Towards Perceptual Intelligence: Statistical Modeling of Human Individual and Interactive Behaviors

by

Nuria M. Oliver

B.S., Universidad Politécnica de Madrid (1992)
M.S., Universidad Politécnica de Madrid (1994)

SUBMITTED TO THE PROGRAM IN MEDIA ARTS AND SCIENCES, SCHOOL OF ARCHITECTURE AND PLANNING, IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2000

Author_____

Program in Media Arts and Sciences
April 28, 2000

Certified by_____

Alex P. Pentland
Academic Head and Toshiba Professor of Media Arts and Sciences
Media Laboratory, MIT
Thesis Advisor

Accepted by_____

Stephen A. Benton
Chairman, Department Committee on Graduate Students
Program in Media Arts and Sciences

Towards Perceptual Intelligence: Statistical Modeling of Human Individual and Interactive Behaviors

by

Nuria M. Oliver

Submitted to the Program in Media Arts and Sciences
on April 28, 2000, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This thesis presents a computational framework for the automatic recognition and prediction of different kinds of human behaviors from video cameras and other sensors, via *perceptually intelligent systems* that automatically sense and correctly classify human behaviors, by means of *Machine Perception* and *Machine Learning* techniques. In the thesis I develop the statistical machine learning algorithms (dynamic graphical models) necessary for detecting and recognizing individual and interactive behaviors. In the case of the interactions two Hidden Markov Models (HMMs) are coupled in a novel architecture called Coupled Hidden Markov Models (CHMMs) that explicitly captures the interactions between them. The algorithms for learning the parameters from data as well as for doing inference with those models are developed and described. Four systems that experimentally evaluate the proposed paradigm are presented: (1) LAFTER, an automatic face detection and tracking system with facial expression recognition; (2) a Tai-Chi gesture recognition system; (3) a pedestrian surveillance system that recognizes typical human to human interactions; (4) and a SmartCar for driver maneuver recognition.

These systems capture human behaviors of different nature and increasing complexity: first, isolated, single-user facial expressions, then, two-hand gestures and human-to-human interactions, and finally complex behaviors where human performance is mediated by a machine, more specifically, a car. The metric that is used for quantifying the quality of the behavior models is their accuracy: how well they are able to recognize the behaviors on testing data. Statistical machine learning usually suffers from lack of data for estimating all the parameters in the models. In order to alleviate this problem, synthetically generated data are used to bootstrap the models creating 'prior models' that are further trained using much less real data than otherwise it would be required. The Bayesian nature of the approach let us do so.

The predictive power of these models lets us categorize human actions very soon after the beginning of the action. Because of the generic nature of the typical behaviors of each of the implemented systems there is a reason to believe that this approach to modeling human behavior would generalize to other dynamic human-machine systems. This would allow us to recognize automatically people's intended action, and thus build control systems that dynamically adapt to suit the human's purposes better.

Thesis Advisor: Alex P. Pentland
Title: Academic Head and Toshiba Professor of Media Arts and Sciences
Media Laboratory, MIT

Thesis Committee

Thesis Advisor _____

Alex P. Pentland
Academic Head and Toshiba Professor of Media Arts and Sciences
Media Laboratory, MIT

Thesis Reader _____

Tommi Jaakkola
Assistant Professor
Department of Computer Science, AI Lab, MIT

Thesis Reader _____

John Hansman
Associate Professor
Department of Aeronautics and Astronautics, MIT

Thesis Reader _____

Warren P. Seering
Professor
Department of Mechanical Engineering, MIT

Acknowledgments

Gratitude is the heart's memory

Thank you to my advisor, Prof. Alex Pentland, for being not just an academic advisor but a parent. Thank you, Sandy, for your support during all these years, for your wise guidance, for always finding time to listen to me, for helping me grow in the scientific community, for your kind words and your so valuable advice. Sandy, during my interview, almost five years ago, you told me that 'I should tell my parents not to worry, because you would take care of me'. Your words reached my heart. I believed you and I did tell my parents. You have certainly done it. Thank you.

Thank you to my thesis readers, Prof. Tommi Jaakkola, Prof. Warren Seering and Prof. John Hansman for your advice, your constructive feedback and, of course, for reading this document.

Special thanks to Prof. Michael Jordan, for sharing so much wisdom and knowledge. This thesis work has been, with no doubt, inspired by his research and lectures.

Thank you to all the past and current members of Vismod and specially to all Pentlandians, for bearing with me during all this time and for creating such an stimulating working environment. Vismod will be, finally, much quiter now... Thank you to my office mates, Jacob and Egon. Thank you for your patience with me. Egon, special thanks for being such a generous heart, always willing to help.

Thank you to all the members of the SmartCar team, my UROP students: Jared, Daniel, Nick, Andrew, Gustavo, Natasha, Chethan, Rooz, Supriya, Janette, Matt, Adam. All of them have been actively and enthusiastically working on the car video annotation. I would not have been able to finish on time without their help.

Thank you to all the members of GMD First Berlin, for offering me the opportunity of spending two certainly enriching and educating months in Berlin.

Thank you to my sponsors, 'La Caixa Foundation', 'Motorola', 'Volvo' and the other members of the 'CC++' Special Interest Group at the MIT Media Lab for enabling the research presented in this thesis, by proving the financial support.

Thank you to my other friends in Boston: Namita, Claudia, Natalia, Nitin, Marina. Thank you to Mark, for sharing with me the secret of a spiritual life in the materialistic world of business. Thank you to Kai, for his truthful friendship that will certainly last forever.

Thank you to Ali for being so patient, sweet, kind and supportive always. Very special thanks to Yael and Ernesto, my very good friends. Thank you for always understanding me, for being so patient with my lack of time, for listening to me, for cheering me up in the hardest moments and always believing in me. Thank you for being my family in Boston.

Thank you to Barbara, for the best days in 1998. Thank you for being such a great co-worker and a friend. Thank you for sharing so many unforgettable moments that I will never forget. I look forward to working with you again.

Thank you to my friend Arthur, who asked me to dance and became my long-life friend. Thank you to my dearest soul mates: Bernhard, for our golden Berlin days; Carmen, for bringing so much positive energy, joy and hope to the very worst weeks of graduate school. Carmen, 'we have made it together!'; Dario, always there, at the other side of the big ocean, ready to listen, to help, to understand; Beatriz, my friend since primary school, for being always close to me, despite the distance and the silence.

Thank you to my "sibling-friends", Mari Mar and Miguel. I owe them so much, since my undergraduate years in Madrid. We have followed very different paths in life. However, we have always stayed together, close in our hearts. Thank you for filling my life with love, energy, unforgettable memories, and dreams. I would not be "me" without you.

My warmest thanks to my so dear co-worker and friend, Betty Lou McClanahan, without whom my last months at the lab would have been gray and lonely. I owe you so much, Betty Lou. Thank you for discovering so many new worlds to me. Thank you for opening the challenging doors of the world of driving, cars and racing; thank you for our unforgettable adventures; thank you for your sweetness and kindness; thank you for caring so much for me; thank you for so many laughs and unforgettable moments. Memories that will last forever.

Thank you to Miguel and Tony for brightening my days –specially the darkest ones– and bringing so much excitement, energy, happiness and love to my life. Thank you for making my heart smile so many times. Without you I would have never been able to walk in, through and out the path of graduate life. Thank you to Flamingo and Chris, for the gift of so many San Francisco dreams, and for the most magic and unique "first-last" week of the Millenium.

Each friend represents a world in us, a world possibly not born until they arrive, and it is only this meeting that a new world is born. Thank you all for filling my life with so

many rich worlds. Worlds forever. Worlds that make me who I am. Thank you all, my friends, for the luxury of a life that I do not deserve. Without you, my life would definitely be empty, and I would die.

Finally and most importantly, thank you to my family and specially to my siblings and my parents, M. Angeles and Jose Luis. Thank you for giving me the roots to always know where home is, and for giving me the wings that let me fly and explore the world. Thank you for always believing in me, even when I did not. Thank you for always being close to me, despite the physical distance. Thank you for looking over my shoulder, for caring for me, everywhere, anywhere, always. Thank you for sacrificing your lives for me. Thank you for the gift of the richest of the educations, for teaching me such strong values and principles. Thank you for the priviledge of a happy life. My work is, as you know, all because of you and for you.

Contents

List of Figures

List of Tables

Chapter 1

The Problem of Human Behavior Modeling

Introduction and Motivation

Over the last decade there has been growing interest within the computer vision and machine learning communities in the problem of analyzing human behavior from sensor information, such as video ([53],[25],[183], [39], [159], [97],[42], [67]). These systems typically consist of a low- or mid-level computer vision system to detect and segment a moving object — human or car, for example — and a higher level interpretation module that classifies the motion into 'atomic' behaviors such as, for example, a smile, a pointing gesture, or a car turning left.

However, there have been relatively few efforts to understand human behaviors that have substantial extent in time, particularly when they involve interactions between several agents. This level of interpretation is the goal of the thesis, with the intention of building systems that can deal with increasingly more complex human behaviors, from single-user facial expressions to interactive driving behaviors where complex interactions with the surrounding traffic take place.

In this thesis I propose a computational framework for human behavior recognition and prediction via *Perceptually Intelligent Systems* that automatically sense and correctly classify real human behaviors by means of Machine Perception and Machine Learning Techniques. The proposed framework could be psychologically plausible at a general level, addresses many of the problems that current behavior theories suffer from and it is evaluated

with experimental data of increasing behavioral complexity collected in four different domains:

1. Individual, isolated behaviors in the LAFTER [171] (Lips and Face TrackER) system: a real-time system for face detection, tracking and facial expression recognition (see figure 1-1 [1]),

2. Body gestures in a Tai-Chi real-time gesture recognition system,

3. Human to human interactive behaviors in a visual surveillance system for detection and recognition of human-to-human interactions [173] (see figure 1-1),

4. Multi-agent interactive behaviors when mediated by a machine in the MIT Media Lab *SmartCar* testbed. More specifically recognition of driver's behaviors at a tactical level, with emphasis on how the context (road lanes, surrounding traffic) affects the driver's performance (see figure 1-2).

As can be seen in figure 1-3 the proposed model's architecture is composed of a hierarchy of two layers. At the bottom (first layer) there is the Perceptual System, composed of cameras and other sensors. The signals captured by the sensors are the input to a Kalman Filter. At the top (second layer) there is the behavior models via dynamic graphical models. The Kalman filtered variables are the observations of the dynamic graphical models (HMMs or CHMMs) at the second layer. The proposed architecture includes a *bottom-up* stream of information provided by the various sensors, and a *top-down* information flow through the predictions provided by the behavior models. Consequently a Bayesian approach –such as the one followed here– offers a mathematical framework for both combining the observations (bottom-up) with complex behavioral priors (top-down) to provide expectations that would be fed back to the perceptual system.

From a practical viewpoint, there are many motivations and potential applications of these systems:

1. LAFTER: video-conferencing, real-time computer graphics animation, "visual speech" recognition and "virtual windows" for visualization. Of particular interest is its ability for accurate, real-time classification of the user's mouth shape without constraining

[1] Appendix 2 contains the color version of the figures that have color

head position; this ability makes possible (for the first time) real-time facial expression recognition in unconstrained office environments.

2. Visual Surveillance (pedestrian interaction recognition): visual surveillance systems, anomaly detection, automatic video parsing and interpretation.

3. Smart Car: drivers' assistants, emergency countermeasure systems, and realistic tactical reasoning modules for car simulators.

Perceptual Intelligence

The computational tasks involved in the systems developed in this thesis combine elements of AI/machine learning and perceptual computing (computer vision, signal processing) yielding to a new research area called *Perceptual Intelligence*, which brings together perception and cognition in the same framework. Two hundred years ago, Kant provocately suggested an intimate connection between perception and concepts. "Concepts without percepts", he wrote, "are empty; percepts without concepts are blind". However, traditional research in Artificial Intelligence has tried to model concepts while ignoring perception, even though high-level perceptual processes lie at the heart of human cognitive abilites. Cognition cannot succeed without processes that build up appropriate representations. Conceptual processes should, thus, be studied in conjunction with the perceptual substrate on which they rest, and with which they are tightly coupled. On the other hand, our perception of any given situation is guided by constant top-down influence from the conceptual level. Without this conceptual influence, the representations that result from such perception will be rigid, inflexible, and unable to adapt to the problems provided by many different contexts. The flexibility of human perception derives from constant interaction with the conceptual level. I would argue that perceptual processes cannot be separated from other cognitive processes even in principle, and therefore traditional AI models cannot be defended by supposing the existence of a 'representation module' that supplies representations ready-made. Recognizing the centrality of perceptual processes makes AI more difficult, but much more interesting. Integrating perceptual processes into a cognitive model leads to flexible representations, and flexible representations lead to flexible actions. This is precisely the goal at the heart of Perceptual Intelligence.

The framework presented in this thesis focuses on perceptually intelligent systems that

understand certain aspects of human behavior, i.e. *'behavioral systems'*. Building these systems presents challenging problems in at least two domains: from a *Perceptual Computing* viewpoint, it requires, for example, real-time, accurate and robust detection and tracking of the objects of interest in an unconstrained environment; from a *Machine Learning and Artificial Intelligence* perspective behavior models for interacting agents are needed to interpret the set of perceived actions and in many situations detect anomalous behaviors or potentially dangerous situations. Moreover, all the processing modules need to be integrated in a consistent manner.

My approach to modeling human individual and interactive behaviors is to use supervised statistical machine learning techniques to teach the system to recognize normal single-person behaviors, two-hand body gestures, common person-to-person interactions, and driver maneuvers. More specifically the focus is on the interactions between different agents, in how the contextual information affects the performance and in predicting what is the most likely action to happen next. There are a number of important AI problems involved in such tasks: (1) Decision-making has to take place in real-time; (2) the sensors are noisy, with significant errors in the estimation of the position of the face, body, hands or other cars. Moreover, some of the objects might not be detected at all; (3) the world is only partially observable –vehicles, for example, might be occluded and all driver's intentions are hidden; (4) finally a successful system should have a very small false alarm rate. This is particularly important in the visual surveillance system.

Taxonomy

The modeling approach developed in the thesis follows the taxonomy proposed by Pentland ([180]): channels, time scale and intentionality. Figure 1-4 illustrates the taxonomy and the regions of the space that the work of this thesis covers.

- **Channels:** The domain is typically broken down into several channels of information: face, whole-body, car internal signals, voice, pressure, etc. These channels are mostly used in a complementary or redundant manner. In general however they should be considered together, as a multi-dimensional manifold. For example, voice, gesture and facial expressions are intimately bound together and should be integral part of the same system. In the work of this thesis I have incorporated channels of different nature: face and mouth, two hands, whole-body, surrounding traffic, road lanes, driver's

gaze and car internal signals.

- **Time Scale:** Each channel carries relevant information at a wide range of time scales. At the longest scale, are semi-permanent *physical attributes* like facial shape and appearance, vocal pitch, body shape and gait. These long-term characteristics are all useful for identification, and are predictive of variables such as age, sex or area of origin. At shorter time scales are *goal-directed behaviors* which typically have durations ranging from a few seconds to minutes or even hours. Examples are getting out of a car and walking to a building, or changing lanes while driving. Behaviors are in turn composed of a multi-modal sequence of individual *actions*, with a shorter time expand, such as frowning, pointing or starting to turn the steering wheel before changing lanes. These individual actions are often broken into 'microactions' such as the facial action units of the FACS system [62]. However it is uncertain whether such microactions constitute an important level of representation. Humans are normally unaware of these microactions (they would correspond, for example, to automatic, reflex, unconscious acts). Therefore we are unable to independently and consciously control them. These observations support the argument that microactions are more likely to be a convenient accounting system for psychologists rather than something intrinsic to the structure of the behavioral phenomena.

- **Intentionality:** The intentionality scale ranges from simple phenomena in which intentionality does not need to be considered to behaviors of increasingly complex intentionality. The testbeds developed in this thesis explore the intentionality axis, starting with simple individual facial expressions and ending with complex multi-agent car interactions. Moreover, the increasing behavioral complexity of the systems yields to longer time scales and an increasing number of multi-modal channels. Simple physical observations –the traditional focus of computer vision– typically do not involve intentionality. The shape or appearance of a face, the body pose, the body shape and dimensions, the acceleration in a car are all simple physical observations.

  The first level at which intentionality must be considered is observation of *direct behaviors*. These are behaviors that have only the intention of directly influencing the surrounding physical environment, and include mechanic activities such as direct manipulation, construction, cleaning, etc. To interpret such behaviors it is normally

necessary to know about both the person's (agent) movements and the objects in the surrounding environment, because the movements' *intended purpose* is to manipulate the object.

In contrast, communicative behaviors have the intention of influencing another agent, something often referred to as higher-order intentionality. Included are most expressions and gestures, even unconscious ones since these have evolved to serve an important role in interpersonal communication. The ability to avoid questions of intentionality is a great advantage for todays' applications, but as we move towards more generally competent systems we will have to directly confront the problem of interpreting intentionality. One area where consideration of intentionality is difficult to avoid is viewpoint. In most vision applications there have traditionally been only two viewpoints: external (third person perspective) and object-centered (first person perspective). However there is an important "second person perspective", where the observed persons are interacting with you (first person) and their intentions *toward* you become a primary consideration. And it is precisely this second person perspective that some of the testbeds developed in this thesis have to deal with, by modeling pedestrian interactions or how the surrounding cars' actions affect the driver's performance and vice versa.

To recognize communicative behaviors it is usually necessary to know something about the context, for instance, if there are other people (agents) present and what is the goal of the interaction. The systems developed in this thesis model these kind of behaviors. For example, the gesture of extending an arm and finger together could be a pointing gesture (communicative behavior), an action for pushing a button (direct behavior) or even an unconscious muscle stretch (non-intentional behavior). It is the presence and relative location of a button or a human observer that differentiates these three behaviors. Therefore, the *context* is *crucial* element for correctly interpreting intentional behaviors. This is one of the emphasis of this thesis.

Contributions

The main contributions of this thesis are consequence of the modeling approach proposed in my work on Perceptual Intelligence. Namely, the combination of Perceptual Computing

with Statistical Machine Learning (dynamic graphical models or DynPINs) for recognizing human behaviors of increasing complexity in different domains. More specifically the proposed framework emphasizes the context through the interactions between the agents as an important element of behavior modeling. In the proposed taxonomy (see section 1) the domains explored in this thesis proceed along the "intentionality" axis, with increasing complexity in the nature of their typical behaviors. Some of the key contributions are:

1. Perception: Blob-based computer vision methodology for face, mouth and pedestrian tracking; Kalman Filters for robust tracking; active camera control via a PD controller; off-line and on-line EM algorithms for blob parameter estimation; eigen-background for pedestrian detection.

2. Machine Learning: dynamic graphical models for human behavior recognition and prediction: HMMs for individual behaviors and CHMMs for interactive behaviors; flexible and interpretable priors using synthetic data generated by synthetic behavioral agents.

The proposed model has been validated in 4 systems with real human data:

1. LAFTER: Active camera real-time system for human face and mouth detection and tracking, and real-time face expression recognition system using HMMs.

2. Tai-Chi gesture recognition: CHMMs for two-hand real-time gesture recognition.

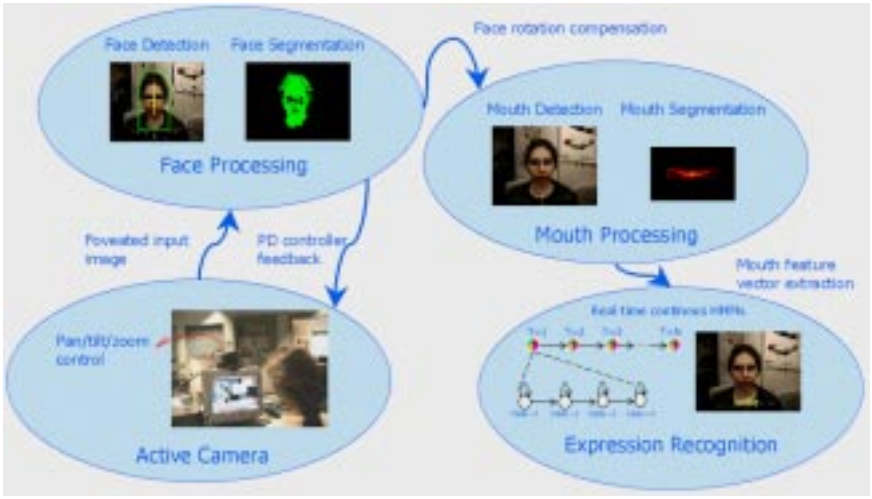
3. Pedestrian Surveillance: pedestrian tracking and interaction recognition, and flexible and interpretable prior behavior models by means of synthetic agents.

4. SmartCar: data acquisition and playback software and hardware, graphical models for driver maneuver recognition and prediction at a tactical level, analysis of the most relevant features, and driver maneuver prediction on average **1 second** before the maneuver takes place.

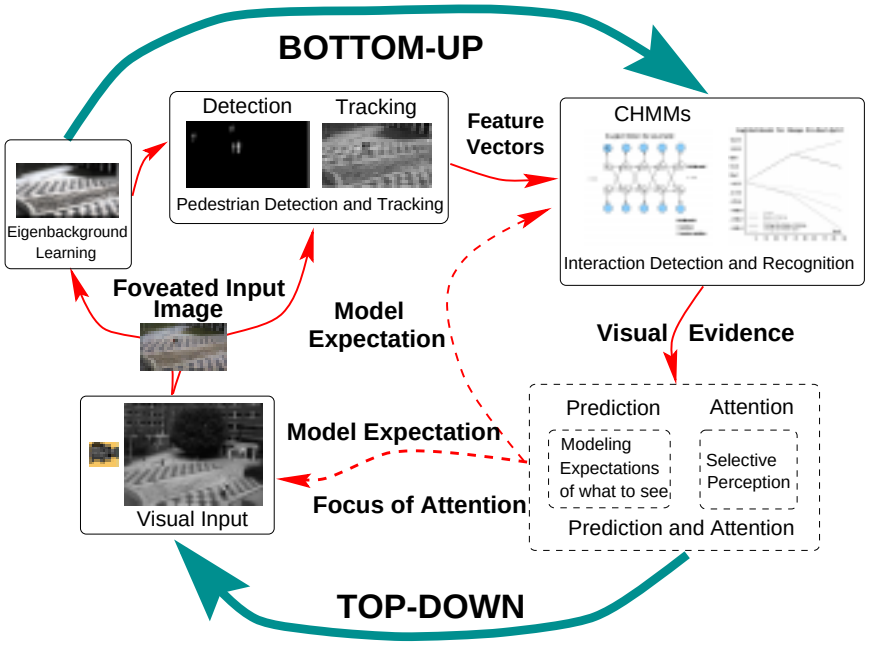
Thesis Structure

This thesis is structured as follows. Chapter 2 describes relevant theories of human behavior from Psychology and Philosophy, with emphasis on those theories that deal with time, causality and dynamics. First, the organization of action and the frame problem

are presented. Second, specific behavior theories that are relevant to this thesis work are described in some detail: trace analysis, GOMs (Goals, Operators, Methods and Selection Rules), Soar, automated and semi-automated analysis, functionalism, state-based models, and dynamic systems theory. Chapter 3 describes in detail the perceptual aspects of each of the systems developed in the thesis. Depending on the domain, different perceptual input modalities have been used: (1) In the case of facial expression recognition, an active camera looking at the user's face; (2) in the framework of pedestrian interactions recognition, a static camera with wide field-of-view watching a dynamic outdoor scene; (3) in the driver domain, multiple sensors of different nature are used: internal sensors of the car's internal state –acceleration, steering wheel angle, gear, speed and break pedal action–, and cameras for the visual context. The mathematical framework for learning from data individual, person-to-person or multi-agent interactive behaviors is presented in chapter 4. A detailed description of the theory behind graphical models and dynamic graphical models is presented. Chapter 5 describes the experiments that validate the mathematical approach described in chapter 4. I have developed four major testbeds for modeling human behavior in real situations. These systems are evaluated by their recognition accuracy on testing data. In the case of interacting behaviors, the performance of the CHMMs (see section 4.11) is compared to that of HMMs, a state-of-the-art competitive learning architecture. Some applications of the systems are also presented. Finally, chapter 6 summarizes this thesis work, highlights its major contributions and sketches future lines of research.

