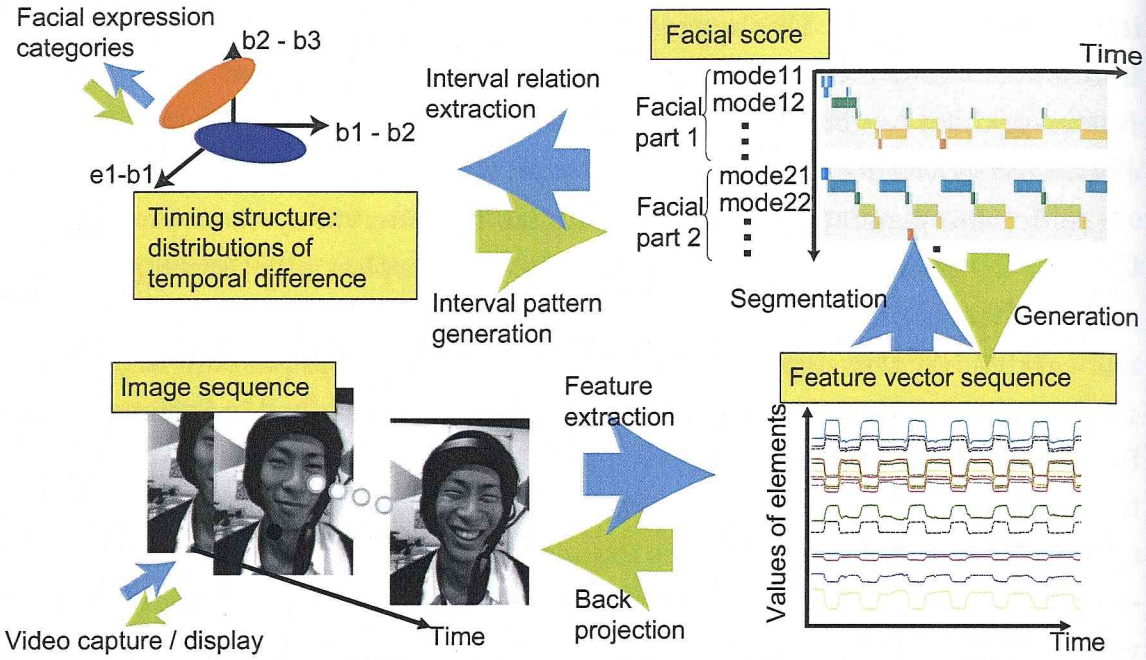


Figure 4.1: Flow of facial expression generation and recognition using the facial score.

The automation of the above process provides applications of learning, generating, and recognizing facial expression in detail using computers. The goal in this thesis is to propose a method for automatically obtaining the facial score and to evaluate the effectiveness of the facial score for facial expression recognition. We compare the timing structure of intentional smiles with that of spontaneous smiles for the evaluation; in human communication it makes sense to make a distinction between the two smiles, but most previous computer systems have classified these smiles into the same category.

In the next subsection, we describe some related work that studies dynamic properties of facial expression. In Section 4.2, we introduce facial scores as a description of timing structures in faces, and describe a method to obtain facial score. In Section 4.3, we describe a method to represent and extract timing structures from a facial score. In Section 4.4, we evaluate the effectiveness of the facial scores. In this evaluation, we first obtain facial scores automatically from captured real data including two expression categories: intentional and spontaneous smiles. Then, we examine the effectiveness of timing structure to separate these two categories of smiles. Finally, in Section 4.5 we discuss the advantage and disadvantage of the proposed representation.

4.1.2 Related Work

In psychological experiments, evaluation by playing back facial expressions on videotape to subjects has suggested the following knowledge of dynamic aspects of facial movement. Bassili video-recorded the face that was covered with black makeup and numerous white spots, and found that it is possible to distinguish facial expression to a certain degree of accuracy merely from motion of the white spots by playing back the video [Bas78].

As a study concentrating on a more specific part of facial motion, Koyama, et al. created CG animations with the temporal relations between eye and mouth movement controlled, and showed laughter can be classified into pleasant, unpleasant, and sociable types based on the temporal difference [NKN98]. As a study of analyzing solitary and social smiles, Schmidt, et al. indicated temporally consistent lip movement patterns based on the evaluation of the relationship between maximum velocity and amplitude [SCT03]. Hence, the importance of dynamic aspect in facial expression has been emphasized by many studies. However, an appropriate representation that maintains spatio-temporal structures in facial actions is still under study.

4.2 Facial Scores

4.2.1 Definition of Facial Scores

A facial score is a representation that describes motion patterns of each facial component and temporal relations between the movements. In this chapter we define the following notations:

Facial parts and facial-part sets. Facial parts represent isolable facial components. We define facial part sets as $\mathcal{P} = \{P_1, ..., P_{N_p}\}$ where N_p is the number of facial parts described by facial scores. For instance, elements of facial-part sets include mouths, right eyes, left eyes, right eyebrows, and left eyebrows.

Modes and mode sets. Modes represent simple motions of facial parts. We define mode sets as $\mathcal{M}^{(a)} = \{M_1^{(a)}, ..., M_{N_a}^{(a)}\}$ where N_a is the number of modes of a facial part $P_a (a \in \{1, ..., N_p\})$. For instance, elements of mode sets of a mouth part include "opening", "remain open", "closing", and "remain closed".

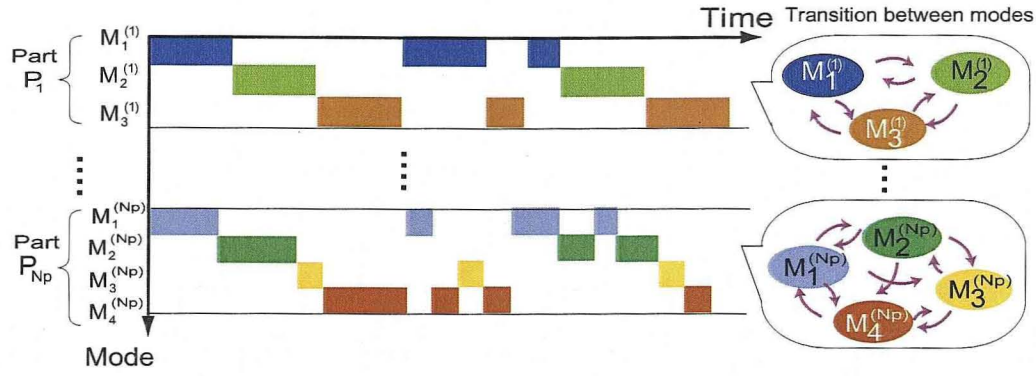


Figure 4.2: Facial scores. The vertical axis represents modes of facial parts, and the horizontal axis represents time. The transition of the motion of each facial part is described based on intervals along the temporal axis.

Intervals and interval sets. Intervals represent temporal ranges of modes. We define interval sets as $\mathcal{I}^{(a)} = \{I_1^{(a)}, ..., I_{K_a}^{(a)}\}$ where K_a is the number of intervals into which time series data of a facial part P_a is segmented. Intervals $I_k^{(a)} (k \in \{1, ..., K_a\})$ have beginning times $b_k^{(a)} \in \{1, ..., T\}$, ending times $e_k^{(a)} \in \{1, ..., T\}$, and labels of modes representing the events $m_k^{(a)} \in \mathcal{M}^{(a)}$ as attributes where T is the length of time series data of a facial part P_a.

Facial scores. We define a facial score as a set that comprises all the interval sets of each facial parts $\{\mathcal{I}^{(1)}, ..., \mathcal{I}^{(N_p)}\}$. Figure 4.2 shows a conceptual figure of a facial score. The vertical axis represents modes of facial parts, and the horizontal axis represents time. The transition of the motion of each facial part is described based on intervals along the temporal axis. For each facial part, intervals with the same mode are depicted by the same color and aligned at the same row. Thus, the facial score describes timing structures among motions of the facial parts.

4.2.2 Facial Parts in Facial Scores

To recognize facial expression based on timing structures, we need to treat multiple facial areas where their movements are able to occur independently. Because the facial motion is produced by muscular action, a straight forward definition is to choose each muscle as a different part. However, some facial skin can be moved by multiple muscle action; moreover, some muscles are hard to control independently. We therefore use appearance-based definition.

Ekman, et al. have revealed that the difference in the facial appearance of basic emotions (happiness, surprise, fear, anger, disgust, and sadness) results from the combination of the three facial areas (around the eyebrows, eyes, and mouth) where their movements can be observed individually in appearance [EF75]. We basically follow the Ekman's definition; that is, we use the three areas; in addition, we treat areas around the eyebrows and eyes on the left and right as different facial parts. This is because the asymmetric movements of each eyebrow and eye can be observed in real facial expression.

It is important to select useful features that can express subtle changes of movements in the five facial areas. Here, we defines feature vectors as coordinates of feature points shown in Figure 5 (a), which can extract information of movement directly. We consider that transient features such as furrows also provide effective information in recognition of subtle facial expression, and that changes of the feature points can represent them indirectly; for instance, movement of feature points on the nose implies nasolabial furrows.

Therefore, we define elements of facial part sets $\mathcal{P}$ as right eyebrow, left eyebrow, right eye, left eye, nose, and mouth. A feature vector $z^{(a)}$ of a facial part P_a is represented by the following $2n_a$-dimensional column vector:

$$x^{(a)} = (z_{x_1}^{(a)}, z_{y_1}^{(a)}, ..., z_{xn_a}^{(a)}, z_{y_{n_a}}^{(a)})^\top, \tag{4.1}$$

where n_a is the number of feature points of a facial part P_a, and let $(z_{xp}^{(a)}, z_{y_p}^{(a)})$ be coordinates of a feature point number $p \in \{1, ..., n_a\}$.

4.2.3 Modes in Facial Scores

As we defined in Subsection 4.2.1, each complex movement of a facial part is composed of simple motion categories, which we refer to as modes. Therefore, a movement can be partitioned into a sequence of temporal intervals by modes.

Modes are classified into two large categories by the velocity of feature vectors: stationary poses and dynamic movements. For the modes with movement, we use motions that have stable dynamics as the lowest-level representation, whereas humans sometimes classify a cyclic motion as one category. Therefore, our facial score represents a cyclic motion as a sequence of monotonic motions. For example, the open and close action of the mouth is represented as the following sequence of four modes: "opening", "remain open", "closing", and "remain closed".

AUs used in FACS are the most common units to describe facial movements. Although AUs are suitable to distinguish emotional facial expressions by their combinations, we do not use AUs as the modes in our facial scores for two reasons. First, a method of AU tracking is still a challenging research topic for computer vision. Second, AUs sometimes do not maintain sufficient dynamic information in facial actions. As a result, AU-based CG animation systems sometimes generate unnatural facial actions.

In contrast, our approach takes a bottom-up learning method to find modes rather than using predefined motion categories, as we described in Subsection 4.1.1. That is, all the modes are extracted by the clustering of dynamics from captured real data.

For a generative model of simple dynamics in each mode, we exploit the interval systems introduced in Chapter 2. The dynamics of the mode $M_i^{(a)} (i \in \{1, ..., N_a\})$ in a facial part P_a is therefore modeled by the following linear dynamical system:

$$x_t^{(a)} = F^{(a,\,i)} x_{t-1}^{(a)} + g^{(a,\,i)} + \omega_t^{(a,\,i)}, \tag{4.2}$$

where $x_t^{(a)}$ is a internal state vector in a feature space at time t, $F^{(a,\,i)}$ is a transition matrix, which differs from other modes' matrices, $g^{(a,\,i)}$ is a bias term, $\omega^{(a,\,i)}$ is a process noise of the system that has a multivariate Gaussian distribution with mean vector 0 and covariance matrix $Q^{(a,\,i)}$.

As a result, complex motion in each facial part is described based on the transition of linear dynamical systems, such as a hybrid dynamical system that we described in Chapter 2. Therefore, the proposed model can be considered as a concurrent process of multiple hybrid dynamical systems, where each hybrid dynamical system is applied to model dynamics in each part.

The extraction of mode is based on the clustering technique that we proposed in Section 3.3 in the previous chapter. We will briefly review the method here. Given a sequence of feature vectors, we first find a initial segmentation based on the velocity. We then merge the nearest dynamical system pairs iteratively based on agglomerative hierarchical clustering. A linear dynamical system, in general, can generate not only stable motions, which start from an initial shape and converge to a specific shape, but cyclic or oscillating motions. To extract only the stable motions, we proposed a method to provide a constraint on eigenvalues of the transition matrices.

4.3 Timing Structures in Facial Scores

Using facial scores defined in the previous sections, we can represent temporal relations among motions in facial parts; we refer to the relations as timing structures of the face. In this section, we describe a method to represent and extract timing structures from a facial score.

4.3.1 Definition of Timing Structures

We first concentrate on modeling timing structure between two parts a and b. Let $I_{(i)}$ be an interval I_k that has mode $M_i \in \mathcal{M}$ in part P_a (i.e.,$m_k = M_i$), and let $b_{(i)}, e_{(i)}$ be its beginning and ending time points, respectively. (We omit index k, which denotes the order of the interval.) Similarly, let $I'_{(p)}$ be an interval that has mode $M'_p \in \mathcal{M}'$ in the range $[b'_{(p)}, e'_{(p)}]$ of part P_b. The temporal relation of two modes becomes the quaternary relation of the four temporal points $R(b_{(i)}, e_{(i)}, b'_{(p)}, e'_{(p)})$.

Here we break up the quaternary relation $R(b_{(i)}, e_{(i)}, b'_{(p)}, e'_{(p)})$ into the following four binary relations:

$$R_{bb}(b_{(i)}, b'_{(p)}), \quad R_{be}(b_{(i)}, e'_{(p)}),$$
$$R_{eb}(e_{(i)}, b'_{(p)}), \quad R_{ee}(e_{(i)}, e'_{(p)}).$$

Let us define timing structure as the relation R that can be determined by a combination of these four binary relations above with respect to all the mode pairs $(M_i, M'_p) \in \mathcal{M} \times \mathcal{M}'$.

Considering temporal ordering relations $R_<, R_=, R_>$, which are often used in temporal logic [All83, All84, PMB97, PB97, Mas98], for these binary relations, we get 3^4 relations for R. Because of $b_{(i)} \leq e_{(i)}$ and $b'_{(p)} \leq e'_{(p)}$, it can be reduced to 13 relations as shown in Figure 4.3(a). Although these categories enable us to represent temporal structures among multiple events, such as overlaps between two intervals, they are insufficient for us to describe the difference of timing structures in facial expressions; that is, it is often important to analyze two motions in different facial parts are synchronized or not. We therefore utilize not only temporal order of events but metric information (i.e., scales and degree of temporal differences) among beginning and ending times of multiple intervals.

To extend the 13 categories using metric information, we use the temporal difference of two time points as the relation R_D, which can be represented by

metric $D \in \mathbf{R}$. Using this metric relation, we can define the first-order timing structure such as

$$D_{bb} = b_{(i)} - b'_{(p)} \qquad D_{be} = b_{(i)} - e'_{(p)}$$
$$D_{eb} = e_{(i)} - b'_{(p)} \qquad D_{ee} = e_{(i)} - e'_{(p)}$$

for R_{bb}, R_{be}, R_{eb}, and R_{ee}, respectively.

We can also define the second-order timing structure as the combination of two relations above. For example, the relation of R_{bb} and R_{ee} is represented by a point $(D_{bb}, D_{ee}) \in \mathbf{R}^2$ in a two-dimensional space if we use temporal difference $b_{(i)} - b'_{(p)}$ and $e_{(i)} - e'_{(p)}$ for the metric of R_{bb} and R_{ee}, respectively. Figure 4.3(b) shows the two-dimensional space, where the horizontal and vertical axes represent the difference between the beginning times and the difference between the ending times, respectively. The interval pairs in the figure denote the typical temporal relations in each area of the two-dimensional space.

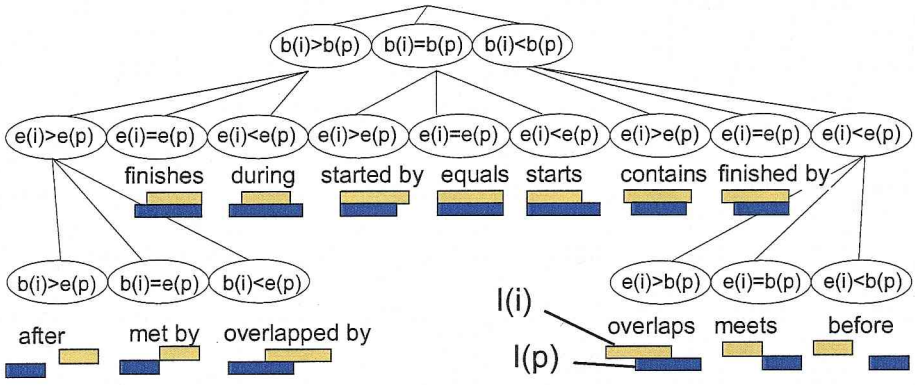
Note that, if we use the third-order timing structure that is defined by combination of three relations above, then all the temporal relations between the two intervals can be defined including the duration length.

4.3.2 Distributions of Timing Structures

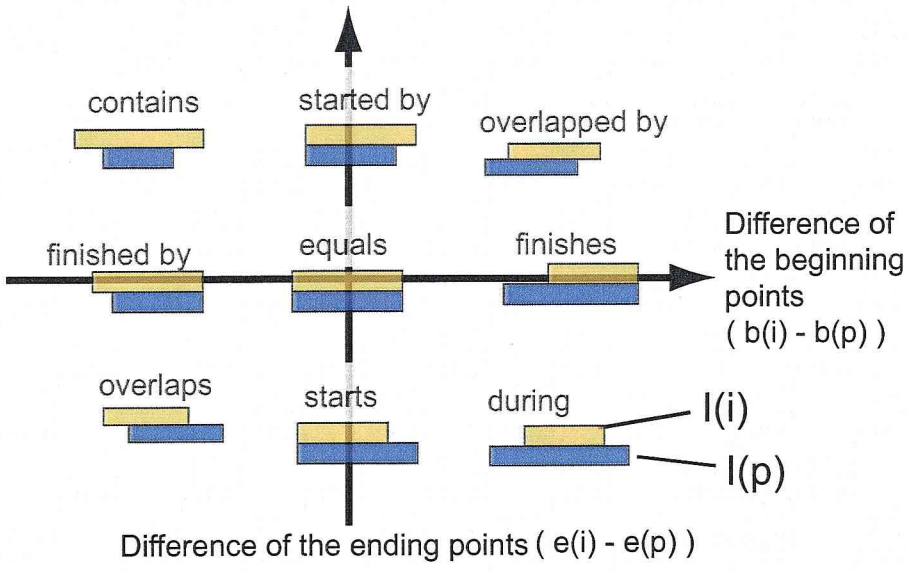
Using temporal differences between beginning and ending times, we can represent the distribution of first-order timing structure as four distributions $H(b_{(i)} - b_{(p)}), H(e_{(i)} - e_{(p)}), H(b_{(i)} - e_{(p)})$ and $H(e_{(i)} - b_{(p)})$, where $H(D)$ is a one-dimensional distribution of variable D. We can also represent the second-order timing structure as six two-dimensional distributions $H(b_{(i)} - b_{(p)}, e_{(i)} - e_{(p)}), H(b_{(i)} - b_{(p)}, b_{(i)} - e_{(p)}), H(b_{(i)} - b_{(p)}, e_{(i)} - b_{(p)}), H(e_{(i)} - e_{(p)}, b_{(i)} - e_{(p)}), H(e_{(i)} - e_{(p)}, e_{(i)} - b_{(p)})$ and $H(b_{(i)} - e_{(p)}, e_{(i)} - b_{(p)})$, where $H(D_1, D_2)$ is a two-dimensional distribution of variables D_1, $D_2 \in \mathbf{R}$. Representations of the third-order timing structures become three-dimensional distributions in the same manner.

4.4 Experiments

In this section, we evaluate the effectiveness of our representation by examining the separability between intentional smiles and spontaneous smiles using obtained facial scores from captured data.



(a) Temporal ordering relations.



(b) Temporal difference relation.

Figure 4.3: Temporal relations of two intervals. (a) The temporal order of beginning and ending time provides 13 relations of the two intervals. (b) The horizontal and vertical axes denote the difference between beginning points $b_{(i)} - b'_{(p)}$ and the difference between ending points $e_{(i)} - e'_{(p)}$, respectively.

4.4.1 Configuration of the Experiments

Intentional and spontaneous smiles of six subjects (we use ID A to F to distinguish them) were captured in 480 × 640 at 60 fps as the input image sequences. Then, we downsampled the images to 240 × 320 resolution. We used a camera system that was composed of a helmet and a camera fixed in front of the helmet to concentrate on the analysis of front faces. The camera system enabled us to capture front face images without self-occlusion even if large head motion occurred.

The subjects were instructed to begin with a neutral expression, make a smile, and return to a neutral expression again. Intentional smiles were captured by instructing the subjects to force a smile during they were watching disgusting movie that have been standardized by Gross [GL95]. Spontaneous smiles were captured during they were watching Japanese-standup comedy (Manzai). Figure 4.4 (b) shows part of a captured face image sequence. The number of intentional smiles was 50 for all the subjects. The number of spontaneous smiles was different among the subjects: subject A, B, C, D, E, and F made 37, 39, 30, 38, 31, and 29 expressions, respectively.

4.4.2 Facial Feature Tracking

We tracked feature points in facial image sequences using the active appearance model (AAM) [CET98]. The AAM contains a statistical model of correlations between shape and grey-level appearance variation. The AAM-based feature point tracking consists of two stages. We first build an AAM model using a training set of face images and its feature points given manually. Then, we can use the model to extract facial feature points in novel images. Due to the trained model, AAM can search the feature points rapidly and robustly (see Appendix D for details).