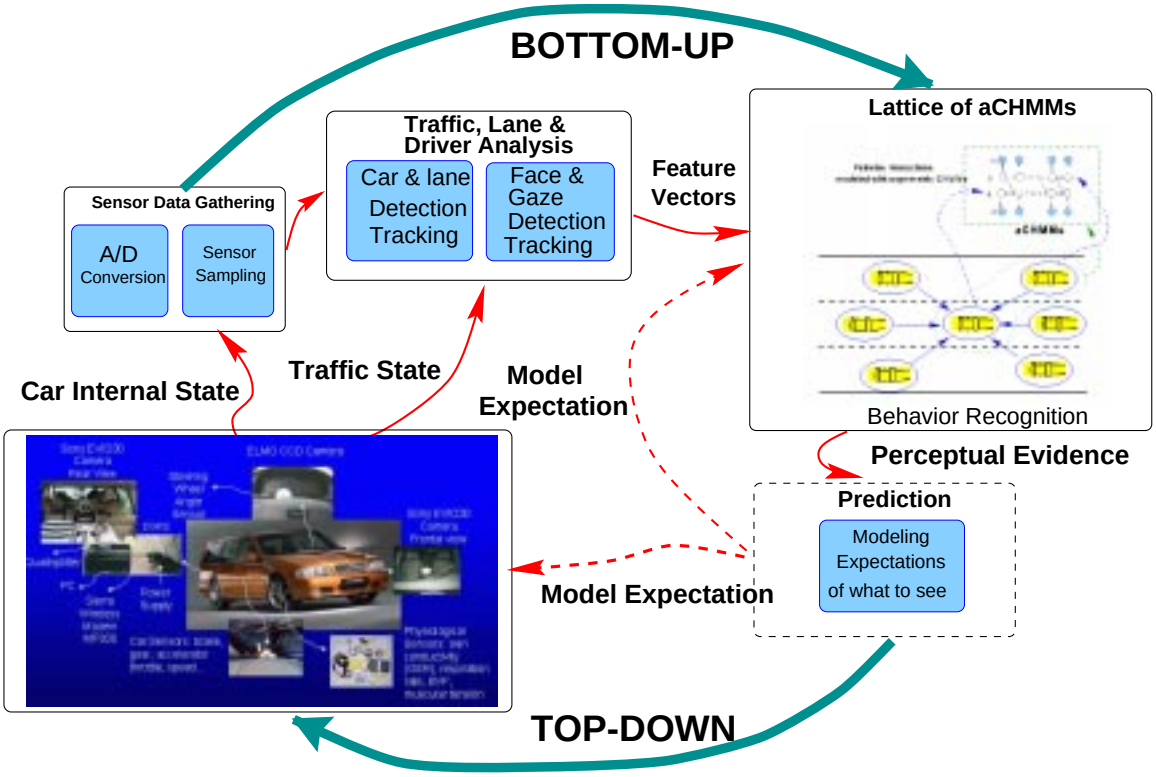


Architecture of LAFTER



Architecture of the Visual Surveillance system

Figure 1-1: System architecture for LAFTER and the visual surveillance systems



Architecture of the SmartCar testbed platform

Figure 1-2: SmartCar System architecture

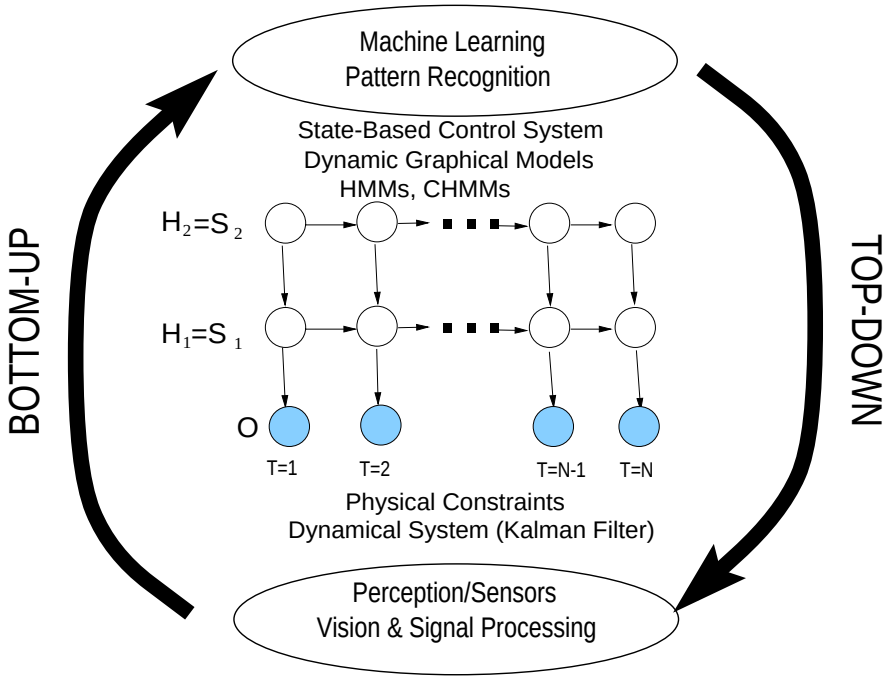


Figure 1-3: Proposed computational model for human behavior recognition and prediction

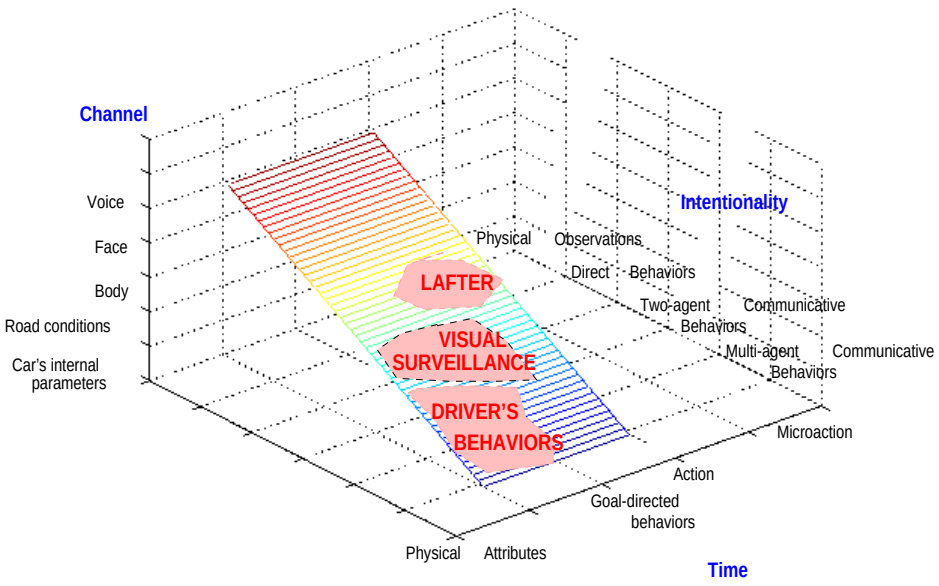


Figure 1-4: Taxonomy

Chapter 2

Related Human Behavior Models in Psychology and Philosophy

This chapter describes relevant theories of human behavior from Psychology and Philosophy. Given that this thesis work focuses on human behavior modeling by means of dynamic graphical models, this chapter emphasizes particularly those theories that deal with time, causality and dynamics. First, the organization of action and the frame problem are presented. Second, specific behavior theories that are relevant to this thesis work are described in some detail: trace analysis, GOMs (Goals, Operators, Methods and Selection Rules), Soar, automated and semi-automated analysis, functionalism, state-based models, and dynamic systems theory.

Some of the issues raised by these psychological and philosophical theories of human behavior are addressed in this thesis work from a computational viewpoint. For example, philosophers and psychologists have proposed finite state automata for explaning human behavior. The behavior recognition framework proposed in this thesis is a non-deterministic version of finite state automata based on dynamic graphical models (explained in chapter 4). These models capture the internal dynamics of the system in a probabilistic, statistically driven manner.

2.1 Organization of Action

The nature of intelligence lies in the organization principles that enable living organisms to make rapid adjustments of patterns of action in response to the environment. No movement

in nature is random, it always serves the purpose of "adapting" the state of the system to the external conditions. No matter how intelligent a living being's action appears to be, that action satisfies the same general principle. The reason human actions look more complex than the actions of inanimated matter is because of the complexity of the human machine, i.e. of the brain's neural circuitry. The subtleties of goal, intent, purpose are but consequences of the hierarchical synthesis of intermediate units. The elementary units of behavior (reflex, oscillator, servomechanism, i.e. external stimulus to internal signal to muscle contraction) are "catalyzed" by units at the higher levels of the system. Gallistel describes the interaction principles that govern the units of behavior (reciprocal facilitation, reciprocal inhibition, chaining, superimposition, acceleration/deceleration, corollary discharge, etc). The goal is to explain how an action that looks like a whole can be decomposed in many coordinated lower-level levels. The computational models of human behavior proposed in this thesis reflect this hierarchical structure where long, complex behaviors can be expressed as the succession of shorter and simpler actions (states in a HMM, for example).

2.1.1 The Frame Problem

According to McCarthy ([146],[148], [147]), knowledge representation must satisfy three fundamental requirements: ontological (must allow one to describe the relevant facts), epistemological (allow one to express the relevant knowledge) and heuristic (allow one to perform the relevant inference). Artificial Intelligence can be defined as the discipline that studies what can be represented in a formal manner (epistemology) and computed in an efficient manner (heuristic). McCarthy developed a situation calculus where temporally limited events, or *situations* (snapshots of the world at a given time), are represented by associating a situation of the world (set of facts that are true) to each moment in time. Actions and events are functions from states to states. An interval of time is a sequence of situations, a *chronicle* of the world. The history of the world is a partially ordered sequence of states and actions, where the states are permanent and the actions change. Each situation is expressed in a formula of first-order predicate logic. Causal relations between two situations can then be computed. A state is expressed by means of a logical expression that relates objects in that state. An action is expressed by a function that relates each state to another state. McCarthy's *frame problem* states that it is not possible to represent what does not change in the universe as a result of an action. There are two complementary paradoxes associated

with the frame problem: the *ramification problem* (infinite things change because one can go into greater and greater detail of description) and the *qualification problem* (the number of preconditions to an action is also infinite). Predicate circumscription consists of adding an axiom that states what is abnormal to the theory of what is known. Circumscription deals with default inference by minimizing abnormality. The objects that can be shown to have a certain property, from what is known of the world, are all the objects that satisfy that property (or, the only individuals for which that property holds are those individuals for which it must hold).

Causal organization is central to the explanation of behavior. A system's behavior is determined by its underlying causal organization. Given a pattern of causal interaction between substates of a system, for instance, there will be a computational description that captures that pattern. Computational descriptions of this kind provide a general framework for the explanation of behavior. The behavior models proposed in this thesis (dynamic graphical models) intrinsically capture causal relationships between the variables in the system. In the next sections I will describe in certain detail theories of human behavior that have played and play an important role within the Psychology and Philosophy communities. I will compare and contrast these theories with the computational model of human behavior proposed in this thesis.

2.2 Behavior Theories

This section describes behavior theories proposed and developed in Psychology and Philosophy viewpoint. Only theories that are relevant to this thesis work are presented. Time, causality and dynamics lie at the core of human behavior in general, and in particular of the models proposed in this thesis. In consequence, I will focus on the role that time, causality and dynamics play in these theories, starting with the simplest models and finishing with the most sophisticated models, where time and dynamics are central.

2.2.1 Tracing and GOMS

Tracing is a rigorous form of sequential protocol analysis. It has become increasingly popular in various fields and under various appellations. To name a few, cognitive scientists have employed trace-based protocol analysis to develop and refine cognitive process mod-

els; researchers in human-computer interaction have employed tracing, especially sequence comparison techniques, to study the fits of user models; and builders of intelligent tutoring systems have utilized model tracing or tracking to determine the user's solution path through a student model of the domain.

Newell and Simon ([164],[223]) provided arguably the most influencial contribution to the methodological foundations of tracing. Their work formalized the notion of the problem space and illustrated how subject protocols could help determine a subject's particular solution path through the space. They also demonstrated how one can test process models by mapping their predictions directly onto the observable actions of human subjects. This is closely related to the notion of *perceptually intelligent systems* and to the validation method used in this thesis.

One of the first implementations of Newell's et al proposed framework are GOMS. GOMS is a family of techniques proposed by Card, Moran, and Newell ([41]), for modeling and describing human task performance. GOMS is an acronym that stands for Goals, Operators, Methods, and Selection Rules, the components of which are used as the building blocks for a GOMS model. Goals represent the goals that a user is trying to accomplish, usually specified in a hierarchical manner. Operators are the set of atomic-level operations with which a user composes a solution to a goal. Methods represent sequences of operators, grouped together to accomplish a single goal. Selection Rules are used to decide which method to use for solving a goal when several are applicable.

Once the GOMS model has been developed, predictions of learning and performance can be obtained. A GOMS description is also a way to characterize a set of design decisions from the point of view of the user, which can make it useful during, as well as after, design. It is also a description of what the user must learn, and so can act as a basis for training and reference documentation.

Actually carrying out a GOMS analysis involves defining and then describing in a formal notation the four basic elements, i.e. the user's Goals, Operators, Methods, and Selection Rules. The hardest elements to identify and define are the Goals and Methods. The Operators are mostly determined by the hardware and lowest-level software of the system, such as whether it has a mouse, for example. Thus the Operators are fairly easy to define. The Selection Rules can be subtle, but usually they are involved only when there are clear multiple methods for the same goal. In a good design, it is clear when each Method should

be used, so defining the Selection Rules is (or should be) relatively easy as well.

Identifying and defining the user's Goals is often difficult, because it requires detailed examination of the task that the user is trying to accomplish, often going beyond just the specific system to the context in which the system is being used. This is especially important in designing a new system, because a good design is one that fits not just the task considered in isolation, but also how the system will be used in the user's job context.

One critical process involved in doing a GOMS analysis is deciding what and what not to describe. The mental processes of the user can be of incredible complexity; trying to describe all of them would be hopeless. However, many of these complex processes have nothing to do with the design of the interface, and so do not need to be analyzed. For example, the process of reading is extraordinarily complex; but usually, design choices for a user interface can be made without any detailed consideration of how the reading process works. We can treat the user's reading mechanisms as a "black box" during the interface design. We may want to know how much reading has to be done, but rarely do we need to know how it is done. So, we will need to describe when something is read, and why it is read, but we will not need to describe the actual processes involved. A way to handle this in a GOMS analysis is to bypass the reading process by representing it with a *dummy* or place holder operator. Making the choices of what to bypass is an important, and sometimes difficult, part of the analysis. This is related to feature selection problem, where the most relevant, meaningful features to the particular task should be considered. It is, definitely, an open question how to determine those features. The main goal is to find a small number of relatively predictive features rather than very large number of features that, taken in the proper but untractably complex combination, are entirely predictive of the class label. Irrelevant and redundant features cause problems by adding noise to the learning algorithm and therefore obscuring the distributions of the small set of truly relevant features for the task at hand. Two purposes are served by reducing the set of features considered by an algorithm: first, from a purely computational viewpoint, we can considerably decrease the running time of the induction algorithm; second, and more importantly, the accuracy of the increasing model is increased ([126],[111]).

Psychological Basis

The cognitive architecture that inspired GOMS techniques is the so called Model Human Processor (MHP) [41]. According to the Model Human Processor, representation of human cognition consists of separate components for cognitive, motor, and perceptual processors (and associated buffers), as well as for long and short-term memory. The components of GOMS map onto this model in one form or another. For instance, control in the MHP is central to the cognitive processor, where execution of methods and selection rules is assumed to take place. Likewise, the execution of operators can be seen as the issuance of commands by the cognitive processor to the other components. The two-layer model proposed in this thesis (see figure 1-3) includes perceptual, cognitive and motor modules: cameras and other sensors, together with computer vision and signal processing modules at the perceptual level; active control system for a camera at the control level; and dynamic graphical models or Dynamic Probabilistic Networks (DynPINs) at the cognitive level.

Uses of GOMS

From a research standpoint, GOMS provides a framework for modeling aspects of human performance and cognition. From an applied perspective, GOMS provides a rich set of techniques for evaluating human performance on any system where people interact with machines. GOMS analysis can provide much insight into an system's usability, such as, task execution time, task learning time, operator sequencing, functional coverage, functional consistency, and aspects of error tolerance. Some type of GOMS analysis can be conducted at almost any stage of system development, from design and allocation of function to prototype design, detailed design, and training and documentation for operation and maintenance. Such analysis is possible for both new designs and redesigns of existing systems.

Varieties of GOMS

Card [41] defined a sufficiently broad framework for GOMS that allows room for multiple analysis and modeling techniques at many different levels. Of the many such techniques they proposed and that others have proposed since, several are in currently in common use: **(1) Keystroke Level Model (KLM)** The simplest GOMS technique is the Keystroke

Level Model (KLM) ([41]). It deals mainly with observable events and is organized as a single stream of sequential operators. KLM is easy to learn and can quickly provide crude task-execution times. **(2) Card, Moran, and Newell GOMS (CMN-GOMS)** Also from the original Card, Moran, and Newell proposal, CMN-GOMS added hierarchical structure to KLM. Tasks are organized as a series of goals and subgoals and operators are organized into subroutines called methods. CMN-GOMS can provide task execution times and affords a better view of the task structure. **(3) Natural GOMS Language (NGOMSL)** NGOMSL ([123]) was developed as a formally defined version of CMN-GOMS based on cognitive complexity theory (CCT). It has a more structured hierarchy than CMN-GOMS and a well-defined analysis methodology for developing models. In addition to the execution time and task structure information provided by CMN-GOMS, its CCT roots allow for learning time predictions, as well. **(4) Cognitive Perceptual Motor GOMS (CPM-GOMS)** CPM-GOMS ([110]) is also based on CMN-GOMS with an emphasis on parallel activities. Where other GOMS techniques assume that humans do one thing at a time, CPM-GOMS assumes as many operations as possible will happen at any given time subject to constraints of the cognitive, perceptual, and motor processes. Models are developed using PERT charts and execution time is derived from the critical path. In one application to manuscript editing, they traced user actions with GOMS model predictions and observed sequences with a sequence- distance metric. Their work stressed the need to utilize quantitative measures of a trace's goodness of fit along with traditional quantitative analyses. The models proposed in this thesis also provide such quantitative metrics.

Among other efforts to carry highlight the significant methodological contributions of Newell's et al work, I would emphasize Ohlsson's ([169]) trace analysis. He has formalized Newell's et al methodology as a three-step process: (1) subject's problem space identification and construction; (2) subject's solution path identification by making use of the sequential information in the protocol; (3) subject's strategy hypothesis by inventing problem-solving heuristics that can reproduce the subject's solution path. The formulation proposed in this thesis, via dynamic graphical models provides a mathematically sound framework for carrying out the three previous steps: (1) the problem space is determined by the model: the graph structure, the selected features, etc, (2) the solution path is given by well-defined inference and MAP estimation algorithms defined over the graph, (3) and the subject's solution path is reproduced by the models, given that they are generative and learnt from

real data.

2.2.2 Soar

Ritter ([206]) examined tracing at length and specified a methodology, trace-based analysis, for testing process models' predictions through comparison with verbal and non-verbal protocols. His formulation of trace-based protocol analysis, reminiscent of Ohlsson's trace analysis, comprised the following steps: (1) using a process model, generate a sequence of predicted actions; (2) compare the model predictions to empirical data by forming a mapping between the predicted action sequence and the observed action sequence; (3) analyze the fit of the model to the data to see where the model can be improved; (4) refine the model and iterate. There is great overlap between this methodology and the machine learning framework used in this thesis. The main difference is that the behavior models in this thesis are automatically learned from data, whereas Ritter's models are manually defined and revised. In the proposed learning framework, the previous steps are automatically included in the training procedure, where the models parameter estimation is performed in terms of maximizing the likelihood of the data given the model.

Soar is an architecture for human cognition expressed in the form of a production system. It involves the collaboration of a number of researchers including Allen Newell, John Laird and Paul Rosenbloom and others at different institutions. The theory builds upon earlier efforts involving Newell such as GPS ([165]) and GOMS ([41]). Like the latter model, Soar is capable of simulating actual responses and response times.

Using their Soar/MT system, Ritter and Larkin ([207]) have successfully developed a process model for users of a computer interface. The predictions of the cognitive model in Soar/MT require that the model's sequence of predicted actions be deterministic, whereas the human behavior recognition methods developed in this thesis allow for non-deterministic action traces using a statistical framework.

The principal element in Soar is the idea of a problem space: all cognitive acts are some form of search task. Memory is unitary and procedural; there is no distinction between procedural and declarative memory. Chunking is the primary mechanism for learning and represents the conversion of problem-solving acts into long-term memory. Soar exhibits a variety of different types or levels of learning: operators (e.g., create, call), search control (e.g., operator selection, plans), declarative data (e.g., recognition/recall), and tasks (e.g.,

identify problem spaces, initial/goal states). Soar is capable of transfer within or across trials or tasks.

Scope/Application: Newell ([166]) has positioned Soar as the basis for a unified theory of cognition and attempts to show how it explains a wide range of past results and phenomena. For example, he provides interpretations for response time data, verbal learning tasks, reasoning tasks, mental models and skill acquisition. In addition, versions of Soar have been developed that perform as intelligent systems for configuring computer systems and formulating algorithms.

2.2.3 Automated and Semi-automated Analysis

Even thought the tracing protocols presented so far constitute a milestone towards the explanation and modeling of human behavior and task performance, they are manual: their definition, implementation and practical operation are full responsibility of the researcher, being, thus, extremely tedious. In consequence, several researchers have attempted to automate the process. For instance, Waterman and Newell ([257]) developed PAS-II, a system for the automated analysis of verbal reports. PAS-II mapped subjects' verbal protocols onto a problem behavior graph, which describes the trajectory of a subject's solution through a problem space. PAS-II allowed the user to interactively take part in the analysis: the user 'can provide answers to subproblems the system is unable to solve, correct processing errors, and even maintain control over the processing sequence' ([257]).

Although PAS-II and other similar systems represented a significant attempt at automating protocol analysis, the systems, as the authors themselves admit, constitute only one component task of the larger picture of protocol analysis.

In another attempt to automate tracing, Smith et al. ([224]) employed cognitive grammars to represent cognitive strategies and parse verbal, keystroke, video and action protocols. Using a cognitive theory of writing, they implemented three types of cognitive grammars for an expository writing task: a production rule grammar, an augmented transition network, and an episode grammar. All three grammars could successfully parse subject protocols into a parse tree of higher-order cognitive actions, symbolizing the model's interpretation of the observed behavior. One of the most important drawbacks of such methods is that the parsing must be complete and exact, with no allowance for deviations from

the predicted model sequences. Real data, however, it's generally too noisy for such interpretation. The process of recognizing real human behavior, gathered by noisy sensors in real situations, must incorporate robust methods that tolerate noise and unexpected or unpredicted actions. As it is described in chapter 4, the behavior models proposed in this thesis, by means of dynamic graphical models, offer a mathematically sound framework for incorporating uncertainty, noise and missing data.

Automated tracing has been successfully employed in intelligent tutoring systems. For instance, Anderson et al. ([7]) describe how model tracing can map student actions in a tutoring system to the predictions of an ACT-R cognitive model. Model tracing in such systems assumes that each student actions corresponds to a unique problem-solving strategy and cannot backtrack in the case of ambiguous actions. It is also limited to the analysis of non-noisy actions such as key presses or mouse clicks.

Several other researchers have investigated automated and semi-automated techniques that do not implement tracing per se but do highlight common goals to the above work and this thesis. Lallement ([132]) used decision trees to classify data from an air-traffic controller task. His work showed that such machine learning techniques can provide automated analysis that is more consistent and faster than analysis by hand. Sanderson et al's ([215]) MacSHAPA system allows for sequential protocol analysis, including sequence comparisons, Fisher cycles, Markov transition statistics, and lag sequential analysis.

Cognitive Process Models

One of the requirements of tracing techniques is a cognitive process model that can generate predicted action sequences to be matched up against observed action protocols. There are a number of modeling systems that allow for such models, including ACT-R ([8]), Soar ([166]), EPIC ([124]), GOMS ([41]), and E-Z Reader ([204]).

Limited effort has been invested into finding automated methods of generating appropriate process models for a task domain ([134], [75]). For a given problem space, these systems infer the conditions under which operators can apply using positive and negative training examples. While the systems do address part of the modeling problem, they still require full specification of the problem space (composed of representation and operators), which in itself is a major component of the modeling process. However such efforts suggest that the future of automated modeling seems promising. This thesis contributes in this area

by proposing and testing with non-simulated data a model of human behavior recognition and prediction. The proposed model is generative, predictive and automatically learnt from data.

2.2.4 Functionalism and the Theory of Mind

In the cognitive scientific as well as the philosophical community, one of the most popular account of people's understanding of mental-state language is the "theory of mind" theory, according to which naive speakers, even children, have a theory of mental states and understand mental words solely in terms of that theory. The most precise statement of this position is the philosophical doctrine of functionalism. Functionalism says that the crucial or defining feature of any type of mental state is the set of *causal relations* it bears to (1) environmental or proximal inputs, (2) other types of mental states, and (3) behavioral outputs. The term "functionalism" is broad enough to incorporate a very rich spectrum of different doctrines. In particular, there is what is called *scientific functionalism* (psychofunctionalism, in Block's terms [23]), according to which it is a scientific fact that mental states are functional states. That is, mental states have functional properties (i.e., causal relations to inputs, other mental states, and outputs) and should be studied in terms of their functional properties. It seems clear that mental states have functional properties; and therefore mental states should be studied (at least in part) in terms of these properties. But this doctrine does not entail that ordinary people understand or represent mental words as designating functional properties only. Another variation of functionalism is *representational functionalism*, or RF, where the focus is the psychological realization of analytic or commonsense functionalism. It is in contrast to a more abstract traditional philosophical approach. In RF, one considers it as a psychological hypothesis, i.e., a hypothesis about how the cognitive system represents mental words. This form of functionalism is interpreted as hypothesizing that the cognitive representation associated with each mental predicate M represents a distinctive set of functional properties, or functional role, FM. The doctrine holds that folk wisdom embodies a theory, or a set of generalizations, which articulate an elaborate network of relations of three kinds: (A) relations between distal or proximal stimuli (inputs) and internal states, (B) relations between internal states and other internal states, and (C) relations between internal states and items of overt behavior (outputs). Here is a sample of such laws due to Churchland ([45]). Under heading (A) (relations between

inputs and internal states) we might have [1]:

'When the body is damaged, a feeling of pain tends to occur at the point of damage. When no fluids are imbibed for some time, one tends to feel thirsty. When a red apple is present in daylight (and one is looking at it attentively), one will have a red visual experience'.

Under heading (B) (relations between internal states and other internal states) we might have:

'Feelings of pain tend to be followed by desires to relieve that pain. Feelings of thirst tend to be followed by desires for potable fluids. If one believes that P, where P elementarily entails Q, one also tends to believe that Q'.

Under heading (C) (relations between internal states and outputs) we might have:

'Sudden sharp pains tend to produce wincing. States of anger tend to produce frowning. An intention to curl one's finger tends to produce the curling of one's finger'.

According to RF, each mental predicate picks out a state with a distinctive collection, or syndrome, of relations of types (A), (B) and/or (C)). The term pain, for example, picks out a state which tends to be caused by bodily damage, tends to produce a desire to get rid of that state, and tends to produce wincing, groaning, etc. The content of each mental predicate is given by its unique set of relations, or functional role, and nothing else. In other words, RF attributes to people a purely relational concept of mental states.

There are slight variations and important additional nuances in the formulations of functionalism. Some formulations, for example, talk about the causal relations among stimulus inputs, internal states, and behavioral outputs. Others merely talk about transitional relations, i.e., one state following another. The dynamic graphical models used in this thesis (HMMs and CHMMs) offer a formal framework for representing both causal relations –via the graph structure– and the transitional relations –via the transition probability matrices between adjacents states– (see chapter 4).

One important problem of RF concerns how a subject can determine which functional type a given state-token instantiates. There is a clear threat of combinatorial explosion: too many other internal states will have to be type-identified in order to identify the target state. This problem is not easily quantified with precision, because we lack an explicitly formulated and complete functional theory, so we don't know how many other internal states

[1] From [43]

are directly or indirectly invoked by any single functional role. The problem is particularly acute, because under standard formulations of functionalism, beliefs, desires and other propositional attitudes have strong holistic properties. A given belief may causally interact with quite a large number of other belief tokens and desire tokens. To type-identify that belief, it looks as if the subject must track its relations to each of these other internal states, their relations to further internal states, and so on until each path terminates in an input or an output. The combinatorial explosion of possible relations and interactions makes the system untractable. For each desire or goal-state there are indefinitely many beliefs with which it could combine to produce a further desire or subgoal. Similarly, for each belief there are infinitely many possible desires with which it could combine to produce a further desire or subgoal, and infinitely many other beliefs with which it could combine to produce a further belief. If the type-identification of a target state depends on tracking all of these relations until inputs and outputs are reached, clearly it is unmanageably complex. At a minimum, we can see this as a challenge to an RF theorist, a challenge which no functionalist has tried to meet, and one which looks pretty forbidding. Here the possibility of partial matching may assist the RF theorist to account for mental-state classification.

There are two crucial features of a functionalist viewpoint that are relevant to Perceptual Intelligence. The first feature is pure relationalism. RF claims that the way subjects represent mental predicates is by relations to inputs, outputs, and other internal states. The other internal-state concepts are similarly represented. Thus, every internal-state concept is ultimately tied to external inputs and outputs. Perception plays, therefore, a crucial role and is tightly coupled to internal representations. What is deliberately excluded from our understanding of mental predicates, according to RF, is any reference to the phenomenology or experiential aspects of mental events (unless these can be expressed in relationalist terms). No intrinsic character of mental states are appealed to by RF in explaining the subject's basic conception or understanding of mental predicates. The second crucial feature of RF is the appeal to nomological (lawlike) generalizations in providing the links between each mental-state concept and suitably chosen inputs, outputs, and other mental states. Thus, if subjects are to exemplify RF, they must mentally represent laws of the appropriate sort. How are these laws acquired? Are they learned? Does empirical research on "theory of mind" support either of these two crucial features? The answer is unknown, because, at the moment, very few of the leading researchers in these topics, if any, construe a "theory

of mind" in quite the sense specified here. They usually endorse vaguer and weaker views.

2.2.5 State-based Models of human Behavior

I have presented so far classical and relational functionalism, where mental states are functional states. Putnam –a classical functionalist– has argued that computational functionalism cannot serve as a foundation for the study of the mind, as every ordinary open physical system implements every finite-state automaton. Chalmers [44], [43] argues, on the other hand, that Putnam's argument fails, but that it points out the need for a better understanding of the bridge between the theory of computation and the theory of physical systems. It also raises questions about the classes of automata that can serve as a basis for understanding the mind. Chalmers develops an account of implementation, linked to an appropriate class of automata, such that the requirement that a system implements a given automaton places a very strong constraint on the system. This clears the way for computation to play a central role in the analysis of mind. These theories are directly relevant to this thesis work: all the models developed in this thesis are stochastic state-based computational models, as opposed to deterministic finite state automata.

According to Chalmers [44], [43] 'it is sufficient to require that a system reliably transits through a sequence of states $s_1, s_2, ...$, irrespective of environmental conditions. This is not a difficult requirement: most clocks satisfy it, for instance. Probably most physical systems satisfy such a requirement; perhaps we might find reliable sequences like this in patterns of radiation decay. In any case, let us say a physical system contains a clock if it has a subsystem that reliably transits through a sequence like this. A system containing a clock will circumvent the first objection. If we define the states $s_1, s_2, ...$ of the system as those states containing the relevant states of the clock, then the transition from s_n to s_{n+1} will be reliable. If disjunctive states a, b, and so on are defined appropriately, then the transitions between these will satisfy the appropriate strong conditionals. Moreover we need to make sure that the system has sufficient extra states to map onto formal states that are not manifested on a given run'. Chalmers claims that we can do this by ensuring that the system contains a dial: that is, a subsystem with an arbitrary number of different states, such that when it is put into one of those states it stays in that state.

In particular, to put stronger constraints on structure Chalmers argues that one needs to move to Finite State Automata (FSAs) with inputs and perhaps with outputs. The

addition of input changes the formalism from trivial to non-trivial. Where there is input, there can be branching behavior. A formal state can be succeeded by various different formal states, depending on the input. Furthermore, the presence of input gives the formalism a kind of combinatorial structure. Later states depend not just on a single state, but on a combination of state and input. Once more, perception and cognition are tightly coupled, as in the two-layer computational model proposed in this thesis, where "perception" lies at the bottom and "cognition" lies at the top.

This formalism is much more appropriate for capturing the dynamics of cognitive systems. Humans do not have a single path of states along which their lives are determined. Even if they do, as some fatalistic views might suggest, this path does not exhaust their description. For any given sequence of states that a human goes through, there is always the case that if things in the world had gone slightly differently, they would have functioned in an interestingly different way. Omitting this potentiality leaves out a vital part of the description of human functioning. A wind-up toy or perhaps a videotape of my life could go through the same sequence of states, but it would not be a cognitive system. Cognition requires at least the possibility of functioning in more than one way. A statistical, non-deterministic approach seems therefore appropriate, as the one taken in this thesis.

Even simple FSAs with inputs and outputs are not a rich enough model to capture the kind of complex structure that computation and cognition involve. The trouble is that the internal states of these FSAs are monadic, lacking any internal structure, whereas the internal states of most computational and cognitive systems have all sorts of complex structure. Generally these states are divisible into components which interact locally and globally according to complex principles. Just as the structure of the system is not captured by a monadic state description, neither are the state-transitions captured by a monadic state-change. There may be all sorts of local dependencies that go into the functioning of such a system. Thus a hierarchical architecture, as the one presented in figure 1-3, seems the most appropriate.

Chalmers claims that often the state-transitions of a FSA will be defined in terms of local dependencies, as when a substate depends only on a few neighboring substates and perhaps on a few inputs rather than on the entire previous state and input vectors (this will be so for cellular automata and Turing machines, for instance). In this case, we can require that the appropriate restricted conditional holds: that is, if the physical system is

in the (few) specified previous substates and receiving the specified inputs, this causes it to transit appropriately. We are therefore requiring a Markov condition in the system's dynamic behavior. Once more the connection to this thesis work is very direct, for the models developed in this thesis are all first order Markov models.

In summary, the model proposed in this thesis captures the input-output nature proposed by Chalmers. The perceptual system gathers contextual, multimodal input data (video, audio, car signals) and processes it. The machine learning modules implemented by use of dynamic graphical models (HMMs and CHMMs) model the internal dynamics of the system in a probabilistic, statistically driven manner. The first order Markov condition of the models formalizes the fact that the entire history of the system is represented and summarized by the previous state. Of course, this is a strong assumption in the system's dynamics. Higher order Markov chains might be necessary for capturing longer term causal relationships.

To which degree can a computational framework model the complexity of human behavior? Isn't it, perhaps, a pretentious aspiration of computer scientists? The theory of computation is often thought to underwrite the theory of mind. In cognitive science, it is widely believed that intelligent behavior is enabled by the fact that the mind or the brain implements some abstract automaton: perhaps a Turing machine, a program, an abstract neural network, or a finite-state automaton. From a formal viewpoint, the ambitions of artificial intelligence rest on a related claim of computational sufficiency, holding that there is a class of automata such that any implementation of an automaton in that class will possess a mind. A similar claim is often made about many specific mental properties, including properties characteristic of human mentality: that is, it is claimed that there exists a class of automata such that any implementation of an automaton in that class will have the mental property in question. In this way, it is hoped that computation will provide a powerful formalism for the replication and explanation of mentality.

In the case of this thesis, there is a well-motivated formalism, graphical models, and an associated implementation, a particular graphical structure, parameters and inference algorithms, in order to model (recognize and predict) some real-life behavior. Of course, it is critical that the proposed architecture have enough expressive power to capture the intrinsic properties of the real-life situation, i.e. the conditional independencies encoded in the graph structure should mirror as closely as possible the causal organization of the real

system being modeled. In this way, a bridge would be established between the computational theories and the physical systems of everyday life. This school of thought opens the way to a computational foundation for the theory of mind.

Note that this only accounts for half of the problem. Moreover, for the easy half. The harder part is to take advantage of this bridge, showing that the physical properties that a computational description formalizes are the properties in virtue of which minds arise. It is not implausible that minds arise in virtue of causal organization, but neither is it obvious. It is also plausible but not obvious that a specific graph structure can capture the precise causal organization (perhaps continuous, perhaps even non-computable) on which mentality depends.

2.2.6 Dynamicist Theory of Cognition

Traditionally there have been two opposed theories of cognition: (1) The cognitive, *computational hypothesis* and (2) the more empirical *connectionist approach*. The former is inspired in Thomas Hobbes'es ([95]) model of the mechanisms of mental operation. According to Hobbes, perhaps thought is nothing but symbolic computation, the rule-governed manipulation of symbols inside the head. Seventeenth-century speculation became twentieth-century science. Hobbes'es idea evolved into the computational hypothesis, according to which cognitive agents are basically digital computers. Perhaps the most famous rendition is Newell and Simon's ([165]) doctrine that 'A physical symbol system has the necessary and sufficient means for general intelligent action.' They proposed this hypothesis as a *law of qualitative structure*, comparable to the cell doctrine in biology or plate tectonics in geology. It expresses the central insight of the research paradigm which has dominated cognitive science for some forty years.

However, and specially in recent years, the empirical –Humean– alternative has been gaining momentum. One of the most notable developments has been the rise of connectionism, which models cognition as the behavior of dynamical systems ([225]), and often understands those models from a dynamical perspective. Equally significant is the emergence of cognitive neuroscience, and within it, the increasing prevalence of dynamical theorising. Dynamics forms the general framework for growing amounts of work in psychophysics, perception, motor control, developmental psychology, cognitive psychology, situated robotics and autonomous agents research, artificial intelligence, and social psychology. It is central

to a number of general approaches, such as ecological psychology, synergetics, and morphodynamics.

Since the emergence of connectionism in the 1980s, connectionism and symbolicism have been the two main paradigms of cognitive science ([18]). However, in recent years, a new approach to the study of cognition has challenged their dominance; that new approach is called dynamicism. There have been a series of papers and books [242], [248] that have advanced the claim that cognition is not best understood as symbolic manipulation or connectionist processing, but rather as *complex, dynamical interactions of a cognitive agent with its environment.* The dynamicist approach to cognitive modeling employs concepts developed in the mathematical field of dynamical systems theory. They claim that cognitive models should be embedded, low-dimensional, complex, described by coupled differential equations, and non-representational. Dynamicists have criticized both symbolicism and connectionism and have decided to dismiss these theories of cognition and instead wish to propose a *'radical departure from current cognitive theory'*, one in which 'there are no structures', as opposed to connectionist approaches, and 'there are no rules' ([242]), as opposed to symbolicist approaches. This new conception of cognitive functioning is intended to replace the currently dominant theories of connectionism and symbolicism. In summary, the dynamical hypothesis is the unifying essence of dynamical approaches to cognition. It is encapsulated in the simple premise that *cognitive agents are dynamical systems.*

Through their discussion of the dynamicist hypothesis, dynamicists identify those certain kinds of dynamical systems which are suitable to describing cognition. Specifically, they are: 'state-determined systems whose behavior is governed by differential equations... Dynamical systems in this strict sense always have variables that are evolving continuously and simultaneously and which at any point in time are mutually determining each other's evolution' ([248]) – in other words, systems governed by coupled nonlinear differential equations. Thus the dynamicist hypothesis has determined that a dynamicist model must have a number of component behaviors, they must be: deterministic; generally complex; described with respect to the independent variable of time; of low dimensionality; and intimately linked ([248]).

Some of the key features that dynamic graphical models share with the dynamicist approach are: they focus on the dynamic aspects of systems; they decompose a system in terms of their variables and the states they can be in; they are computational and quan-

titative in space, time, or both; they deal with dimensionality reduction and parameter estimation. However, they differ also in some crucial points: the models proposed in this thesis are stochastic, statistical, data-driven models versus the deterministic nature of dynamical systems theory; they have a well defined structure that captures the causal relations of the variables, versus the lack of structure of dynamical systems; the systems dynamics is modeled by transition probabilities and not by differential equations as in the dynamical approach; they treat time discretely whereas dynamical systems theory was designed to describe continuous temporal behaviors. I will present in more detail the dynamicists viewpoint to better understand the similarities and differences between dynamical systems and dynamic graphical models.

Dynamical Systems Theory

The branch of mathematics called dynamical systems theory describes the natural world with essentially geometrical concepts. Concepts commonly employed by dynamicists include: state space, path or trajectory, topology, and attractor. The state space of a system is simply the space defined by the set of all possible states that the system could ever pass through. A trajectory plots a particular succession of states through the state space and is commonly equated with the behavior of the system. The topology of the state space describes the "attractive" properties of all points of the state space. Finally, an attractor is a point or path in the state space towards which the trajectory will tend when in the neighborhood of that attractor. Employing these concepts, dynamicists attempt to predict the behavior of a cognitive system if they are given the set of governing equations (which will define the state space, topology and attractors) and a state on the trajectory. The fact that dynamical systems theory employs a novel set of metaphors for thinking about cognition is paramount. These metaphors offer a perspective on cognition that is instrumental in understanding some of the problems of cognitive science. The dynamical hypothesis states that cognitive agents are dynamical systems. This hypothesis has two major components. The *nature hypothesis*: a claim about the nature of cognitive agents themselves; it specifies what they are (i.e., dynamical systems); and the *knowledge hypothesis*: a claim about cognitive science: namely, that we can and should understand cognition dynamically.

There are numerous practical and theoretical advantages of dynamical systems theory descriptions of cognition. The most obvious advantage is that dynamical systems theory is a

proven empirical theory, as opposed to purely theoretical symbolic approaches to cognition. Thus, the differential equations used in formulating a description of a cognitive system can be analyzed and (often) solved using known techniques. One result of having chosen this mathematical basis for a description of cognition is that dynamicists are bound to a deterministic view of cognition. The behavior modeling framework introduced in this thesis, based on graphical models, is non deterministic, but probabilistic.

Another advantage is the disposition of dynamical descriptions to exhibit complex and chaotic behavior. Dynamicists convincingly argue that human behavior, the target of their dynamical description, is quite complex and in some instances chaotic ([247], [242]).

I will describe in the following some key concepts not only in the dynamicist framework, but also in the framework of the dynamic graphical models (DynPINs) proposed in this thesis.

Systems

Systems are here taken to be sets of interdependent variables. A variable is simply some entity that can change, i.e., be in different states at different times. Variables are interdependent when the way any one changes depends on others, and change in others depends on it. The state of the system is simply the state or value of all its variables at a time; the behavior of the system consists of transitions between states. In dynamic graphical models, a system is composed of a graph structure and its parameters. The graph nodes are random variables, some observed and some hidden. Their interdependencies are captured by the graph structure, more specifically the absence of edges (links) between variables represents conditional independence assumptions.

Often, change in a system depends on factors outside the system itself (e.g., the force of gravity), referred to here as parameters. Sometimes, changes in a parameter depend in turn on the system itself. For example, the position of the moon both depends upon, and affects, the position of the planets. This kind of reciprocal, direct dependence is known as *coupling*. System variables and coupled parameters can be regarded as forming a larger system. This illustrates the semi-arbitrariness of systems. It is always up to us to nominate a set of concrete variables as the system we will study. Reality determines whether that set is in fact a system, and how it behaves.

All systems in the current sense change in time. In general, time is just some intrinsically

ordered set, or order, serving to provide orderings over other things. The real time of concrete systems is the set of instants at which things can actually happen, ordered by temporal priority (before/after). Concrete events are paired with instants or periods of time, and hence stand in temporal relations with each other. Abstract systems are not situated in real time at all, and so must take some other set as their time set; usually, it is the positive integers or the real numbers. The mathematical rule imposes orderings over states of the system by pairing them with members of this set.

Roughly, systems are quantitative when there are distances in state or time, such that these distances matter to behavior. This can be true in progressively deeper ways, giving rise to progressively more substantial senses in which a system can count as dynamical. Systems can be quantitative in state, time or the relationship of both:

1. Quantitative in state. First, there can be distances between any two overall states of the system, such that the behavior of the system depends on these distances. More precisely, a system is quantitative in state when there is a metric over the state set such that behavior is systematically related to distances as measured by that metric. Such systems will be governed by a rule compactly specifying this distance-dependent change. For example, the transition matrices in a HMM or CHMM describe how the system changes by telling us the probability of each state at time $t+1$ given the state at time t, i.e. $p(s_{t+1}|s_t)$. Standardly, the relevant quantitive properties of state sets are derived from quantitive properties of the variables. Quantitive variables can be either abstract or concrete.

2. Quantitative state/time interdependence. A system is quantitative in time when time is a quantity, i.e., there is a metric over the time set, such that system behavior is systematically related to distances as measured by that metric. At least in cognitive science practice, systems that are quantitative in time are also quantitative in space, and these properties are interdependent. That is, the behavior of the system is such that amounts of change in state are systematically related to amounts of elapsed time. Such systems are governed by a rule specifying a quantitative relationship between change in state, elapsed time, and current state. In concrete systems, this rule captures *causal organization*; that is, the system changes as it does because system variables have the quantitative properties in terms of which the rule is expressed. When both

state and time are quantitative, the system exhibits rates of change. Systems that are interdependently quantitative in state and time are governed by rules specifying the rate of change in terms of current state. Dynamic graphical models model the state of random variables over time, where time is a discrete variable and the state is either discrete or continuous. They are, therefore, quantitative in time and state.

3. Rate dependence. Third, some systems are such that their rates of change depend on current rates of change. In these systems, variables include both basic variables and the rates of change of those variables. Systems whose behavior is governed by rules most compactly expressed as sets of higher-order differential equations are quantitative in this sense. None of the human behavior models built in this thesis captures explicitly rates of change in its variables.

In what follows, a system is taken to be dynamical to the extent that it is quantitative in one of the above senses. At least three considerations support this approach. First, it reflects the actual practice of cognitive scientists in classifying systems as dynamical or not, or as more or less dynamical. Second, it fits comfortably with existing definitions. Third, it is cast in terms of deep, theoretically significant properties of systems. For example, a system that is quantitative in state is one whose states form a space such that states are positions in that space, and behaviors are paths or trajectories. Thus quantitative systems support a geometric perspective on system behavior.

Dimensionality

In order to avoid the difficult analyses of high-dimensional dynamical systems, dynamicists have claimed that accurate descriptions of cognition are achievable with low-dimensional descriptions. The aim of dynamicists is to 'provide a low-dimensional model that provides a scientifically tractable description of the same qualitative dynamics as is exhibited by the high-dimensional system (the brain)' ([247], p. 28).

The dimension of a dynamical systems model is simply equal to the number of parameters in the system of equations describing a model's behavior. Thus, a low dimensional model has few parameters and a high dimensional model has many parameters. The dimensionality of a system refers to the size of its state space. Therefore, each axis in the state space corresponds to the set of values a particular parameter can have.

The low dimensionality of dynamicist systems is a feature which contrasts the dynamicist approach with that of the connectionists. By noting that certain dynamical systems can capture very complex behavior with low dimensional descriptions, dynamicists have insisted that complex cognitive behavior should be modeled via this property. Thus, dynamicists avoid the difficult analyses of high dimensional systems, necessary for understanding connectionist systems. However, it also makes the choice of equations and variables very difficult, because the most relevant, informative variables need to be chosen. From a Machine Learning perspective, this problem is known as *feature selection*, where a small number of relatively predictive features is prefered over a very large number of features that, taken in the proper but untractably complex combination, are entirely predictive of the class label. Irrelevant and redundant features cause problems by adding noise to the learning algorithm and therefore obscuring the distributions of the small set of truly relevant features for the task at hand. Two purposes are served by reducing the set of features considered by an algorithm: first, from a purely computational viewpoint, we can considerably decrease the running time of the induction algorithm; second, and more importantly, the accuracy of the model is increased ([126],[111]).

Parameter Estimation

By adopting a purely dynamicist approach and thus necessitating the use of collective parameters, it becomes impossible to identify the underlying mechanisms that affect behavior. In contrast, connectionism provides a reasonably simple unit (the neuron or node) to which behavior can ultimately be referred. Similarly, symbolicism provides fundamental symbols to which we can appeal. In both of these instances, understanding global behavior is achieved through small steps, modeling progressively more complex behavior and allowing a "backtrace" when necessary to explain a behavior. With dynamical equations, on the other hand, no such progression can be made. The model is general to such an extent as to lose its ability to explain from where the behaviors it is producing are coming, because of its intrinsic lack of representation. The framework proposed in this thesis, via dynamic graphical models, offers a compact way of encoding conditional dependencies between the variables via the graph structure in as efficient a manner as possible.

Coupling

The linked, or coupled, nature of a system of equations implies that changes to one component (most often reflected by changes in a system variable) have an immediate effect on other parts of the system. Thus, there is no representation passing between components of such a system, rather the system is linked via the inclusion of the same parameter in multiple equations. The ability of such systems of equations to model "cognitive" behaviors has prompted theorists, like van Gelder, to insist that the systems being modeled similarly have no need of representation ([247], [248]). In a way, "coupling" thus replaces the idea of "representation passing" for dynamicists. A central contribution of this thesis is the explicit incorporation of context in the dynamic graphical models through causal coupling and interactions between several generative processes. Therefore, there is a clear representation of what coupling means.

Embeddedness

Dynamicist systems also have a special relation with their environment in that they are not easily distinguishable from their surroundings: 'In this vision, the cognitive system is not just the encapsulated brain; rather, since the nervous system, body, and environment are all constantly changing and simultaneously influencing each other, the true cognitive system is a single unified system embracing all three' ([248], p. 373). Since the environment is also a dynamical system, and since it is affecting the cognitive system and the cognitive system is affecting it, the environment and cognitive system are strongly coupled. Such embeddedness of the cognitive system makes a precise distinction between the system and the system's environment very difficult – in other words, the system boundaries are obscure. But this fact, dynamicists claim, is not only a good reflection of how things really are, it is a unique strength of the dynamicist approach ([248], p. 25). Coupling amongst not only the equations describing a cognizing system, but also between those describing the environment and those describing the system results in complex "total system" behaviors.

System Boundaries

An important distinction between dynamicism and either symbolicism or connectionism is the dynamicists' unique view of representation; to be a truly dynamicist model, there should

be no representation. In contrast, symbolicist models are fundamentally dependent on symbolic representations, so clearly they are inadequate. Similarly, connectionists represent concepts (via either distributed representation or local symbolic representation) in their simplified networks. But dynamicists decry the use of representation in cognitive models ([86], [242],[247]).

In the late 1950s there was extensive debate over the behaviorist contention that representation had no place in understanding cognition. One of the best known refutations of this position was given by Chomsky in his 1959 review of B. F. Skinner's book Verbal Behavior. Subsequently, behaviorism fell out of favor as it was further shown that the behaviorist approach was inadequate for explicating even basic animal learning ([241], p. 231). The reasons for the behaviorist failure was its fundamental rejection of representation in natural cognitive agents.

Thus, it is not easy to convincingly deny that representation plays an important role in cognition. It seems obvious that humans use representation in their dealings with the world around them. For example, people seem to have the ability to rotate and examine objects in their head. It seems they are manipulating a representation ([128], [129]). More striking perhaps is the abundant use of auditory and visual symbols by humans everyday to communicate with one another. Exactly where these ever-present communicative representations arise in the dynamicist approach is uncertain. It will evidently be a significant challenge, if not an impossibility, for dynamicists to give a full account of human cognition, without naturally accounting for the representational aspects of thinking. Though dynamicists can remind us of the impressive behaviors exhibited by Brooks' ([36]) dynamical robots, it is improbable that the insect-like reactions of these sorts of systems will scale to the complex interactions of mammalian cognition.

Time

Dynamical systems theory was designed to describe continuous temporal behaviors, thus the dynamicist commitment to this theory provides for a natural account for behavioral continuity. Though the question of whether or not all intelligent behavior is continuous or discrete is a matter of great debate among psychologists ([152], [155]), dynamical systems models possess the ability to describe both. So, relying on the assumption that behavior is 'pervaded by both continuities and discrete transitions' ([247], p. 14) as seems reasonable

[46], [61] dynamicism is in a very strong position to provide good cognitive models based on its theoretical commitments.

One of the greatest strengths of the mathematics of dynamical systems theory is its inherent ability to effectively model complex temporal behavior. It is a unanimous judgement among the paradigms that the temporal features of natural cognitive agents must be adequately accounted for in a good cognitive model ([166], [46], [247]). Not only do dynamicists address the temporal aspect of cognition, they make this aspect the most important. The reasons for espousing this theoretical commitment are obvious: we humans exist in time; we act in time; and we cognize in time – real time. Therefore, dynamical systems theory, which has been applied successfully in other fields to predict complex temporal behaviors, should be applied to the complex temporal behavior of cognitive agents. Whether or not we choose to subscribe to the dynamicist commitment to a particular type of dynamical model, they convincingly argue that we cannot remove temporal considerations from our models of cognition – natural cognition is indeed inherently temporal in nature.

Fundamentally, dynamicists believe that the other approaches to cognition 'leave time out of the picture' ([247], p. 2). They view the brain as continually changing as it intersects with information from its environment. There are no representations, rather there are 'state-space evolutions in certain kinds of non-computational dynamical systems' ([247], p. 1). The temporal nature of cognition does not rely on "clock ticks" or on the completion of a particular task, rather it is captured by a continual evolution of interacting system parts which are always reacting to, and interacting with the environment and each other. These temporal properties can be captured with relatively simple sets of differential equations.

At the highest level, there are a number of general characteristics of a broadly dynamical perspective on some natural phenomenon. The following stand out particularly strongly when the subject is cognition and the contrast is with a computational approach, such as the graphical models presented in this thesis:

1. Change versus state. Change and state are like two sides of one coin. Nevertheless, theoretical perspectives can differ in their primary emphasis or focus. Dynamicists are interested, in the first instance, in how things change; states are the medium of change, and have little intrinsic interest. Computationalists, by contrast, focus primarily on states; change is just what takes you from one state to another. Dynamic graphical models focus both on the states of the variable and on their changes over time, i.e.,

the dynamics of the system, via the transition matrices.

2. Geometry versus structure. How are states of a system conceptualised? Computationalists focus on internal structure, and in particular on internal combinatorial or syntactic structure - how basic pieces are combined to form structured wholes. Dynamicists, by contrast, understand a state geometrically, in terms of its position with respect to other states and features of the system's dynamical landscape such as basins of attraction. In other words, they focus on where the state is, rather than what it is made up of.

3. Structure in time. Sophisticated cognition demands structural complexity in the cognitive system. How is that structure implemented? Computationalists tend to think of it as a static structure that it is all present at one time - and of cognition as simple transformations of static structures. Dynamical Systems Theory suggests an alternative. Systems with simple states - perhaps just one variable - can behave in very complex ways. This enables dynamicists to think of cognitive structure as laid out temporally, much like speech as opposed to the written word. Cognition is then seen as the simultaneous, mutually influencing unfolding of complex temporal structures.

4. Timing versus order. Dynamicists tend to be interested in how behaviors happen in time, whereas computationalists are interested in what the behavior is, regardless of timing details. Computationalists focus on which states the system passes through, whereas dynamicists focus relatively more on when it passes through them.

5. Parallel versus serial. Dynamicists tend to think of systems as operating in parallel, i.e., all aspects changing interdependently at the same time. Computationalists, by contrast, tend to think of systems as serial: most variables remain unchanged in any given state transition. For a dynamicist, change is standardly global; for a computationalist, change is standardly local. Dynamic graphical models exploit the modularity of the graph structure to perform local calculations that ensure a globally consistent representation of the overall probability distribution. Because of their modularity, the inference and MAP estimation algorithms are parallelizable.

6. Ongoing versus Input/Output. Computationalists standardly think of a process as starting with an input to the system. The task for the systems to produce an ap-

propriate output, and it does so via a sequence of internal operations culminating in the system halting with that output. Dynamicists, by contrast, think of processes as always ongoing, not starting anywhere and not finishing anywhere. The goal is not to map an input at one time onto an output some later time, but to constantly maintain appropriate change.

7. Interaction: state-setting or coupling? How does a cognitive system interact with other things, such as the environment? Computationalists standardly think of interaction as setting state; the system changes in its own way from that state, until new input resets state again. Dynamicists recognize an alternative: interaction can be a matter of parameters influencing the shape of change. Input is conceived as an ongoing influence on their direction of change, and output as ongoing influence on something else, just as radio set is continuously modified by an incoming signal and at the same times delivering its sound. Sometimes interaction is a matter of coupling two systems simultaneously shaping each other's change. The coupled HMMs (CHMMs) architecture proposed in this thesis is intented to capture causal interactions between two generating processes.

8. Representations. Computationalists take representations to be static configurations of symbol tokens. Dynamicists conceive representations very differently. They build their representations using the basic entities of Dynamical Systems Theory, such as parameter settings, system states, attractors, trajectories, or bifurcation structures (e.g., [185]). Currently, most dynamicists make use of only the simplest models possible, mostly due to computational limitations. As dynamical modeling increases in mathematical sophistication, we can expect representations to take even more exotic forms.

   Unlike digital computers, dynamical systems are not inherently representational. A small but influential contingent of dynamicists have found the notion of representation to be dispensable or even a hindrance for their particular purposes. Dynamics forms a powerful framework for developing models of cognition which side-step representation altogether. The assumption that cognition must involve representations is based in part on inability to imagine how any non-representational system could possibly exhibit cognitive performances. Within the dynamical approach, such systems can

be not only imagined, they can be modeled and constructed (see, e.g., [20], [19], [71], [263]).

Some objections

As any other theory of human cognition and behavior, the dynamical systems approach suffers from some important objections. Among them, we find:

1. **Structure objection:** One of the most important objections to the dynamical systems approach to cognition is known as the structure objection: 'Sophisticated cognitive performances require complex internal structures. The dynamical approach is taking a huge step backwards in trying to replace symbolic representations with quantities. To explain high level cognition, dynamical systems will have to implement computational mechanisms'.

   Almost everyone now agrees that most kinds of cognitive performance can only be explained by reference to complex structures internal to the system responsible for those performances. Still, it remains an open question what form those structures might take. Symbolic cognitive models advocate that they are the kind of structures found in digital computers, i.e., symbol structures ([165]) or "classical" combinatorial representations ([69]). Lying behind this idea is an assumption that the kinds of complex structures required cannot exist in any system except by instantiating digital symbol structures.