Figure 4.4 shows an example of tracked feature points [1]. The number of feature points used in the AAM was set to 5 on each eyebrow, 8 on each eye, 11 on the nose, 8 on the mouth, and 13 on the jaw line (refer to Figure 4.4 (a)). Although the jaw line was not represented as one of the facial parts, it was used for improving tracking accuracy. Therefore, the dimensionality of feature vectors in the eyebrows (left/right), the eye (left/right), the nose, and the mouth were 10, 16, 22, and 16, respectively.

Figure 4.4 (c) shows part of a face image sequence with tracked feature points;

[1] Feature points were tracked using the AAM-API that Stegmann (Technical University of Denmark) developed [SG02].

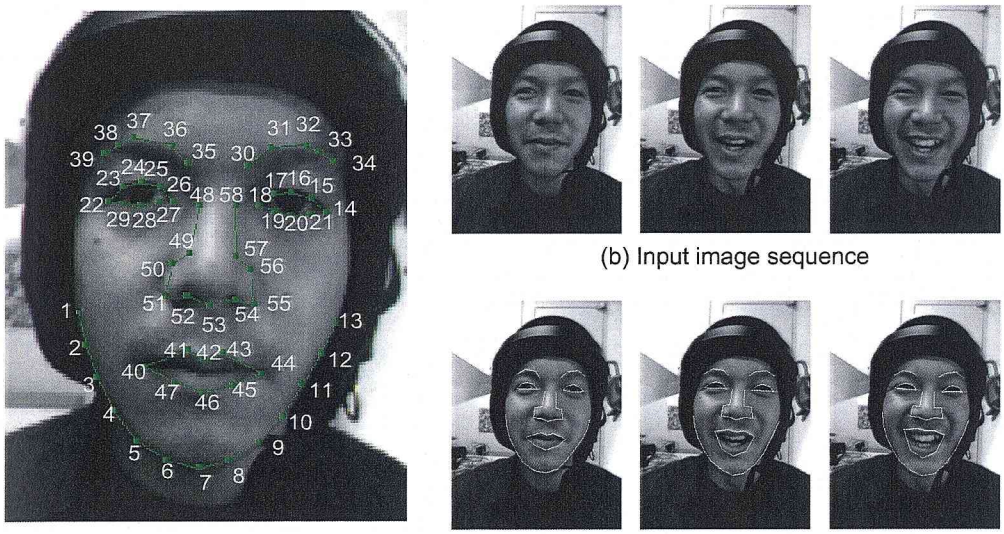


Figure 4.4: (a) A training image to build active appearance models. (b) Part of a captured face image sequence. (c) Part of a face image sequence with tracked feature points.

the frames correspond to the images shown in Figure 4.4 (b). Comparison of the corresponding images demonstrates precise detection of feature points in changes of facial expression.

4.4.3 Automatic Acquisition of Facial Scores

As we described in Subsection 4.2.3, the obtained feature vectors of each facial part were segmented into modes using the clustering of linear dynamical systems that we proposed in Section 3.3. Figure 4.5 is an example of the segmentation result of the mouth part. The vertical axes of the top, the middle and the bottom subfigures represent x-coordinates of feature points, y-coordinates of feature points and the transition of modes respectively. The horizontal axis of each subfigure represents time.

Figure 4.6 shows an example of the facial score that describes dynamic characteristics of all facial parts during intentional smiles. This figure suggests that the movement of each smile can be segmented into the following four modes: two stationary modes ("neutral" and "smiling") and two dynamic modes ("onset" and "offset" of smiling).

Figure 4.7 and Figure 4.8 shows the facial score of an intentional smile and a natural smile, respectively. We see that the beginning and ending timing of the in-

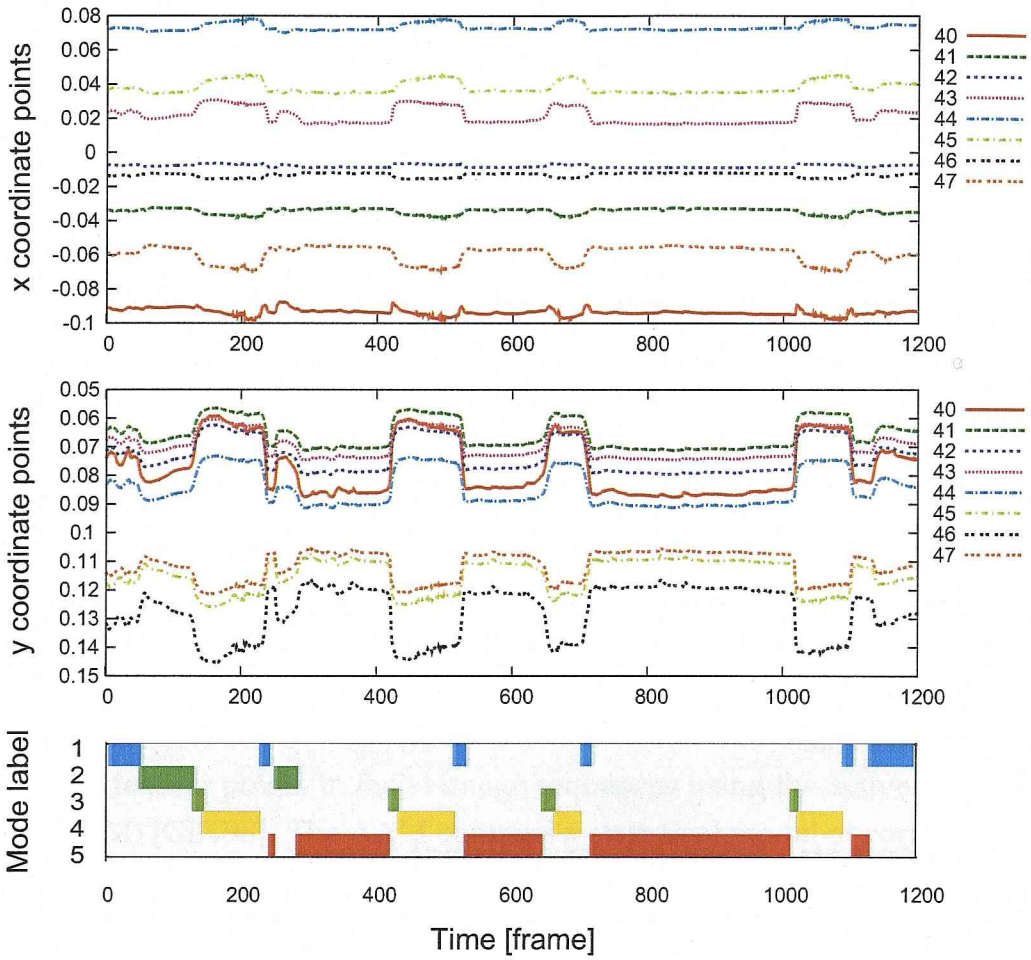


Figure 4.5: The correspondence of the mouth part of an obtained facial score from spontaneous smiles with the feature vector series. The vertical axes of the top, the middle and the bottom subfigures represent x-coordinates of feature points, y-coordinates of feature points and modes respectively, and the horizontal axes of each subfigure represent time. The numbers of legends in the top and middle correspond to numbers that represent labels of feature points in Figure 4.4 (a). For example, the mode 4 and 5 represent "remain open" and "remain closed", respectively.

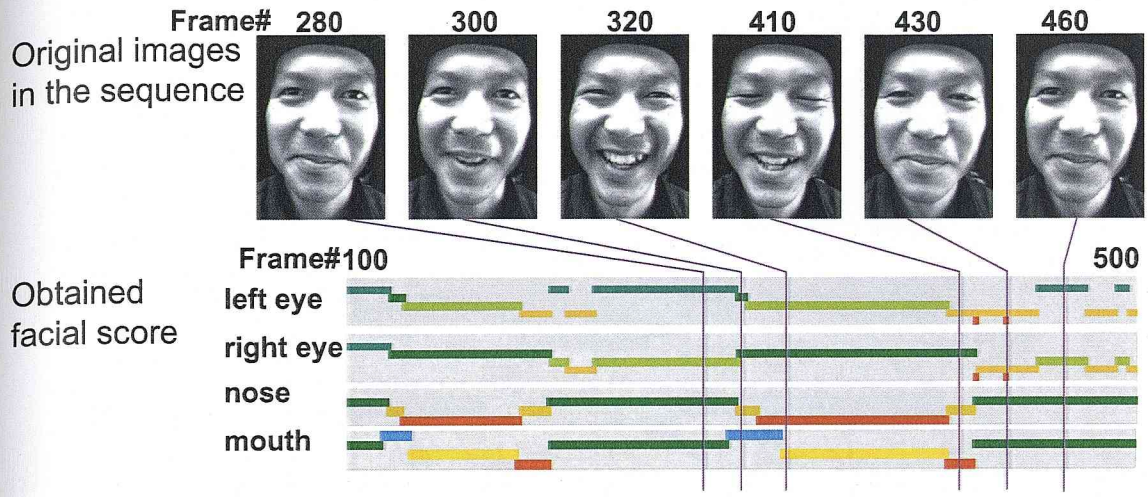


Figure 4.6: An example of obtained facial scores from intentional smiles (left and right eyebrows are omitted).

tervals are different in these expressions. Especially, the motions of the intentional smile are synchronized compared to the natural smile. In the next subsection, we evaluate the differences of these two smiles based on the comparison of timing structures.

Because of the limitation of the movie length and the capacity of the capturing system, we obtained facial expressions using several sessions. Then, we acquired facial scores from each of the sessions automatically. We however manually found the correspondence of modes among these sessions. In addition, we merged multiple intervals based on human observation in case that one mode was divided into multiple modes.

Despite that we could replace this manual operation with an automatic training method such as the expectation-maximization algorithm of the interval system described in Section 3.4, we chose to check and modify the segmentation results manually because we wanted to verify the effectiveness of timing structures rather than to evaluate the precision of the EM algorithm. To distinguish these two problems and to concentrate on verifying the effectiveness of timing structures, we postulated that the clustering algorithm provided a set of candidates for the segmentation, and we selected one of the candidates manually.

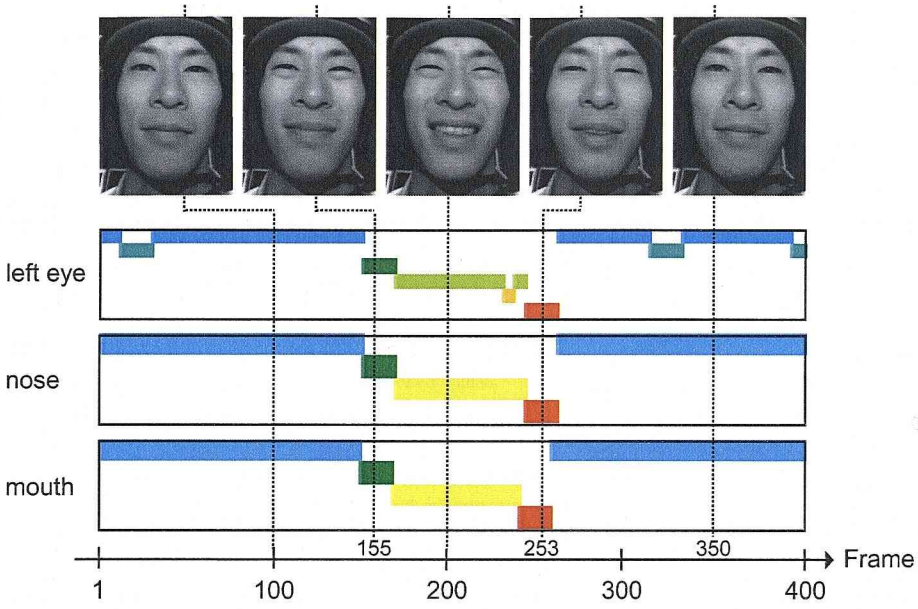


Figure 4.7: The facial score of an intentional smile.

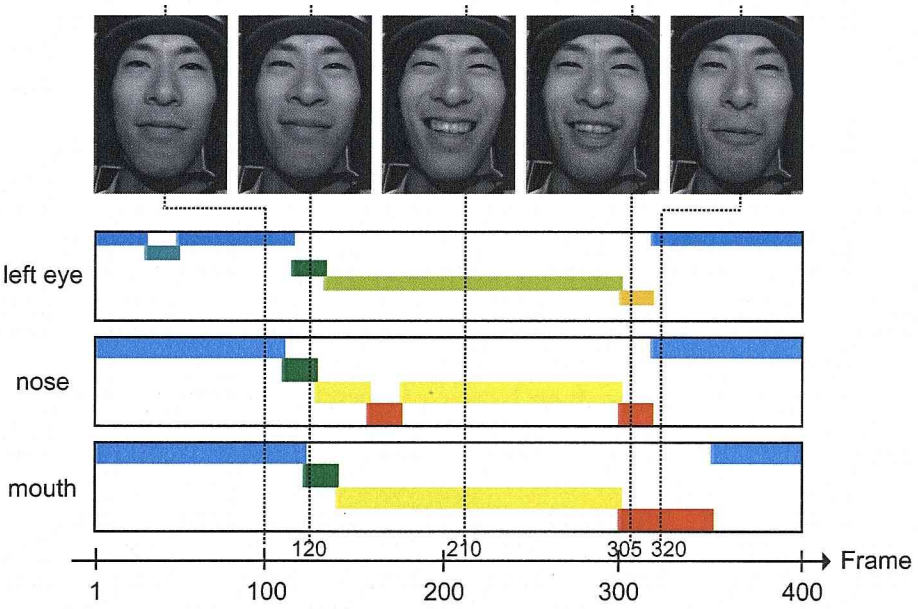


Figure 4.8: The facial score of a spontaneous smile.

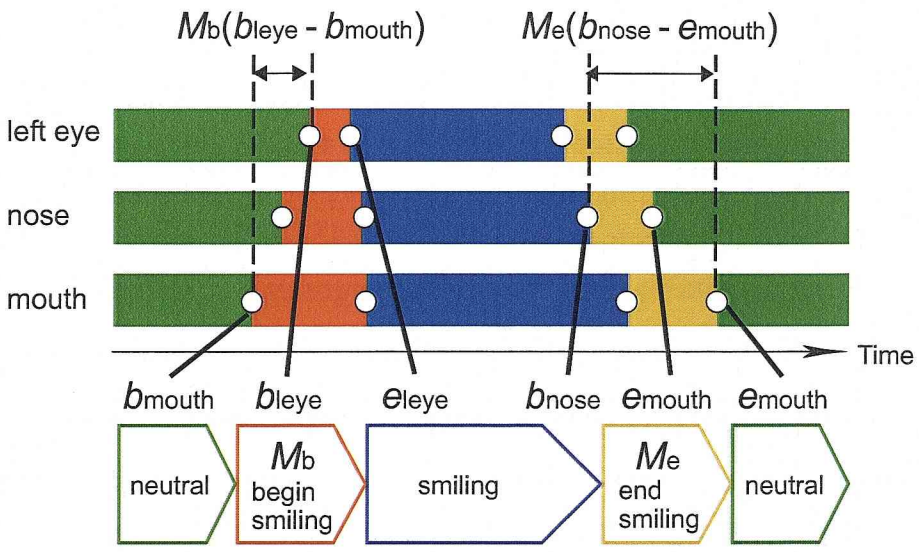


Figure 4.9: Onset mode M_b and offset mode M_e.

4.4.4 Comparison of Timing Structures between Intentional and Spontaneous Smiles

Modes of Smile Onset and Offset

To evaluate the separability of intentional and spontaneous smiles using extracted facial scores, we defined the following two modes which is selected from automatically extracted modes:

M_b: onset motion of smiles (from neutral to smiling)

M_e: offset motion of smiles (from smiling to neutral)

To simplify the evaluation, we used a facial score that consists of three facial parts: left eye, nose, and mouth. In addition, since the duration lengths of stationary modes such as "neutral" and "smiling" closely depend on the context of the expression, we concentrate on analyzing timing structures among dynamic modes M_b and M_e (see Figure 4.9).

Let b_{leye} and e_{leye} be the beginning and ending time points of the left eye motion in its facial score. Similarly, let b_{nose} and e_{nose} be those of the nose motion, and b_{mouth} and e_{mouth} be those of the mouth motion, respectively. Then we extract temporal differences between such time points; for example, we use $M_b(b_{\text{nose}} - b_{\text{mouth}})$, which denote the temporal difference between the beginning of nose motion and that of mouth motion during the onset of a smile.

Analysis of Timing Structures

To analyze the intentional and natural facial expression, we exploited the representation of timing structure described in Section 4.3. We first used the first-order timing structure analysis using one-dimensional distributions as preliminary experiments. Since this preliminary experiments showed that any single temporal difference cannot discriminate the two smile categories (i.e., intentional and spontaneous), we employed a pair of temporal differences (i.e., the second-order timing structure) as a distinguishing feature. That is, a feature to characterize each shot of smile is represented by a point in the two-dimensional space whose axes denote a selected pair of temporal differences. Since there are many possibilities for the combination of temporal differences, for each pair of temporal differences, we calculated the Maharanobis generalized distance between a pair of distributions of two smile categories, and selected such pair of temporal differences that the two distributions took the largest distance. Note that since smiling actions may differ from person to person, we extracted a distinguishing feature for each subject person.

Figure 4.10 shows the experimental results for six persons. Each subfigure shows the selected two-dimensional space and the distributions of intentional and natural smile categories for each subject. Each point denotes a single expression. From this figure, we observe that we can discriminate the distributions of two smile categories using their dynamic features.

To evaluate the effectiveness of timing structure for discriminating intentional and natural smiles, we calculated recognition rate of each smile for each subject based on leave-on-out method [DHS00]. First, we trained a linear discriminate boundary plane using support vector machines [Bur98]; we then discriminated the test data. The result is shown in Table 4.1. We see that all the recognition rates for all the subjects are in the ranges from 79.4% to 100%. Hence, the timing structure provides an enough feature for distinguishing intentional and natural smile categories.

Differences among the Subjects

From Figure 4.10, we see that the extracted axes are different among the six subjects. Especially, the axes that correspond to duration lengths of onset or offset motions (e.g., $M_b(b_{\text{nose}} - e_{\text{mouth}})$) were extracted from five subjects excluding subject C.

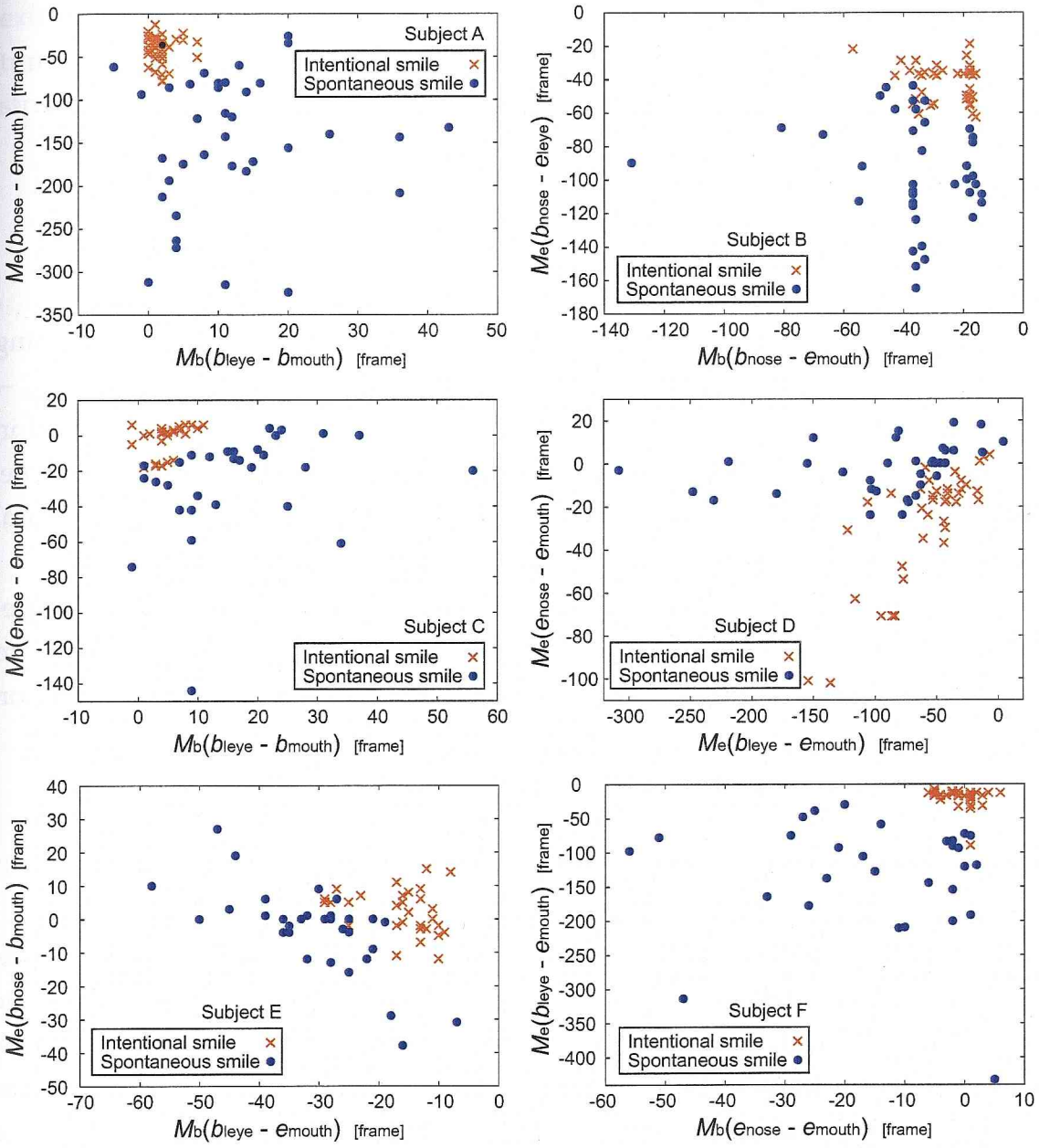


Figure 4.10: Timing structure with the longest distance between intentional and spontaneous smiles distribution.

Table 4.1: Accuracy of discrimination between intentional and spontaneous smiles based on timing structures in Figure 4.10.

Subject	**Intentional** (%)	**Spontaneous** (%)
A	100	83.8
B	100	79.4
C	82.4	96.4
D	85.1	79.7
E	85.3	90.3
F	96.6	93.1

For subject A and C, the axis that denotes the difference of the beginning points between the left eye and mouth parts in the onset motion (i.e., $M_b(b_{\text{leye}} - b_{\text{mouth}})$) were extracted. This feature corresponds to the feature that is used in the psychological experiments conducted by Nishio, et al. [NKN98]. Therefore, intentional smiles of subject A and C might be easily discriminated from natural smiles by human.

Consequently, the personality of facial expression becomes significant especially for discriminating intentional and spontaneous expression compared to emotion recognition in which most of the existing work attempts to find common factor of facial expression.

4.5 Discussion

In this chapter, we proposed a facial score as a novel facial expression representation. The score describes timing structures in faces by assuming that dynamic movement of each facial part yields changes of facial expression. Using the score, we provided a framework for recognizing fine-grained facial expression categories. In our evaluation, the scores were acquired from captured real image sequences including intentional and spontaneous smiles automatically, and we confirmed that movement of facial parts was expressed based on temporal intervals each of which is characterized by linear dynamical system, and the effectiveness of the timing structure for discriminating the two smile categories.

To emphasize the characteristics of the proposed representation, we focused on using only timing structures. However, other features of movement such as scale, speed and duration, which provide further information on recognizing fa-

cial expression, should be taken into account in practical systems. We also need to discuss specificity and generality of timing structures: some structures may exist as general features determined by physical muscle constraints, and the other may exist as subject-specific features produced by personal habits. Directions for future work are to tackle these problems and to evaluate the effectiveness of timing structure using a large number of captured sequences.

Chapter 5

Modeling Timing Structures in Multimedia Signals

In this chapter, we propose a model to represent timing structures in multimedia signals, and exploit the model to generate a media signal from another related signal. The difference from the previous chapter is that we here show a general framework for modeling and utilizing mutual dependency among media signals based on the temporal relations among hybrid dynamical systems rather than only apply the system to each media signal and analyze the temporal structures among dynamic events.

5.1 Timing Structures in Multimedia Signals

Measuring dynamic human actions such as speech and musical performance with multiple sensors, we obtain multiple media signals across different modalities. We human usually sense and feel cross-modal dynamic features fabricated by multimedia signals such as synchronization and delay. For example, it is well-known fact that the simultaneity between auditory and visual patterns influences human perception (e.g., the McGurk effect [MM76]), and we can find some psychological studies about the audio-visual simultaneity (e.g., [FSKN04]).

On the other hand, modeling cross-modal structures is also important to realize multimedia systems (Figure 5.1); for example, human computer interfaces such as audio-visual speech recognition systems [NLP+02] and computer graphic techniques such as generating a media signal from another related signal (e.g., lip motion generation from input audio signals [Bra99]). Articulated motion model-

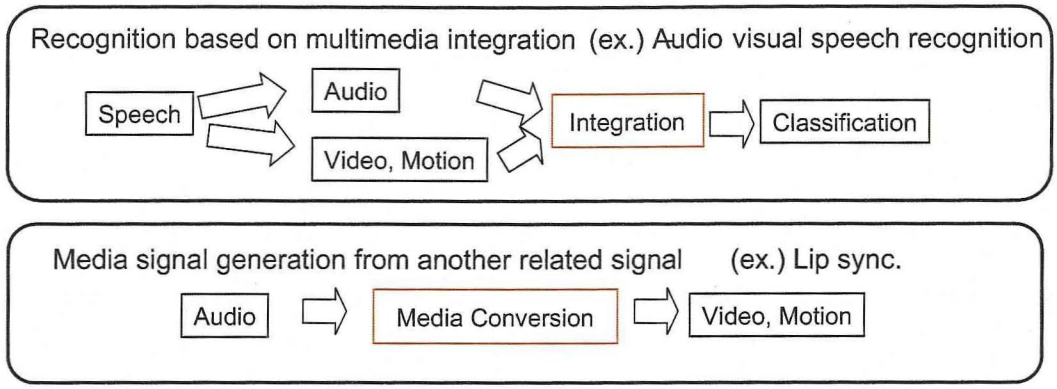


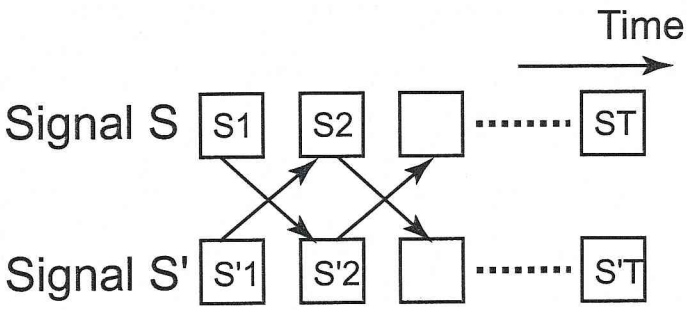
Figure 5.1: Applications of modeling cross-modal structures.

ing can also exploit this kind of temporal structures because motion timing among each different part plays an important role to realize natural motion generation.

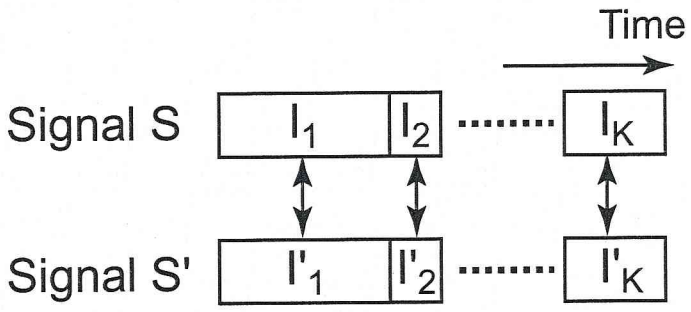
Dynamic Bayesian networks, such as coupled hidden Markov models (HMMs) [BOP97], are one of the most well-known methods to integrate multiple media signals [NLP+02]. These models describe relations between concurrent (co-occurred) or adjacent states of different media data (Figure 5.2(a) and 5.2(b)). A coupled HMM can be categorized into a frame-wise method because it models the frequency of state pairs that occur in adjacent frames. Although this frame-wise representation enables us to model short term relations or interaction among multiple processes, they are not well-suited to describe systematic and long-term cross-media relations. For example, an opening lip motion is strongly synchronized with an explosive sound /p/, while the lip motion is loosely synchronized with a vowel sound /e/; in addition, the motion always precedes the sound (Figure 5.3 left). We can see such an organized temporal difference in music performances also; performers often make preceding motion before the actual sound (Figure 5.3 right).

In this chapter, we propose a novel model that directly represents this important aspect of temporal relations, what we refer to as *timing structure*, such as synchronization and mutual dependency with organized temporal difference among multiple media signals (Figure 5.2(c)).

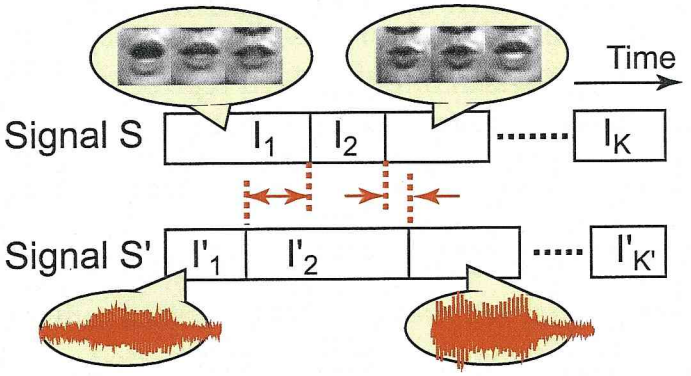
First, we assume that each media signal is described by a finite set of "modes" (i.e., primitive temporal patterns) similar to the previous chapter; we apply an interval-based hybrid dynamical system (interval system) to represent signal patterns in each media based on the modes. Then, we introduce a *timing structure model*, which is a stochastic model for describing temporal structure among in-



(a) Frame-wise modeling (adjacent time relations)



(b) Frame-wise modeling (co-occurrence)



(c) Timing based modeling

Figure 5.2: Temporal structure representation in multimedia signals.

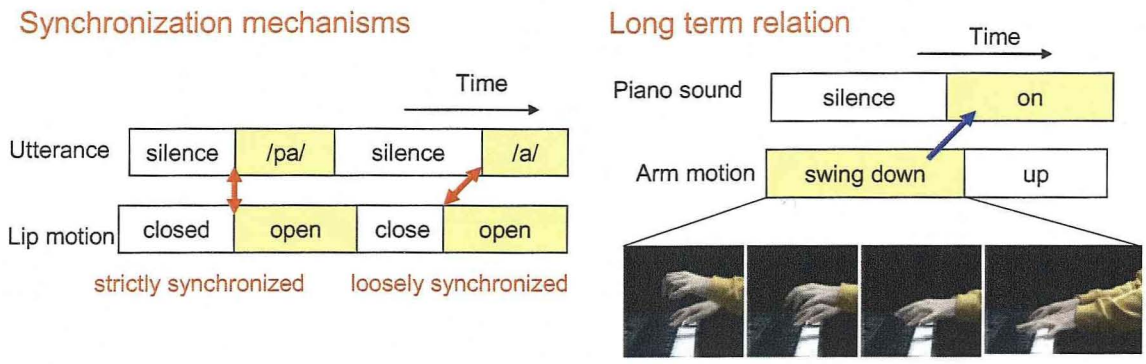


Figure 5.3: Open issues of existing multimedia co-occurrence models.

tervals in different media signals. The model explicitly represents temporal difference among beginning and ending points of intervals, it therefore provides a framework of integrating multiple interval systems across modalities as we will see in the following sections. Consequently, we can exploit the timing structure model to wide area of multimedia systems including human machine interaction systems in which media synchronization plays an important role. In the experiments, we verified the effectiveness of the method by applying it to media signal conversion that generates a media signal from another media signal.

As we described in Chapter 1, segment models [ODK96] can also be candidate models. Despite we use interval systems for experiments in this chapter, the timing structure model, which is proposed in this chapter, can be applicable for every model that provides an interval-based representation of media signals, where each interval is a temporal region labeled by one of the modes.

5.2 Modeling Timing Structures in Multimedia Signals

5.2.1 Temporal Interval Representation of Media Signals

To define timing structure, we assume that each media signal is represented by a single interval system, and the parameters of the interval system are estimated in advance (see [ODK96, LWS02], for example). Then, each media signal is described by an interval sequence. In the following paragraphs, we introduce some terms and notations for the structure and the model definition.

Media signals. Multimedia signals are obtained by measuring dynamic event with N_s sensors simultaneously. Let S_c be a single media signal. Then, multimedia signals become $\mathcal{S} = \{S_1, \cdots, S_{N_s}\}$. We assume that S_c is a discrete signal that is sampled by rate ΔT_c.

Modes and Mode sets. Mode $M_i^{(c)}$ is the property of temporal variation occurred in signal S_c (e.g., "opening mouth" and "closing mouth" in a facial video signal). We define a mode set of S_c as a finite set: $\mathcal{M}^{(c)} = \{M_1^{(c)}, \cdots, M_{N_c}^{(c)}\}$. Each mode is represented by a sub model of the interval system (i.e., linear dynamical systems).

Intervals. Interval $I_k^{(c)}$ is a temporal region that a single mode represents. Index k denotes a temporal order that the interval appeared in signal S_c. Interval $I_k^{(c)}$ has properties of beginning and ending time $b_k^{(c)}, e_k^{(c)} \in \mathbf{N}$ (the natural number set), and mode label $m_k^{(c)} \in \mathcal{M}^{(c)}$. Note that we simply refer to the indices of sampled order as "time". We assume signal S_c is partitioned into interval sequence $\mathcal{I}^{(c)} = \{I_1^{(c)}, ..., I_{K_c}^{(c)}\}$ by the interval system, where the intervals have no gaps or (i.e., $b_{k+1}^{(c)} = e_k^{(c)} + 1$ and $m_k^{(c)} \neq m_{k+1}^{(c)}$).

Interval representation of media signals. Interval representation of multimedia signals is a set of interval sequences: $\{\mathcal{I}^{(1)}, ..., \mathcal{I}^{(N_s)}\}$.

5.2.2 Definition of Timing Structure in Multimedia Signals

In this chapter, we concentrate on modeling timing structure between two media signals S and S'. (We use the mark " ′ " to discriminate between the two signals.)

Let us use notation $I_{(i)}$ for an interval I_k that has mode $M_i \in \mathcal{M}$ in signal S (i.e.,$m_k = M_i$), and let $b_{(i)}, e_{(i)}$ be its beginning and ending time points, respectively. (We omit index k, which denotes the order of the interval.) Similarly, let $I'_{(p)}$ be an interval that has mode $M'_p \in \mathcal{M}'$ in the range $[b'_{(p)}, e'_{(p)}]$ of signal S'. Then, the temporal relation of two modes becomes the quaternary relation of the four temporal points $R(b_{(i)}, e_{(i)}, b'_{(p)}, e'_{(p)})$. If signal S and S' has different sampling rate, let the cycles be ΔT and $\Delta T'$, we have to consider the relation of continuous time such as $b_{(i)}\Delta T$ and $b'_{(i)}\Delta T'$ on behalf of $b_{(i)}$ and $b'_{(i)}$. In this subsection, we just use $b_{(i)} \in \mathbf{R}$(the real number set) for both continuous time and the indices of discrete time to simplify the notation.

Similar to Subsection 4.3.1 in the previous chapter, we can define timing structure as the relation R that can be determined by the combination of four binary

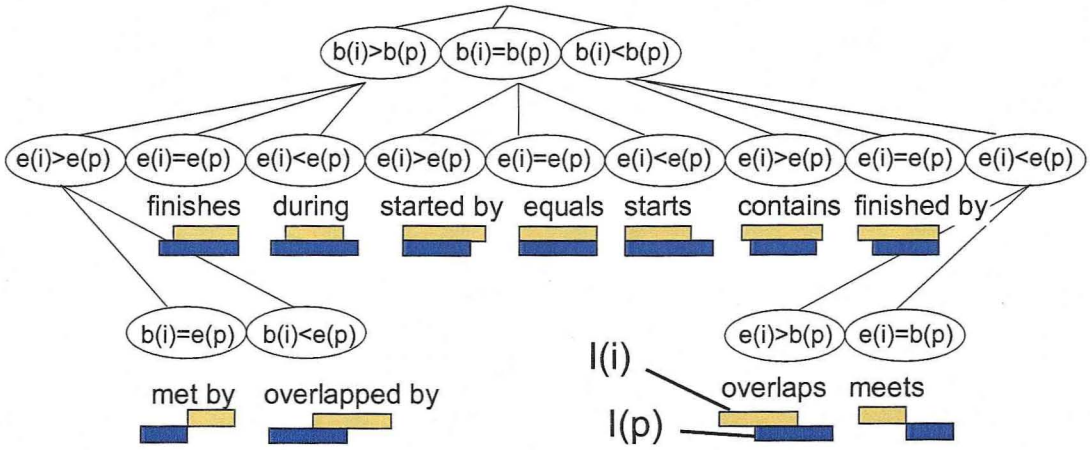


Figure 5.4: Overlapped interval relations (Subset of Figure 4.3(a)).

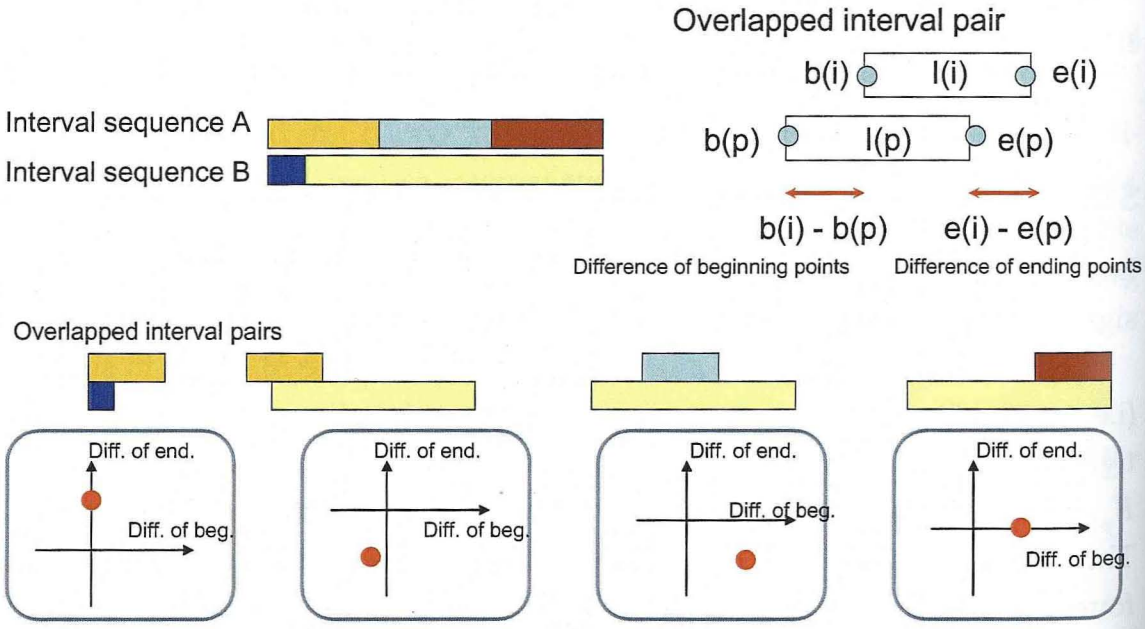


Figure 5.5: Examples of the metric relations. Four points represent temporal relations of two modes that appeared in overlapped intervals.

relations: $R_{bb}(b_{(i)}, b'_{(p)})$, $R_{be}(b_{(i)}, e'_{(p)})$, $R_{eb}(e_{(i)}, b'_{(p)})$, $R_{ee}(e_{(i)}, e'_{(p)})$. In the following, we specify the four binary relations that is suitable for modeling temporal structure in media signals (e.g., temporal difference between sound and motion).

We first introduce metric relations for R_{bb} and R_{ee} by assuming that R_{be} and R_{eb} is $R_{\leq}$ and $R_{\geq}$, respectively (i.e., the two modes have overlaps), as shown in Figure 5.4. This assumption is natural when the influence of one mode to the other modes with long temporal distance can be ignored. For the metric of R_{bb} and R_{ee}, we use temporal difference $b_{(i)} - b'_{(p)}$ and $e_{(i)} - e'_{(p)}$, respectively; the relation is represented by a point $(D_b, D_e) \in \mathbf{R}^2$ (see also Figure 4.3(b)).

Figure 5.5 shows some examples of the relations. There are three modes in interval sequence A, and two modes in interval sequence B. The four figures below represent the relations of mode pairs that appear in the overlapped interval pairs.

In the next subsection, we model this type of temporal metric relation using two-dimensional distributions. As a result, the model provides framework to represent synchronization and co-occurrence.

5.2.3 Modeling Timing Structures

Temporal Difference Distribution of Mode Pairs

To model the metric relations that described in the previous subsection, we introduce the following distribution for every mode pair $(M_i, M'_p) \in \mathcal{M} \times \mathcal{M}'$:

$$P(b_k - b'_{k'} = D_b, e_k - e'_{k'} = D_e | m_k = M_i, m'_{k'} = M'_p, [b_k, e_k] \cap [b'_{k'}, e'_{k'}] \neq \phi). \quad (5.1)$$

We refer to this distribution as a *temporal difference distribution* of the mode pair. As we described in Subsection 5.2.2, the domain of the distribution is $\mathbf{R}^2$.

Because the distribution explicitly represent the frequency of the metric relation between two modes (i.e., temporal difference between beginning points and the difference between ending points), it provides significant temporal structures for two media signals. For example, if the peak of the distribution comes to the origin, the two modes tend to be synchronized each other at the beginning and ending points, while if $b_k - b_{k'}$ has large variance, the two modes loosely synchronized at their onset timing.