   However, as dynamical cognitive science has matured, it has become apparent that dynamical systems can incorporate combinatorial structures in various ways without merely implementing their digital counterparts ([249]). For example, arbitrarily many structures can be mapped onto states of a dynamical system, such that these states can then be used as the basis of systematic processing (e.g., ([188])). Other work has found combinatorial structure in the attractor basins of appropriate dynamical systems ([168]), or in the trajectories induced by sequences of bifurcations ("attractor chaining", [247]). The possibilities have really only begun to be explored. The dynamical approach is not vainly attempting to do without complex internal structures. Rather, it is in the process of dramatically reconceiving how they might be instantiated.

2. **"The Not Cybernetics Again!" Objection:** The dynamical approach has been blamed for being just a new face of cybernetics. Cybernetics was famously defined by Wiener, one of its creators, as 'the science of communication and control in man and machine' but it soon developed into an even wider enterprise: a kind of general, non-reductionistic study of systems, particularly self-sustaining systems in their environments see [178]. Throughout its brief ascendancy, cybernetics enthusiastically embraced anything of conceivable relevance to complex systems, including information theory, communication theory, automata theory, neurophysiology, systems theory, game theory and control theory.

   Dynamics was certainly a part of cybernetics, and the Dynamical Hypothesis is sometimes traced back to a leading cyberneticist, H. Ross Ashby. Still, the original cybernetics proposal implied little about the contemporary dynamical approach. Therefore they differ in fundamental ways. The Dynamical Hypothesis is, by comparison, tightly defined. It is concerned with cognition specifically, rather than systems generally, and is defined in terms of a core commitment to a single framework. The dynamical approach is not more closely connected to cybernetics than it does to other disciplines with ancestral links to cybernetics, such as computational neuroscience and artificial intelligence. Moreover, much more powerful tools are available today, such that the bulk of Dynamical Systems Theory has been developed in the period since cybernetics.

3. **The "Humans Compute" Objection:** Humans can do arithmetic in their heads. At least some cognitive activity is specifically digital computation. Therefore, the Dynamical Hypothesis cannot be the whole truth about cognition. If it is granted that mental arithmetic and similar processes are, literally, digital symbol manipulation inside the head, then the Dynamical Hypothesis should indeed graciously concede. The general truth of the Dynamical Hypothesis is compatible with certain special activities counting as exceptions. However, we are also implicitly assuming that mental arithmetic consists of symbol manipulation. Certainly, it seems like symbol manipulation: numerals, lines, etc. are "seen in the mind's eye". It does not follow that there are symbols in the head, i.e., that the states and processes that subserve such seeing actually instantiate symbols and their manipulations. Imagining the Eiffel Tower does not entail that one has the Eiffel Tower, or even a picture of it, inside one's head

([213], Ch.8). We must not confuse the content of experience with the mechanisms implementing it. As usual, the question turns out to be the empirical one: in the long run, what kind of models provide the best account ofhe mechanisms underlying the relevant kind of cognitive performance?

The intuitive appeal of a dynamical systems theory description of many systems' behaviors is quite difficult to resist. It seems to make sense to think of the behavior of cognitive systems in terms of an "attraction" to a certain state (e.g. some people seem to be disposed to being happy). However, can such metaphorical descriptions of complex systems actually provide us with new insights, integrate previously unrelated facts, or in some other way lead to a deeper understanding of these systems?

In science it is necessary to provide a model. By a model, I mean a precise description of the properties of the system being modeled. The more important properties of the source that are exactly demonstrated by the model, the better the model. differentiate between model and analogy in science,

There is no real explanation provided by the psychological applications of dynamical systems theory to the phenomenology or intentionality of cognition. These supposed models do not provide new insights in clinical experimental psychology. They do not reveal any details about what is being described (i.e. cognition). There are no consistent and explicit mappings between dynamical systems theory and human behavior.

The human behavior models proposed in this thesis are statistical, generative and learned from data. It would claim that it is possible to learn generative models (parameters of) from data and to generate a model which produces data that seems appropriate. From a strictly cognitive viewpoint, and since we have no explicit map between the concepts of clinical psychology and those of dynamical systems theory, the data is meaningful only in its mathematical context, not in a cognitive one.

Even in the most rigorous of dynamical models, and despite the application of nonlinear differential equations in their model, there is very little empirical evidence, if any, about how the model relates to cognition.

In sum, the concepts of dynamical systems theory provide an interesting method of thinking about cognitive systems, but they have not yet been shown to be successfully transferable to rigorous definitions of human behavior or cognition. The fuzziness of clinical psychology does not allow for quantification of mechanisms in dynamical systems theory

terms. Furthermore, even some physiological processes do not seem to lend themselves to precise quantitative dynamicist descriptions that are able to provide the predictive or explanative powers expected of good models (c.f. [246]).

Critics may claim that a dynamical systems approach to cognition is simply not new – as early as 1970, Simon and Newell were discussing the dynamical aspects of cognition ([165]). In 1991, Giunti showed that the symbolicist Turing Machine is a dynamical system ([247]), so it could be concluded that there is nothing to gain from introducing a separate dynamicist paradigm for studying cognition. However, Turing Machines and connectionist networks have also been shown to be computationally equivalent yet these approaches are vastly disparate in their methods, strengths, and philosophical commitments ([69], p. 10). Similarly, though Turing Machines are dynamical in the strictest mathematical sense, they are nonetheless serial and discrete. Hence, symbolicist models do not behave in the same ideally coupled, dynamical and continuous manner as dynamicist systems are expected to. Dynamicist systems can behave either continuously or discretely, whereas Turing Machines are necessarily discrete. Furthermore, they are not linked in the same way to their environment, and the types of processing and behavior exhibited is qualitatively different. For these reasons, dynamicists believe their approach will give rise to fundamentally superior models of cognition. Biological evidence and the symbolicists' practical difficulties lend support to many of the dynamicists criticisms ([166], [46], [247]).

2.3 Conclusions

In this chapter I have presented Psychological and Philosophical theories of human action, and behavior. First, the frame problem has been introduced as one of the first explanations of the organization of action; then several behavior theories have been described with special emphasis in the way they formalize time and how do they relate to the statistical behavior models proposed in this thesis (see chapter 4).

I have pointed out the major connections between the computational model proposed in this thesis and the behavior theories proposed in Psychology and Philosophy. To summarize them:

1. The Model Human Processor distinguishes three separate components in human cognition: cognitive, motor and perceptual processors. The computational model proposed

in this thesis is a two-layer model (see figure 1-3) that includes perceptual, cognitive and motor modules: cameras and other sensors, together with computer vision and signal processing modules at the perceptual level; active control system for a camera at the control level; and dynamic graphical models or Dynamic Probabilistic Networks (DynPINs) at the cognitive level.

2. The methodology derived from Newell's work has been formalized in three steps: (1) subject's problem space identification and construction; (2) subject's solution path identification by making use of the sequential information in the protocol; (3) subject's strategy hypothesis by inventing problem-solving heuristics that can reproduce the subject's solution path. Similarly, the formulation proposed in this thesis, via dynamic graphical models provides a mathematically sound framework for carrying out the three previous steps: (1) the problem space is determined by the model: the graph structure, the selected features, etc, (2) the solution path is given by well-defined inference and MAP estimation algorithms defined over the graph, (3) and the subject's solution path is reproduced by the models, given that they are generative and learnt from real data.

3. It has been acknowledged that most of the models proposed are very sensitive to noise, missing data and variability among different subjects. These problems are alleviated by the proposed framework, because dynamic graphical models offer a mathematically sound framework for incorporating uncertainty, noise and missing data.

4. Some formulations of functionalism talk about the causal relations among stimulus inputs, internal states, and behavioral outputs. Others merely talk about transitional relations, i.e., one state following another. The dynamic graphical models used in this thesis (HMMs and CHMMs) offer a formal framework for representing both causal relations –via the graph structure– and the transitional relations –via the transition probability matrices between adjacents states– (see chapter 4).

5. There are many key features shared between the dynamicist approach to cognition and the dynamic graphical models framework proposed in this thesis. Among them: both of them focus on the dynamic aspects of systems; they decompose a system in terms of their variables and the states they can be in; they are computational and quantitative in space, time, or both; they deal with dimensionality reduction and parameter

estimation. However, they differ also in some crucial points: the models proposed in this thesis are stochastic, statistical, data-driven models versus the deterministic nature of dynamical systems theory; they have a well defined structure that captures the causal relations of the variables, versus the lack of structure of dynamical systems; the systems dynamics is modeled by transition probabilities and not by differential equations as in the dynamical approach; they treat time discretely whereas dynamical systems theory was designed to describe continuous temporal behaviors.

6. In the dynamicist approach, a model is general to such an extent as to lose its ability to explain from where the behaviors it is producing are coming, because of its intrinsic lack of representation. The framework proposed in this thesis, via dynamic graphical models, offers a compact way of encoding conditional dependencies between the variables via the graph structure in as efficient a manner as possible.

Most, if not all, of the previously described psychological and philosophical models are manually built, relatively abstract, not directly learnt from human behaviors in real situations and not predictive. Finally there is a lack of rigorous mechanisms for evaluating the performance of the models. As a consequence, there are very few systems able to perceive and understand aspects of human behavior. The human behavior modeling framework proposed in this thesis solves most of these problems: the model parameters are automatically learnt from real data, collected in real situations. The models are generative, predictive, and are validated according to their recognition accuracy on test data. With the proposed framework I have built four systems that perceive and understand certain human behaviors. Because of the generality of the proposed model, I would claim that many more systems in other domains could be successfully built using it.

Chapter 3

Perceptual Input

'Concepts without percepts are empty; percepts without concepts are blind'.

Kant

3.1 Introduction and Motivation

In chapter 1 I have presented the concept of *Perceptual Intelligence*, a new discipline which brings together perception and cognition in the same framework. High-level perception –the process of making sense of complex data at an abstract, conceptual level– is fundamental to human cognition. Prior to the twentieth century, theories of knowledge were inherently perceptual. For over 2,000 years, theorists viewed higher cognition as inherently perceptual. Since Aristotle (4th century BC) and Epicurus (4th century BC), theorists saw the representations that underlie cognition as imagistic. For example, two hundred years ago, Kant provocately suggested an intimate connection between perception and concepts. 'Concepts without percepts', he wrote, 'are empty; percepts without concepts are blind'. After being widely accepted for two millennia, this view withered with mentalism in the early twentieth century. At that time, behaviorists and ordinary language philosophers successfully banished mental states from consideration in much of the scientific community, arguing that they were unscientific and led to confused views of human nature ([271]). Because perceptual theories of mind had dominated mentalism to that point, attacks on mentalism often included a critique of perception. As a result, perceptually-based theories of cognition disappeared within the theories of cognition. Moreover, developments in logic, statistics, and programming languages have inspired theories that rest on principles fundamentally

different from those underlying perception. Traditional research in Artificial Intelligence has tried to model concepts while ignoring perception, even though high-level perceptual processes lie at the heart of human cognitive abilites. Conversely, perception diverged from cognition, focusing primarily on bottom-up sensory mechanisms and ignoring top-down effects.

However, cognition cannot succeed without processes that build up appropriate representations [17]. Cognition is inherently perceptual, sharing systems with perception at both the cognitive and the neural levels. Much research in neuroscience has established that categorical knowledge is grounded in sensory-motor regions of the brain (for reviews see [52], [1], in press). Damage to a particular sensory-motor region disrupts the conceptual processing of categories that use this region to perceive physical exemplars. For example, damage to the visual system disrupts the conceptual processing of categories whose exemplars are primarily processed visually, such as birds. These findings strongly suggest that categorical knowledge is not amodal –purely cognitive–. This influence is not unidirectional: cognition does not become more perceptual while perception remains unaffected. Perception is not an entirely modular system with cognition lying outside it. Because perception shares systems with cognition, bottom-up activation of perceptual systems engages cognitive processes immediately. Bottom-up information may dominate conflicting top-down information but fuse with consistent top-down information.

According to [17] there are six core properties in any conceptual system: (1) perceptual symbols are neural representations in sensory-motor areas of the brain; (2) they represent schematic components of perceptual experience, not entire holistic experiences; (3) they are multimodal, arising across the sensory modalities, proprioception, and introspection; (4) related perceptual symbols become integrated into a simulator that produces limitless simulations of a perceptual component (e.g., red, lift, hungry); (5) frames organize the perceptual symbols within a simulator, and (6) words associated with simulators provide linguistic control over the construction of simulations.

Conceptual processes should, thus, be studied in conjunction with the perceptual substrate on which they rest, and with which they are tightly coupled. On the other hand, our perception of any given situation is guided by constant top-down influence from the conceptual level. Without this conceptual influence, the representations that result from such perception will be rigid, inflexible, and unable to adapt to the problems provided by

many different contexts. The flexibility of human perception derives from constant interaction with the conceptual level. I would argue that perceptual processes cannot be separated from other cognitive processes even in principle, and therefore traditional AI models cannot be defended by supposing the existence of a "representation module" that supplies representations ready-made. Recognizing the centrality of perceptual processes makes AI more difficult, but much more interesting. Integrating perceptual processes into a cognitive model leads to flexible representations, and flexible representations lead to flexible actions. This is precisely the goal at the heart of Perceptual Intelligence.

Fortunately, there are today research efforts towards developing perceptual theories of cognition in psychology, philosophy, cognitive sciences, artificial intelligence, and linguistics ([211],[195],[151], [133], [105], [84], [47], [17], [14]). The computational model of human behavior developed in this thesis reflects the connection between perception and cognition, as depicted in figure 1-3. In each system, the cognitive modules (behavior models) are statistically learnt from observed data through the perceptual modules. The behavior modeling framework is presented in chapter 4. In this chapter, I will describe in detail the perceptual aspects of each of the testbeds developed in the thesis. In the context of the proposed model, this chapter deals with the bottom-most layer of the model, as it appears boxed in figure 3-1.

Depending on the domain, different perceptual input modalities have been used: (1) In the case of facial expression recognition, an active camera looking at the user's face; (2) in the framework of pedestrian interactions recognition, a static camera with wide field-of-view watching a dynamic outdoor scene; (3) in the driver domain, multiple sensors of different nature are used: internal sensors of the car's internal state –acceleration, steering wheel angle, gear, speed and break pedal action–, and cameras for the visual context –front and rear traffic, driver's face and gaze, and driver's viewpoint.

3.2 Visual Input and Representation of Visual Data

First, I will proceed to explain the computer vision processing involved in LAFTER and in the pedestrian surveillance system.

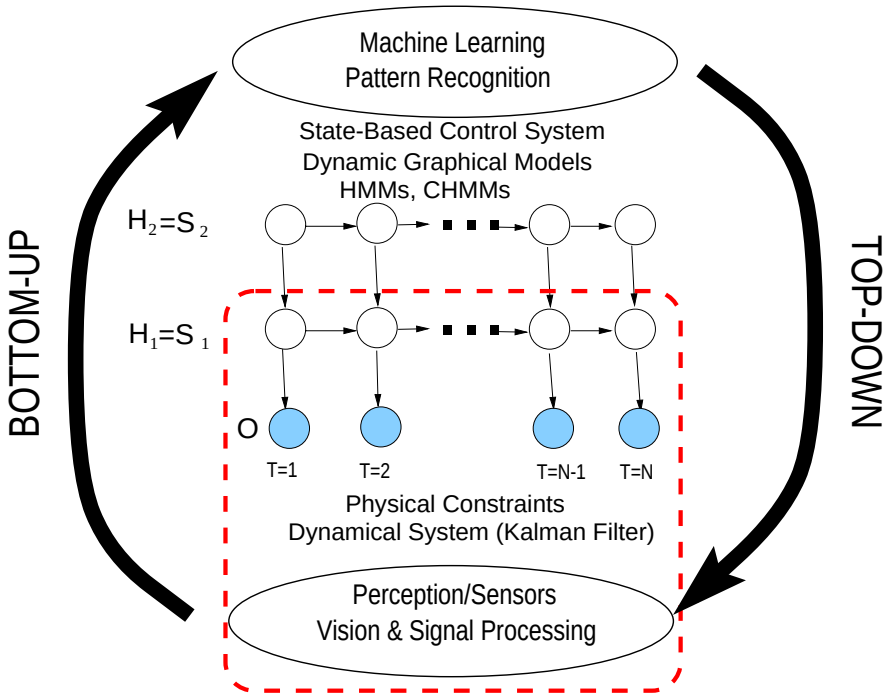


Figure 3-1: The perceptual system occupies the lowest level in the proposed model

Blobs: A Probabilistic Representation

The notion of "blobs" as a representation for image features has a long history in computer vision [181, 122, 24, 272], and has had many different mathematical definitions. In the usage of this thesis work it is a compact set of pixels that share a visual property that is not shared by the surrounding pixels. This property could be color, texture, brightness, motion, shading, a combination of these, or any other salient spatio-temporal property derived from the signal (the image sequence). In this thesis blobs are a coarse, locally-adaptive encoding of the images' spatial and color/texture/motion/etc. properties. A prime motivation for the interest in blob representations is the discovery that they can be reliably detected and tracked even in complex, dynamic scenes, and that they can be extracted in real-time without the need for special purpose hardware. These properties are particularly important in applications that require tracking people, and recently they have been used in 2-D blob tracking for real-time whole-body human interfaces [272] and real-time recognition of American Sign Language hand gestures [232].

One can represent shapes in both 2-D and 3-D by their low-order statistics. Clusters of 2-D points have 2-D spatial means and covariance matrices, which will be denoted by $\bar{q}$ and C_q. The blob spatial statistics are described in terms of their second-order properties.

For computational convenience it will be interpreted as a Gaussian model. The Gaussian interpretation is not terribly significant, because I also keep a pixel-by-pixel *support map* showing the actual occupancy.

Like other representations used in computer vision and signal analysis, including superquadrics, modal analysis, and eigen-representations, blobs represent the global aspects of the shape and can be augmented with higher-order statistics to attain more detail if the data supports it. The reduction of degrees of freedom from individual pixels to blob parameters is a form of regularization which allows the ill-conditioned problem to be solved in a principled and stable way.

For both 2-D and 3-D blobs, there is a useful physical interpretation of the blob parameters in the image space. The mean represents the geometric center of the blob area (2-D) or volume (3-D). The covariance, being symmetric semi-definite positive, can be diagonalized via an eigenvalue decomposition: $C = \Phi L \Phi^T$,where Φ is orthonormal and L is diagonal.

The diagonal L matrix represents the size of the blob along independent orthogonal object-centered axes and Φ is a rotation matrix that brings this object-centered basis in alignment with the coordinate basis of C. This decomposition and physical interpretation is important for estimation, because the shape L can vary at a different rate than the rotation Φ. The parameters must be separated so they can be treated appropriately.

Face Detection in LAFTER by Maximum Likelihood Estimation

The blob features are modeled as a mixture of Gaussian distributions in the color (or texture, motion, etc.) space. The algorithm that is generally employed for learning the parameters of such a mixture model is the *Expectation-Maximization (EM)* algorithm of Dempster *et al* [58], [200]. I refer the reader to section 4.7.3 in chaper 4 for a detailed description of the EM algorithm.

In the LAFTER system the input data vector d is the normalized R,G,B content of the pixels in the image, $\mathbf{x} = (\tilde{r}, \tilde{g}) = (\frac{r}{r+g+b}, \frac{g}{r+g+b})$. Work by Wren *et al* [272], or that of Schiele *et al* or Hunke *et al* [217, 98] have shown that use of normalized or chromatic color information $(\tilde{r}, \tilde{g}) = (\frac{r}{r+g+b}, \frac{g}{r+g+b})$ can be reliably used for finding flesh areas present in the scene despite wide variations in lighting. The color distribution of each of the blobs is modeled as a mixture of Gaussian probability distribution functions (*pdf's*) that are iteratively estimated using EM. One can perform a maximum likelihood decision criterium after the

clustering is done because human skin forms a compact, low dimensional (approximately 1D) manifold in color space. Two different clustering techniques, both derived from EM are employed: an off-line training process and an on-line adaptive learning process.

In order to determine the mixture parameters of each of the blobs, the unsupervised EM clustering algorithm is computed off-line on hundreds of samples of the different classes to be modeled (in our case, face, lips and interior of the mouth), in a similar way as it is done for skin color modeling in [106]. When a new frame is available the likelihood of each pixel is computed using the learned mixture model and compared to a likelihood threshold. Only those pixels whose likelihood is above the threshold are classified as belonging to the model. Figure 3-2 illustrates LAFTER's face detection and segmentation processes.

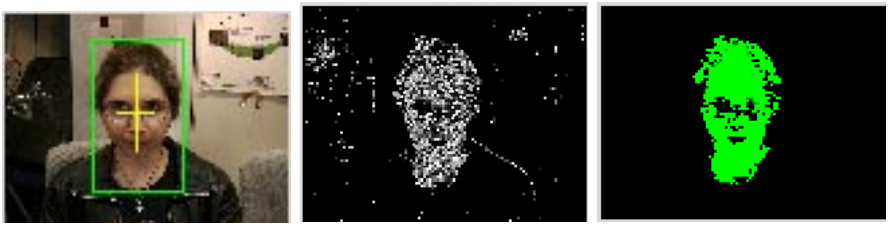


Figure 3-2: Face detection, per-pixel probability image computation and face blob growing

Adaptive Modeling via EM

Even though general models make the system relatively user-independent, they are not as good as an adaptive, user-specific model would be. I therefore use adaptive statistical modeling of the blob features to narrow the general model, so that its parameters are closer to the specific users' characteristics.

The first element of this adaptive modeling is to update the model priors as soon as the user's face has been detected. Given n independent observations $x_i = (\tilde{r}_i, \tilde{g}_i)$, $i = 1 \ldots n$ of the user's face, they are modeled as being samples of a Normal distribution in color space with mean the sample mean μ_{user} and covariance matrix, Σ_{user} The skin color prior distribution is also assumed to be Normal $p(x|\mu_{general}, \Sigma_{general}) = N(\mu_{general}, \Sigma_{general})$ whose parameters have been computed from hundreds of samples of different users. By applying Bayesian integration of the prior and user's distributions a Normal posterior distribution

$N(\mu_{post}, \Sigma_{post})$ is obtained, whose sufficient statistics are given by:

$$\begin{aligned} \Sigma_{post} &= [\Sigma^{-1}_{general} + \Sigma^{-1}_{user}]^{-1} \\ \mu_{post} &= \Sigma_{post}[\Sigma^{-1}_{general} * \mu_{general} + \Sigma^{-1}_{user} * \mu_{user}] \end{aligned} \tag{3.1}$$

Equation 3.1 corresponds to the computation of the posterior skin color probability distribution from the prior (general) and the user's (learned from the current image samples) models.

This update of skin model occurs only at the beginning of the sequence, assuming that the blob features are not going to drastically change during run time. To obtain a fully adaptive system, however, one must also be able to handle second-to-second changes in illumination and user characteristics.

Therefore an *on-line* Expectation-Maximization algorithm [194, 243] is utilized to adaptively model the image characteristics. Both the background and the face are modeled as a mixture of Gaussian distributions with mixing proportions π_i and K components:

$$p(x/\Theta) = \sum_{i}^{K} \pi_i \frac{e^{-1/2(x-\mu_i)^T \Sigma_i^{(-1)}(x-\mu_i)}}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \tag{3.2}$$

The unknown parameters of such a model are the sufficient statistics of each Normal distribution (μ_i, Σ_i), the mixing proportions π_i and the number of components of the mixture K.

The incremental EM algorithm is data-driven, i.e., it estimates the distribution from the data itself. Two update algorithms are needed for this purpose: A criterium for adding new components to the current distribution as well as an algorithm for computing the sufficient statistics of each Normal Gaussian component.

The sufficient statistics are updated by computing an on-line version of the traditional EM update rules. If the first n data points have already been computed, the parameters when data point $(n+1)$[1] is read are estimated as follows: First, the posterior class probability $p(i|x^{n+1})$ or *responsibility (credit)* h_i^{n+1} for a new data point x^{n+1} is computed:

$$h_i^{n+1} = \frac{\pi_i^n p(x^{n+1}/\theta_i^n)}{\sum_j \pi_j^n p(x^{n+1}/\theta_j^n)} \tag{3.3}$$

[1]Superscript n will refer in the following to the estimated parameters when n data points have already been processed

This responsibility can be interpreted as the probability that a new data point x^{n+1} was generated by component i. Once this responsibility is known, the sufficient statistics of the mixture components are updated, weighted by the responsibilities:

$$\pi_i^{n+1} = \pi_i^n + \frac{h_i^{n+1} - \pi_i^n}{n} \tag{3.4}$$

$$\mu_i^{n+1} = \mu_i^n + \frac{h_i^{n+1}}{n * w_i^n}(x^{n+1} - \mu_i^n) \tag{3.5}$$

$$\sigma_i^{2(n+1)} = \sigma_i^{2(n)} + \frac{h_i^{n+1}}{n * w_i^n}((x^{n+1} - \mu_i^n)^2 - \sigma_i^{2(n)}) \tag{3.6}$$

where σ_i is the standard deviation of component i and w_i^{n+1} is the *average* responsibility of component i per point: $w_i^{n+1} = w_i^n + \frac{h_i^n - w_i^n}{n}$. The main idea behind this update rules is to distribute the effect of each new observation to all the terms in proportion to their respective likelihoods.

A new component is added to the current mixture model if the most recent observation is not sufficiently well explained by the model. In particular, if the last observed data point has a very low likelihood with respect of each of the components of the mixture, i.e. if it is an outlier for all the components, then a new component is added with mean the new data point and weight and covariance matrix specified by the user. The threshold in the likelihood can be fixed or stochastically chosen. In the latter case the algorithm would randomly choose whether to add a component or not given an outlier. There is a maximum number of components for a given mixture as well.

The foreground models are initialized with the off-line unsupervised learned *a priori* mixture distributions described above. In this way, the algorithm quickly converges to a mixture model that can be directly related to the *a priori* models' classes. The background models are not initialized with an a *priori* distribution but learned on-line from the image.

MAP segmentation

Given these models, a MAP foreground-background decision rule is applied to compute *support maps* for each of the classes, that is, pixel-by-pixel maps showing the class membership of each model. Given several statistical blob models that could potentially describe some particular image data, the membership decision is made by searching for the model with the Maximum A Posteriori (MAP) probability.

Once the class memberships have been determined, the statistics of each class are then updated via the EM algorithm, as described above. This approach can easily be seen to be a special case of the MDL segmentation algorithms developed by Darrell and Pentland [55, 54] and later by Ayer and Sawhney [11].

Visual Surveillance System: Figure Segmentation by Eigenbackground Subtraction

The first step in the visual surveillance system is to reliably and robustly detect and track the pedestrians in the scene. 2-D *blob features* are used for modeling each pedestrian.

In the system the main cue for clustering the pixels into blobs is motion, because there is a static background with moving objects. To detect these moving objects an eigenspace that models the background is adaptively built. This eigenspace model describes the range of appearances (e.g., lighting variations over the day, weather variations, etc.) that have been observed. The eigenspace could also be generated from a site model using standard computer graphics techniques.

The eigenspace model is formed by taking a sample of N images and computing both the mean μ_b background image and its covariance matrix C_b. This covariance matrix can be diagonalized via an eigenvalue decomposition $L_b = \Phi_b C_b \Phi_b^T$, where Φ_b is the eigenvector matrix of the covariance of the data and L_b is the corresponding diagonal matrix of its eigenvalues. In order to reduce the dimensionality of the space, in principal component analysis (PCA) only M eigenvectors (eigenbackgrounds) are kept, corresponding to the M largest eigenvalues to give a Φ_M matrix. A principal component feature vector $I_i - \Phi_{M_b}^T X_i$ is then formed, where $X_i = I_i - \mu_b$ is the mean normalized image vector.

Note that moving objects, because they don't appear in the same location in the N sample images and they are typically small, do not have a significant contribution to this model. Consequently the portions of an image containing a moving object cannot be well described by this eigenspace model (except in very unusual cases), whereas the static portions of the image can be accurately described as a sum of the the various eigenbasis vectors. That is, the eigenspace provides a robust model of the probability distribution function of the background, but not of the moving objects.

Once the eigenbackground images (stored in a matrix called Φ_{M_b} hereafter) are obtained, as well as their mean μ_b, each input image I_i can be projected onto the space expanded by

Figure 3-3: Background mean image, blob segmentation image and input image with blob bounding boxes

the eigenbackground images $B_i = \Phi_{M_b} X_i$ to model the static parts of the scene, pertaining to the background. Therefore, by computing and thresholding the Euclidean distance (distance from feature space DFFS [154]) between the input image and the projected image we can detect the moving objects present in the scene: $D_i = |I_i - B_i| > t$, where t is a given threshold. In the following, I will refer to D_i as a *motion mask*. Figure 3-3 depicts typical examples of the background mean image, the blob (pedestrian) segmentation image and the input image with bounding boxes around each of the pedestrians. Note that it is easy to *adaptively* perform the eigenbackground learning, in order to compensate for changes such as big shadows. The motion mask D_i is the input to a clustering connected component algorithm that produces blob descriptions that characterize each person's shape. I have also experimented with modeling the background by using a mixture of Gaussian distributions at each pixel, as in Pfinder [273]. However I finally opted for the eigenbackground method because it offered good results and less computational load.

Tracking by Kalman Filtering

Kalman filters have extensively been used in control theory as stochastic linear estimators. The Kalman filter was first introduced by R. Kalman [118] for discrete systems and by Kalman and Bucy [117] for continuous-time systems. The objective is to design an estimator that provides estimates of the non-observable estate of a system taking into account the known dynamics and the measured data. Recall that the Kalman Filter is the "best linear unbiased estimator" in a mean squared sense and that for Gaussian processes, the Kalman filter equations corresponds to the optimal Bayes' estimate (for a more detailed description, see section 4.4.1 in chapter 4).

In the LAFTER system to ensure stability of the MAP segmentation process, the spatial parameters for each blob model are filtered using a zero-order Kalman filter. For each blob two independent, zero-order filters are maintained, one for the position of the blob centroid

and another for the dimensions of the blob's bounding box. The MAP segmentation loop now becomes:

1. For each blob predict the filter state vector, $X^* = \hat{X}$ and covariance matrix, $C^* = \hat{C} + (\Delta t)^2 W$, where the matrix W measures the precision tolerance in the estimation of the vector X and depends on the kinematics of the underlying process.

2. For each blob new observations Y (e.g., new estimates of blob centroid and bounding box computed from the image data) are acquired and the Mahalanobis distance between these observations (Y,C) and the predicted state $(\hat{X}, \hat{C})$ is computed. If this distance is below threshold, the filters are updated by taking into account the new observations:

$$\hat{C} = \left[C^{*^{-1}} + C^{-1} \right]^{-1} \tag{3.7}$$

$$\hat{X} = \hat{C} \left[C^{*^{-1}} X^* + C^{-1} Y \right]^{-1} \tag{3.8}$$

   Otherwise a discontinuity is assumed and the filters are reinitialized: $\hat{X} = X^*$ and $\hat{C} = C^*$.

A generalized version of this technique is employed in [51] for fusing several concurrent observations. This Kalman filtering process is used in the tracking of all of the blob features. In my experience the stability of the MAP segmentation process is substantially improved by use of the Kalman filter, specially given that LAFTER's real-time performance yields small errors in the predicted filter state vectors. Moreover, smooth estimates of the relevant parameters are crucial for preventing jittering in the active camera, as described in section 3.2.

In the visual surveillance application, the trajectories of each blob are computed and saved into a *dynamic track memory*. Each trajectory has associated a first order Kalman filter that predicts the blob's position and velocity in the next frame.

In order to handle occlusions as well as to solve the correspondence between blobs over time, the appearance of each blob is also modeled by a Gaussian *pdf* in RGB color space. When a new blob appears in the scene, a new trajectory is associated to it. Thus for each blob the Kalman-filter-generated spatial *pdf* and the Gaussian color *pdf* are combined to form a joint (x, y) image space and color space *pdf*. In subsequent frames the Mahalanobis distance is used to determine the blob that is most likely to have the same identity.

Active Camera Control in LAFTER

Because LAFTER already maintains a Kalman filter estimate of the centroid and bounding box of each blob, it is a relatively simple matter to use these estimates to control an active camera so that the face of the user always appears in the center of the image and with the desired size. LAFTER uses an abstraction of the camera control parameters, so that different camera/motor systems (currently the Canon VCC1 and Sony EVI-D30) can be successfully used in a transparent way. In order to increase tracking performance, the camera pan-tilt-zoom control is done by an independent light-weight process (thread) which is started by the main program. The Sony EVI-D30 camera is capable of doing panoramic (left-right) and tilt (up-down) rotations about two orthogonal axes, lens zooming within the range of 5.4 $\sim$ 64.8 mm, and auto focus. The rotation ranges are $\pm 100^o$ (pan) and $\pm$ 25^o (tilt). The ranges of the angles of view of the lens are roughly 3.3^o $\sim$ 36.6^o (vertical) and 4.4 $\sim$ 48.8^o (horizontal).

The current estimation of the position and size of the user's face provides a reference signal to a PD controller which determines the tilt, pan and zoom of the camera so that the target (face) has the desired size and is at the desired location. The zoom control is relatively simple, because it just has to be increased or decreased until the face reaches the desired size. Pan and tilt speeds are controlled by $S_c = \frac{C_e * E + C_d * \frac{dE}{dt}}{F_z}$, where C_e and C_d are constants, E is the error, i.e. the distance between the face current position and the center of the image, F_z is the zoom factor, and S_c is the final speed transmitted to the camera.

The zoom factor plays a fundamental role in the camera control because the speed with which the camera needs to be adjusted depends on the displacement that a fixed point in the image undergoes for a given rotation angle, which is directly related to the current zoom factor. The relation between this zoom factor and the current camera zoom position follows a non-linear law which needs to be approximated. In our case, a second order polynomial provides a good approximation. Figure 3-4 illustrates the processing flow of the PD controller.

Speed, Accuracy, and Robustness

Running LAFTER on a single SGI Indy with a 200Mhz R4400 processor, the average frame rate for tracking is typically 25 Hz. When mouth detection and parameter extraction are

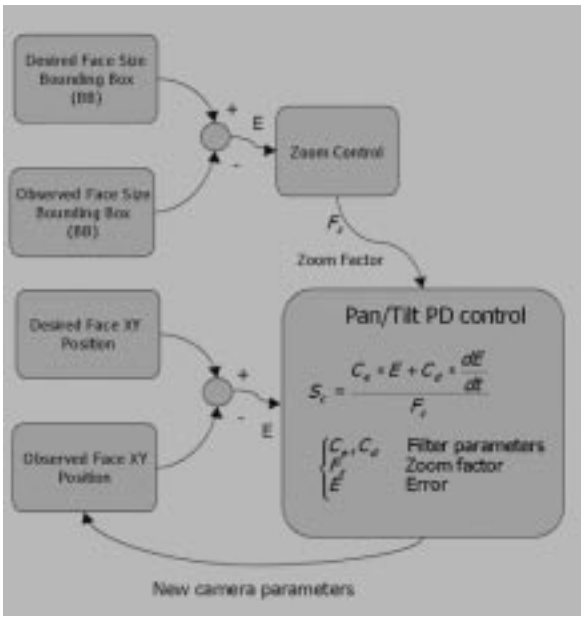


Figure 3-4: PD Controller

added to the face tracking, the average frame rate is 14 Hz.

To measure LAFTER's 3D accuracy during head motion, the RMS error was measured by having users make large cyclic motions along the X, Y, and Z axes respectively, with the true 3D position of the face being determined by manual triangulation. In this experiment the camera actively tracked the face position, with the image-processing/camera-control loop running at a nearly constant 18hz. The image size was 1/6 full resolution, i.e. 106x80 pixels, and the camera control law varied pan, tilt, and zoom to place the face in the center of the image at a fixed pixel resolution. Figure 3-5 illustrates the active-camera tracking system in action. The RMS error between the true 3D location and the system's output was computed in pixels and is shown in table 3.1. Also shown is the variation in apparent head size, e.g., the system's error at stabilizing the face image size. As can be seen, the system gave quite accurate estimates of 3D position. Perhaps most important, however, is the robustness of the system. LAFTER has been tested on hundreds of users at many different events, each with its own lighting and environmental conditions. Examples are the *Digital Bayou*, part of SIGGRAPH '96, the *Second International Face & Gesture Workshop (October 96)* or several open houses at the Media Laboratory during the years 1996 to 1998. In all cases the system failed in approximately $5-7\%$ of the cases, when the users had dense beard, extreme skin color or clothing very similar to the skin color models.

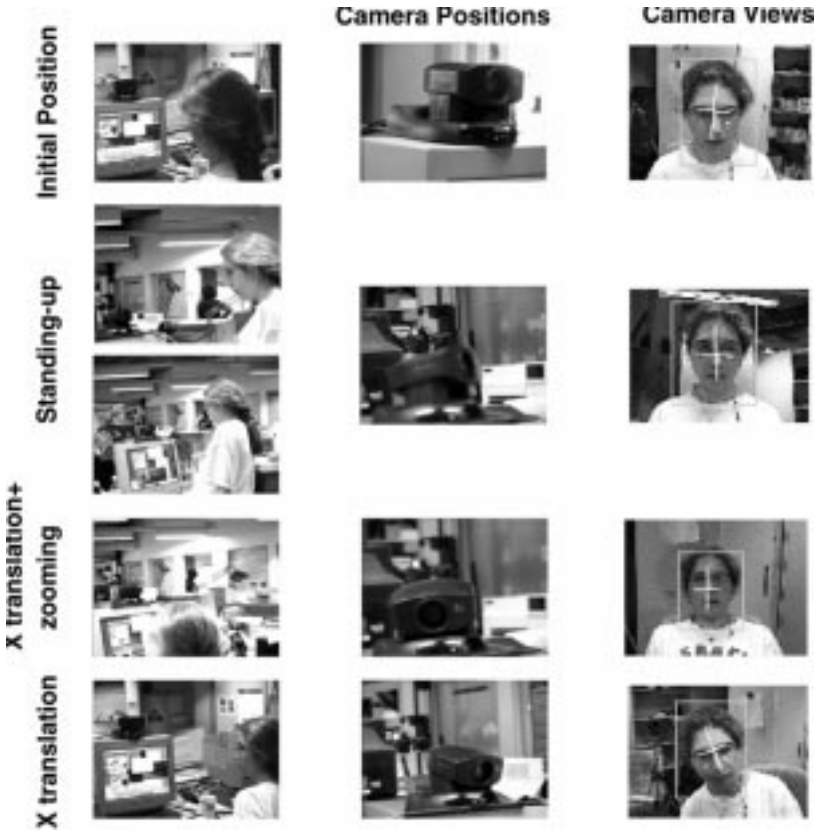


Figure 3-5: Active camera tracking

Multi-resolution Processing in LAFTER: Mouth Extraction and Tracking

Once the face location and shape parameters are known (center of the face, width, height and image rotation angle), anthropometric statistics are used to define a bounding box within which the mouth must be located.

The mouth is modeled using the same principles as the face, i.e. through a second-order mixture model that describes both its chromatic color and spatial distribution. However to obtain good performance a more finely detailed model of the face region surrounding the mouth is needed. The face model that is adequate for detection and tracking might not be adequate for accurate mouth shape extraction.

The system, therefore, acquires image patches from around the located mouth [2] and builds a Gaussian mixture model. In the current implementation, skin samples of three different facial regions around the mouth are extracted during the initialization phase and

[2]The mouth extraction and processing is performed on a Region of Interest (ROI) extracted from a full resolution image (i.e. 640x480 pixels) whereas the face detection and processing is done on an image of 1/6 full resolution, i.e. 106x80 pixels

	Translation Range	X RMS Error (pixels)	Y RMS Error (pixels)
Static Face	0.0 cm	0.5247 (0.495 %)	0.5247 (0.6559 %)
X translation	$\pm$76 cm	0.6127 (0.578 %)	0.8397 (1.0496 %)
Y translation	$\pm$28 cm	0.8034 . (1.0042 %)	1.4287 (1.7859 %)
Z translation	$\pm$78 cm	0.6807 (0.6422 %)	1.1623 (1.4529 %)

	Width Std (pixels)	Height Std (pixels)	Size change (pixels)
Zooming	2.2206 (2.09 %)	2.6920 (3.36 %)	Max. size: 86x88 Min. size: 14x20

Table 3.1: Translation and zooming active tracking accuracies.

their statistics are computed, as is depicted in figure 3-6. The second image in the same figure is an example of how the system performs in the case of facial hair. The robustness

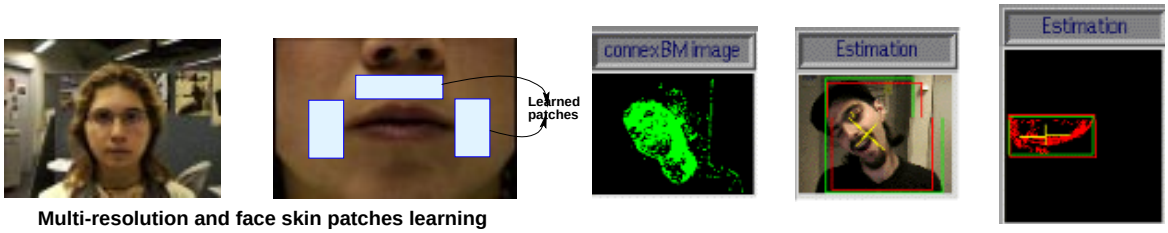


Figure 3-6: Multi-resolution mouth extraction, skin model learning. Head and mouth tracking with rotations and facial hair

of the system is increased by computing at each time step the linearly predicted position of the center of the mouth. A confidence level on the prediction is also computed, depending on the prediction error. When the prediction is not available or its confidence level drops below a threshold, the mouth's position is reinitialized.

3.3 Perception in the SmartCar

Smart cars

Our SmartCar is an automobile equipped with sensors and computers. The sensors enable the car to perceive the road, potential hazards, other vehicles and its own internal state

(acceleration throttle, gear, brake pedal activity, speed); the computers gather data from these sensors and process the information to recognize the current action that the driver and eventually the surrounding cars are doing, and to predict what will be their most likely next action.

Computers, unlike humans, are not subject to fatigue, boredom or distractions. Single vehicle roadway departure crashes (caused by driver inattention or impairment) account for almost 15000 deaths in the US annually [256]. Therefore a system able to augment the driver in these capacities could potentially have a substantial impact in reducing the number of accidents caused by such factors.

There have been different proposed designs for Intelligent Highway Systems (IHS): some advocate for completely computer-controlled vehicles (autonomous navigation) while others propose mixed approaches, where the vehicles have some intelligence to assist the human drivers. The latter approach is the one pursued in this thesis: the purpose of the SmartCar is to *augment* the driver as opposed to substitute for him.

Perceptual Issues At the tactical level[3] it is usually expected that perception systems can reliably track *traffic entities* such as vehicles, lanes and exits –along with information about the entity as the speed of a vehicle or the distance to an exit. Thus previous work on tactical driving modeling [203, 130, 50] has generally assumed that the smart vehicle has complete perfect knowledge of its surroundings. However there are several perceptual effects that are important at the tactical level:

1. **Blindspots**: The smart vehicle's sensors might not be able to scan all around the vehicle, or may only provide limited information (such as presence/absence of objects). To tackle blindspots the vehicle should make some assumptions about the contents of the unknown region.

2. **Occlusion**: Even if 360 degrees coverage is available, on-board sensors cannot see through opaque objects. This problem is specially severe when the smart vehicle is surrounded by large vehicles such as trucks or buses. Assuming that occluded regions are free of obstacles is risky.

[3] In section 5.5.2 of chapter 5 the driving taxonomy is described. It consists of three levels: strategic, tactical and operational.

3. **Sensor noise**: Measurements from real sensors are noisy for a variety of reasons. Physical phenomena (e.g. specular reflections), software limitations (e.g. range buckets) and unreliable tracking all introduce uncertainty into the observed world attributes.

4. **Sensor limits**: Current sensing technology is able to provide sufficiently accurate measurements of range and relative velocity of other vehicles (using computer vision, radar, sonar or optical flow) within a reasonable sensor range. At further ranges, errors in bearing may make object-to-lane mapping unreliable. Moreover, higher order derivatives of position such as acceleration or jerk are very noisy and cannot be assumed to be available.

To collect driving data in real situations (see section 5.5.5 in chapter 5), I have instrumented a Volvo V70XC, generously donated by Volvo for research purposes. The goal is to design and implement a data gathering platform for acquiring real-time driving maneuvers. In particular, I have collected driving maneuvers at a tactical level, as described in the experimental part of this thesis (section 5.5). In general, there are at least three different aspects relevant to the level of driving that is the subject of this thesis (tactical-level driving):

1. SmartCar physical self-state: information sensed from the speedometer, acceleration throttle, steering wheel angle sensor (rotary potentiometer), brake pedal, gear and GPS unit.

2. Road state: including road geometry and exit information.

3. Traffic state: relative speeds, direction and distances of the surrounding traffic.

The sensors installed in the SmartCar provide information about the internal state of the car (brake, gear, acceleration throttle, steering wheel angle and speed), the driver's face and gaze, the surrounding traffic and the road lanes' positions. I have not analyzed the road geometry, marks or exit information. Figure 3-7 illustrates the instrumentation that I have installed in the car.

The instrumented Volvo has the following sensors:

1. Frontal and rear wide field-of-view Sony EVI-D30 cameras mounted respectively on the frontal dash-board and on a tripod in the trunk (see figure 3-8 (a) and (b))

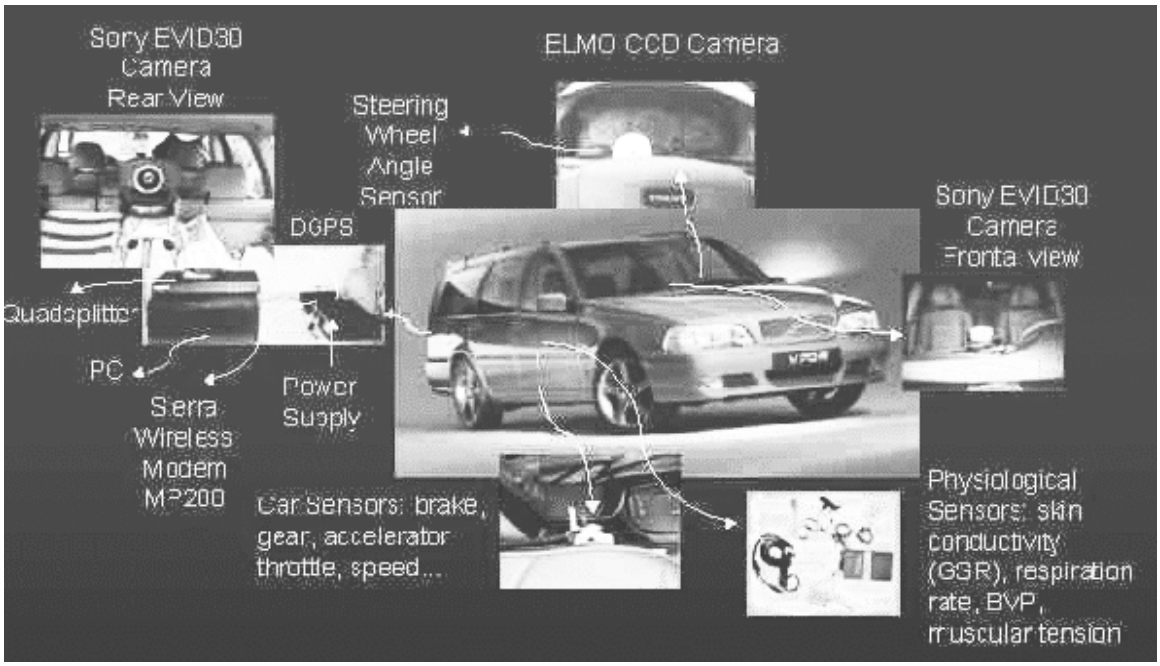


Figure 3-7: SmartCar (Volvo V70XC)

2. ELMO CCD QN401E color camera mounted on the steering wheel to record the driver's facial expressions, head pose and gaze (see figure 3-8 (c))

3. ELMO CCD QN401NE color camera mounted on a pair of glasses worn by the driver to record the driver's viewpoint (see figure 3-8 (d))

4. Steering wheel angle sensor using a rotary potentiometer mounted on the steering wheel (see figure 3-8 (c))

5. Gear, acceleration throttle, brake pedal action and speed coming from the car internal data bus

6. GPS unit

All the video signals are combined in a quadsplitter whose output is recorded using a Sony GV-A500 Hi8 Video Walkman recorder.

The first version of the data acquisition system was self-designed and built. The hardware consisted of PIC microcontrollers, A/D converters and additional electronic components to perform the analog-to-digital conversion, sampling and synchronization of all the car signals. The user interface was built on top of Sun Microsystem's COMMAPI (Communications API), a native library that allows serial-port communications in Java. The host

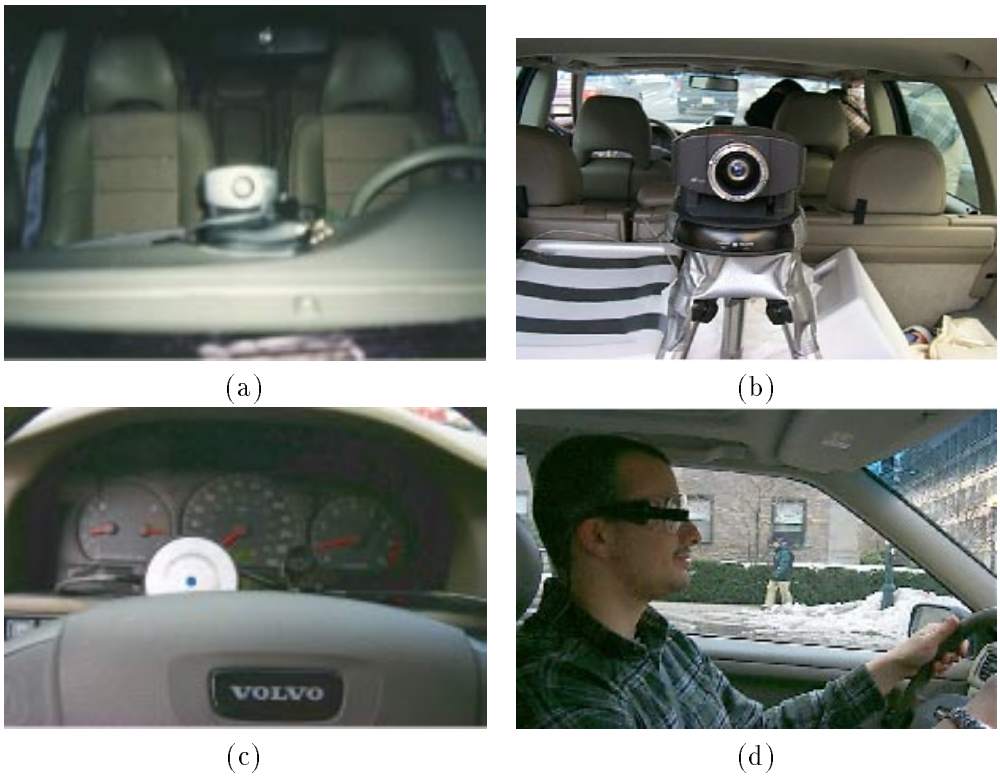


Figure 3-8: SmartCar sensors: (a) Front and rear wide-field-of-view cameras (b) Steering wheel sensor and driver's face camera (c) Driver's viewpoint camera

machine performed data collection by polling via the serial port each sensor independently, waiting for its response and logging the returned data to a file. Since all the sensors shared the same serial port, each sensor had to be addressed by a unique 8-bit ID. Finally, another Java application let the user read the log files with the recorded data and play them back in a graphical, intuitive fashion to facilitate data analysis. Even though this first prototype was operative, the system turned out to be too fragile for an automotive application. Therefore, I had to re-design the entire data acquisition hardware and software.

The current implementation of the hardware and software for acquiring in real-time car state data has been developed using National Instruments products. The hardware obtains its inputs from sources of three different nature as shown in table 3.2. All the car signals are connected to a Sony VAIO PCG-N505VE Intel Celeron microprocessor laptop computer via a PCMCIA Data Acquisition Card (DAQCard-AI-16XE-50) by National Instruments (http://www.ni.com). The analog signals are digitized (16 bits) and sampled at 150 scans/sec. The digital signals are sampled using the same card at 150 scans/sec. All the

signals can be directly connected to one of these boards, except for the speed given that it consists of a 12 pulse-per-revolution signal. Therefore for this signal a frequency-to-voltage converter is needed to convert it to analog.

Signal	Nature	Description
Speed	Analog	12 pulse per wheel revolution, square wave
Acceleration	Analog	Linear 0-12 V
Brake Pedal	Digital	Boolean (0=brake is off, 1=brake is on)
Gear	Digital	2-bit
Steering wheel angle	Analog	Up to 3 revolutions
GPS	Digital	NMEA ASCII string

Table 3.2: Sensor signals in the Smart Car.

The laptop and VCR are synchronized to guarantee the temporal alignement of the acquired signals.

The software for data acquisition and playback has been developed in LabVIEW. LabVIEW is a powerful programming environment used in engineering and scientific environments. LabVIEW is based on a functional programming language known as G, developed by National Instruments. It is based on graphics instead of written lines of text. The icon based programming structure is based on logical sequencing of images and is essentially independent of written language.

In LabVIEW, programs are referred to as VIs. VI stands for virtual instrument. I have developed graphical LabVIEW programs for calibrating the car signals, acquiring (triggering the acquisition and annotating the driving maneuvers as they take place), post-processing, analyzing and visualizing data. Figure 3-9 depicts one part of the entire data acquisition LabVIEW environment that I have developed in this thesis. The data acquisition system runs on the Sony VAIO laptop.

The contextual information is acquired via the video signals. I have developed a video processing graphical environment that let's the user record, playback and annotate the video signals coming from the front, rear and face driver cameras. Figure 3-10 depicts one screenshot of the program. Table 3.3 contains the information that was manually annotated for each frame of the maneuvers.