To estimate the distribution, we collect all pairs of overlapping intervals that have the same mode pairs (Figure 5.6). Since training data is usually finite when

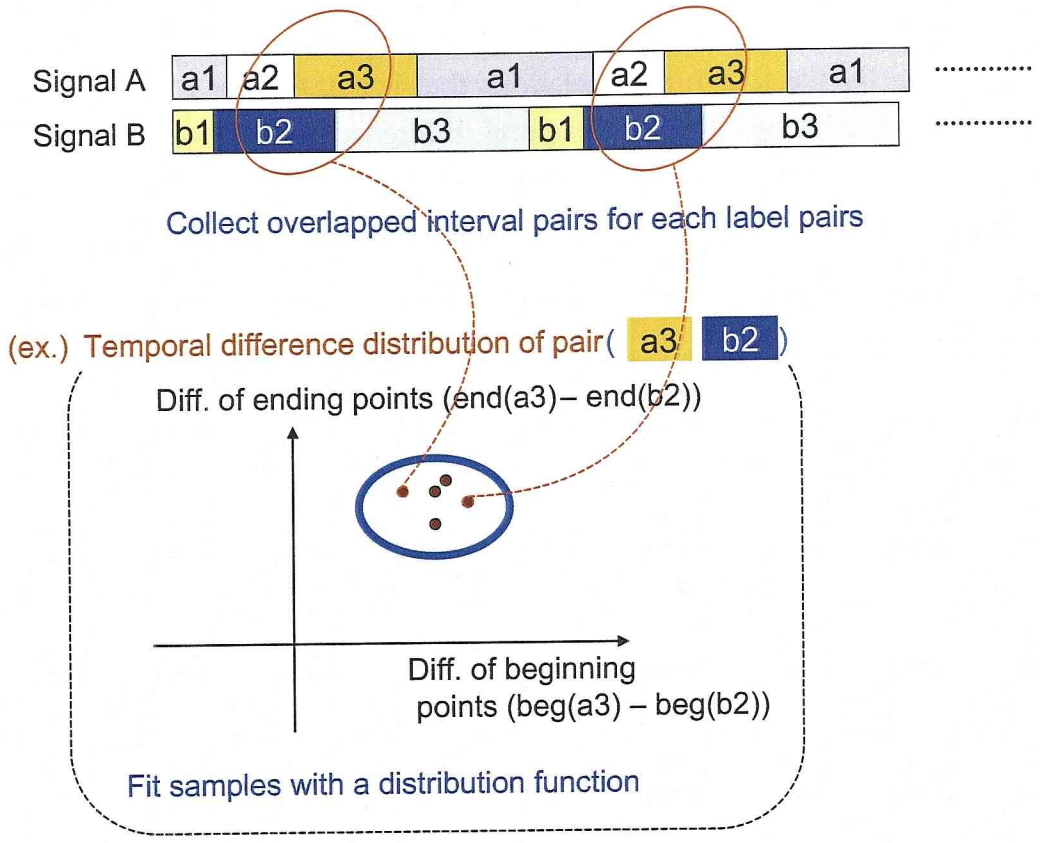


Figure 5.6: Learning of a timing structure model.

we use the model in real applications, we fit a density function such as Gaussian or its mixture models to the samples.

Co-occurrence Distribution of Mode Pairs

As we see in Equation (5.1), the temporal difference distribution is a probability distribution under the condition of the given mode pair. To represent frequency that each mode pair appears in the overlapped interval pairs, we introduce the following distribution:

$$P(m_k = M_i, m_{k'} = M'_p \mid [b_k, e_k] \cap [b'_{k'}, e'_{k'}] \neq \phi). \tag{5.2}$$

We refer to this distribution as *co-occurrence distribution* of mode pairs. The distribution can be easily estimated by calculating a mode pair histogram from every overlapped interval pairs.

Transition Probability of Modes

Using Equation (5.1) and (5.2), we can represent timing structure that is defined in Subsection 5.2.2. Although timing structure models temporal metric relations between media signals, temporal relation in each media signal is also important. Therefore, similar to previously introduced interval systems, we use the following transition probability of adjacent modes in each signal:

$$P(m_k = M_j | m_{k-1} = M_i) \, (M_i, M_j \in \mathcal{M}). \tag{5.3}$$

5.3 Media Signal Conversion Based on Timing Structures

Once we estimated the timing structure model that introduced in Section 5.2 from simultaneously captured multimedia data, we can exploit the model for generating one media signal from another related signal. We refer to the process as *media signal conversion*, and introduce the algorithm in this section.

The overall flow of media signal conversion from signal S' to S is as follows (see also Figure 5.7):

1. A reference (input) media signal S' is partitioned into an interval sequence $\mathcal{I}' = \{I'_1, ..., I'_{K'}\}$.

2. A media interval sequence $\mathcal{I} = \{I_1, ..., I_K\}$ is generated from a reference interval sequence $\mathcal{I}'$ based on the trained timing structure model. (K and K' is the number of intervals in $\mathcal{I}$ and $\mathcal{I}'$, and $K \neq K'$ in general.)

3. Signal S is generated from $\mathcal{I}$.

The key process of this media conversion lies in step 2. Since the methods of step 1 and 3 have been already introduced in Chapter 2, we here propose a novel method for step 2: a method that generates one media interval sequence from another related media interval sequence based on the timing structure model. In the following subsections, we assume that the two media signals S, S' have the same sampling rate to simplify the algorithm.

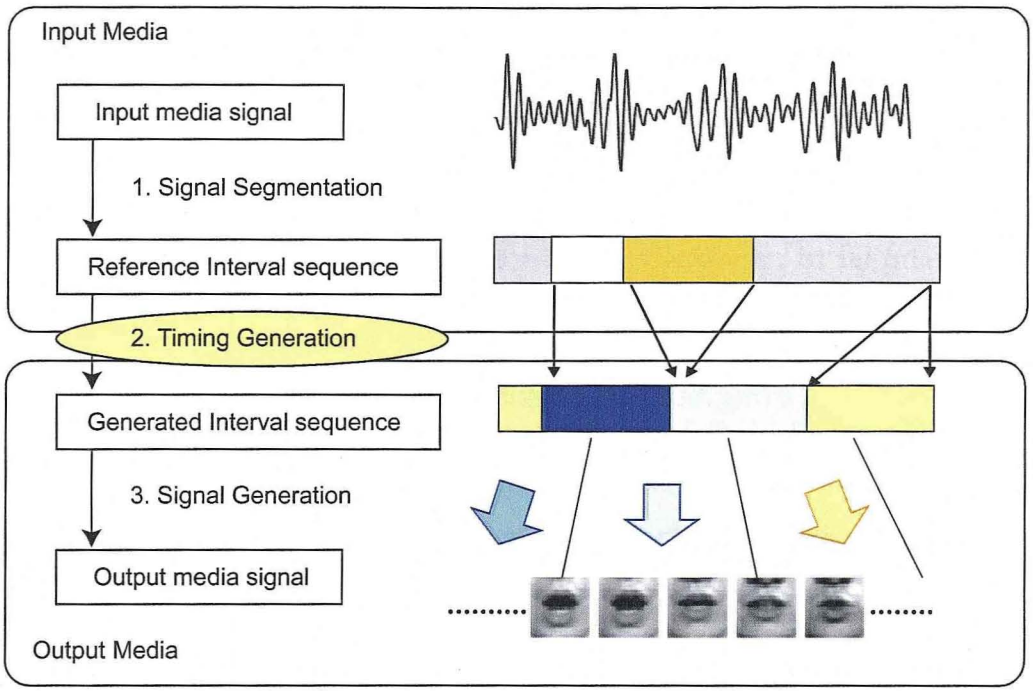


Figure 5.7: The flow of media conversion.

5.3.1 Formulation of Media Signal Conversion Problem

Let Φ be the timing structure model that is learned in advance (i.e., all the parameters described in Subsection 5.2.3 is estimated). Then, the problem of generating an interval sequence $\mathcal{I}$ from a reference interval sequence $\mathcal{I}'$ can be formulated by the following optimization:

$$\hat{\mathcal{I}} = \arg\max_{\mathcal{I}} P(\mathcal{I}|\mathcal{I}', \Phi). \tag{5.4}$$

In the equation above, we have to determine the number of intervals K and their properties, which can be described by triples (b_k, e_k, m_k) $(k = 1, ..., K)$, where $b_k, e_k \in [1, T]$ and $m_k \in \mathcal{M}$. Here, T is the length of signal S', and $\mathcal{M}$ is the mode set, which is estimated simultaneously with the signal segmentation. If we search for all the possible interval sequences $\{\mathcal{I}\}$, the computational cost would increase exponentially as T becomes longer. We therefore use a dynamic programming method to solve Equation (5.4), where we assume that generated intervals have no gaps or overlaps; thus, pairs $< e_k, m_k >$ $(k = 1, ..., K)$ are required to be estimated under this assumption (see Subsection 5.3.2 for details).

We currently do not consider online media signal conversion because it re-

quires a trace back step that finds partitioning points of intervals from the final to the first frame of the input signal. If online processing is necessary, one of the simplest method is dividing input stream comparatively longer range than the sampling rate and apply the following method to each of the divided range.

5.3.2 Interval Sequence Generation via Dynamic Programming

To simplify the notation, we omit the model parameter variable Φ in the following equations. Let us use notation $f_t = 1$ that denotes the interval "finishes" at time t, which is similar to the notation that we introduced in Subsection 2.3.2. Then, $P(m_t = M_j, f_t = 1|\mathcal{I}')$, which is the probability when an interval finishes at time t and the mode of time t becomes M_j in the condition of the given interval sequence $\mathcal{I}'$, can be calculated by the following recursive equation:

$$\begin{aligned} & P(m_t = M_j, f_t = 1|\mathcal{I}') \\ = & \sum_{\tau} \sum_{i(\neq j)} \left\{ \begin{array}{r} P(m_t = M_j, f_t = 1, l_t = \tau | m_{t-\tau} = M_i, f_{t-\tau} = 1, \mathcal{I}') \\ \times P(m_{t-\tau} = M_i, f_{t-\tau} = 1|\mathcal{I}') \end{array} \right\}, \end{aligned} \tag{5.5}$$

where l_t is a duration length of an interval (i.e., it continues l_t at time t) and m_t is a mode label at time t. The lattice in Figure 5.8 depicts the path of the above recursive calculation. Each pair of arrows from each circle denotes whether the interval "continues" or "finishes", and every bottom circle sums up all the finishing interval probabilities.

The following dynamic programming algorithm is deduced directly from the recursive equation (5.5):

$$\begin{aligned} E_t(j) &= \max_{\tau} \max_{i(\neq j)} \underline{P(m_t = M_j, f_t = 1, l_t = \tau | m_{t-\tau} = M_i, f_{t-\tau} = 1, \mathcal{I}')} E_{t-\tau}(i), \\ & \text{where} \quad E_t(j) \triangleq \max_{m_1^{t-1}} P(m_1^{t-1}, m_t = M_j, f_t = 1|\mathcal{I}'). \end{aligned} \tag{5.6}$$

$E_t(j)$ denotes the maximum probability when the interval of mode M_j finishes at time t, and is optimized for the mode sequence from time 1 to $t-1$ under the condition of given $\mathcal{I}'$. The probability with underline denotes that interval I_k with a triple $(b_k = t - \tau + 1, e_k = t, m_k = M_j)$ occurs just after the interval I_{k-1} that has mode $m_{k-1} = M_i$ and ends at $e_{k-1} = t - \tau$. We refer to this probability as an *interval transition probability*.

We recursively calculate the maximum probability for every mode that fin-

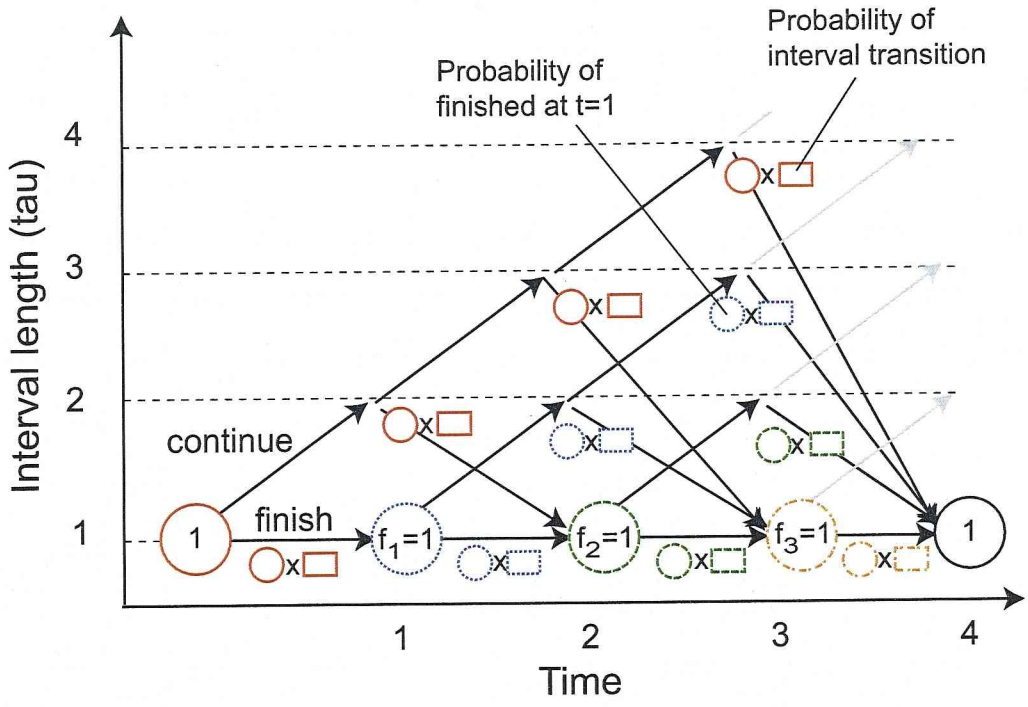


Figure 5.8: Lattice to search optimal interval sequence (num. of mode =2). We assume that $\sum_j P(m_T = M_j, f_T = 1|\mathcal{I}') = 1$

ishes at time $t(t = 1, ..., T)$ using Equation (5.6). After the recursive calculation, we find the mode index $j^* = \arg\max_j E_T(j)$. Then, we can get the duration length of the interval that finishes at time T with mode label M_{j*}, if we preserve τ that gives the maximum value at each recursion of Equation (5.6). Repeating this trace back, we finally obtain the optimized interval sequence and the number of intervals.

The remaining problem for the algorithm is the method of calculating the interval transition probability. As we see in the next subsection, this probability can be estimated from a trained timing structure model.

5.3.3 Calculation of Interval Transition Probability

As we described in previous subsection, the interval transition probability appeared Equation (5.6) is the transition from interval I_{k-1} to I_k. To simplify the notation, we here replace $t - \tau + 1$ with B_k. Let $e_{\min} = B_k$ and $e_{\max} = \min(T, B_k + l_{\max} - 1)$ be the minimum and maximum values of e_k, where $l_{\max}$ is the maximum length of the intervals. Let $I'_{k'}, ..., I'_{k'+R} \in \mathcal{I}'$ be reference inter-

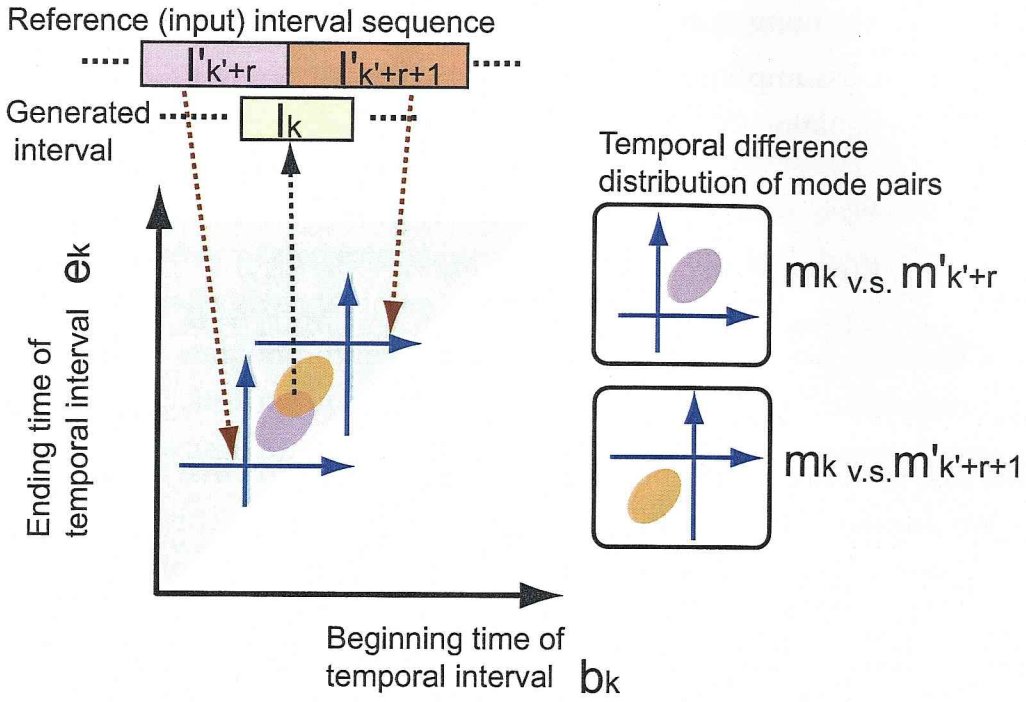


Figure 5.9: An interval probability calculation from the trained timing structure model.

vals that are possible to overlap with I_k. Assuming that the reference intervals are independent of each other (this assumption empirically works well), the interval transition probability can be calculated by the following equation:

$$\begin{aligned} & P(m_t = M_j, f_t = 1, l_t = \tau | m_{t-\tau} = M_i, f_{t-\tau} = 1, \mathcal{I}') \\ = \; & P(m_k = M_j, e_k, e_k \in [e_{\min}, e_{\max}] | m_{k-1} = M_i, b_k = B_k, I'_{k'}, ..., I'_{k'+R}) \\ = \; & \prod_{r=0}^{R} \{ \mathrm{Rect}(e_k, e_k \in [e_{\min}, b'_{k'+r} - 1]) \\ & + \kappa_r P(m_k = M_j, e_k, e_k \in [b'_{k'+r}, e_{\max}] | m_{k-1} = M_i, b_k = B_k, I'_{k'+r}) \} , \quad (5.7) \end{aligned}$$

where $\mathrm{Rect}(e, e \in [a, b]) = 1$ in the range $[a, b]$; else 0. Since the domain of e_k is $[e_{\min}, e_{\max}]$, Rect is out of range when $r = 0$, and $b'_{k'} = e_{\min}$. κ is a normalizing factor: $\kappa_r = 1\,(r = 0)$ and

$$\kappa_r = P(m_k = M_j, e_k, e_k \in [b'_{k'+r}, e_{\max}] | b_k = B_k, m_{k-1} = M_i)^{-1} \quad (r = 1, ..., R).$$

In the experiments, we assume κ_r is uniform for (m_k, e_k); thus, $\kappa_r = N(e_{\max} -$

$e_{\min}+1$) (N is the number of modes).

Using some assumptions that we will describe later, we can decompose the probability in Equation (5.7) as follows:

$$\begin{aligned} & P(m_k = M_j, e_k, e_k \in [b'_{k'+r}, e_{\max}] \mid m_{k-1} = M_i, b_k = B_k, I'_{k'+r}) \\ = \; & P(e_k \mid e_k \in [b'_{k'+r}, e_{\max}], m_k = M_j, b_k = B_k, I'_{k'+r}) \\ & \times P(m_k = M_j | e_k \in [b'_{k'+r}, e_{\max}], m_{k-1} = M_i, b_k = B_k, I'_{k'+r}) \\ & \times P(e_k \in [b'_{k'+r}, e_{\max}] \mid m_{k-1} = M_i, b_k = B_k) \end{aligned}$$

The first term is the probability of e_k under the condition that I_k overlaps with $I'_{k'+r}$. We assume that it conditionally independent of m_{k-1}. This probability can be calculated from Equation (5.1). Here, we omit the details of the deduction, and just make an intuitive explanation using Figure 5.9. First, an overlapped mode pair in I_k and $I'_{k'+r}$ provides a relative distribution of $(b_k - b'_{k'+r}, e_k - e'_{k'+r})$. Since $I'_{k'+r}$ is given, the relative distribution is mapped to the absolute time domain (the upper triangle region). Normalizing the distribution of (b_k, e_k) for $e_k \in [b'_{k'+r}, e_{\max}]$, we obtain the probability of the first term. The second term can be calculated using Equation (5.2) and (5.3). For the third term, we assume that the probability of $e_k \geq b'_{k'+r}$ is independent of $I'_{k'+r}$. Then, this term can be calculated by modeling temporal duration length l_t. In the experiments, we assumed uniform distribution of e_k and used $(e_{\max} - b'_{k'+r})/(e_{\max} - e_{\min} + 1)$.

The computational cost of interval transition probabilities strongly depends on the maximum interval length $l_{\max}$. If we successfully estimate the modes, $l_{\max}$ becomes comparatively small (i.e., balanced among modes) than the total input length. Thus, the cost becomes reasonable.

5.4 Experiments

To evaluate the descriptive power of the proposed timing structure model and the performance of the media conversion method, we first used simulated data for the verification of the interval generation algorithm described in Subsection 5.4.1. We then conducted the experiment that examines the overall media conversion flow shown in Figure 5.7 using audio and video data, and evaluated the precision of lip video generation from an input audio signal in Subsection 5.4.2.

5.4.1 Evaluation of Interval Sequence Generation Algorithm Using Simulated Data

To verify the interval generation algorithm described in Subsection 5.3.2, we input an interval sequence that comprised two modes $\mathcal{M}' = \{M'_1, M'_2\}$ and attempted to generate another related interval sequence that comprised $\mathcal{M} = \{M_1, M_2, M_3\}$ based on manually given temporal difference distributions.

Each of the temporal difference distribution was assumed to be a Gaussian function. Let $\mu_{i,p}$ be a mean vector of the temporal difference distribution of mode pair (M_i, M'_p) where $M_i \in \mathcal{M}$ (i.e., mode set for generating interval sequences) and $M'_p \in \mathcal{M}'$ (i.e., mode set for input interval sequences). Mean vectors $\mu_{i,p}$ $(i = 1,2,3, p = 1,2)$ were manually decided as follows:

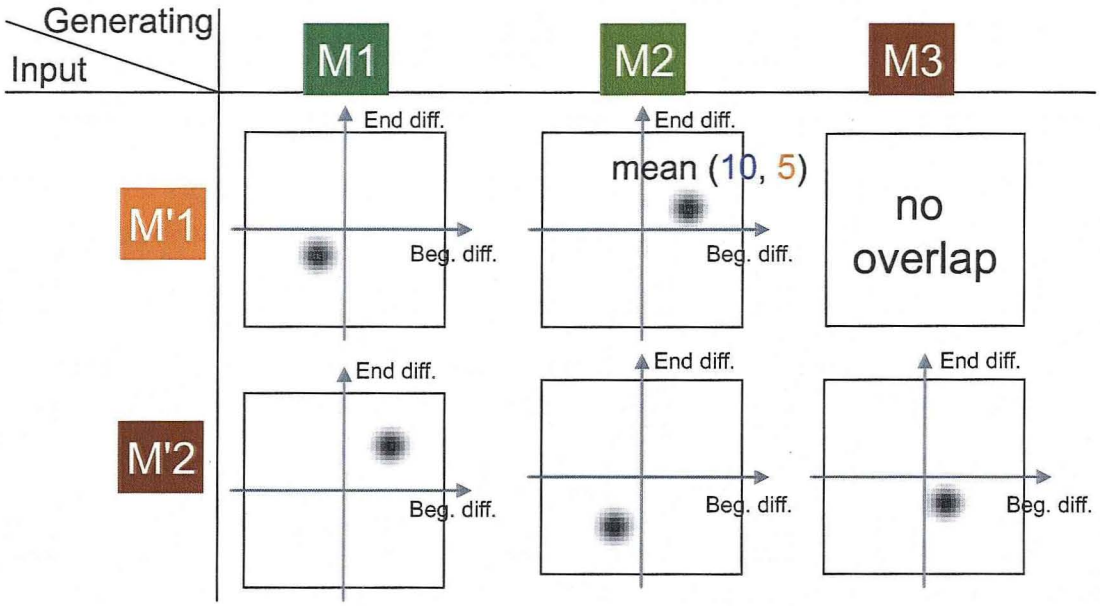
$$\begin{array}{lll} \mu_{1,1} = (-5,-5), & \mu_{2,1} = (10,5), & \mu_{3,1} : \text{not available}, \\ \mu_{1,2} = (10,10), & \mu_{2,2} = (-5,-10), & \mu_{3,2} = (5,-5), \end{array} \tag{5.8}$$

where mode pair (M_3, M'_1) was assumed to have no overlap. All the variances were set to be 4 and all the covariances were assumed to be zero. Figure 5.10 (a) shows the assumed temporal difference distributions.

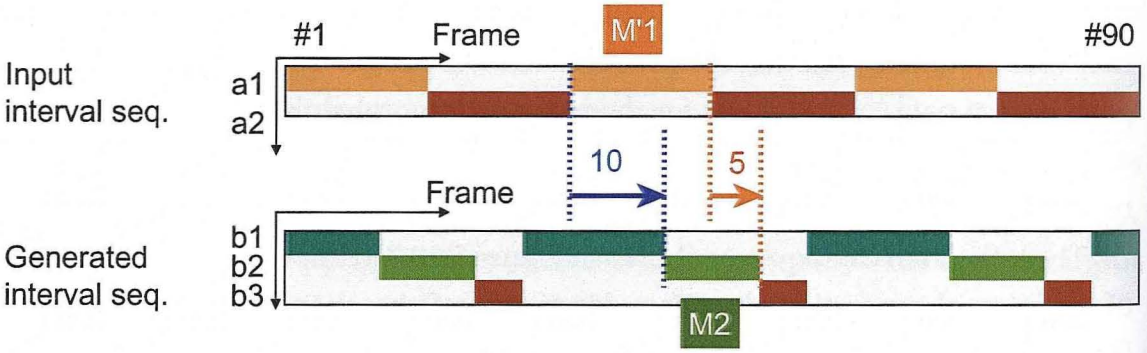
As for co-occurrence distribution defined in Equation (5.2), uniform distribution was assumed. That is, the probabilities were set to be 0.2 for all the mode pairs except pair (M_3, M'_1). As for mode transition probabilities, transition probabilities from M_1 to M_2, M_2 to M_3, and M_3 to M_1 were set to be one, and the remaining were set to be zero for generating cyclic transition of modes.

Then, the interval sequence shown in Figure 5.10 (b) (upper) was used as input of the interval generation algorithm described in Subsection 5.3.2. Figure 5.10 (b) (bottom) shows the generated interval sequence using the algorithm. We see that the temporal differences between beginning and ending points correspond to the elements of mean vectors $\mu_{i,p}$ of Gaussian distributions. For example, we see that the mode M_2 always begins ten frames after the beginning point of M'_1 begins, and finishes five frames after the ending point of M'_1.

We also examined other simulated data using different conditions such as the number of input modes is larger than that of generating modes. In those several experiments, we checked that the proposed algorithm always generated interval sequences in which each temporal interval of modes was determined so as to maximize the probability in Equation (5.4) with respect to given parameters. Consequently, the proposed timing structure model, especially temporal differ-



(a) Manually given temporal difference distributions (Gaussian).



(b) The input and generated interval sequences.

Figure 5.10: Verification of interval sequence generation from another related interval sequence described in Subsection 5.3.2 using manually given timing distributions.

ence distributions, successfully determines the temporal relation among modes appeared in input and generated interval sequences.

5.4.2 Evaluation of Image Sequence Generation from an Audio Signal

We applied the media conversion method described in Section 5.3 to the application that generates image sequences from an audio signal.

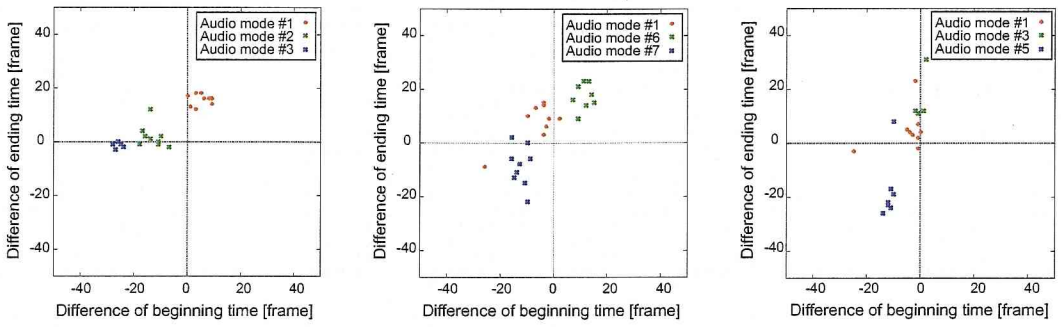
Feature Extraction

A continuous utterance of five vowels /a/,/i/,/u/,/e/,/o/ (in this order) was captured using mutually synchronized camera and microphone. This utterance was repeated nine times (18 sec.). The resolution of the video data was 720×480 and the frame rate was 60 fps. The sampling rate of the audio signal was 48 kHz (downsampled to 16 kHz in the analysis). Then, we applied short-term Fourier transform to the audio data with the window step of 1/60msec; thus, the frame rate corresponds to the video data.

Filter bank analysis was used for the audio feature extraction. We obtained 1134 frames of audio feature vectors each of which had dimensionality of 25, which corresponded to the number of filter banks. As for the video feature, a lip region in each video image was extracted by the active appearance model (AAM) [CET98] described in Section 4.4 (see also Appendix D for details). Then, the lip regions were downsampled to 32×32 pixels and the principal component analysis (PCA) was applied to the downsampled lip image sequence. Finally, we obtained 1134 frames of video feature vectors each of which had dimensionality of 27, which corresponded to the number of used principal components.

Learning the Timing Structure Model

Using the extracted audio and visual feature vector sequences as signal S' and S, we estimated the number of modes, parameters of each mode, and the temporal partitioning of each signal. We used linear dynamical systems for the models of modes. To estimate the parameters, we exploited hierarchical clustering of the dynamical systems described in Section 3.3. The estimated number of modes was 13 and 8 for audio and visual modes, respectively. The segmentation results are



(a) visual mode #1 (b) visual mode #5 (c) visual mode #7

Figure 5.11: Scattering plots of temporal difference between overlapped audio and visual modes. Visual mode #1, #5, and #7 corresponds to lip motion /o/ → /a/, /e/ → /o/, and /a/ → /i/, respectively

shown in Figure 5.12 (the first and second rows). Because of the noise, some vowels were divided into several different audio modes.

Temporal difference distributions of Equation (5.1), co-occurrence distributions of Equation (5.2), and mode transition probabilities of Equation (5.3) were estimated from the two interval sequences obtained in the segmentation process. Figure 5.11 shows the scattered plots of the samples that are temporal difference between beginning points and ending points of the overlapped modes appeared in the two interval sequences. Each chart shows samples of one visual mode to typical (two or three) audio modes. We see that the beginning motion from /a/ to /i/ synchronized with the actual sound (right chart) compared to the motion from /o/ to /a/ (left) and from /e/ to /o/ (middle). Applying Gaussian mixture models to these distributions, we estimated the temporal difference distributions. The numbers of the mixtures were manually determined.

Evaluation of Timing Generation

Based on the estimated cross-media timing-structure, we applied the media conversion method in Section 5.3. We used an audio signal interval sequence included in the training data of the interval system as an input (reference) media data (top row in Figure 5.12) and converted it into a video signal interval sequence (third row in Figure 5.12).

Then, to verify the performance of the media conversion method, we first compared the converted interval sequence with the original one, which was generated

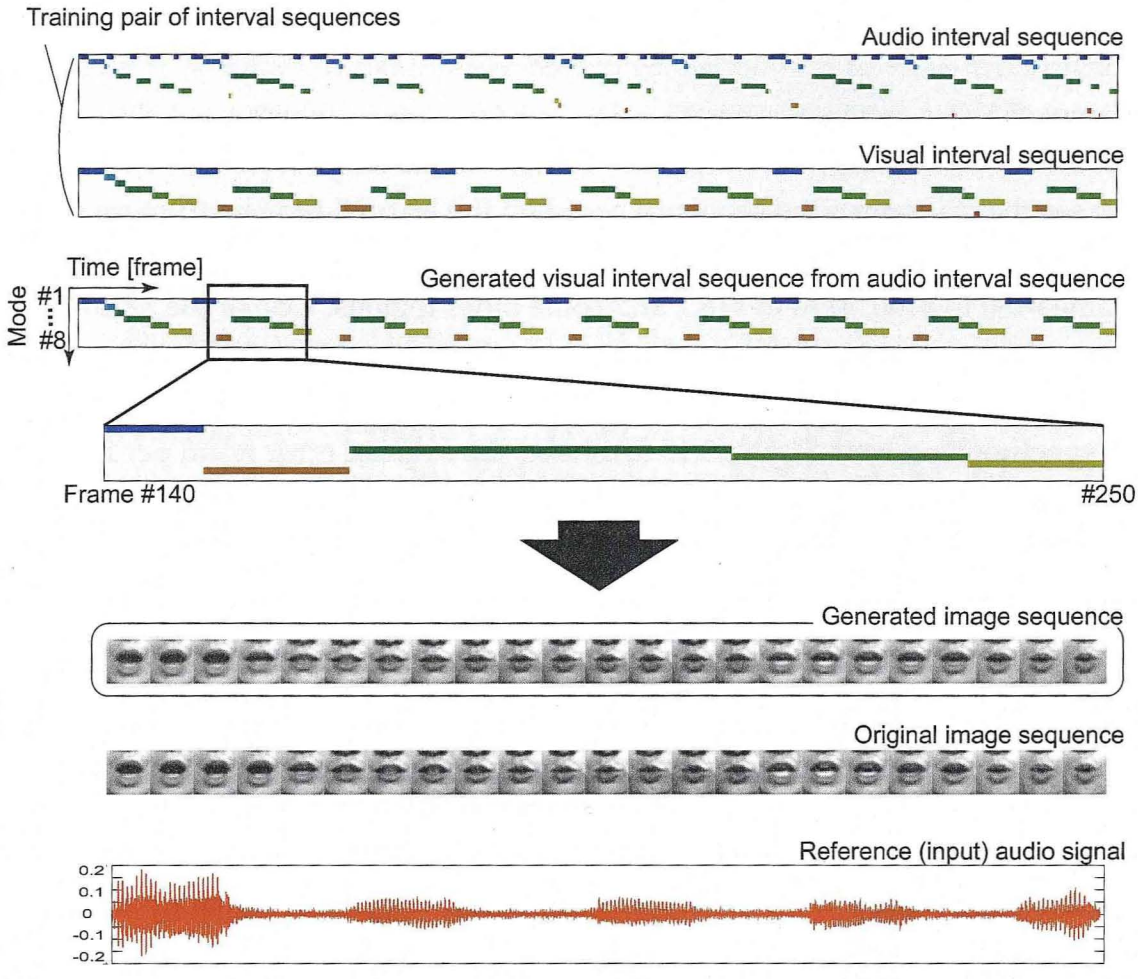


Figure 5.12: Generated visual interval sequence and an image sequence from the audio signal.

from the video data measured simultaneously with the input audio data (second row in Figure 5.12). Moreover, we also compared the pair of video data: one generated from the converted interval sequence (third bottom row in Figure 5.12) and the originally captured one (second bottom row in Figure 5.12), where images from frame #140 to #250 were shown in Figure 5.12. As for step 3, these image sequences were decoded from the visual feature vectors by the linear combination of principal axes (eigenvectors of PCA) and feature vectors (principal components). We also see the visual motion precedes the actual sound by comparing to the wave data (in the bottom row). From these data, the media conversion method seemed to work very well.

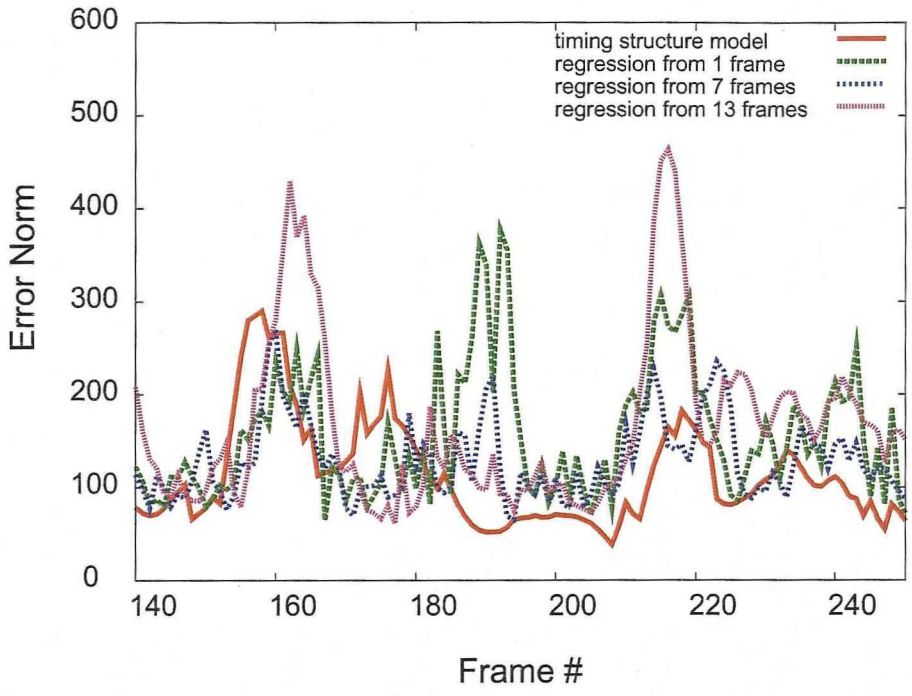
To quantitatively compare our method with others, we generated feature vec-

tor sequences based on several regression models. Seven regression models were constructed; each of the models estimated visual feature vector y_t from $2a+1$ frames of audio feature vectors $y_{t-a}, y_{t-a+1}, ..., y_t, ..., y_{t+a}$, where $a = 1, 2, ..., 7$. Figure 5.13 (a) shows the error norm of each frame in the range of frame # 140 to #250. We see that the generated sequence based on the learned timing structure model has small error values compared to other regression models except the range of frame #150 to #160, #170 to #180, and some other regions. One of the reasons the error of the timing-based method was larger than regression models is that these regions corresponded to such as vowel /i/, so the sound and visual motion might be synchronized well. Figure 5.13 (b) shows the average error norm per frame of each model. All the generated frames were used to calculate the average values. We see that the timing-based model provide the smallest error compared to the regression models.

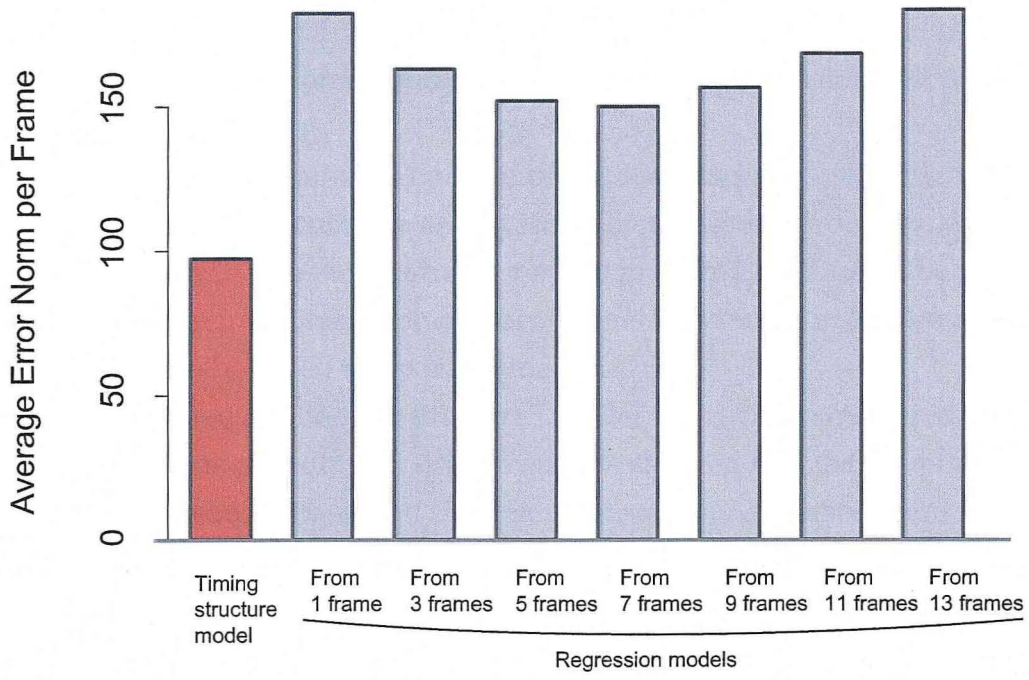
5.5 Discussion

We proposed a timing structure model that explicitly represents dynamic features in multimedia signals using temporal metric relations among intervals. The experiment shows that the model can be applied to one media signal from another signal across the modalities.

Although this is a preliminary result of evaluating the proposed timing models, its basic ability for representing temporal synchronization is expected to be useful for wide variety of areas. For example, human machine interaction systems including speaker tracking and audio-visual speech recognition, computer graphics such as generating motion from another related audio signals, and robotics such as calculating motion of each joint based on input events. We will discuss these points in Chapter 6 as feature work.



(a) Error norm of each frame in the range of frame #140 to #250.



(b) Average error norm (per frame).

Figure 5.13: Error norm of each frame and its average per frame between generated sequences and original sequence.

Chapter 6

Conclusion

6.1 Summary

In this thesis, we proposed a novel computational model, named an interval-based hybrid dynamical system, to model dynamic events and structures. As we described in Chapter 2, we exploited temporal intervals as an interface between dynamical systems, which is suitable for describing physical phenomena (consider time as physical metric entity), and discrete-event systems, which is suitable for describing human subjective or intellectual activities (consider time as ordinal state transition).

To overcome the paradoxical nature of the learning process, which requires to solve temporal segmentation and system identification problems simultaneously, we proposed a two-step learning method in Chapter 3. Due to the proposed method, we can extract linear dynamical systems that model primitive dynamics of the event from the given temporal signals.

In Chapter 4, we applied the proposed model to describe structured dynamic events that consists of multipart primitives. We showed that the systems can analyze dynamic features based on the timing structures extracted from temporal intervals. We examined the effectiveness of the timing structure analysis to discriminate fine-grained facial expression categories such as intentional and spontaneous smiles of which existing methods had difficulty to represent the difference.

In Chapter 5 we proposed a "timing structure model" that directly represents timing structures in multimedia signals, such as synchronization and mutual dependency with organized temporal differences among temporal patterns of media signals. Experiments on simultaneously captured audio and video data showed that time-varying signals of one media signal can be generated from an-

other related media signal using the trained timing structure model.

In the next section, we show some open issues that we were not able to cope with in this thesis, which can be divided into two aspects: (1) the extension of the proposed computational model, and (2) the situations that the model can be applicable including multiparty interaction.

6.2 Future Work

6.2.1 Extension of the Interval-Based Hybrid Dynamical System

In this subsection, we show some directions that the proposed interval-based hybrid dynamical system should be extended for future work.

(a) Non-linear Dynamical Systems

The selection of appropriate dynamical models depends on the nature of signals and the design policies of users. We chose to use linear dynamical systems because most of the continuously changing human motions can be approximated by linear dynamics. This is because the motions are generated by the expansion and contraction of muscles, and are controlled to be stable (e.g. no oscillation). However, nonlinear dynamical systems sometimes can be more reasonable choice for modeling time-varying patterns such as consonants in human speech. One of the straight forward methods to extend our model is the use of "kernel methods", which are major approach for the nonlinear data analysis. The kernel methods convert nonlinear algorithms in the original data space into linear algorithms in higher (or infinite) dimensional feature space. For example, kernel principal component analyses utilize inner products in the higher dimensional space [HTF01]. We need further discussion to give a guideline to select models.

Some motion generation researches in robotics design the overall system as a nonlinear dynamical system rather than a hybrid dynamical system. For example, Okada represented the motion of robots as a cyclic attractor in the configuration space [Oka95], and modeled the switching process between cyclic attractors in the configuration space based on continuous dynamics [ONN03]. Recurrent neural networks (RNN) also utilize its nonlinear dynamics to represent complex motion. Ogata et al. use the RNN with parametric bias (RNNPB) to extracting [OOK+05], which change its internal dynamics based on the additional input to the network (parametric bias). Morita et al. proposed RNN with non-monotonous function

for modeling temporal pattern recognition [MMM02] and extended the model to represent symbolic contexts of patterns using selective desensitization of some of elements [MMM02, MMMS04].