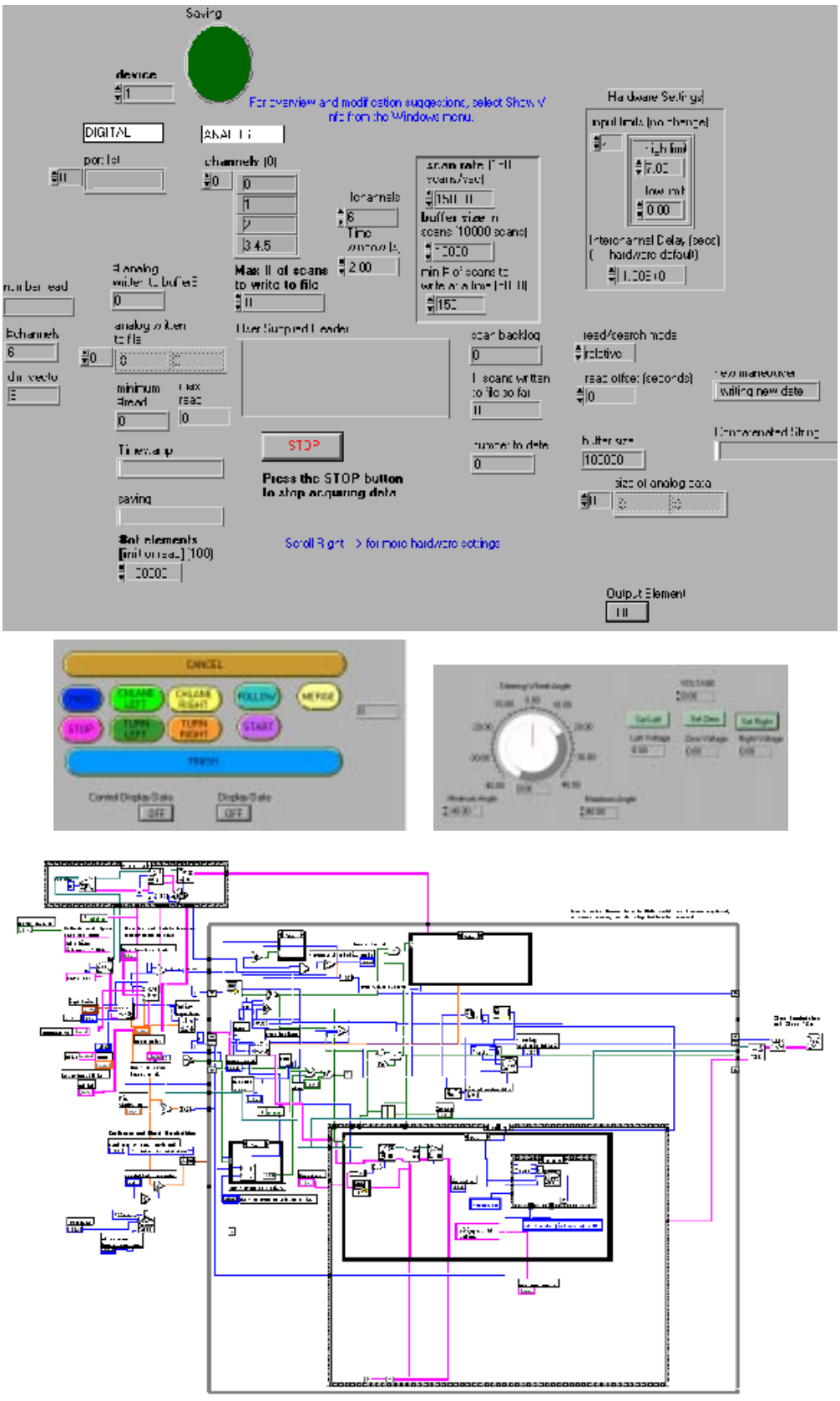
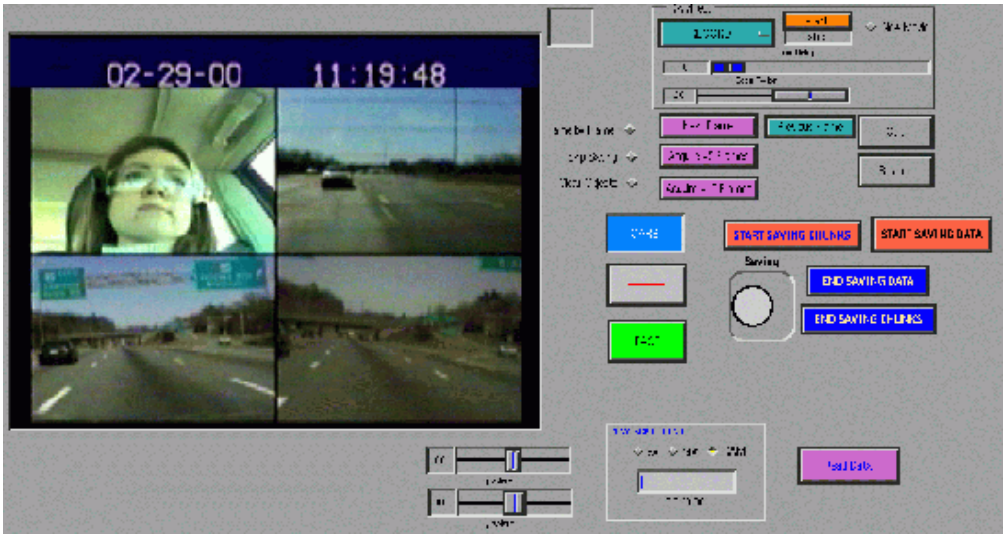


Figure 3-9: Example of LabVIEW graphical user interface and diagram.

	Front and rear traffic	Driver's face	Road lanes
Position	Right/Left/Same	Front/Rear-view mirror Right mirror/Left mirror Right/Left	Right/Left
Relative Speed	Slower/Same/Faster		
Relative Distance	Far/Medium/Close		
Direction	Same/Opposite		
Representation	Rectangle	Rectangle	line

Table 3.3: Information from the video annotation process



(a)

(b)

Figure 3-10: Graphical User Interface for video signals annotation: (a) Input image (b) Annotated image.

Chapter 4

Graphical Models For Human Behavior Modeling

This chapter describes the mathematical framework for learning from data individual, person-to-person and potentially multi-agent interactive behaviors. This chapter, thus, describes the upper-most layer of the human behavior model proposed in this thesis. Figure 4-1 highlights this layer within the model. I would claim in this thesis, in a similar way as it has been proposed in [184], that many human behaviors can be accurately described as a set of dynamic models (e.g. Kalman filters) sequenced together by a Markov chain. From this perspective, the human is considered as a device with a large number of internal mental states, each with its own particular control behavior and interstate transition probabilities. A canonical example of this type of model would be a bank of standard linear controllers (e.g. Kalman filters plus a simple control law), each using different dynamics and measurements, sequenced together with a Markov network of probabilistic transitions. The states of the model can be hierarchically organized to describe both short-term and long-term behaviors. In this chapter I will develop the theory behind these kind of models. First, I will present the general theory of dynamic graphical models. Then I will describe in detail the specific dynamic graphical model architectures used in this thesis for modeling human interactive behaviors.

In order to build effective computer models of human behaviors one needs to address the question of how knowledge can be mapped onto computation to dynamically deliver consistent interpretations. From a strict computational viewpoint there are two key problems

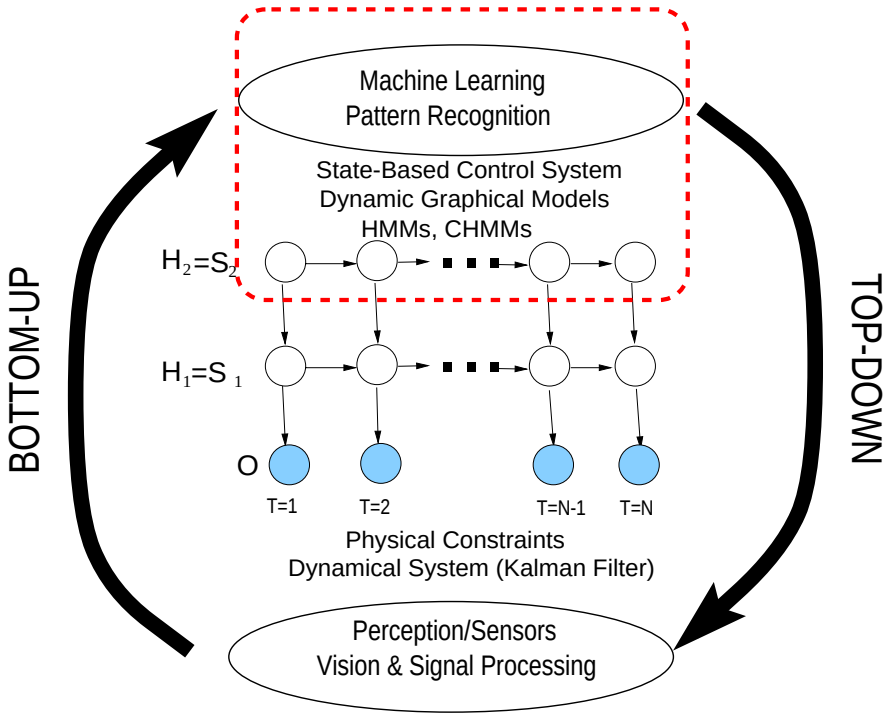


Figure 4-1: The perceptual system occupies the lowest level in the proposed model

when processing the continuous flow of feature data coming from a stream of input video: (1) Managing the computational load imposed by frame-by-frame examination of all of the agents and their interactions. For example, the number of possible interactions between any two agents of a set of N agents is $N * (N - 1)/2$. If naively managed this load can easily become large for even moderate N; (2) Even when the frame-by-frame load is small and the representation of each agent's instantaneous behavior is compact, there is still the problem of managing all this information over time.

Statistical directed acyclic graphs (DAGs) or probabilistic inference networks (PINs) [38, 89] can provide a computationally efficient solution to these problems. Moreover I propose that DAGs offer a sufficiently expressive and adequate framework for building models of human individual and interactive behaviors. Hidden Markov Models (HMMs) and their extensions, such as the architecture used in this thesis, namely Coupled Hidden Markov Models (CHMMs) [28, 30], can be viewed as a particular, simple case of temporal PIN or DAG. PINs consist of a set of random variables represented as nodes as well as directed edges or links between them. They define a mathematical form of the joint or conditional *pdf* between the random variables. More importantly from a human behavior perspective, they constitute a simple graphical way of representing causal dependencies

between variables. It has been remarked in chapter 2 that causality plays a crucial role in human behavior understanding. The absence of directed links between nodes implies a conditional independence. Moreover there is a family of transformations performed on the graphical structure that has a direct translation in terms of mathematical operations applied to the underlying *pdf*. Finally they are modular, i.e. one can express the joint global *pdf* as the product of local conditional *pdfs*.

In the following sections I will describe the basic theory behind PINs. Some of the material can be found in [227], [81] and [28]. The major contributions of this thesis in this area are: (1) the use of dynamic graphical models in perceptual systems for modeling individual, person-to-person or multi-agent real behaviors; (2) the proposal and use of a new graphical structure called Coupled Hidden Markov Models (CHMMs) that specifically captures causal influence between generative processes; (3) the proposal and use of a two-layer hierarchical architecture with a Kalman Filter at the lowest level and a Hidden Markov Model –or extension– at the upper level (see figure 1-3); (4) the validation of the proposed models with extensive human behavior data collected in real situations.

I will describe the models from two different perspectives: from the viewpoint of dynamic bayesian networks (graphical models) and from the traditional viewpoint of HMMs and extensions.

4.1 Background and Notation

For multivariate statistical modeling applications, such as the recognition of the human behaviors, the identification and manipulation of relevant conditional independence assumptions is a useful tool for model building and analysis. There has been a considerable amount of work exploring the relationships between conditional independence in probability models and structural properties of the associated graphs. In particular, the *separation properties* of a graph can be directly related to *conditional independence* properties in a set of associated probability models.

The analysis and manipulation of HMMs, CHMMs and related structures can be facilitated by exploiting the relationship between probability models and graphs. The major advantages to be gained are in:

- **Model description:** A graphical model provides a natural and intuitive framework

for representing dependencies between random variables. In particular, the structure of the graphical model clarifies the conditional independencies in the associated probability models, allowing model assessment and revision.

- **Computational efficiency:** The graphical model framework is a powerful basis for specifying efficient algorithms for computing quantities of interest in the probability model, e.g., calculation of the probability of observed data given the model. These inference algorithms can be specified automatically once the initial structure of the graph is determined.

In the following will refer to both probability models and graphical models. Each is composed of:

1. *Structure*: The structure of the model consists of the specification of a set of *conditional independence relations* for the probability model, or a set of *(missing) edges* in the graph for the graphical model. The graph not only allows to understand the dependencies between variables, but also serves as the backbone for efficiently computing marginal and conditional probabilities that may be required for inference and learning.

2. *Parameters*: The parameters of both probability and graphical models consist of the specification of the joint probability distribution: in factored form for the probability model and defined locally on the nodes of the graph in the graphical model.

There are three basic problems that are usually addressed in statistical machine learning:

1. The *inference* problem deals with computing the posterior probabilities of variables of interest given observable data and given a particular specification of the probabilistic model. The conditional independence relations derived from the absence of arcs in a graphical model can be exploited to obtain efficient algorithms for computing marginal and conditional probabilities. For *singly* connected graphs, in which the underlying undirected graph has no loops, there exist a number of equivalent algorithms, such as *belief propagation*, *junction tree*, *JLO* (described in detail in section 4.5), and the *Dawid* algorithms. For *multiply connected* networks, in which there can be more than one undirected path between any two nodes, the junction tree, JLO or Dawid algorithms can be used, but not belief propagation. However, there has been recent

progress on using belief propagation in graphs with a single loop, leading to a new algorithm called "loopy belief propagation" [158]. Just recently Weiss and Freeman [261] analyze the behavior of belief propagation in graphs of arbitrary topology when the nodes in the graph describe jointly Gaussian random variables. The authors give an analytical formula relating the true posterior probabilities with those calculated using loopy belief propagation.

2. The related task of *MAP identification* is the determination of the most likely state of a set of unobserved variables, given observed variables and the probabilistic model.

3. The *learning or estimation* problem is that of determining the parameters (and possibly structure) of the probabilistic model from data.

4.2 Notation and Background

Let $U = X_1, X_2, \ldots, X_N$ represent a set of discrete-valued random variables. Even though I will develop in this chapter the theory for discrete-valued random variables, many of the results generalize directly to continuous and mixed sets of random variables ([139], [265]). Let lower case x_1 denote one of the values of variable X_1. The notation $\sum_{x_1}$ means the sum over all possible values of X_1. Let $p(x_i)$ be shorthand for the particular probability $p(X_i = x_i)$, whereas $p(X_i)$ represents the probability function for X_i (i.e. a table of probability values, since X_i is assumed to be discrete), $1 \leq i \leq N$. The full joint distribution function is $p(U) = (X_1, X_2, \ldots, X_N)$, and $p(u) = (x_1, x_2, \ldots, x_N)$ denotes a particular value assignment for U.

If A, B and C are disjoint sets of random variables, the conditional independence relation $A \perp B|C$ is defined such that A is independent of B given C, i.e. $p(A, B|C) = p(A|C)p(B|C)$. Conditional independence is symmetric. Note also that marginal independence (no conditioning) does not in general imply conditional independence, nor does conditional independence in general imply marginal independence ([265]).

With any set of random variables U we can associate a graph G defined as $G = (V, E)$. The set of vertices or nodes in the graph are denoted by V such that there is a one-to-one mapping between the nodes in the graph and the random variables, i.e., $V = X_1, X_2, \ldots, X_N$. The set of edges is denoted by $E = e(i, j)$, where i and j are shorthand for the nodes X_i and X_j, $1 \leq i, j \leq N$. Edges of the form $e(i, i)$ are not of interest

and thus are not allowed in the graphs discussed in this thesis.

If the edges are ordered such that $e(i,j)$ means that the edge is directed from node i to node j, i is a *parent* of its *child* j. An *ancestor* of a node i is a node which has as a child either i or another ancestor of i. A subset of nodes A is an *ancestral* set if it contains its own ancestors. A *descendant* of i is either a child of i or a child of a descendant of i.

Two nodes i and j are *adjacent* in G if the set of all edges, E, contains the edge $e(i,j)$. A *path* is a sequence of distinct nodes $\{1,2,\ldots,m\}$ such that there exists an edge for each pair of nodes $\{l, l+1\}$ on the path. A graph is *singly-connected* if there exists only one path between any two nodes in the graph. A *cycle* is a path such that the beginning and ending nodes on the path are the same. A directed cycle is a cycle of directed edges which all point in the same direction.

If E contains only undirected edges then the graph G is an *undirected graph*, UG. If E contains only directed edges and no directed cycles, then G is an *directed acyclic graph*, DAG. If E contains a mixture of directed and undirected edges, then it is referred to as a *mixed or chain* graph. There exists a theory for graphical independence models involving mixed graphs ([265]) but mixed graphs will not be discussed further in this thesis.

For an UG, G, a subset of nodes C *separates* two other subsets of nodes A and B if every path joining every pair of nodes $i \in A$ and $j \in B$ contains at least one node from C. For $DAGs$ and mixed graphs analagous, but somewhat more complicated, separation properties exist.

A cycle is *chordless* if no other than successive pairs of nodes in the cycle are adjacent. A graph G is *triangulated* if and only if the only chordless cycles in the graph contain no more than three nodes. Thus, if one can find a chordless cycle of length four or more, G is not triangulated.

A graph G is *complete* if there are edges between all pairs of nodes. The *cliques* of G are the largest subgraphs of G that are complete. A *clique tree* of G is a tree of cliques such that there is a one-to-one node correspondence between the cliques of G and the nodes of the tree.

4.3 Probabilistic Independence Networks (PINs)

So far I have described some properties and definitions of graphs, without alluding to the underlying probability model represented by the graph. In this section I will review briefly the relation between a probabilistic independence network structure $G = (V, E)$ and a probability model $p(U) = p(X_1, X_2, \ldots, X_N)$. The results in this section are largely summarized versions of material in [179] and [265].

A probabilistic independence network structure (PIN structure) G is a graphical statement of a set of conditional independence relations for a set of random variables U. *Absence* of an edge, $e(i, j)$ in G implies some independence relation between the associated variables, X_i and X_j. Thus, a PIN structure G is a particular way of specifying some independence relationships present in the probability model $p(U)$. We say that G implies a *set* of probability models $p(U)$, denoted as $\mathcal{P}_\mathcal{G}$, i.e., $p(U) \in \mathcal{P}_\mathcal{G}$. In the reverse direction, a particular model $p(U)$ embodies a particular set of conditional independence assumptions which may or may not be representable in a consistent graphical form. One can derive all of the conditional independence properties and inference algorithms of interest for U without reference to graphical models. However, as has been emphasized in the statistical and AI literature, and it is reiterated in this thesis in the context of HMMs and extensions, there are distinct advantages to be gained from using the graphical formalism.

4.3.1 Undirected Probabilistic Independence Networks (UPINs)

A $UPIN$ is composed of both a $UPIN$ structure and $UPIN$ parameters. A $UPIN$ *structure* specifies a set of conditional independence relations for a probability model in the form of an *undirected* graph. $UPIN$ *parameters* consist of numerical specifications of a particular probability model consistent with the $UPIN$ structure. Terms used in the literature to describe $UPINs$ of one form or another include Markov random fields (MRFs), Markov networks, Boltzmann machines and log-linear models.

Conditional independence semantics of UPIN structures Let A, B and S be any disjoint subsets of nodes in an undirected graph UG, G. G is an undirected probabilistic network structure ($UPIN$ structure) for $p(U)$ if for any A, B and S such that S separates A and B in G, the conditional independence relation $A \perp B|S$ holds in $p(U)$. The set of all conditional independence relations implied by separation in G constitute the global *Markov*

properties of G. Figure 4-2 shows a simple example of a $UPIN$ structure for 7 variables.

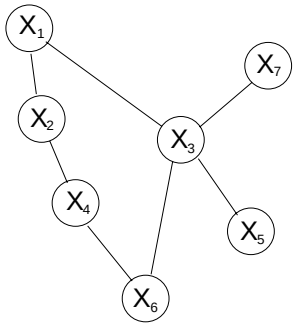


Figure 4-2: UPIN structure G which captures a particular set of conditional independence relationships among the set of variables $\{X_1, \ldots, X_N\}$. For example, $X_5 \perp \{X_1, X_2, X_3, X_4, X_6\} | \{X_3\}$

.

Thus, separation in the $UPIN$ structure implies conditional independence in the probability model, i.e., it constraints $p(U)$ to belong to a set of probability models $\mathcal{P}_\mathcal{G}$ which obey the Markov properties of the graph. Note that a complete UG is trivially a $UPIN$ structure for any $p(U)$ in the sense that there are no constraints on $p(U)$. G is a *perfect undirected map* for p if G is a $UPIN$ structure for p and all the conditional independence relations present in p are represented by separation in G. For many probability models p there are no perfect undirected maps. A weaker condition is that a $UPIN$ structure G is *minimal* for a probability model $p(U)$ if the removal of any edge from G implies an independence relation which is not present in the model $p(U)$, i.e. the structure without the edge is no longer a $UPIN$ structure for $p(U)$. Minimality is not equivalent to perfection (for $UPIN$ structures) since, for example, there exist probability models with independencies which can not be represented as $UPINs$ except for the complete $UPIN$ structure. For example, if X_1 and X_2 are marginally independent, but conditionally dependent given X_3 (see figure 4-5, for an example), then the complete graph is the minimal $UPIN$ structure for $\{X_1, X_2, X_3\}$ but it is not perfect because of the presence of an edge between X_1 and X_2.

Probability functions on UPIN structures Given a $UPIN$ structure G, the joint probability distribution for U can be expressed as a simple factorization:

$$p(u) = p(x_1, \ldots, x_N) = \prod_{V_C} a_C(x_C) \tag{4.1}$$

where V_C is the set of cliques of G, x_C represents a value assignment for the variables in a particular clique C, and the $a_C(x_C)$ are non-negative clique functions. The clique functions represent the particular *parameters* associated with the $UPIN$ structure. This corresponds directly to the standard definition of a *Markov random field* [101]. The clique functions reflect the relative "compatibility" of the value assignments in the clique.

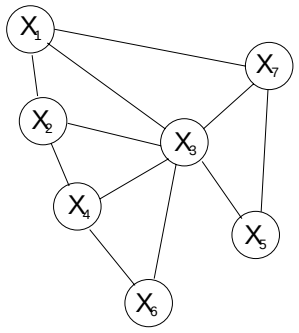


Figure 4-3: A triangulated version of the UPIN structure G from figure 4-2

A model p is said to be *decomposable* if it has a minimal $UPIN$ structure G which is triangulated. Figure 4-3 illustrates an example. A $UPIN$ structure G can be converted to a *junction tree*, which is a tree of cliques of G arranged such that the cliques satisfy the *running intersection property*: each node in G which appears in any two different cliques also appears in all the cliques on the path between these two cliques. Associated with each edge in the junction tree there is a *separator* S, such that S contains the variables in the intersection of the two cliques that it links. Given a junction tree representation, one can factorize $p(U)$ as the product of clique marginals over separator marginals ([179]):

$$p(u) = \frac{\prod_{C \in V_C} p(x_C)}{\prod_{S \in V_S} p(x_S)} \tag{4.2}$$

where $p(x_C)$ and $p(x_S)$ are the marginal (joint) distributions for the variables in clique C and separator S respectively, and V_C and V_S are the set of cliques and separators in the junction tree.

This product representation is central to the results in the rest of this chapter. It is the basis of the fact that globally consistent probability calculations on U can be carried out in a purely local manner. The mechanics of these local calculations are described later in this chapter. At this point it is sufficient to note that the complexity of the local inference algorithms scales as the sum of the sizes of the state-spaces of the cliques. Thus, local clique

updating can make probability calculations on U much more tractable than using "brute force" inference, if the model decomposes into relatively small cliques.

Many probability models of interest may be not decomposable. However, we can define a *decomposable cover* G^T for p such that G^T is a triangulated, but not necessarily minimal, $UPIN$ structure for p. Since any $UPIN$ G can be triangulated simply by addition of the appropriate edges, one can always identify at least one decomposable cover of G^T. However, a decomposable cover may not be minimal in that it can contain edges which obscure certain independencies in the model p. For example, the complete graph is a decomposable cover of *all* possible probability models p over the variables. For efficient inference, the goal is to find a decomposable cover G^T such that G^T contains as few extra edges as possible over the original $UPIN$ structure G. Later in this chapter I will discuss an algorithm for finding decomposable covers for arbitrary PIN structures. All *singly-connected* $UPIN$ structures imply probability models $\mathcal{P}_\mathcal{G}$ which are decomposable.

Note that, given a particular probability model p and a $UPIN$ G for p, the process of adding extra edges to G to create a decomposable cover does not change the underlying probability model p, i.e., the added edges are a convenience for manipulating the graphical representation, but the underlying numerical probability specifications remain unchanged.

An important point is that decomposable covers have the running intersection property and thus can be factored as in equation 4.2: thus local clique updating is also possible with non-decomposable models via this conversion. Once again, the complexity of such local inference scales with the sum of the size of state-spaces of the cliques of the decomposable cover.

In summary, any UPIN structure can be converted to a junction tree permitting inference calculations to be carried out purely locally on cliques.

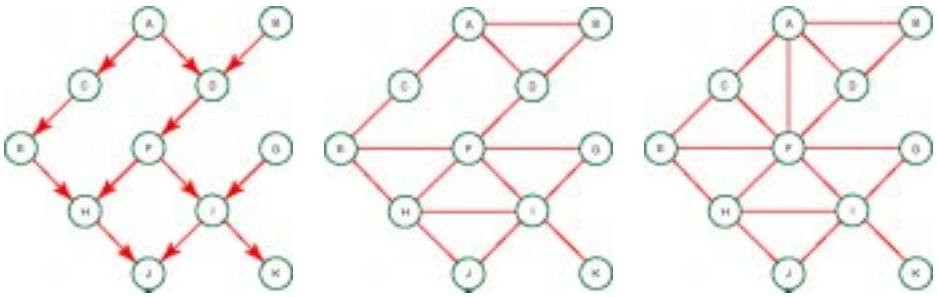
4.3.2 Directed Probabilistic Independence Networks (DPINs)

A $DPIN$ is composed of both a $DPIN$ structure and $DPIN$ parameters. A $DPIN$ *structure* specifies a set of conditional independence relations for a probability model in the form of a *directed graph*. $DPIN$ *parameters* consist of numerical specifications of a particular probability model consistent with the $DPIN$ structure. $DPINs$ are referred to in the literature using different names, including Bayes networks, belief networks, recursive graphical models, causal belief networks, and probabilistic causal networks.

Conditional Independence Semantics of $DPIN$ Structures

A $DPIN$ structure is a DAG $G^D = (V, E)$ where there is a one-to-one correspondence between V and the elements of the set of random variables $U = X_1, \ldots, X_N$.

The moral graph G^M of G^D is defined as the undirected graph obtained from G^D by placing undirected edges between all non-adjacent parents of each node and then dropping the directions from the remaining directed edges. See figure 4-4 for an example of how to obtain the moral graph from a DAG. The term "moral" was coined to denote the "marrying" of "unmarried" (non-adjacent) parents.



(a) (b) (c)

Figure 4-4: (a) A DPIN structure G^D which captures a set of independence relationships among the set $\{A, B \ldots, K\}$. (b) The moral graph G^M for G^D, where the parents of every node have been linked. (c) The triangulated graph.

Let A, B and S be any disjoint subsets of nodes in G^D. G^D is a $DPIN$ structure for $p(U)$ if for any A, B and S such that S separates A and B in G^D, the conditional independence relation $A \perp B|S$ holds in $p(U)$. This is the same definition as for a $UPIN$ structure, except that separation has a different interpretation in the directed context: *S separates A from B in a directed graph if S separates A from B in the moral (undirected) graph of the smallest ancestral set containing A, B and S* [136]. It can be shown that this is equivalent to the statement that a variable X_i is independent of all other nodes in the graph except for its descendants, given the values of its parents. Thus, as with a $UPIN$ structure, the $DPIN$ structure implies certain conditional independence relations, which in turn imply a set of probability models $p \in P_{G^D}$. Figure 4-4 contains a simple example of the steps to follow to obtain a triangulated graph from a $DPIN$.

4.3.3 Probability Functions on DPINs

A basic property of a $DPIN$ structure is that it implies a direct factorization of the joint probability distribution $p(U)$:

$$p(u) = \prod_{i=1}^{N} p(x_i|pa(x_i)) \tag{4.3}$$

where $pa(x_i)$ are the value assignment for the parents of the node X_i. A probability model p can be written in its factorized form in a trivial manner by the conditioning rule. Consequently there are many possible $DPIN$ structures consistent with a particular probability model p, potentially containing extra edges which hide true conditional independence relations. Thus, one can define *minimal $DPIN$ structures* for p in a manner exactly equivalent to that of $UPIN$ structures: deletion of an edge in a minimal $DPIN$ structure G^D is a *perfect $DPIN$ structure G for p* if G^D is a $DPIN$ structure for p and all the conditional independence relations present in p are represented by separation in G^D. As with $UPIN$ structures, minimal does not imply perfect for $DPIN$ structures. For example, the $UPIN$ in figure 4-5 (b) encodes the independence relations: $X_1 \perp X_3|X_2, X_4$ and $X_2 \perp X_4|X_1, X_3$. However, the minimal $DPIN$ structure contains an edge from X_4 to X_2.