(b) Modeling Transition Process between Dynamical Systems in adjacent Intervals

The state in the internal state space often changes discontinuously when the automaton changes the dynamical systems. To model co-articulated dynamics such as phonemes in speech data and to generate smooth motion, we need transitional process modeling between two dynamical systems (e.g., the modulation of dynamics by the preceding dynamics). A straightforward method is to model the interpolation of two dynamics in adjacent intervals. Although the interpolation provides low-cost method to smoothing two dynamics, it sometimes generates unnatural motion at the joint of the two intervals. Li et al. proposed to set the end constraints of a synthesized segments [LWS02]. They deduced a block-banded system of linear equation from the constrains, and realized smooth motion texton synthesis by solving the equation.

(c) Modeling Temporal Structures among More Than Three Signals

While we concentrated on a timing structure model in two media signals in Chapter 5, we can apply the model to represent the structures among more than three media signals by defining pairs of signals and constructing timing structure models for each of the pairs similar to coupled HMMs [BO97]. On the other hand, we will be required to introduce other timing structure models if we consider a problem specific causality between signals. For example, we can introduce a unobservable interval sequence that controls a generation timing of observable media patterns. This model might be applied to a large area of human (animal) behavior because many of muscular motions are controlled by unobservable spike signals from a brain with physical delay [Pop85].

(d) Modeling Complex Structures of Dynamic Events

We exploited a simple finite state automaton as a discrete-event model in order to concentrate on modeling human body actions and motions, which have relatively simple grammatical structures (represented by a regular grammar) compared to natural languages. To model languages (e.g., human speech and signs) and other

complex situations (e.g., human communication and strategies), more complex grammatical structures such as N-gram models and context free grammars will be required together with its parameter estimation method.

There are several aspects of structures to extend our model. In the following, we show some of the existing approaches in machine learning and computer vision to represent complex structures.

Context-free grammars. A stochastic context-free grammar (SCFG), which defines probabilities of each of productions in a context-free grammar, is used for modeling grammatical structures among primitive events. Ivanov and Bobick use the SCFG model to recognize dynamic situations of parking area and human gestures [IB00]. Moore and Essa extended the model to detect errors and to recover the detected errors, and applied for modeling behavior of players involved in card game situation [ME02].

Layered structures. A hierarchical HMM (HHMM) was proposed by Fine, Singer and Tishby in machine learning community, and some computer vision applications were realized based on the model [NBVW03, BPV04]. The model is the extension of HMMs that each of states is capable to have not only output probability but a child HMM. As a result, the model can represent layered structure of events based on the recursive definition of HMMs. The HHMM is the simplified model of SCFG, therefore, the computation cost of probabilistic inference in HHMM is lower than that of SCFG.

Higher-order Markov models. Variable-length N-gram model was also proposed by Ron and Tishby [RST96]. They used a prediction suffix tree to represent and construct a variable-length N-gram model from an input symbol sequence. The model can be converted to a finite state automaton in which each state corresponds to a sequence of symbols. Galata et al. applied the model to represent long-term context of human behavior and provided some preliminary results [GJH01].

A key issue when we introduce layered structures of discrete states into the interval-based hybrid dynamical system is how to determine the layer in which temporal intervals and temporal relations among the intervals are defined. As we assumed in this thesis, a set of modes (dynamic primitives) corresponds directly to a set of discrete states, and the modes mapped one-to-one to the discrete-states. An intuitive extension is to consider a sequence of modes (linear dynamical systems) as a single discrete state based on the variable-length N-gram model. In a

sense, the sequence of modes constitute a "phrase" of dynamic primitives, and the discrete-state transition determines the sequence of the phrases. In this case, we can regard duration of phrase as an interval, and can introduce duration lengths of phrases. We can use a prediction suffix tree to train this model after the set of modes are determined by the clustering method proposed in Chapter 3.

6.2.2 Modeling Multiparty Interaction

In this thesis, we concentrated on applying the interval-based hybrid dynamical system to model a single human behavior rather than multiparty interaction, because our first concern is to see the effectiveness of the proposed model for modeling and learning dynamic events and structures from multimodal signals (see Figure 6.1 left). Extending the proposed scheme to model multiparty interaction and to realize human-machine interaction systems, we have to aim at finding key features of interaction dynamics or protocols in human-human communication, and exploiting the found features for natural and smooth human-machine communication (Figure 6.1 right).

We discuss how the proposed framework of the system can be applicable for modeling multiparty interaction and what are insufficient for our current system in the following paragraphs.

Timing Structures in Speech Conversation

In human conversation, there exist many lexical, prosodic and syntactic elements that help create the dialog structure [Shi05]. Especially, utterance timing (such as transition interval in Figure 6.1 upper-right) and speaking tempo can help to add a smooth, tense, lively or relaxing tone to the dialog. It is with this tone that dialog can evoke feelings of pride, sorrow, fear, and enjoyment. Since we humans are exposed to a large amount of timing structures in speech dialog during their development, in other words, we are professional to recognize and generate timings; the users are sensitive to unnatural utterance timing and speaking tempo of speech dialog systems.

As for the analysis of the timing structures and their effects in human speech dialogs, Ichikawa and Sato showed that many of backchannel utterances occur within about 0.4 second after the keyword appears in the speech of the others [IS94]. Nagaoka et al. analyzed dialog of operators and customers in a telephone shopping situation, and showed that utterance timings are one of the es-

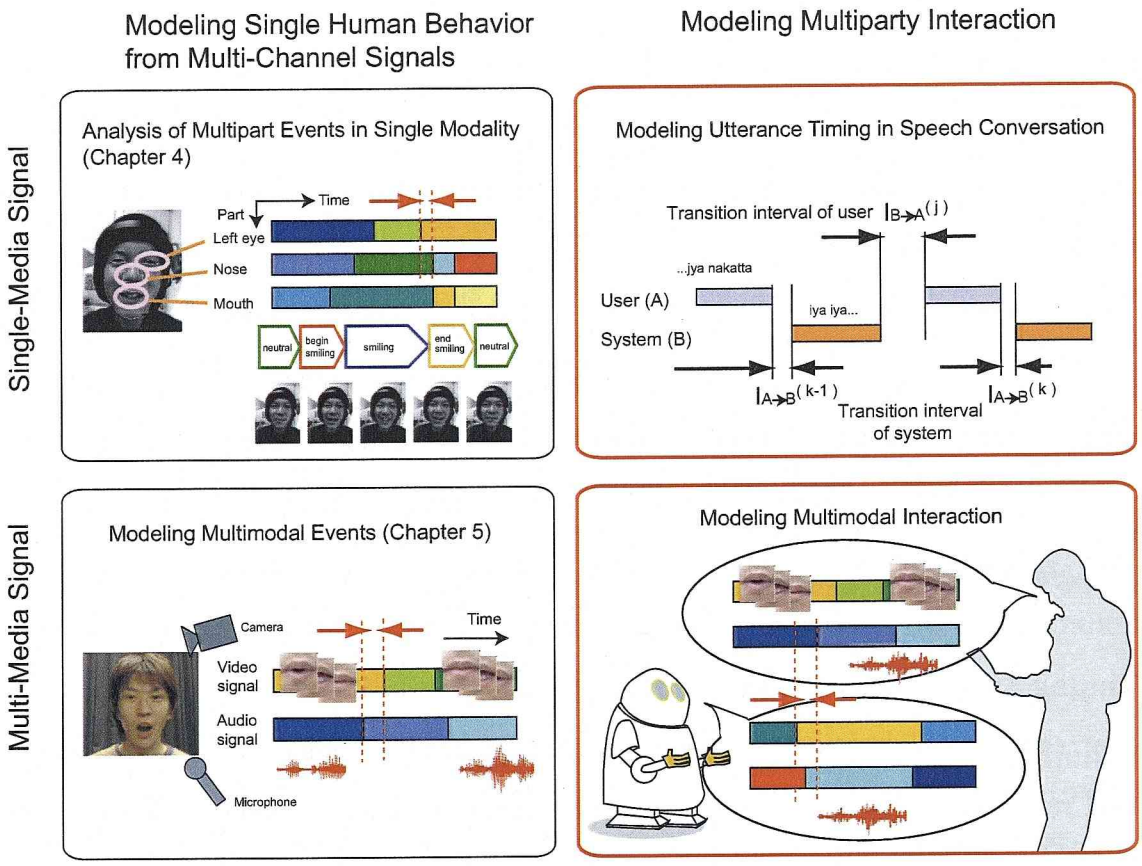


Figure 6.1: Extension to Multiparty Interaction. We investigated left half of the figure in this thesis. (See also Figure 1.11.)

sential cues for determining an impression about the speaker [NDKN02]. As for dialog systems that control utterance timing, Okato proposed the use of pitch patterns in utterances to estimate the timing of backchannel responses. Kitaoka et al. developed a response timing generator for speech dialog systems based on the use of pitch and power patterns, power, utterance lengths, and some lexical information [KTNN05]. Fujie developed a robot that makes backchannel feedbacks with overlaps based on the use of pitch and power patterns [FFK05].

Although the existing approaches generate timing of backchannel utterances, non-backchannel utterances also have significant features in its generation timings. We have to use more fine categories of utterances based on the functions or purpose of the speech in order to realize dialog systems that communicate purpose, attitude and feeling of the spoken word. The use of dialog acts (e.g. "question", "statement", "opinion", etc.), which correspond to illocutionary acts in speech acts [Sea86], will help to analyze and categorize backchannel and non-backchannel utterances based on the functions of them [JSFC98, SRC+00, DBCS04]. Once the categories of dialog acts are defined, we can learn the timing structure model for each of the dialog acts, and can apply the timing generation algorithm described in Chapter 5.

Timing Structures in Multimedia Interaction

Another disadvantage of existing timing generation applications is that the systems have no unified framework to integrate multimodal information captured as different media signals. For example, human utterance timings are defined by not only audio information but visual features such as facial expressions and lip motions of others. We can also see that the facial expression of one person affects the others expressions in our daily communication.

As we described in this thesis, the interval-based hybrid dynamical system has a capability of modeling the mutual dependency among multiple signals. We can therefore exploit this system to realize human-machine interaction systems by extending the system to model timing structures among more than three signals (see Subsection 6.2.1 (c)). The use of nonlinear dynamical systems and more complex structures should be considered to represent these general interaction patterns (see Subsection 6.2.1 (a) and (d)).

In addition, a timing structure model is useful to estimate internal state of humans considering the result of the facial expressions analysis in Chapter 4. Estimation of human internal states including intensions, interests, emotions, and

other unobservable mental events are essential for providing appropriate information to others. Because these mental events often affect our body via a neural system deliberately or involuntary, we can observe distinctive dynamic features of signals from multiple modalities that is sufficient for estimating internal states of others. Especially, we remark the following points for taking advantage of timing structure models for the internal state estimation:

- Internal states affects the timing structure (e.g., pause, tempo, and rhythms) of pitch and power in speech, gestures, eye gaze, gait motion, and other observable media signals from human activities.

- The timing of human reaction is affected by his or her internal states; for example, we can see some time lag of the response when the person concentrated on other things.

From the timing structures above, information systems will understand user's aims and situations, and will provide kind and timely guidance, which are the most important functions for realizing human-centered communication.

Consequently, the extension of the interval-based hybrid dynamical systems can be a fundamental basis of interaction systems that share a sense of time with humans based on the integration of physical and subjective time as we described in Chapter 1.

6.2.3 Hybrid Computing in Robotics

In this subsection, we show how the proposed concept of the interval-based hybrid dynamical system can be applicable to the area of robotics, especially the degree of freedom (DOF) of robots becomes very high (e.g., humanoids).

The existing methods to realize robots that control the body motion can be categorized into model-based approaches and behavior-based approaches. A model-based approach calculates and plans body motions and actions at the inside of computers based on the knowledge of the real world and robot bodies of robots (e.g., reasoning agents [RN02, PS99] and inverse-dynamics based controls [KYHH05]). A behavior-based approach exploits the interaction between robot bodies and the environment to emerge robot actions without modeling the real world or body (e.g., subsumption architectures [Bro86, Bro91] and a passive walk [CRTW05]). The integration methods of two approaches, which use different computing resources, are under the investigation [YK02]. This stream is

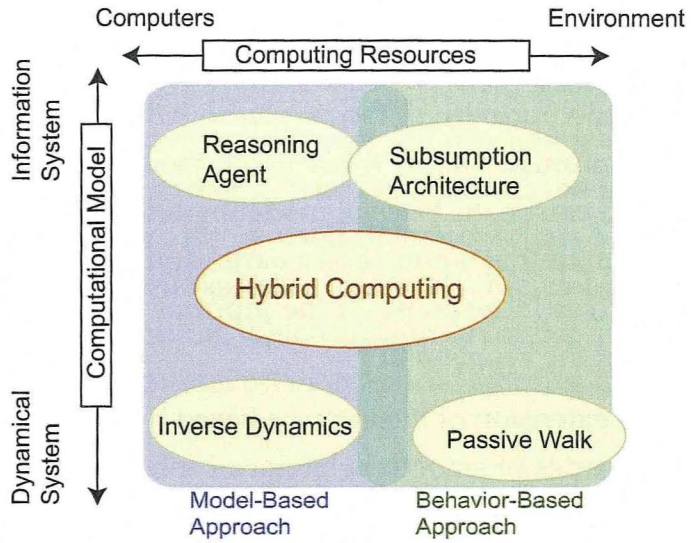


Figure 6.2: A Map of Robotics Approaches.

regarded as an integration of the horizontal direction in Figure 6.2.

On the other hand, these approaches can be divided based on different aspect depicted by the vertical direction in Figure 6.2: dynamical systems and information systems. For example, both the inverse-dynamics-based control and agent reasoning can be regarded as model-based approaches, which utilize the knowledge of the world including body of robots, the representation of the system is however completely different. That is, the inverse-dynamics-based control exploits dynamical systems and the agent reasoning uses information systems.

There exist some methods to integrate these two systems. The characteristics of these methods are that they first define some action primitives such as "move left hand up" and "move right leg forward", and then integrates these action primitives based on information systems such as finite state machines; meanwhile, each of primitives are realized by calculation of torque based on dynamical systems. The methods however have disadvantages that the primitives become too coarse and abstract to generate smooth motion. This is because the primitives are defined manually, and the temporal scale of each primitive cannot become smaller than human recognizable scales (Problem A). It is also difficult to define enough number of primitives when the DOF of robots becomes very high such as in humanoids (Problem B). Moreover, appropriate energy input timings are important to realize dynamic motion similar to human [KOT+04] (Problem C).

As we described in Chapter 3, the interval-based hybrid dynamics system pro-

posed in this thesis has a capability of learning a set of dynamic primitives from the given time-varying signals, it has therefore a potential to solve Problem A and B. The size of learned primitives can be controlled to be smaller than our intentional segmentation units. On the other hand, the timing generation method described in Chapter 5 can be the basis of modeling timing structure between the primitives (e.g., motion timing among body parts) and of determining activation timing of each primitives in response to the input or recognized events, which leads to the solution of the Problem C.

Consequently, the extension of the interval-based hybrid dynamical system is expected to realize a series of smooth behavior of robots, including humanoids, in dynamic situations where a robot contacts with objects and humans.

6.2.4 Relation to Human Consciousness

In this subsection, we consider the relation of the concepts in the interval-based hybrid dynamical system to "human consciousness"[1]. We do not intend to discuss what consciousness is and where the consciousness exists in our human body; consciousness comprises many aspects, and the definition of consciousness often differs from person to person. Our motivation here is focusing on one important aspect of human consciousness, the function of "temporal coordination", to bring some crucial issues as extending interval-based hybrid dynamical systems.

Why we human enjoy rhythms? The reason that the ability to enjoy rhythmic patterns has acquired during the process of evolution may not be only for playing music. We human are required to control and coordinate the timing of a series of actions in response to perceived events. Therefore, the sensitivity to dynamic structures among various events, which we described in Subsection 1.4.2, are indispensable for humans to survive in the real world. Especially, the temporal coordination with consciousness among dynamic events must be advantageous to humans under selection pressure compared to coordination in subconscious.

[1]The discussion in this subsection may also be applicable to animal consciousness. It is interesting to consider the difference of consciousness between humans and animals [Ecc89]; however, this topic is beyond the scope here.

Essential Features of Temporal Coordination with Consciousness

We first consider the essential features that constitute temporal coordination with consciousness. A necessary condition for the function is to handle events (e.g., perceived input and generating actions) apart from the physical-time domain. However, this feature can also be used in subconscious temporal coordination. For example, human sometimes use a clutch of a manual-transmission car without awareness of the operation, which requires dynamic structure among multiple events. We here concentrate on more crucial features of the temporal coordination:

- A single time axis is used in mind for coordinating among multiple events (e.g., action and utterance). While multiple processes can be unconsciously activated in parallel [Lib04] (e.g., control of multiple body parts), the unified time in mind work as coordinator to maintain the consistency among the processes.

- Crucial time points that exist in various abstraction levels are dynamically selected, logically combined, and coordinated. While multiple time points of discrete events are recognizable, some points are crucial to achieve an overall action (e.g., "knacks" of robot action [KOT+04]). We pay attention to those crucial time points as the occasion demands. Once the crucial time points are coordinated, the dynamic structure among discrete events in lower abstraction levels is also coordinated unconsciously.

Hence, we can handle dynamic structures that have non-fixed patterns by exploiting these features.

Learning of Structure among Discrete Events in Multiple Abstraction Levels

As for learning of a novel action such as a gymnastic exercise, crucial points are also variable. We human first find the temporal ordering relations among sensory information (e.g., visual input) and muscular activation. We then search the timing among perceived and generating events to realize the best performance of the action.

Once the action is acquired, we can orchestrate the control of multiple body parts in response to perceived input without awareness if the structure among events is fixed or simple enough; meanwhile, the learning phase requires awareness of fine-grained events that determine the performance of the action. In other

words, dynamic structures of acquired actions are push down to subconscious domain for concentrating on obtaining and realizing more complex or compositional actions that have structures in higher abstraction levels.

Direction to Extend the Interval-Based Hybrid Dynamical System

Compared to the temporal coordination with human consciousness described above, the interval-based hybrid dynamical system proposed in this thesis is quite restricted. As we discussed in Subsection 6.2.1, the interval system finds temporal points of discrete events based on linear dynamics and it controls only a single level of dynamic structure among those time points. As a more important issue, the timing generation method proposed in Chapter 5 is only able to control the temporal position of discrete events that have simple static distributions. In a sense, the system handles the subconscious coordination of events.

Considering temporal coordination in human, we anticipate the following features are essential to design information systems that fulfill the enough functions of human-machine interaction (Subsection 6.2.2) and robotics (Subsection 6.2.3):

- The mechanism that dynamically finds the crucial points in multiple abstraction levels based on the context and situation
- Temporal coordination function that maintains consistency of lower abstraction levels
- Learning method that reuses the dynamic structures obtained in the past learning to construct more complex structures

The function of temporal coordination in human consciousness also affects the number of events of which human is aware, and may influence the length of cognitive time in the experience, which attract many scientists' interest [Tsu87]. We believe the design of the computational model that has satisfactory functions to continue and survive in the real situations is necessary not only for engineering purpose but also for understanding the mechanisms of mind process, such as cognitive sense of time, in humans and animals. We hope the concept of the interval-based hybrid dynamical system serve as the first step of these objectives.

Appendix A

Matrix Formulas

A.1 Basics

Let A, B, and C be arbitrary matrices, and suppose that the number of rows and columns are chosen appropriately. Then we can use the following relations:

$$A(B+C) = AB+AC, \tag{A.1}$$
$$(A+B)^\top = A^\top + B^\top, \tag{A.2}$$
$$(AB)^\top = B^\top A^\top, \tag{A.3}$$
$$(A^{-1})^\top = (A^\top)^{-1}. \tag{A.4}$$

Let $\lambda_1, ..\lambda_n$ be the eigenvalues of matrix A that are sorted by descending order. We can use the following relations as for the determinant $|A|$:

$$|AB| = |A||B|, \tag{A.5}$$
$$|A^{-1}| = \frac{1}{|A|}, \tag{A.6}$$
$$|BAB^{-1}| = |B||A|\frac{1}{|B|} = |A|, \tag{A.7}$$
$$|A| = \prod_i \lambda_i, \tag{A.8}$$
$$|A^\top| = |A|, \tag{A.9}$$
$$|aA| = a^n|A|, \tag{A.10}$$
$$|-A| = (-1)^n|A|. \tag{A.11}$$

As for the trace $\mathrm{tr}(A)$, we can use

$$\begin{aligned} \mathrm{tr}(A+B) &= \mathrm{tr}(A)+\mathrm{tr}(B), && \text{(A.12)} \\ \mathrm{tr}(A) &= \sum_i \lambda_i, && \text{(A.13)} \\ \mathrm{tr}(ABC) &= \mathrm{tr}(BCA)=\mathrm{tr}(CAB). && \text{(A.14)} \end{aligned}$$

As for the rank and condition number of matrix A, there exist the following relations:

$$\begin{aligned} \mathrm{rank}(A) &= \mathrm{rank}(A^\top A)=\mathrm{rank}(AA^\top), && \text{(A.15)} \\ \text{condition number}(A) &= \sqrt{\frac{\lambda_1}{\lambda_n}}. && \text{(A.16)} \end{aligned}$$

A.2 Differential Formulas of Matrices

Suppose $x = [x_1, ..., x_n]^\top \in \mathbf{R}^n$ and $X = [x_{ij}]$ is $n \times m$ matrix. We define the following notations to denote partial differential of each vector (matrix) element.