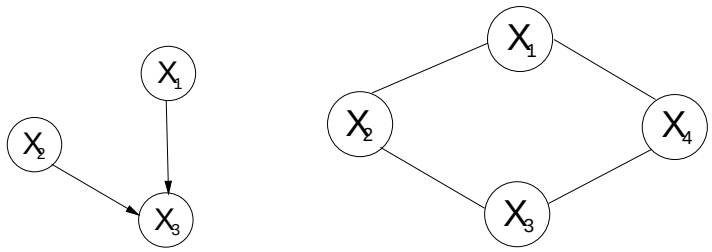


Figure 4-5: (a) The $DPIN$ structure to encode the fact that X_3 depends on X_1 and X_2, but $X_1 \perp X_2$. For example, consider that X_1 and X_2 are two independent coin flips and that X_3 is a bell which rings when the flips are the same. There is no perfect $UPIN$ structure which can encode these dependence relationships. (b) A $UPIN$ structure which encodes $X_1 \perp X_3|X_2, X_4$ and $X_2 \perp X_4|X_1, X_3$. There is no perfect $DPIN$ structure that can encode these dependencies.

4.3.4 Differences between Directed and Undirected Graphical Representations

It is important to emphasize that directed and undirected graphs possess different conditional independence semantics. There are common conditional independence relations which have perfect $DPIN$ structures but no perfect $UPIN$ structure and vice-versa. See figure 4-5 for an example.

Does a $DPIN$ structure have the same Markov properties as the $UPIN$ structure obtained by dropping all the directions on the edges in the $DPIN$ structure? The answer is *yes, if and only if the $DPIN$ structure contains no subgraphs where a node has two or more non-adjacent parents* [265], [179]. In general it can be shown that if a $UPIN$ structure G for p is decomposable (triangulated) then it has the same Markov properties as some $DPIN$ structure for p.

On a more practical level, $DPIN$ structures are frequently used to encode causal information, i.e. to formally represent the belief that X_i preceeds X_j in some causal sense, e.g. temporally $DPINs$ have found application in causal modeling in applied statistics and artificial intelligence. Their popularity in these fields stems from the fact that the joint probability model can be specified directly via equation 4.3, i.e. via the specification of conditional probability tables or functions [231]. In contrast, $UPINs$ must be specified in terms of clique functions (as in equation 4.1) which may not be easy to work with (cf. [78, 153, 250] for examples of ad hoc design of clique functions in image processing). $UPINs$ are more frequently used in problems such as image analysis and statistical physics where associations are thought to be correlational rather than causal. Causality is central in human behavior modeling (see chapter 2). Therefore the behavior models developed and proposed in this thesis are $DPIN$ temporal structures.

4.3.5 From DPINs to Decomposable UPINs

The moral $UPIN$ structure G^M obtained from the $DPIN$ structure G^D does not imply any new independence relations which are not present in G^D. As with triangulation, however, the additional edges may obscure conditional independence relations which are implicit in the numeric specification of the original probability model p associated with the $DPIN$ structure G^D. Furthermore, G^M might not be triangulated (decomposable). By the addi-

tion of appropriate edges, the moral graph can be converted to a non-unique triangulated graph G^T, namely a decomposable cover for G^M. In this manner, for any probability model p for which G^D is a $DPIN$ structure, one can construct a decomposable cover G^T for p.

This mapping from $DPIN$ structures to $UPIN$ structures was first discussed in the context of efficient inference algorithms by Lauritzen and Spiegelhalter ([137]). The advantage of this mapping derives from the fact that analysis and manipulation of the resulting $UPIN$ is considerably more direct than dealing with the original $DPIN$. Furthermore, it has been shown that many of the inference algorithms for $DPINs$ are in fact special cases of inference algorithms for $UPINs$ and can be considerably less efficient ([219]).

4.4 Dynamic Probabilistic Independence Networks (DynPINs)

In time series modeling, we observe the values of certain variables at different instants of time. The assumption that an event can cause another event in the future, but not vice-versa, simplifies the design of DPINs for time series: directed arcs should flow forward in time. Assigning a time index t to each variable, one of the simplest causal models for a sequence of observed data $O = o_1, o_2, \ldots, o_{T-1}, o_T$ is a *first order Markov Model*, or $MM(1,1)$, in which each variable is directly influenced only by the previous variable (see figure 4-6):

$$P(o_1, o_2, \ldots, o_{T-1}, o_T) = P(o_1)P(o_2|o_1)\ldots P(o_{T-1}|o_T) \tag{4.4}$$

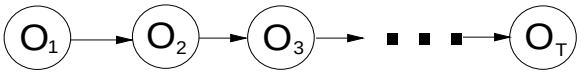


Figure 4-6: A dynamic graphical model representing a first-order Markov process $MM(1,1)$

These models do not represent direct dependencies between observables over more than one time step. Having observed $O = o_1, o_2, \ldots, o_{t-1}, o_t$ the model will only make use of o_t to predict the value of o_{t+1}. One simple way of adding more memory to the system is by allowing higher order interactions between variables. For example a r^{th} order Markov model allows arcs from $O = o_{t-r}, \ldots, o_{t-1}$ to o_t. Another way to extend simple Markov

models is by making the observations depend on a hidden variable H, which we will call the *state variable*, and making the sequence of states be a Markov process (see figure 4-7). A classic model of this kind is the linear-Gaussian state-space model, also known as the Kalman filter.

4.4.1 DynPINs for Kalman Filters

I have briefly described Kalman Filters in section 3.2 of chapter 3, as one of the elements of the perceptual (bottom-most) level of the proposed human behavior model. For completeness, I will present in this section Kalman Filters from the perspective of dynamic graphical models. In a state-space model the sequence of D-dimensional real-valued T observation vectors $O = o_1, o_2, \ldots, o_{T-1}, o_T$, is modeled by assuming that at each time step o_t was generated from a K-dimensional real-valued hidden state variable H_t, and that the sequence of $H = h_1, h_2, \ldots, h_T$ define a first-order Markov process. See figure 4-7 for the graph structure of such models.

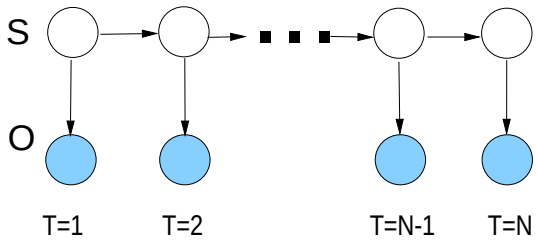


Figure 4-7: Graphical representation of a state-space model

The well known simple first order state-space model obeys the following two conditional independence relations:

$$H_t \perp H_1, O_1, \ldots, H_{t-2}, O_{t-2}, O_{t-1} | H_{t-1}, \qquad 2 \leq t \leq T \tag{4.5}$$

$$O_t \perp H_1, O_1, \ldots, H_{t-1}, O_{t-1} | H_t, \qquad 2 \leq t \leq T \tag{4.6}$$

Therefore the joint probability distribution $P(O_t, H_t)$ is given by

$$P(O_t, H_t) = P(H_1)P(O_1|H_1)\prod_{t=2}^{T} P(H_t|H_{t-1})P(O_t|H_t) \tag{4.7}$$

In the case of Kalman filters, the state transition function that provides H_t given H_{t-1}

can be decomposed into deterministic and stochastic components:

$$H_t = f_t(H_{t-1}) + w_t \tag{4.8}$$

where f_t is the deterministic transition function to obtain the mean of H_t given H_{t-1}, and w_t is a zero-mean random noise vector. Similarly the the continuous observation vector O_t is given by:

$$O_t = g_t(H_t) + v_t \tag{4.9}$$

If both transition f_t and output g_t functions are linear and time-invariant and the distribution of the states and observation noise variables is Gaussian, the model becomes a *linear-Gaussian state-space* model, more commonly known as *Kalman filter*:

$$H_t = AH_{t-1} + w_t \tag{4.10}$$

$$O_t = CH_t + v_t \tag{4.11}$$

where A is the state transition matrix and C is the observation matrix.

Often the observations are divided into a set of predictor or input variables U_t and output or response variables, leading to *input-output models*. Again, assuming linearity and Gaussian noise we can write the state transition function as

$$H_t = AH_{t-1} + BU_t + w_t \tag{4.12}$$

4.4.2 DynPINs for Hidden Markov Models (HMM(1,1))

As I have already stated, dynamic graphical models lie at the heart of the upper-most layer in the human behavior model proposed in this thesis (see figure 4-1). Hidden Markov Models (HMMs) and extensions (CHMMs) decompose the behaviors in a sequence of discrete, non-observed states (which could be mapped the mental state of the human performing the action) with probabilistic transitions between states and observations. Note that the observations at the upper-most level are the predictions of the Kalman Filter from the previous (bottom-most) level. This section describes HMMs from the perspective of dynamic graphical models.

In Hidden Markov (HMMs) modeling problems ([196]) we are interested in the set

of random variables $U = H_1, O_1, H_2, O_2, \ldots, H_{T-1}, O_{T-1}, H_T, O_T$, where H_t is a discrete valued hidden variable at index t, and O_t is the corresponding continous or discrete-valued observed variable at index t, $1 \leq t \leq T$ (the results here can be directly extended to continuous-valued observables). The index i denotes a sequence from 1 to T, for example, discrete time steps. Note that O_t is considered univariate for convenience: the extension to the multivariate case, with d observables is straightforward but it is omitted here for simplicity since it does not affect the conditional independence relationships in the HMM.

The well known simple first order HMM obeys the following two conditional independence relations[1]:

$$H_t \perp H_1, O_1, \ldots, H_{t-2}, O_{t-2}, O_{t-1} | H_{t-1}, \qquad 2 \leq t \leq N \tag{4.13}$$

$$O_t \perp H_1, O_1, \ldots, H_{t-1}, O_{t-1} | H_t, \qquad 2 \leq t \leq N \tag{4.14}$$

We will refer to this "first-order" hidden Markov probability model as $HMM(1,1)$: the notation $HMM(K,J)$ is defined such that the model has state memory of depth K and contains J separate underlying state processes. The notation will be clearer in later sections when I will discuss extensions to the $HMM(1,1)$, such as Coupled Hidden Markov Models (CHMM) that, under this notation, become $HMM(1,2)$.

Construction of a PIN for $HMM(1,1)$ is particularly simple. In the undirected case, assumption 1 requires that each state H_t is only connected to H_{t-1} from the set $\{H_1, O_1, \ldots, H_{t-2}, O_{t-2}, O_{t-1}\}$. Assumption 2 requires that O_i is only connected to H_t. The resulting $UPIN$ structure for $HMM(1,1)$ is shown in figure 4-8. This graph is singly-connected and thus implies a decomposable probability model p for $HMM(1,1)$, where the cliques are of the form $\{H_t, O_t\}$ and $\{H_{t-1}, H_t\}$. In section 4.5 the joint probability distribution is expressed as a product function in the junction tree, thus leading to a junction tree definition of the familiar forward-backward (Baum-Welch) and Viterbi inference algorithms. The junction tree for a $HMM(1,1)$ is depicted in figure 4-8 (b).

In the directed case the connectivity for the DPIN structure is the same. It is natural to choose the directions of the edges between H_{t-1} and H_t as going from $t-1$ to t because time goes forward and not backwards. The reverse direction could also be chosen without

[1] Note that these two conditions are identical to the already given in equation 4.5 for a state-space model, as expected, because the graph structure is identical in both cases.

changing the Markov properties of the graph, but violating the causality principle of physical systems. The directions on the edges between the hidden state variables H_t and the observables O_t must be chosen as going from H_t to O_t rather than in the reverse direction (see figure 4-9, (a)). If the reverse arrows were chosen (as shown in figure 4-9, (b)) it would imply that O_t is marginally independent of H_{t-1} which is not true in the $HMM(1,1)$ probability model. The proper direction of the edges implies the correct relation, namely that O_t is *conditionally independent* of H_{t-1} given H_t. The log probability of the model (hidden and observed nodes) is given by:

$$\log P(O_t, H_t) = \log P(H_1) + \sum_{t=1}^{T} \log P(O_t|H_t) + \sum_{t=1}^{T} \log P(H_t|H_{t-1}) \tag{4.15}$$

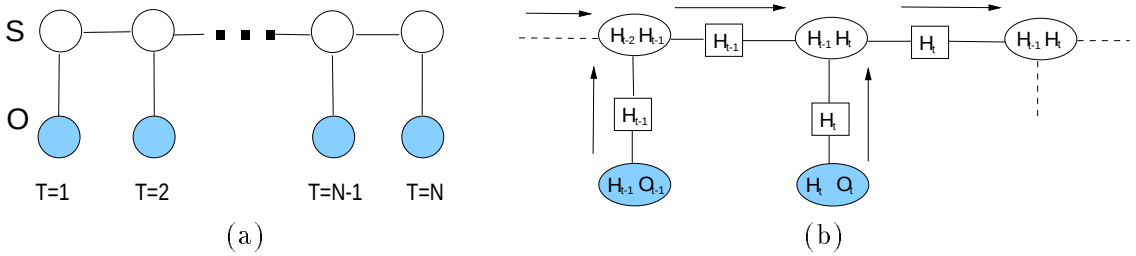


Figure 4-8: (a) A UPIN for a single process, 1^{st} order HMM, $HMM(1,1)$. (b) The corresponding junction tree.

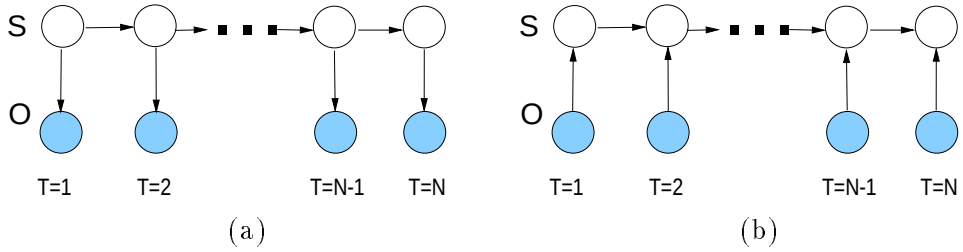


Figure 4-9: (a) A DPIN structure for the $HMM(1,1)$ probability model, (b) a DPIN structure which is not a DPIN for the $HMM(1,1)$ probability model

The DPIN structure for $HMM(1,1)$ does not possess a subgraph with non adjacent parents. As stated earlier this implies that the independence properties of the DPIN structure are the same as those of the corresponding UPIN structure obtained by dropping the directions from the edges in the DPIN structure, and thus they both result in the same junction tree structure (see figure 4-8, (b)). Thus, for the $HMM(1,1)$ probability model,

the minimal directed and undirected graphs possess the same Markov properties, i.e. imply the same conditional independence relations. Furthermore, both PIN structures are perfect maps for the directed and undirected cases respectively.

4.5 Inference and MAP algorithms for DPINs

Inference and MAP algorithms for DPINs and UPINs are quite similar. In the case of UPINs some subtleties are involved that are not encountered in DPINs. All the graphical models employed in this thesis are dynamic DPINs (DynPINs). Therefore I will only describe one of the inference algorithms for DPINs. In particular, I will present the algorithm developed by Jensen, Lauritzen and Olesen [108] that I will refer to as the JLO algorithm hereon. The original JLO algorithm applies to discrete variables. However extensions for Gaussian and Gaussian-mixture distributions are discussed in Lauritzen and Spiegelhalter [138]. There is also a closely related algorithm to the JLO algorithm solves the MAP identification problem with the same complexity as the original JLO inference algorithm ([56]). It is the so called Dawid's propagation algorithm and it is described in section 4.8.

The JLO algorithm is a strict generalization of the well-known forward-backward and Viterbi algorithms for HMM(1,1) in that they can be applied to arbitrarily complex graph structures (and thus a large family of probabilistic graphical models beyond $HMM(1,1)$) and can handle missing data and partial inference in a straightforward manner.

There are many variations of the original JLO algorithm. For example, Pearl ([179]) describes related versions of these algorithms in his early work. It can be shown ([219]) that all known exact algorithms for inference on DPINs are equivalent at some level to the JLO algorithm.

The JLO algorithm consists of two steps:

1. **Construction Step:** It involves a series of sub-steps where the original graph is moralized and triangulated, a junction tree is formed, and the junction tree is initialized.

2. **Propagation Step:** The junction tree is used in a local message-passing manner to propagate the effects of observed evidence, i.e. to solve the inference and MAP problems.

The construction step needs to be carried out just once per graph. The propagation step is carried out every time a new inference for the given graph is requested.

The Construction Step for the JLO Algorithm: from DPIN Structures to Junction Trees The goal of the construction step is to build a junction tree (JT) from the DPIN structure. The construction step is composed of two sub-steps: (1) first the original graph is moralized giving a moral graph G^M (figure 4-10 (b)); (2) second the moral graph is triangulated to obtain a decomposable cover G^T (figure 4-10 (c)). The algorithm operates in a greedy manner based on the fact that a graph is triangulated if and only if all of its nodes can be eliminated, where a node can be eliminated whenever all of its neighbors are pairwise linked. Whenever a node is eliminated, it and its neighbors define a clique in the JT that is eventually constructed. Thus, we can triangulate a graph and generate the cliques for the JT by eliminating nodes in some order, adding links if necessary. If no node can be eliminated without adding links, then we choose the node that can be eliminated by adding the links that yield the clique with the smallest state-space ([109]). Note that the time complexity of the JLO algorithm for a junction tree with N_C cliques and $s(C_i)$ states in the joint state-space clique C_i (i.e. the product over each variable in C_i of the number of states of each variable) is $O(\sum_{i=1}^{N_C} s(C_i))$. Therefore the most efficient inference corresponds to the JT with the smallest cliques. (3) After triangulation the JLO algorithm constructs a JT from G^T, i.e. a clique tree satisfying the running intersection property. The JT construction goes as follows: define the weight of a link between two cliques as the number of variables in their intersection. Then, a tree of cliques will satisfy the running intersection property if and only if it is a spanning tree of maximal weight. Thus, the JLO algorithm constructs a JT by choosing successively a link of maximal weight unless it creates a cycle. The JT constructed from the cliques defined by the DPIN structure triangulation in figure 4-10 (c) is shown if figure 4-10 (d).

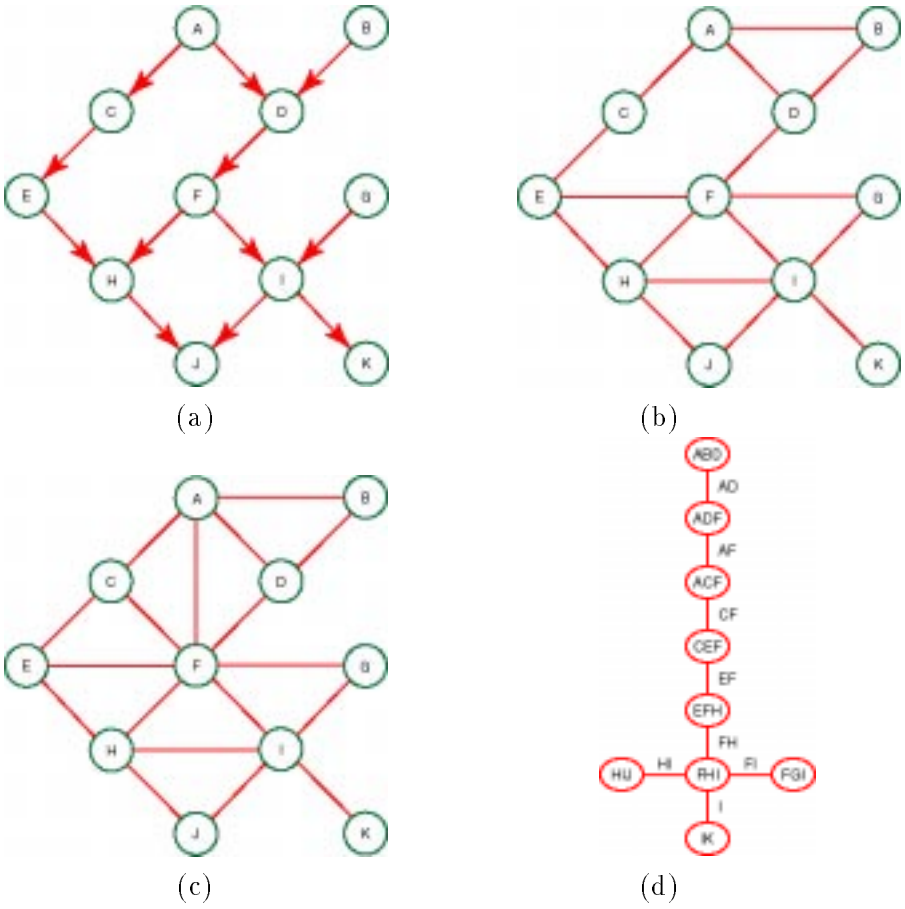


Figure 4-10: (a) A DPIN structure G^D. (b) The moral graph G^M for G^D, where the parents of every node have been linked. (c) The triangulated graph G^T where the nodes have been linked to satisfy the running intersection property. (d) The corresponding junction tree (JT).

In a graph with N nodes, the worst-case complexity is $O(N^3)$ for the triangulation heuristic and $O(N^2 logN)$ for the maximal spanning tree portion of the algorithm. This construction step is carried out only once as an initial step to convert the original graph to a JT representation.

Potential Functions in the Junction Tree The next step is to take the numeric probability specifications as defined on the directed graph G^D (given by equation 4.3) and convert this information into the general form for a JT representation of p (see equation 4.1). This is achieved by noting that each variable X_i is contained in at least one clique in the JT. The procedure thus is as follows: assign each X_i to just one such clique and for each clique define the potential function $a_C(C)$ to be either the product of $p(X_i|pa(X_i))$ or 1 if not variables are assigned to that clique. The initial values of the separator potentials in equation 4.1 are set to 1.

A schedule of local message passing can be defined which converges to a *globally consistent marginal representation* for p, i.e. the potential on any clique or separator is the marginal for that clique or separator. Thus, via local message passing, one can go from the initial potential representation defined above to a marginal representation:

$$p(u) = \frac{\prod_{C \in V_C} p(x_C)}{\prod_{S \in V_S} p(x_S)} \tag{4.16}$$

Once these calculations are finished the JT is initialized. The most interesting calculation, however, is the ability to propagate information through the graph given some observed data and the initialized JT, e.g. to calculate the posterior probabilities of some variables of interest.

Local Message Propagation in Junction Trees using the JLO Algorithm In general $p(U)$ can be expressed as

$$p(u) = \frac{\prod_{C \in V_C} a_C(x_C)}{\prod_{S \in V_S} b_S(x_S)} \tag{4.17}$$

where the a_C and b_S are non-negative potential functions (the potential functions could be the initial marginals described above, for example). $K = (a_C : C \in V_C, b_S : S \in S_C)$ is a representation for $p(U)$. A factorizable function $p(U)$ can admit many different representa-

tions, i.e. many different sets of clique and separator functions which satisfy equation 4.17 given a particular $p(U)$.

The JLO algorithm carries out globally consistent probability calculations via local message-passing on the JT, i.e. probability information is passed between neighboring cliques and clique and separator potentials are updated based on this local information. Cliques and separators are updated such that K is at all times a representation for $p(U)$, i.e. equation 4.17 holds at all times. Eventually the propagation converges to the marginal representation given the initial model and the observed evidence.

The message passing proceeds as follows: given two adjacent cliques C_i and C_j and given S_k the separator between them, we define

$$b^*_{S_k}(x_{S_k}) = \sum_{C_i \setminus S_k} a_{C_i}(x_{C_i}) \tag{4.18}$$

and

$$a^*_{C_j}(x_{C_j}) = a_{C_j}(x_{C_j})\lambda_{S_k}(x_{S_k}) \tag{4.19}$$

where

$$\lambda_{S_k}(x_{S_k}) = \frac{b^*_{S_k}(x_{S_k})}{b_{S_k}(x_{S_k})} \tag{4.20}$$

with $\lambda_{S_k}(x_{S_k})$ being the *update factor*. Passage of a flow corresponds to updating the neighboring clique with the probability information contained in the originating clique and it is illustrated in figure 4-11. This flow induces a new representation $K^* = (a^*_C : C \in V_C, b^*_S : S \in S_C)$ for $p(U)$.

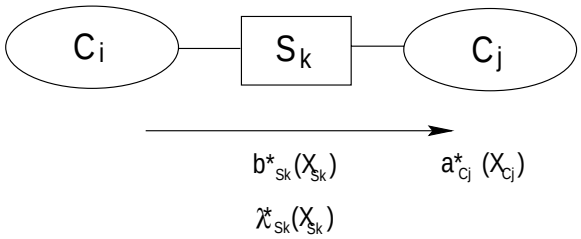


Figure 4-11: Message passing algorithm from clique C_i to clique C_j via the separator S_k

A schedule for the message passing flow can be defined such that all the cliques are eventually updated with all relevant information and JT tree reaches an *equilibrium state*. The most direct scheduling scheme consists of two steps: one node of the tree is defined as

the root of the JT. During the *collection phase* the messages flow along all edges towards the root clique (if a node has more than one incoming flow, the flows are absorbed sequentially). After the collection is completed, the *distribution phase* involves passing messages from the root to the rest of the nodes in the JT along the same edges. There are at most two flows along any edge in the tree in a non-redundant schedule. The directionality of the message passing flow in the JT does not have to coincide in principle with the edge directionality in the original DPIN structure.

The JLO Algorithm for Inference given Observed Evidence In the case of observing some *evidence*, i.e. the value of some variables in the graphical model, one needs to update the marginal probabilities of all nodes in the JT to incorporate this new evidence. Consider that we observe evidence $e = \{X_i = x_i^*, X_j = x_j^*, \ldots\}$ and $U^e = \{X_i, X_j, \ldots\}$ denotes the set of variables that have been observed. Let $U^h = U$, with U^h denote the set of hidden or unobserved variables and u^h a value assignment for the hidden nodes U^h.

To compute the posterior probability of the hidden nodes given the observations, $p(U^h|e)$ we define an evidence function $g^e(x_i)$ such that

$$g^e(x_i) = \begin{cases} 1 \quad if \quad x_i = x_i^* \\ 0 \qquad \text{otherwise} \end{cases} \tag{4.21}$$

Instead of directly computing $p(U^h|e)$ we compute $f^*(u) \propto p(u^h|e)$, given by

$$f^*(u) = p(u) \prod_{U^e} g^e(x_i) \tag{4.22}$$

The steps to obtain $f^*(u)$ from the message passing algorithm are: (1) *Entering evidence into the cliques*, by assigning each observed variable $X_i \in U^e$ to one particular clique that contains it. We C^E denote the set of all cliques into which evidence is entered in this manner. (2) For each $C \in C^E$ we compute $g_C(x_C) = \prod_{\{i:X_i \text{ is entered into C}\}} g^e(x_i)$. (3) $f^*(u) = p(u) \times \prod_{C \in C^E} g_C(x_C)$. The effects of these modifications can be propagated throughout the JT using the collect and distribute schedule. The x_C^h denote a value assignment of the hidden (unobserved) variables in clique C. Once the message passing is finished, a new final representation K_f^* is obtained such that the local potential on each clique –or separator– is $f^*(x_C) = p(x_C^h, e)$, i.e. the joint probability of the local unobserved clique variables and

the introduced evidence ([108]).