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

$$\frac{\partial f}{\partial X} = \begin{bmatrix} \frac{\partial f}{\partial x_{11}} & \cdots & \frac{\partial f}{\partial x_{1m}} \\ \vdots & & \vdots \\ \frac{\partial f}{\partial x_{n1}} & \cdots & \frac{\partial f}{\partial x_{nm}} \end{bmatrix}$$

A.2.1 Formulas

$$\begin{aligned} \frac{\partial}{\partial X}\mathrm{tr}(X^\top A) &= A && \text{(A.17)} \\ \frac{\partial}{\partial X}\mathrm{tr}(X^\top AX) &= (A+A^\top)X && \text{(A.18)} \\ \frac{\partial}{\partial X}\mathrm{tr}(X^\top AXB) &= AXB+A^\top XB^\top && \text{(A.19)} \\ \frac{\partial}{\partial X}\log|X| &= (X^\top)^{-1} && \text{(A.20)} \\ \frac{\partial}{\partial X}|X| &= (X^\top)^{-1}|X| && \text{(A.21)} \end{aligned}$$

A.2.2 Examples

Differential Formulas of Vectors

From Equation (A.17) and Equation (A.18), we obtain the following vector formulas:

$$\begin{aligned} \frac{\partial}{\partial x} x^\top a &= a \\ \frac{\partial}{\partial x} x^\top A x &= (A + A^\top)x = 2Ax \end{aligned}$$

Useful Formulas

$$\begin{aligned} \frac{\partial}{\partial X}||AX+B||^2 &= \frac{\partial}{\partial X}\mathrm{tr}((AX+B)^\top(AX+B)) && \text{(A.22)} \\ &= \frac{\partial}{\partial X}\mathrm{tr}(X^\top A^\top AX + X^\top A^\top B + B^\top AX + B^\top B) && \text{(A.23)} \\ &= 2A^\top AX + 2A^\top B && \text{(A.24)} \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial X}\mathrm{tr}(A^\top XX^\top A) &= \frac{\partial}{\partial X}\mathrm{tr}(X^\top AA^\top X) && \text{(A.25)} \\ &= 2AA^\top X && \text{(A.26)} \end{aligned}$$

Appendix B

Estimation of the Transition Matrices and Bias Vectors of Linear Dynamical Systems

In this appendix, we show how to estimate the transition matrix $F^{*(i)}$ and bias vector $g^{*(i)}$ from the internal state sequence $x_b^{(i)}, .., x_e^{(i)}$ in temporal range $[b, e]$, which is represented by linear dynamical system D_i. The results described here corresponds to Equation (3.5) and Equation (3.6).

As we introduced in Subsection 3.3.3, we use the following notations:

$$X_0^{(i)} \triangleq [x_b^{(i)}, ..., x_{e-1}^{(i)}], \quad X_1^{(i)} \triangleq [x_{b+1}^{(i)}, ..., x_e^{(i)}] \tag{B.1}$$

$$m_0^{(i)} \triangleq \frac{1}{l-1} \sum_{t=b}^{e-1} x_t^{(i)} = \frac{1}{l-1} X_0^{(i)} \underbrace{[1, ..., 1]}_{l-1}^{\top} \tag{B.2}$$

$$m_1^{(i)} \triangleq \frac{1}{l-1} \sum_{t=b+1}^{e} x_t^{(i)} = \frac{1}{l-1} X_1^{(i)} \underbrace{[1, ..., 1]}_{l-1}^{\top}, \tag{B.3}$$

where $l = e - b + 1$. Using the notations above, we can rewrite Equation (3.3) as

the following:

$$\sum_{t=b+1}^{e} ||\epsilon_t||^2 = ||X_1^{(i)} - (F^{(i)}X_0^{(i)} + g^{(i)}[\underbrace{1,...,1}_{l-1}])||^2 \tag{B.4}$$

$$\begin{aligned} &= \mathrm{tr}\left(X_1^{(i)} - (F^{(i)}X_0^{(i)} + g^{(i)}[1,...,1])\right)^\top \left(X_1^{(i)} - (F^{(i)}X_0^{(i)} + g^{(i)}[1,...,1])\right) \\ &= \mathrm{tr}(X_1^{(i)\top}X_1^{(i)}) + \mathrm{tr}\left((F^{(i)}X_0^{(i)} + g^{(i)}[1,...,1])^\top (F^{(i)}X_0^{(i)} + g^{(i)}[1,...,1])\right) \\ &\quad -\mathrm{tr}\left(X_1^{(i)\top}(F^{(i)}X_0^{(i)} + g^{(i)}[1,...,1])\right) - \mathrm{tr}\left((F^{(i)}X_0^{(i)} + g^{(i)}[1,...,1])^\top X_1^{(i)}\right) \end{aligned}$$

Using Equation (B.2) and Equation (B.3) with $\mathrm{tr}(ABC) = \mathrm{tr}(BCA) = \mathrm{tr}(CAB)$ and $(AB)^\top = B^\top A^\top$ (see Appendix A), we obtain

$$\begin{aligned} \sum_{t=b+1}^{e} ||\epsilon_t||^2 &= \mathrm{tr}(X_1^{(i)\top}X_1^{(i)}) + \mathrm{tr}(F^{(i)}X_0^{(i)}X_0^{(i)\top}F^{(i)\top}) + (l-1)\mathrm{tr}(g^{(i)\top}g^{(i)}) \\ &\quad +(l-1)\mathrm{tr}(F^{(i)\top}g^{(i)}m_0^{(i)\top}) + (l-1)\mathrm{tr}(m_0^{(i)}g^{(i)\top}F^{(i)}) \\ &\quad -\mathrm{tr}(F^{(i)}X_0^{(i)}X_1^{(i)\top}) - (l-1)\mathrm{tr}(g^{(i)}m_1^{(i)\top}) \\ &\quad -\mathrm{tr}(X_1^{(i)}X_0^{(i)\top}F^{(i)\top}) - (l-1)\mathrm{tr}(g^{(i)\top}m_1^{(i)}). \end{aligned}$$

If $l-1 >= n$ (i.e., the number of samples is equal to or greater than the dimensionality of state vectors), we can estimate transition matrix $F^{*(i)}$ and bias vector $g^{*(i)}$ by solving the least squares problem of Equation (3.4). To solve this minimization problem, we first differentiate $\sum_{t=b+1}^{e} ||\epsilon_t||^2$ with respect to each of $F^{(i)}$ and $g^{(i)}$:

$$\begin{aligned} \frac{\partial}{\partial F^{(i)}} \sum_{t=b+1}^{e} ||\epsilon_t||^2 &= 2\left\{F^{(i)}X_0^{(i)}X_0^{(i)\top} + (l-1)g^{(i)}m_0^{(i)\top} - X_1^{(i)}X_0^{(i)\top}\right\}, \\ \frac{\partial}{\partial g^{(i)}} \sum_{t=b+1}^{e} ||\epsilon_t||^2 &= 2(l-1)\left\{g^{(i)} + F^{(i)}m_0^{(i)} - m_1^{(i)}\right\}. \end{aligned}$$

Then, we set zero for all the differentiated elements and obtain the following equations:

$$F^{*(i)}X_0^{(i)}X_0^{(i)\top} + (l-1)g^{*(i)}m_0^{(i)\top} - X_1^{(i)}X_0^{(i)\top} = O, \tag{B.5}$$

$$g^{*(i)} + F^{*(i)}m_0^{(i)} - m_1^{(i)} = 0. \tag{B.6}$$

From Equation (B.6), we obtain

$$g^{*(i)} = m_1^{(i)} - F^{*(i)} m_0^{(i)}. \tag{B.7}$$

Substituting this equation into Equation (B.5), we obtain

$$F^{*(i)} X_0^{(i)} X_0^{(i)\top} + (l-1)(m_1^{(i)} - F^{*(i)} m_0^{(i)}) m_0^{(i)\top} - X_1^{(i)} X_0^{(i)\top} = O. \tag{B.8}$$

Using Equation (B.2) and Equation (B.3) again, Equation (B.8) can be replaced by

$$F^{*(i)} X_0^{(i)} X_0^{(i)\top} + (X_1^{(i)}[1,...,1]^\top - F^{*(i)} X_0^{(i)}[1,...,1]^\top) m_0^{(i)\top} - X_1^{(i)} X_0^{(i)\top} = O.$$

Let $\hat{X}_0^{(i)}$ be centerized $X_0^{(i)}$:

$$\hat{X}_0^{(i)} \triangleq X_0^{(i)} - m_0^{(i)}[1,...,1] = [x_b^{(i)} - m_0^{(i)}, ..., x_{e-1}^{(i)} - m_0^{(i)}], \tag{B.9}$$

then we obtain

$$F^{*(i)} X_0^{(i)} \hat{X}_0^{(i)\top} = X_1^{(i)} \hat{X}_0^{(i)\top}. \tag{B.10}$$

Here, we can replace $X_0^{(i)} \hat{X}_0^{(i)\top}$ as $\hat{X}_0^{(i)} \hat{X}_0^{(i)\top}$ using Equation (B.2):

$$\begin{aligned}
\hat{X}_0^{(i)} \hat{X}_0^{(i)\top} &= (X_0^{(i)} - m_0^{(i)}[1,...,1]) \hat{X}_0^{(i)\top} \\
&= X_0^{(i)} \hat{X}_0^{(i)\top} - m_0^{(i)}[1,...,1](X_0^{(i)} - m_0^{(i)}[1,...,1])^\top \\
&= X_0^{(i)} \hat{X}_0^{(i)\top} - (l-1) m_0^{(i)} m_0^{(i)\top} + (l-1) m_0^{(i)} m_0^{(i)\top} \\
&= X_0^{(i)} \hat{X}_0^{(i)\top}
\end{aligned}$$

Similarly,

$$\begin{aligned}
\hat{X}_1^{(i)} \hat{X}_0^{(i)\top} &= (X_1^{(i)} - m_1^{(i)}[1,...,1]) \hat{X}_0^{(i)} \\
&= X_1^{(i)} \hat{X}_0^{(i)} - m_1^{(i)}[1,...,1](X_0^{(i)} - m_0^{(i)}[1,...,1])^\top \\
&= X_1^{(i)} \hat{X}_0^{(i)\top} - (l-1) m_1^{(i)} m_0^{(i)\top} + (l-1) m_1^{(i)} m_0^{(i)\top} \\
&= X_1^{(i)} \hat{X}_0^{(i)\top}
\end{aligned}$$

Thus, we finally get

$$\begin{aligned}
F^{*(i)} \hat{X}_0^{(i)} \hat{X}_0^{(i)\top} &= \hat{X}_1^{(i)} \hat{X}_0^{(i)\top}, \\
\therefore \quad F^{*(i)} &= \hat{X}_1^{(i)} \hat{X}_0^{(i)\top} (\hat{X}_0^{(i)} \hat{X}_0^{(i)\top})^{-1}
\end{aligned} \tag{B.11}$$

On the other hand, if $l-1 < n$ (i.e., the number of samples is smaller than the dimensionality of state vectors), the solution of the minimization problem does not fix. From Equation (B.4), the special solution of the minimization problem becomes

$$\begin{aligned} F^{*(i)} &= X_1^{(i)}(X_0^{(i)\top}X_0^{(i)})^{-1}X_0^{(i)\top} \\ &= X_1^{(i)}(I-A)((I-A)X_0^{(i)\top}X_0^{(i)}(I-A))^{-1}(I-A)X_0^{(i)\top} \\ &= \hat{X}_1^{(i)}(\hat{X}_0^{(i)\top}\hat{X}_0^{(i)})^{-1}\hat{X}_0^{(i)\top}, \qquad \text{(B.12)} \\ g^{*(i)} &= m_1^{(i)} - F^{*(i)}m_0^{(i)}, \qquad \text{(B.13)} \end{aligned}$$

where $A = [1,...,1]^\top[1,...,1]/(l-1)$. The total prediction error becomes zero, if we use the above solution.

Note that both Equation (B.11) and (B.12) satisfy Equation (3.5), which is described by Moore-Penrose generalized inverse.

Appendix C

Gershgorin's Theorem

The Gershgorin's theorem is a well known method to describe a region in the complex plane $\{z \mid z \in \mathbf{C}\}$ that contains all the eigenvalues of a complex square matrix [Iri03].

Theorem: Let $A = [a_{ij}]$ be an arbitrary $n \times n$ complex square matrix, and define r_i as :

$$r_i \triangleq \sum_{j=1, j \neq i}^{n} |a_{ij}| \quad (i = 1, 2, ..., n).$$

Then, all the eigenvalues of matrix A exist in the union of circles $\cup_{i=1}^{n} C_i$, where

$$C_i = \{z \mid z \in \mathbf{C}, |z - a_{ii}| \leq r_i\}.$$

Corollary: Let $A = [a_{ij}]$ be an arbitrary $n \times n$ complex square matrix, and let $\lambda_i \, (i = 1, ..., n)$ be the eigenvalues of matrix A. The maximum absolute value (spectral radius) of these eigenvalues satisfies the following relation:

$$\max_i |\lambda_i| \leq \max_i \sum_{j=1}^{n} |a_{ij}|. \tag{C.1}$$

Appendix D

Active Appearance Model

The AAM-based feature point tracking consists of two stages. We first build an AAM model using a training set of face images and its feature points given manually. Then, we use the model to extract facial feature points in novel images.

To AAM build the model, we require a training set of images marked with feature points. Figure 4.4 (a) shows an example of a face image labeled with 58 feature points. Let s be a shape vector that represents the coordinate value of feature points. Let g be a grey-level vector that represents the intensity information from the shape-normalized image over the region covered with the mean shape. In the first step, the method applies principal component analysis (PCA) to the data. Any example image can then be approximated using:

$$s = \bar{s} + U_s c_s \ , \ g = \bar{g} + U_g c_g, \tag{D.1}$$

where $\bar{s}$ and $\bar{g}$ are the corresponding sample mean vectors, U_s and U_g are matrices of column eigenvectors of the shape and grey-level, and c_s and c_g are vectors of shape and grey-level parameters, respectively. In the second step, because there may be correlations between the shape and grey-level variation, the method concatenates the vectors c_s and c_g, applies PCA, and obtains a model of the form

$$\begin{bmatrix} W_s c_s \\ c_g \end{bmatrix} = c = \begin{bmatrix} V_s \\ V_g \end{bmatrix} d = Vd, \tag{D.2}$$

where W_s is a diagonal matrix of weights for each shape parameter, allowing for the difference in units between the shape and grey-level models, V is a matrix of column eigenvectors, and d is a vector of appearance parameters controlling both the shape and grey-levels of the model.

Note that the linear nature of the model allows us to express the shape vector s and grey-level vector g directly as functions of d:

$$s = \bar{s} + U_s W_s^{-1} V_s d \ , \ g = \bar{g} + U_g V_g d. \tag{D.3}$$

An example image can be synthesized for a given d by generating the shape-free grey-level image from the vector g and warping it using the feature points described by s. During a training phase we learn the relationship between model parameter displacements and the residual errors induced between a training image and a synthesized image.

The matching process for tracking the feature points is provided as an optimization problem in which we minimize the difference between a target image and an image synthesized by the model.

Bibliography

[AC99] J. K. Aggarwal and Q. Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(3):428–440, 1999.

[ACH+95] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

[AIYK00] K. Aihara, T. Ikeguchi, T. Yamada, and M. Komuro. *Fundamentals and Applications of Chaotic Time Series Analysis*. Sangyo Tosho, 2000.

[Alb72] A. Albert. *Regression and The Moore-Penrose Pseudoinverse*. Academic Press, 1972.

[All83] J. F. Allen. Maintaining knowledge about temporal interval. *Commun. of the ACM*, 26(11):832–843, 1983.

[All84] J. F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23:123–154, 1984.

[AM79] B. D. O. Anderson and J. B. Moor. *Optimal Filtering*. Prentice-Hall, 1979.

[Bas78] J. N. Bassili. Facial motion in the perception of faces and of emotional expression. *Journal of Experimental Psychology: Human Perception and Performance*, 4(3):373–379, 1978.

[BCMS01] A. Bissacco, A. Chiuso, Y. Ma, and S. Soatto. Recognition of human gaits. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 52–57, 2001.

[BD97] A. F. Bobick and J. W. Davis. The recognition of human movements using temporal templates. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(3):257–267, 1997.

[BHJT04] H. Balakrishnan, I. Hwang, J. S. Jang, and C. J. Tomlin. Inference methods for autonomous stochastic linear hybrid systems. *Hybrid Systems: Computation and Control*, pages 64–79, 2004.

[BIR00] B. N. A. Blake, M. Isard, and J. Rittscher. Learning and classification of complex dynamics. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(9):1016–1034, 2000.

[BO97] M. Brand and N. Oliver. Coupled hidden Markov models for complex action recognition. *MIT Media Lab Vision and Modeling TR407*, 1997.

[BOP97] M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999, 1997.

[BPV04] H. H. Bui, D. Q. Phung, and S. Venkatesh. Hierarchical hidden Markov models with general state hierarchy. *Proc. National Conference on Artificial Intelligence*, pages 324–329, 2004.

[Bra95] T. Brants. Estimating HMM topologies. *Logic and Computation*, 1995.

[Bra99] M. Brand. Voice puppetry. *Proc. SIGGRAPH*, pages 21–28, 1999.

[Bre97] C. Bregler. Learning and recognizing human dynamics in video sequences. *Proc. Int. Conference on Computer Vision and Pattern Recognition*, pages 568–574, 1997.

[Bro86] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14–23, 1986.

[Bro91] R. A. Brooks. Intelligence without reason. *Artificial Intelligence*, 47:139–159, 1991.

[Bur98] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Knowledge Discovery and Data Mining*, 2(2):121–167, 1998.

[BW97] A. F. Bobick and A. D. Wilson. A state-based approach to the representation and recognition of gesture. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(12):1325–1337, 1997.

[CET98] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance model. *Proc. European Conference on Computer Vision*, pages 484–498, 1998.

[CR98] T. Chen and R. R. Rao. Audio-visual integration in multimodal communication. *Proceedings of the IEEE*, pages 837–852, 1998.

[CRTW05] S. Collins, A. Ruina, R. Tedrake, and M. Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307:1082–1085, 2005.

[CS94] C. Cedras and Mubarak Shah. A survey of motion analysis from moving light displays. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 214–221, 1994.

[DAJ95] A. A. Desrochers and R. Y. AI-Jaar. *Application of Petri nets in Manufacturing Systems: Modeling, Control, and Performance Analysis.* IEEE Press, 1995.

[DBCS04] R. Dhillon, S. Bhagat, H. Carvey, and E. Shriberg. Meeting recorder project: Dialog act labeling guide. *ICSI Technical Report TR-04-002*, 2004.

[DCWS03] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *Int. Journal of Computer Vision*, 51(2):91–109, 2003.

[DD05] F. Dornaika and F. Davoine. Simultaneous facial action tracking and expression recognition using a particle filter. *Proc. IEEE Int. Conference on Computer Vision*, pages 1733–1738, 2005.

[dFNG98] J. F. G. de Freitas, M. Niranjan, and A. H. Gee. The EM algorithm and neural networks for nonlinear state space estimation. *Cambridge University Engineering Department Technical Report CUED/F-INFENG/TR313*, 1998.

[DHS00] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley-Interscience, 2000.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Statist. Soc. B*, 39:1–38, 1977.

[Dor96] G. Dorffner. Neural networks for time series processing. *Neural Network World*, 6(4):447–468, 1996.

[DP93] T. Darrell and A. Pentland. Space-time gestures. *Proc. IJCAI'93 Looking at People Workshop*, 1993.

[Ecc89] J. C. Eccles. *Evolution of the Brain: Creation of the Self.* Routledge, 1989.

[EF75] P. Ekman and W. V. Friesen. *Unmasking the Face.* Prentice Hall, 1975.

[EP97] I. A. Essa and A. P. Pentland. Coding, analysis, interpretation, and recognition of facial expressions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(7):757–763, 1997.

[FFK05] S. Fujie, K. Fukushima, and T. Kobayashi. Back-channel feedback generation using linguistic and nonlinguistic information and its application to spoken dialogue system. *Proc. EUROSPEECH*, pages 889–892, 2005.

[FH88] R. Futami and N. Hoshimiya. A neural sequence identification network (ansin) model. *IEICE*, J71-D-II(10):2181–2190, 1988.

[FMLM03] G. Ferrari, M. Muselli, D. Liberati, and M. Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39:205–217, 2003.

[FSKN04] W. Fujisaki, S. Shimojo, M. Kashino, and S. Nishida. Recalibration of audiovisual simultaneity. *Nature Neuroscience*, 7(7):773–778, 2004.

[Gav99] D. M. Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82–98, 1999.

[GH96] Z. Ghahramani and G. E. Hinton. Switching state-space models. *Dept. of Computer Science*, 1996.

[GJH01] A. Galata, N. Johnson, and D. Hogg. Learning variable-length Markov models of behavior. *Computer Vision and Image Understanding*, 81(3):398–413, 2001.

[GL95] J. J. Gross and R. W. Levenson. Emotion elicitation using films. *Cognition and Emotion*, 9:89–108, 1995.

[GR99] Z. Ghahramani and S. Roweis. Learning nonlinear dynamical systems using an EM algorithm. *Advances in Neural Information Processing Systems*, 11:599–605, 1999.

[GV89] A. Gollu and P. Varaiya. Hybrid dynamical systems. *Proc. Conference on Decision and Control*, pages 2708–2712, 1989.

[HAJ90] H. D. Huang, Y. Ariki, and M. A. Jack. *Hidden Markov Models for Speech Recognition*. Edinburgh Univ., 1990.

[HTF01] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Wiley-Interscience, 2001.

[HWK01] M. Haruno, D. M. Wolpert, and M. Kawato. Mosaic model for sensorimotor learning and control. neural computation. *Neural Computation*, 13:2201–2220, 2001.

[IB98] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *Int. Journal of Computer Vision*, 29(1):5–28, 1998.

[IB00] Y. A. Ivanov and A. F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):852–872, 2000.

[Iri03] M. Iri. *General Linear Algebra*. Iwanami, 2003.

[IS94] A. Ichikawa and S. Sato. Roles of prosodies in dialogue. *IPSJ SIGSLP (Japanese)*, pages 51–58, 1994.

[Joh73] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14(2):201–211, 1973.

[JR85] B. H. Juang and L. R. Rabiner. A probabilistic distance measure for hidden Markov models. *AT & T Technical Journal*, 64(2):391–408, 1985.

[JSFC98] D. Jurafsky, E. Shriberg, B. Fox, and T. Curl. Lexical, prosodic, and syntactic cues for dialog acts. *Proc. ACL/COLING-98: Workshop on Discourse Relations and Discourse Markers*, pages 114–120, 1998.

[KBM+01] M. Kamachi, V. Bruce, S. Mukaida, J. Gyoba, S. Yoshikawa, and S. Akamatsu. Dynamic properties influence the perception of facial expressions. *Perception*, 30:875–887, 2001.

[KCB95] J. Kosecka, H. I. Christensen, and R. Bajcsy. Discrete event modeling of visually guided behaviors. *International Journal of Computer Vision*, 14(2):179–191, 1995.

[KHS+04] J.-H. Kim, S. Hayakawa, T. Suzuki, K. Hayashi, S. Okuma, N. Tsuchida, M. Shimizu., and S. Kido. Modeling of drivier's collision avoidance behavior based on piecewise linear model. *Proc. IEEE Int. Conference on Decision and Control*, pages 2310–2315, 2004.

[KK05] E. Krumhube and A. Kapps. Moving smiles: The role of dynamic components for the perception of the genuineness of smiles. *Journal of Nonverbal Behavior*, 29(1):3–24, 2005.

[Koh89] T. Kohonen. *Self-Organization and Associative Memory*. Springer-Verlag Berlin Heidelberg, 1989.

[KOT+04] Y. Kuniyoshi, Y. Ohmura, K. Terada, A. Nagakubo, S. Eitoku, and T. Yamamoto. Embodied basis of invariant features in execution and perception of whole body dynamic actions — knacks and focuses of roll-and-rise motion. *Robotics and Autonomous Systems*, pages 189–201, 2004.

[KP98] E. J. Keogh and M. J. Pazzani. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. *Fourth Int. Conference on Knowledge Discovery and Data Mining*, pages 239–241, 1998.

[KS86] R. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–95, 1986.

[KTNN05] N. Kitaoka, M. Takeuchi, R. Nishimura, and S. Nakagawa. Response timing detection using prosodic and linguistic information for human-friendly spoken dialog. *Journal of The Japanese Society for Artificial Intelligence*, 20(3 SP-E):220–228, 2005.

[KYHH05] S. Kajita, K. Yokoi, H. Hirukawa, and K. Harada. *Humanoid Robot*. Ohmsha, 2005.

[Lev86] S. E. Levinson. Continuously variable duration hidden Markov models for automatic speech recognition. *Computer Speech and Language*, 1:29–45, 1986.

[Lib04] B. Libet. *Mind Time*. Harvard University Press, 2004.

[LMZ98] D. A. Langan, J. W. Modestino, and J. Zhang. Cluster validation for unsupervised stochastic model-based image segmentation. *IEEE Trans. on Image Processing*, 7(2):180–195, 1998.

[LWS02] Y. Li, T. Wang, and H.-Y. Shum. Motion texture: A two-level statistical model for character motion synthesis. *Proc. SIGGRAPH*, pages 465–472, 2002.

[Mas98] Y. Masunaga. Re-examination of allen's interval-based temporal logic. *IPSJ*, 39(4):846–856, 1998.

[Mat96] F. Matsuda. *Psychological Time*. Kitaoji Shobou, 1996.

[ME02] D. Moore and I. Essa. Recognizing multitasked activities from video using stochastic context-free grammar. *Proc. National Conference on Artificial intelligence*, pages 770–776, 2002.

[MM76] H. McGurk and J. MacDonald. Hearing lips and seeing voices. *Nature*, pages 746–748, 1976.

[MM98a] M. Morita and S. Murakami. Recognition of sequential patterns by nonmonotone neural networks. *IEICE*, J81-D-II(7):1679–1688, 1998.

[MM98b] M. C. Mozer and D. Miller. Parsing the stream of time: The value of event-based segmentation in a complex real-world control problem. *Lecture Notes in Artificial Intelligence. Springer-Verlag.*, 1387:370–388, 1998.

[MMdB+91] O. Maler, Z. Manna, A. Pnueli (J. W. de Bakker, C. Huizing, W. P. de Roever, and G. Rozenberg). *From Timed to Hybrid Systems*, pages 447–484. Real-Time: Theory in Practice. Springer-Verlag, 1991.

[MMM69] J. McCarthy, P. J. Hayes (B. Meltzer, and D. Michie). *Some Philosophical Problems from the Standpoint of Artificial Intelligence*, pages 463–502. Machine Intelligence 4. Edinburgh University Press, 1969.

[MMM02] M. Morita, K. Matsuzawa, and S. Morokami. A model of context-dependent association using selective desensitization of nonmonotonic neural elements. *IEICE*, J85-D-II(10):1602–1612, 2002.

[MMMS04] M. Morita, K. Murata, S. Morokami, and A. Suemitsu. Information integration ability of layered neural networks with the selective desensitization method. *IEICE*, J87-D-II(12):2242–2252, 2004.

[Mor96] M. Morita. Memory and learning of sequential patterns by nonmonotone neural networks. *Neural Networks*, 9(8):1477–1489, 1996.

[Mur98] K. P. Murphy. Switching kalman filter. *Technical report, U. C. Berkeley*, 1998.

[Mur02] K. P. Murphy. Hidden semi-Markov models (HSMMs). *Informal Notes*, 2002.

[NA94] S. A. Niyogi and E. H. Adelson. Analyzing and recognizing walking figures in xyt. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 469–474, 1994.

[Nak00] S. Nakagawa. A survey on automatic speech recognition. *IEICE*, J83-D-II(2):433–457, 2000.

[Nak01] M. Nakamura. Dance notation. *Art Research (Ritsumeikan ARC)*, 2:89–100, 2001.

[NBVW03] N. T. Nguyen, H. H. Bui, S. Venkatesh, and G. West. Recognising and monitoring high-level behaviours in complex spatial environments. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 620–625, 2003.

[NDKN02] C. Nagaoka, M. Draguna, M. Komori, and T. Nakamura. The influence of switching pauses on interpersonal perception in dialogues. *Human Interface*, pages 1431–1434, 2002.

[NKHM05] M. Nishiyama, H. Kawashima, T. Hirayama, and T. Matsuyama. Facial expression representation based on timing structures in faces. *IEEE International Workshop on Analysis and Modeling of Faces and Gestures (W. Zhao et al. (Eds.): AMFG 2005, LNCS 3723)*, pages 140–154, 2005.

[NKN98] S. Nishio, K. Koyama, and T. Nakamura. Temporal differences in eye and mouth movements classifying facial expressions of smiles. *Proc. IEEE Int. Conference on Automatic Fase and Gesture Recognition*, pages 206–211, 1998.

[NLP+02] A. V. Nefian, L. Liang, X. Pi, X. Liu, and K. Murphy. Dynamic Bayesian networks for audio-visual speech recognition. *EURASIP Journal on Applied Signal Processing*, 2002(11):1–15, 2002.

[NMN97] T. Nishimura, T. Mukai, and S. Nozaki. Spotting recognition of gestures performed by people from a single time-varying image using low-resolution features. *IEICE*, J80-D-II(6):1563–1570, 1997.

[NNYI04] S. Nakaoka, A. Nakazawa, K. Yokoi, and K. Ikeuchi. Leg motion primitives for a dancing humanoid robot. *Proc. IEEE Int. Conference on Robotics and Automation*, pages 610–615, 2004.

[ODK96] M. Ostendorf, V. Digalakis, and O. A. Kimball. From HMMs to segment models: A unified view of stochastic modeling for speech recognition. *IEEE Trans. Speech and Audio Process*, 4(5):360–378, 1996.

[Oka95] M. Okada. A hierarchy of macrodynamical equations for associative memory. *Neural Networks*, 8(6):833–838, 1995.

[OKYI96] Y. Okato, K. Kato, M. Yamamoto, and S. Itahashi. Prosodic pattern recognition for insertion of interjectory responses and its evaluation. *IPSJ SIG-SLP (Japanese)*, pages 33–38, 1996.

[OM95] P. V. Overschee and B. D. Moor. A unifying theorem for three subspace system identification algorithms. *Automata*, 31(12):1853–1864, 1995.

[ONN03] M. Okada, D. Nakamura, and Y. Nakamura. On-line and hierarchical design methods of dynamics based information processing system. *Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pages 954–959, 2003.

[OOK+05] T. Ogata, H. Ohba, K. Komatani, J. Tani, and H. G. Okuno. Extracting mult-modal dynamics of objects using rnnpb. *Journal of Robotics and Mechatronics*, 17(6):681–688, 2005.

[OTN02] M. Okada, K. Tatani, and Y. Nakamura. Polynomial design of the nonlinear dynamics for the brain-like information processing of whole body motion. *Proc. IEEE Int. Conference on Robotics and Automation*, 2002.

[PB97] C. Pinhanez and A. Bobick. Human action detection using pnf propagation of temporal constraints. *M.I.T Media Lab. Technical Report.*, pages 1–9, 1997.

[PH95] D. K. Panjwani and G. Healey. Markove random field models for unsupervised segmentation of textured color images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(10):939–954, 1995.

[PMB97] C. S. Pinhanez, K. Mase, and A. F. Bobick. Interval scripts: A design paradigm for story-based interactive systems. *Proc. CHI*, pages 287–294, 1997.

[PN94] R. Polana and R. Nelson. Low level recognition of human motion (or how to get your man without finding his body parts). *Proc. IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 77–82, 1994.

[Pop85] E. Poppel. *Grenzen des Bewubtseins.* Deutsche Verlags-Anstalt GmbH, Stuttgart, 1985.

[PRCM99] V. Pavlovic, J. M. Rehg, T. Cham, and K. P. Murphy. A dynamic bayesian network approach to figure tracking using learned dynamic models. *Proc. IEEE Int. Conference on Computer Vision*, pages 94–101, 1999.

[PRM00] V. Pavlovic, J. M. Rehg, and J. MacCormick. Learning switching linear models of human motion. *Proc. Neural Information Processing Systems*, 2000.

[PS99] P. Pfeifer and C. Scheier. *Understanding Intelligence.* The MIT Press, 1999.

[Rab89] L. R. Rabiner. A tutorial on hidden markov models and selected applications inspeech recognition. *Proc. IEEE*, pages 257–286, 1989.

[Rao97] R. Rao. Dynamic appearance-based recognition. *Proc. Int. Conference on Computer Vision and Pattern Recognition*, pages 540–546, 1997.

[Ras80] R. F. Rashid. Toward a system for the interpretation of moving light display. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2(6):574–581, 1980.

[RB99] J. Rittscher and A. Blake. Classification of human body motion. *Proc. IEEE Int. Conference on Computer Vision*, pages 634–639, 1999.

[RBL04] J. Roll, A. Bemporad, and L. Ljung. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40:37–50, 2004.

[RN02] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach (2nd edition).* Prentice Hall, 2002.

[Rob94] A. J. Robinson. An application of recurrent nets to phone probability estimation. *IEEE Trans. Neural Networks*, 5(2):298–305, 1994.

[RST96] D. Ron, Y. Singer, and N. Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25(2-3):117–149, 1996.

[SCT03] K. L. Schmidt, J. F. Cohn, and Y.-L. Tian. Signal characteristics of spontaneous facial expressions: Automatic movement in solitary and social smiles. *Biological Psychology*, 65:49–66, 2003.

[Sea86] J. R. Searle. *Speech Acts, an Essay in the Philosophy of Language (Cambridge Univ. Press 1969).* Keisou Shobou, 1986.

[SEL03] M. B. Stegmann, B. K. Ersboll, and R. Larsen. FAME - a flexible appearance modelling environment. *Informatics and Mathematical Modelling*, 2003.

[SG02] M. B. Stegmann and D. D. Gomez. A brief introduction to statistical shape analysis. *Informatics and Mathematical Modelling*, 2002.

[Shi05] E. Shirberg. Spontaneous speech: How people really talk, and why engineers should care. *Proc. EUROSPEECH*, 2005.

[SHST00] H. Segawa, N. Hiraki, H. Shioya, and T. Totsuka. Constraint-conscious smoothing framework for the recovery of 3D articulated motion from image sequences. *Proc. IEEE Int. Conference on Automatic Face and Gesture Recognition*, pages 476–482, 2000.

[SP95] T. Starner and A. Pentland. Visual recognition of american sign language using hidden markov models. *Proc. Int. Workshop on Automatic Face and Gesture Recognition*, pages 189–194, 1995.

[SRC+00] A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, M. Meteer, and C. V. Ess-Dykema. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–371, 2000.

[TKC01] Y. Tian, T. Kanade, and J. F. Cohn. Recognizing action units for facial expression analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):97–115, 2001.

[TKT+00] I. Tomonori, N. Kawakatsu, M. Tanabe, R. Nakajima, and Y. Hayashi. Timed event pattern matching — formalism, machines and applications —. *Computer Software*, 17(5):61–79, 2000.

[TSKO94] K. Takahashi, S. Seki, H. Kojima, and R. Oka. Spotting recognition of human gestures from time-varying images. *IEICE*, J77-D-II(8):1552–1561, 1994.

[Tsu87] N. Tsukahara. *Brain plasticity and memory*. Kinokuniya Shoten, 1987.

[Tur36] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proc. London Math. Soc. Ser. 2*, 42(2198):230–265, 1936.

[Tur50] A. M. Turing. Computing machinery and intelligence. *A Quarterly Review of Psychology and Philosophy*, LIX(236):433–460, 1950.

[UT00] T. Uchiyama and H. Takahashi. Speech recognition using recurrent neural prediction model. *IEICE*, J83-D-II(2):776–783, 2000.

[WBC97] A. D. Wilson, A. F. Bobick, and J. Cassell. Temporal classification of natural gesture with application to video coding. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 948–954, 1997.

[Wie61] N. Wiener. *Cybernetics (2nd edition)*. MIT press, 1961.

[WM00] T. Wada and T. Matsuyama. Multiobject behavior recognition by event driven selective attention method. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):873–887, 2000.

[WYH05] K. Wakiyama, A. Yoshitaka, and T. Hirashima. Information recommendation by collaborative filtering incorporated with gaze detection. *Proc. Workshop on Interactive Systems and Software (WISS)*, 2005.

[YK02] T. Yamamoto and Y. Kuniyoshi. Stability and controllability in a rising motion: A global dynamics approach. *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2467–2472, 2002.

[YKM06] A. Yamaguchi, H. Kawashima, and T. Matsuyama. Event-driven control and its representation for multi-dof robot to learn motion. *The 24th of Annual Conference of the Robot Society of Japan*, page 2D13, 2006.

[YOI92] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden Markov model. *Proc. Int. Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.

[ZG03] S. Zhong and J. Ghosh. A unified framework for model-based clustering. *Journal of Machine Learning Research*, 4(11):1001–1037, 2003.

[ZM95] Y. Zhang and A. K. Mackworth. Constraint nets: A semantic model for hybrid dynamic systems. *Theoretical Computer Science*, 138(1):211–239, 1995.

[ZMI01] L. Zelnik-Manor and M. Irani. Event-based analysis of video. *Proc. IEEE Conference on Computer Vision and Pattern Recognition,* pages 123–130, 2001.

List of Publications

Journal and Letter

1. Hiroaki Kawashima, Takashi Matsuyama, "Multiphase Learning for an Interval-Based Hybrid Dynamical System", IEICE Trans. Fundamentals, Vol.E88-A, No.11, pp.3022–3035, 2005.

2. Hiroaki Kawashima, Masahiro Nishiyama, Takashi Matsuyama, "Facial Expression Description, Generation, and Recognition based on Timing Structure", Information Technology Letters (FIT2005), pp.153–156, 2005 (in Japanese) (Funai Best Paper Award).

3. Hiroaki Kawashima, Kimitaka Tsutsumi, Takashi Matsuyama, "Hybrid Dynamical System for Dynamical Event Segmentation, Learning, and Recognition ", Information Technology Letters (FIT2004), pp.175–178, 2004 (in Japanese) (FIT Best Paper Award).

4. Hiroaki Kawashima, Takashi Matsuyama, "Multi-Viewpoint Gesture Recognition by an Integrated Continuous State Machine", The Transaction of The Institute of Electronics, Information and Communication Engineers, Vol.J85-D-II No.12, pp.1801–1812, 2002 (in Japanese).

International Conference (reviewed)

1. Hiroaki Kawashima, Kimitaka Tsutsumi, and Takashi Matsuyama, "Modeling Timing Structure in Multimedia Signals", 4th International Conference on Articulated Motion and Deformable Objects (F. J. Perales and R. B. Fisher (Eds.): AMDO 2006, LNCS 4069), pp. 453–463, 2006.

2. Masahiro Nishiyama, Hiroaki Kawashima, Takatsugu Hirayama, and Takashi Matsuyama, "Facial Expression Representation based on Timing

Structures in Faces", IEEE International Workshop on Analysis and Modeling of Faces and Gestures (W. Zhao et al. (Eds.): AMFG 2005, LNCS 3723), pp. 140–154, 2005.

3. Hiroaki Kawashima, Takashi Matsuyama, "Hierarchical Clustering of Dynamical Systems based on Eigenvalue Constraints", 3rd International Conference on Advances in Pattern Recognition (S. Singh et al. (Eds.): ICAPR 2005, LNCS 3686), pp. 229–238, 2005.

4. Hiroaki Kawashima, Takashi Matsuyama, "Integrated Event Recognition from Multiple Sources", 16th International Conference on Pattern Recognition (ICPR), Vol.2, pp785–789, 2002.

Book and Article

1. Hiroaki Kawashima, "Feature Extraction (Section 1.3 in Part 2)", Biometric Security Technology Handbook (Biometrics Security Consosium (Eds.)), , pp.65–120, 2006 (in Japanese).

2. Takashi Matsuyama, Hiroaki Kawashima, Kazuhiko Sumi, "Developing Man-Machine Symbiotic Systems", IPSJ Magazine, Vol.47, No.8, pp.851–858, 2006 (in Japanese).

3. Takashi Matsuyama, Akihiro Sugimoto, Yoichi Sato, Hiroaki Kawashima, "Developing Man-Machine Symbiotic Systems", Journal of the Japanese Society for Artificial Intelligence, Vol.19, No.2, pp.257–266, 2004 (in Japanese).

Presentation

1. Takashi Matsuyama and Hiroaki Kawashima, "Modeling Dynamic Structure of Human Verbal and Nonverbal Communication", The Second International Conference on Informatics Research for Development of Knowledge Society Infrastructure, pp.1–8, 2007.

2. Akihiko Yamaguchi, Hiroaki Kawashima, and Takashi Matsuyama, "Event-driven control and its representation for multi-DOF robot to learn motion", The 24th Annual Conference of the Robotics Society of Japan, 2006 (in Japanese).

3. Hiroaki Kawashima, Kimitaka Tsutsumi, and Takashi Matsuyama, "Modeling Timing Structure in Multimedia Signals", Forum on Information Technology(FIT2006), Vol.3, pp.93–96, 2006 (in Japanese).

4. Hiroaki Kawashima and Takuichi Nisimura, "Temporal Pattern Recognition in Computer Vision", IPSJ SIG Technical Reports (2006-CVIM-154), Vol.2006, No.51, pp.197–209, 2006 (in Japanese).

5. Akihiko Yamaguchi, Hiroaki Kawashima, and Takashi Matsuyama, "Event-driven control and its representation —towards a motion learning of multi-DOF robot in dynamic environment—", IPSJ SIG Technical Reports (2006-CVIM-154), Vol.2006, No.51, pp.159–166, 2006 (in Japanese).

6. Masahiro Nishiyama, Hiroaki Kawashima, Takashi Matsuyama, "Facial Expression Recognition Based on Timing Structure in Faces", IPSJ SIG Technical Reports (2005-CVIM-149), Vol.2005, No.38, pp.179–186, 2005 (in Japanese). (Bachelor's Thesis Award).

7. Scoggins Levi, Hiroaki Kawashima, Takashi Matsuyama, "Analysis of the Dynamic Strucutre of Manzai for the Purpose of Good Timing Control", Interaction2005, CD-ROM D-404,2005 (in Japanese).

8. Hiroaki Kawashima, Kimitaka Tsutsumi, Takashi Matsuyama, "A Hybrid Dynamical System for Event Segmentation, Learning and Recognition", Second International Workshop on Man-machine Symbiotic Systems 2004.

9. Hiroaki Kawashima, Kimitaka Tsutsumi, Takashi Matsuyama, "Finding Structure in Lip Image Sequences based on Self-Organization of Dynamical Systems", Information-Based Induction Sciences (IBIS2004), pp.86–93, 2004 (in Japanese).

10. Kimitaka Tsutsumi, Hiroaki Kawashima, Takashi Matsuyama, "Time-varying Event Recognition by An Interval Linear Dynamical System", IPSJ SIG Technical Reports (2004-CVIM-144), Vol.2004, No.40, pp.41–48, 2004 (in Japanese).

11. Takashi Fujie, Hiroaki Kawashima, Takashi Matsuyama, "Dynamical System based Time-varying Pattern Representation and Recognition", IPSJ SIG Technical Reports (2003-CVIM-138), Vol.2003, No.41, pp.81–88, 2003 (in Japanese).

12. Hiroaki Kawashima, Takashi Matsuyama, "Integrated Event Recognition from Multiple Sources", First Int'l Workshop on Man-Machine Symbiotic Systems, 2002.

13. Hiroaki Kawashima, Takashi Matsuyama, "Multi-Viewpoint Gesture Recognition by an Integrated Continuous State Machine", PRMU2001-105, 2001 (in Japanese).