The probability of the observed evidence $p(e)$ is given by $p(e) = \sum_{X_C^h} p(x_C^h, e)$. The *conditional probability* of the local unobserved clique variables given the evidence, $p(x_C^h|e)$ is the normalized to 1 potential function at the clique where the hidden variable belongs to.

Complexity of the Message Passing Algorithm The time complexity of propagation within a junction tree is $O(\sum_{i=1}^{N_C} s(C_i))$ where N_C is the number of cliques in the juntion tree and $s(C_i)$ is the number of states in the joint state-space clique C_i, i.e. the product over each variable in C_i of the number of states of each variable. Therefore the most efficient inference corresponds to the JT with the smallest cliques. Unfortunately the problem of finding optimally small junction trees (i.e. JTs with the smallest maximal clique) is NP-hard. Heuristic algorithms have been found to perform well in practice ([108], [109]).

4.6 Inference and MAP problems in HMMs

Starting from a generative model of the data –a prior model–, the learning problem consists of estimating the parameters of the model that best fit the observed data O. This fit is generally measured by the *likelihood* of the data given the parameters θ, i.e. $\mathcal{L}(O|\theta)$, which can be maximized as a function of the parameters. Bayesian approaches extend this inference process by incorporating a prior distribution over the parameters $p(\theta)$ and requiring that the result of the learning process be a posterior distribution on the parameters.

An equivalent approach can be derived using *information theory* [49]. In this case, the goal of the learner is to communicate the data as efficient as possible to the receiver, thereby producing a compact –compressed– representation of the data. A cost function quantifying the efficiency of this communication process can be derived from the Minimum Description Length (MDL) principle ([116]). Using Shannon's coding theorem, the MDL cost function can be shown to be equal to the posterior probability of the parameters given the data.

Data Maximum Likelihood: In the context of HMMs, the most common inference problem is the calculation of the likelihood of the observed evidence given the model, i.e. $P(o_1, o_2, \ldots, o_T|M)$, where $o_1, o_2, \ldots, o_T$ denote the observed values for the variables $O_1, O_2, \ldots, O_T$. Again, in this section, I will assume that only one model, M, –the ML model– has been selected and therefore a structure and parameters have been

determined. Thus I will omit the conditioning on the model M hereon. The direct method for obtaining the previous probability would be to sum out the unobserved state variables from the full joint probability distribution:

$$p(o_1, o_2, \ldots, o_T) = \sum_{h_1, \ldots, h_T} p(H_1, o_1, \ldots, H_T, o_T) \tag{4.23}$$

where h_t denotes the possible values of the hidden variable H_t.

State Posterior: Another inference calculation of interest is the calculation of $P(h_t = s_i | o_1, o_2, \ldots, o_T)$, for any or all i, namely the probability of a particular hidden state value given the observed evidence. Inferring the posterior state probabilities is useful when the states have direct physical interpretations (as in fault monitoring applications [226]) and is also implicitly required during the standard Baum-Welch learning algorithm for $HMM(1,1)$ (see section 4.6 for a detailed description). The states in the behavior models developed in this thesis seem to correspond to sub-actions that compose a longer behavior. They could be related to the non-observed mental states of the human performing the action. As chapter 2 has stated, a similar explanation of human behavior has been proposed in Psychology and Philosophy.

In general both of these calculations scale as N^T where N is the number of states for each hidden variable. In practice, the forward-backward algorithm ([196], [192]) can perform these inference calculations with much lower complexity, namely TN^2, by using dynamic programming. The likelihood of the observed evidence can be obtained with the forward step of the forward-backward algorithm; calculation of the state posterior probabilities requires both the forward and backward steps. The forward-backward algorithm relies on a *factorization* of the joint probability distribution to obtain locally recursive methods. One important aspect of the graphical modeling approach is that it provides an automatic method for determining such local efficient factorizations, for an arbitrary probabilistic model, if *efficient factorizations exist* given the conditional independence relations specified in the model.

The MAP identification problem in the context of HMMs involves identifying the most likely hidden state sequence given the observed evidence. Just as with the inference problem, the Viterbi algorithm provides an efficient, locally recursive method for solving this problem with complexity TN^2. In the same way as with the inference problem, the graphical

modeling approach provides an automatic technique for determining efficient solutions to the MAP problem for arbitrary models, if an efficient solution is possible given the structure of the model.

The Baum-Welch Algorithm as a Special Case of the JLO Algorithm The JT for a standard simple HMM, HMM(1,1), is depicted in figure 4-12. Usually dynamic graphical models are represented "rolled out" in time. Each hidden clique in the JT (e.g. (H_{t-1}, H_t) in figure 4-12) contains the hidden states at consecutive points in time. Traditionally the Baum-Welch or Forward-Backward algorithm ([196]) are used for doing inference in the model. The inference problem consists of, given a set of values for the observable variables,

$$e = O_1 = o_1, O_2 = o_2, \ldots, O_T = o_T \tag{4.24}$$

inferring the likelihood of the evidence e given the model. This problem can be exactly solved by local propagation in any JT using the JLO inference algorithm.

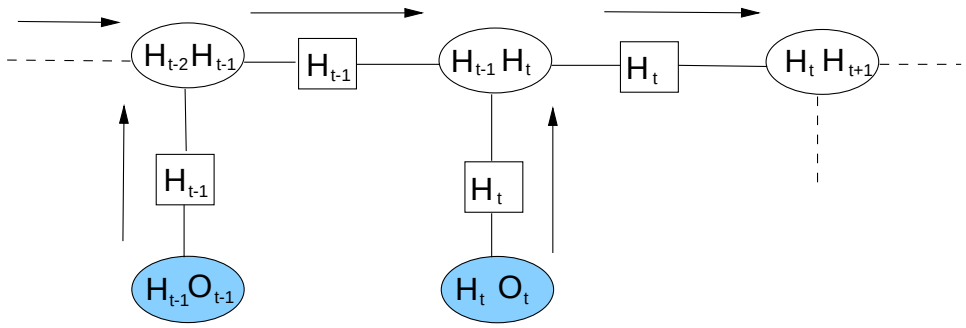


Figure 4-12: Local message passing in a standard single HMM, HMM(1,1) JT during the collect phase on a "left to right" schedule. Ovals represent cliques and squares separators. Arrows indicate the message flow.

First we have to select a root clique in the JT. Let the last clique, (H_{T-1}, H_T) be the root clique. A non redundant message passing schedule consists of (1) *collect step:* recursively passing messages from each observation clique, (O_t, H_t), and previous hidden state clique, (H_{i-2}, H_{i-1}), to the current hidden state clique, (H_{i-1}, H_i), in the appropriate sequence until reaching the root (last) clique. This is the *forward step* of the traditional Forward-Backward algorithm; (2) *propagate step:* distributing messages in the reverse direction from the root clique to the first clique, i.e. the *backward step* of the Forward-Backward algorithm. If we were interested in computing the *likelihood* of the evidence e given the

model, $p(e|O, M)$, the distribute or backward step is not needed since we can marginalize over the local variables in the root clique to obtain $p(e)$.

In the following subscripts on potential functions $f^*(h_t)$ and update factors $\lambda(h_t)$ indicate which variables have been used in deriving that potential or update factor, e.g. f_{O_1} indicates that this potential has been updated based on information exclusively about O_1 and no other variable.

1. Collection Step: Forward Step Let's assume that the JT has been initialized so that the potential function in each clique and separator is the local marginal. Given some observed evidence e, each individual observation, $O = o_t^*$ is entered into each observation clique (O_t, H_t) such that each clique marginal becomes $f^*_{O_t}(h_t, o_t) = p(h_t, o_t^*)$ after entering evidence.

Looking at figure 4-12, the potential on the separator H_t is updated, by definition, to

$$f^*_{O_t}(h_t) = \sum_{o_t} f^*(h_t, o_t) = p(h_t, o_t^*) \tag{4.25}$$

The update factor from this separator flowing into its adjacent clique (H_{t-1}, H_t) is then given by

$$\lambda_{O_t}(h_t) = \frac{p(h_t, o_t^*)}{p(h_t)} = p(o_t^*|h_t) \tag{4.26}$$

This update factor is incorporated into (H_{t-1}, H_t) through

$$f^*_{O_t}(h_{t-1}, h_t) = p(h_{t-1}, h_t)\lambda_{O_t}(h_t) = p(h_{t-1}, h_t)p(o_t^*|h_t) \tag{4.27}$$

Now consider the message flow from clique (H_{t-2}, H_{t-1}) to clique (H_{t-1}, H_t). Let $\Phi_{t,k} = O_t, \ldots, O_k$ denote the set of consecutive observable variables, and $\Phi^*_{t,k} = o_t^*, \ldots, o_k^*$ denote the set of observed values for these variables (evidence), $1 \leq t < k \leq T$. Assume that the potential on the separator H_{t-1} has been updated to

$$f^*_{\Phi_{1,t-1}}(h_{t-1} = s_j) = f^*_{j,\Phi_{1,t-1}} = p^*(h_{t-1} = s_j, \Phi^*_{1,t-1}) \tag{4.28}$$

because of the earlier flows in the schedule. Therefore the update factor on separator H_{t-1}

becomes

$$\lambda_{\Phi_{1,t-1}}(h_{t-1} = s_j) = \lambda_{j,\Phi_{1,t-1}} = \frac{p^*(h_{t-1} = s_j, \Phi^*_{1,t-1})}{p(h_{t-1} = s_j)} \tag{4.29}$$

where the hidden node h_{t-1} is assumed to be in state s_j, $1 \leq j \leq N$.

This gets absorbed into clique (H_{t-1}, H_t) to produce

$$\begin{aligned} f^*_{\Phi_{1,t}}(h_{t-1}, h_t) &= p^*_{O_t}(h_{t-1}, h_t)\lambda_{\Phi_{1,t-1}}(h_{t-1}) && (4.30)\\ & p(h_{t-1}, h_t)p(o^*_t|h_t)\frac{p^*(h_{t-1}, \Phi^*_{1,t-1})}{p(h_{t-1})} && (4.31)\\ & p(o^*_t|h_t)p(h_t|h_{t-1})p^*(h_{t-1}, \Phi^*_{1,t-1}) && (4.32) \end{aligned}$$

Finally, we can calculate the new potential on the separator for the flow from clique (H_{t-1}, H_t) to (H_t, H_{t+1}),

$$\begin{aligned} f^*_{\Phi_{1,t}}(h_t = s_j) = f^*_{j,\Phi_{1,t}} &= \sum_{h_{i-1}} f^*_{\Phi_{1,t}}(h_{t-1}, h_t) && (4.33)\\ &= p(o^*_t|h_t)\sum_{h_{t-1}} p(h_t|h_{t-1})p^*(h_{t-1}, \Phi^*_{1,t-1}) && (4.34)\\ &= p(o^*_t|h_t)\sum_{h_{t-1}} p(h_t|h_{t-1})f^*_{\Phi_{1,t-1}}(h_{t-1}) && (4.35)\\ &= \alpha_t(j) && (4.36) \end{aligned}$$

with $1 \leq j \leq N$.

Equation 4.33 corresponds to the recursive equation (equation 20 in [196]) for the $\alpha_t(j)$ (forward) variables used in the forward step of the Forward-Backward or Baum-Welch algorithm in HMMs. Using a "left-to-right" schedule the updated potential functions on the separators between the hidden cliques, the $f^*_{\Phi_{1,t}}(h_t = s_j) = f^*_{j,\Phi_{1,t}}$ are exactly the $\alpha_{t,j}$ variables. Therefore the JLO algorithm applied to a standard single HMM, HMM(1,1) produces exactly the same local recursive calculations as the forward step in the Forward-Backward algorithm.

Dynamic Programming and State Trellis This forward step can also be seen from a dynamic programming viewpoint. Dynamic programming allows us to collect statistics on a exponential number of possible paths through a single HMM state trellis in polynomial time, because evidence from all paths that share a state a time t can be combined without loss of information. The *state trellis* for a single HMM is a representation of all possible state

sequences that the HMM could go through given some evidence. Figure 4-13 depicts the state trellis for a 3-state HMM: in a state trellis, each column corresponds to possible states for the hidden nodes at a time slice. A path across the trellis multiplies the probabilities associated with each traversed hidden state and transition from one hidden state to the next one. We have seen that an HMM of time length T and with N possible states per hidden variable has a complexity of N^T possible paths. Using dynamic programming, i.e. the forward or collect step, the N^T paths can be explored by tracking only N paths "heads". To collect statistics for estimation of the marginal probabilities of each hidden state given the evidence ($f^*_{O_t}(h_t = s_j, o_t) = p(h_t = s_j, o^*_t) = p_{j,t}$), it is necessary to pass through every state and every transition at every time slice. In consequence, dynamic programming in a trellis of length T and width N takes $O(TN^2)$ time, which is, of course, exactly the same complexity as that given by the JLO algorithm.

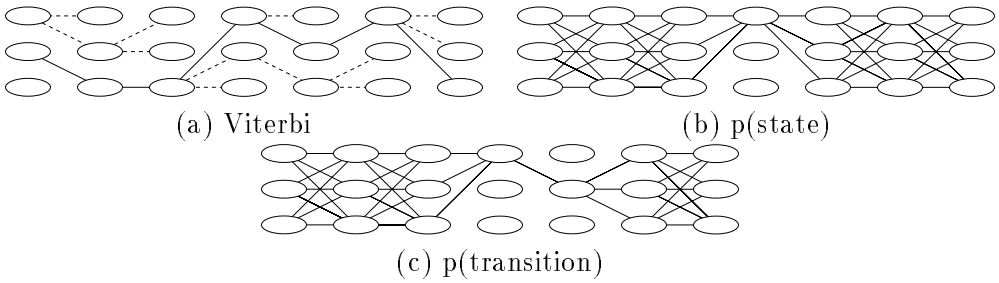


Figure 4-13: State trellis for a single standard 3-state HMM, HMM(1,1). (a) Most likely state path computed by the Viterbi algorithm; (b) probability of a hidden state ($= f^*_{j,\Phi_{1,t}} \lambda^*_{\Phi_{i,t},j} = \alpha_{t,j}\beta_t(j)$); (c) probability of a transition from one hidden state to another.

2. Distribution Step: Backward Step Similarly, the backward step of the Forward-Backward algorithm corresponds to the collect step of the JLO algorithm when the root node is the first node. Therefore, in case the "left-most" hidden clique in the JT, namely (H_1, H_2) is the root clique. We define a message passing flow from right to left. Figure 4-14 illustrates the message flow that takes place. Assume the potential in the hidden clique (H_t, H_{t+1}) has already been updated by earlier messages from the right. Thus, by definition,

$$f^*_{\Phi_{t+1,T}}(h_t, h_{t+1}) = p^*(h_t, h_{t+1}, \Phi^*_{t+1,T}) \tag{4.37}$$

The potential function on the separator, H_t between (H_t, H_{t+1}) and (H_{t-1}, H_t) is given

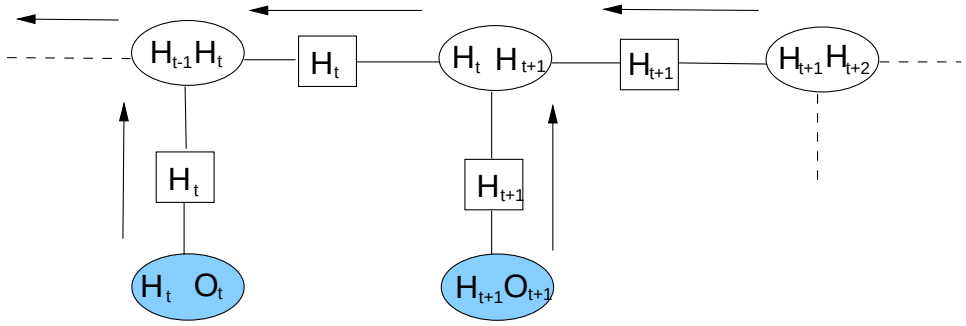


Figure 4-14: Local message passing in a standard single HMM, HMM(1,1) JT during the "right to left" distribution phase. Ovals represent cliques and squares separators. Arrows indicate the message flow.

by

$$
\begin{aligned}
f^*_{\Phi_{t+1,T}}(h_t = s_j) &= f^*_{j,\Phi_{t+1,T}} && (4.38)\\
&= \sum_{h_{t+1}} f^*_{\Phi_{t+1,T}}(h_t, h_{t+1}) && (4.39)\\
&= \sum_{h_{t+1}} p(h_t, h_{t+1}, \Phi^*_{t+1,T}) && (4.40)\\
&= p(h_t) \sum_{h_{t+1}} p(h_{t+1}|h_t) p(o^*_{t+1}|h_{t+1}) p(\Phi^*_{t+1,T}|h_{t+1}) && (4.41)\\
&\quad \text{(by virtue of the conditional independences} && (4.42)\\
&\quad \text{that hold for the HMM}(1,1)) && (4.43)\\
&= p(h_t) \sum_{h_{t+1}} p(h_{t+1}|h_t) p(o^*_{t+1}|h_{t+1}) \frac{p(\Phi^*_{t+2,T}, h_{t+1})}{p(h_{t+1})} && (4.44)\\
&= p(h_t) \sum_{h_{t+1}} p(h_{t+1}|h_t) p(o^*_{t+1}|h_{t+1}) \frac{f^*_{\Phi_{t+2,T}}(h_{t+1})}{p(h_{t+1})} && (4.45)
\end{aligned}
$$

The update factor on the separator H_t yields

$$
\begin{aligned}
\lambda^*_{\Phi_{t+1,T}}(h_t = s_j) &= \lambda^j_{\Phi_{t+1,T}} && (4.46)\\
&= \frac{f^*_{\Phi_{t+2,T}(h_t = s_j)}}{p(h_t = s_j)} && (4.47)\\
&= \sum_{h_{t+1}} p(h_t|h_{t+1}) p(o^*_{t+1}|h_{t+1}) \frac{f^*_{\Phi_{t+2,T}}(h_{t+1})}{p(h_{t+1})} && (4.48)\\
&= \sum_{h_{t+1}} p(h_t|h_{t+1}) p(o^*_{t+1}|h_{t+1}) \lambda^*_{\Phi_{t+2,T}}(h_{t+1}) && (4.49)
\end{aligned}
$$

The recursive formulation for the update factor $\lambda^*_{\Phi_{t+1,T}}(h_t = s_j)$ is exactly the recursive equation (equation 25 in [196]) for the $\beta_t(j)$ variables in the backward step of the Forward-Backward algorithm. Therefore, the JLO inference algorithm is equivalent to the Forward-Backward or Baum-Welch algorithm in traditional single HMMs, HMM(1,1), as expected.

4.7 Learning and ML Inference with Complete Data

At this point I will describe the techniques used for estimating the parameters in a graphical model given the observed data. There is a variety of methods that could be used for parameter estimation: from maximum-likelihood (ML), maximum-a-posteriori (MAP), or full Bayesian methods, to more traditional techniques such as gradient descent, expectation-maximization (EM) or Monte-Carlo sampling, or new techniques such as maximum entropy discrimination [104].

4.7.1 Model Learning

I have previously stated that there are three basic problems that are commonly addressed in statistical machine learning: (1) the inference problem, (2) the MAP state identification problem, and (3) the parameter estimation and model learning problems. In this section I will review very briefly the problem of model learning.

The observations can be used not only for parameter estimation, but also for model selection (graph structure). One solution to this problem is the Bayesian approach. A Bayesian approach to learning starts with some *a priori* knowledge about the model structure (as defined in 4.1) and model parameters. This initial knowledge is represented in the form of a prior probability distribution over model structures and parameters, and updated using the data to obtain a posterior probability distribution over models and parameters. Formally, assuming a prior distribution over model structures $P(M)$ and a prior distribution over parameters for each model structure $P(\theta|M)$ a data set D is used to form a posterior distribution over models using Bayes rule

$$P(M|D) = \frac{\int P(D|\theta, M)P(\theta|M)d\theta P(M)}{P(D)} \tag{4.50}$$

which integrates out the uncertainty in the parameters. Some criteria for selecting the model include the highest posterior probability model or the average predictions of two or

more models weighted by their posterior probabilities.

For a given model structure, we can compute the posterior distribution over the parameters:

$$P(\theta|M,D) = \frac{P(D|\theta,M)P(\theta|M)}{P(D|M)} \tag{4.51}$$

If the data is a time series, i.e. a sequence of observations at consecutive instants of time $D = o_1, o_2, \ldots, o_{T-1}, o_T$ and we wish to predict the next observation, o_{T+1}, given the data and the models, the the Bayesian prediction

$$P(o_{T+1}|D) = \int P(o_{T+1}|\theta,M,D)P(\theta|M,D)P(M|D)d\theta dM \tag{4.52}$$

integrates out the uncertainty in the model structure and parameters.

In this Bayesian approach to learning, usually we assume a single model structure M and we estimate the parameters $\hat{\theta}$ that maximize the likelihood $P(\theta|M,D)$ under that model. In the limit of a large data set and an uninformative (e.g. uniform) prior over the parameters, the posterior $P(\theta|M,D)$ will be sharply peaked around the maxima of the likelihood, and therefore the predictions of a single maximum likelihood (ML) model will be similar to those obtained by Bayesian integration over the parameters.

When there is missing data –such as hidden variables– the exact computation of the Bayesian integral 4.50 is usually intractable. Simple approximations to this integral exist, such as the Bayesian Information Criterion (BIC) described in [218]:

$$\log p(D|M) \approx \log p(D|\hat{\theta},M) - d/2 \log N \tag{4.53}$$

where $\hat{\theta}$ is the ML estimate of the parameters, N is the number of observations and d is the dimension of the model M –typically the number of parameters of M. The first term of this "score" function rewards how well the data fits the model M, whereas the second term punishes model complexity. This score does not depend on the parameter prior and thus can be applied easily. Raftery [198] applies BIC in the context of graphical and other statistical models.

The BIC score is the additive inverse of Rissanen's [116] minimum description length (MDL) principle. Other scores, which can be viewed as approximations to the marginal likelihood, are hypothesis testing ([198]) and cross validation ([73]) (see section 4.7.1).

Structural Risk Minimization [253], [252] minimizes a regularized risk functional $R_{reg}[f]$, which is the weighted sum of the empirical risk functional $R_{emp}[f]$ –given by the normalized negative log likelihood– and a regularization or complexity term $Q[f]$

$$R_{reg}[f] = R_{emp} + \lambda Q[f] \tag{4.54}$$

The regularization term $Q[f]$ is added to prevent overfitting. This regularizer is a convex penalty term on some quantity related to the function to be estimated, f. Two requirements will be imposed on $Q[f]$: it has to be convex and continuous (in order not to alter the optimization problem) and should restrict the function class in such a way that uniform convergence bounds can be stated. In other words, one minimizes $R_{emp}[f]$ while keeping the model complexity fixed by enforcing the upper bound on the measure of complexity –regularization term– $Q[f]$. This is what should be done when following the empirical risk minimization principle. In [175] a new model selection method is presented that exploits the geometry of statistical manifolds in a Structural Risk Minimization framework.

The interested reader would find a comprehensive review of the literature on learning the structure of PINs in [38].

In this thesis I focus on estimating the ML parameters for a model given the model structure. Although this is only an approximation to pure Bayesian learning, in practice full-fledged Bayesian analysis is often impractical. Furthermore there are application areas where there is strong a priori knowledge about the model structure and a single estimate of the parameters provides a more parsimonious and interpretable model than a distribution over the parameters. The technique employed in this thesis for selecting the model structure is *k-fold cross-validation*.

Cross validation Cross-validation and bootstrapping are both methods for estimating generalization error based on "resampling" ([260], [94], [187]). The resulting estimates of generalization error are often used for choosing among various models, such as different graphical model architectures.

In *k-fold cross-validation*, you divide the observed –training– data into k subsets of (approximately) equal size. You estimate the model parameters (train) k times, each time leaving out one of the subsets from training, but using only the omitted subset to compute the chosen error criterion. In this thesis I use likelihood as the evaluating function. If

k equals the sample size, this is called "leave-one-out" cross-validation. "Leave-v-out" is a more elaborate and expensive version of cross-validation that involves leaving out all possible subsets of v cases.

Leave-one-out cross-validation is easily confused with jackknifing. Both involve omitting each training case in turn and retraining the network on the remaining subset. But cross-validation is used to estimate generalization error, while the jackknife is used to estimate the bias of a statistic. In the jackknife, some statistic of interest is computed in each subset of the data. The average of these subset statistics is compared with the corresponding statistic computed from the entire sample in order to estimate the bias of the latter. One can also obtain a jackknife estimate of the standard error of a statistic. Jackknifing can be used to estimate the bias of the training error and hence to estimate the generalization error, but this process is more complicated than leave-one-out cross-validation [60].

Cross-validation can be used simply to estimate the generalization error of a given model, or it can be used for model selection by choosing one of several models that has the smallest estimated generalization error, as in the case of this thesis. For example, cross-validation can be used to choose the dimensionality of the hidden state nodes in a HMM (number of states), or to select a subset of the inputs (subset selection). A subset that contains all relevant inputs will be called a "good" subset, while the subset that contains all relevant inputs but no others will be called the "best" subset. Note that subsets are "good" and "best" in an asymptotic sense (as the number of training cases goes to infinity). With a small training set, it is possible that a subset that is smaller than the "best" subset may provide better generalization error.

Leave-one-out cross-validation often works well for estimating generalization error for continuous error functions such as the mean squared error, but it may perform poorly for discontinuous error functions such as the number of misclassified cases. In the latter case, k-fold cross-validation is preferred. But if k gets too small, the error estimate is pessimistically biased because of the difference in training-set size between the full-sample analysis and the cross-validation analyses. (For model-selection purposes, this bias can actually help; see the discussion in [220].) A value of 10 for k is popular for estimating generalization error. This is the value used in the cross validation methods of this thesis and it is referred to as 10-fold validation.

Leave-one-out cross-validation can also run into trouble with various model-selection

methods. Again, one problem is lack of continuity –a small change in the data can cause a large change in the model selected ([33]). For choosing subsets of inputs in linear regression, [34] found 10-fold and 5-fold cross-validation to work better than leave-one-out. [125] also obtained good results for 10-fold cross-validation with empirical decision trees (C4.5). Values of k as small as 5 or even 2 may work even better if you analyze several different random k-way splits of the data to reduce the variability of the cross-validation estimate.

Leave-one-out cross-validation also has more subtle deficiencies for model selection. [221] shows that in linear models, leave-one-out cross-validation is asymptotically equivalent to Akaike's Information criterion (AIC), but leave-v-out cross-validation is asymptotically equivalent to Schwarz's Bayesian criterion (called SBC or BIC) when $v = N[1 - \frac{1}{(log(N)-1)}]$, where N is the number of training cases. BIC provides consistent subset-selection, while AIC does not. That is, BIC will choose the "best" subset with probability approaching one as the size of the training set goes to infinity. AIC has an asymptotic probability of one of choosing a "good" subset, but less than one of choosing the "best" subset ([236]). Many simulation studies have also found that AIC overfits badly in small samples, and that BIC works well (e.g., [99], [222]). Hence, these results suggest that leave-one-out cross-validation should overfit in small samples, but leave-v-out cross-validation with appropriate v should do better. However, when true models have an infinite number of parameters, BIC is not efficient, and other criteria that are asymptotically efficient but not consistent for model selection may produce better generalization.

4.7.2 ML Estimation with Complete Data

Given a set of independend and identically distributed (i.i.d.) observations $D = o_1, \ldots, o_T$, each of which can be a vector or time series of vectors, then the likelihood of this data set is:

$$P(D|\theta, M) = \prod_{t=1}^{T} P(o_t|\theta, M) \tag{4.55}$$

For convenience in the notation I will drop from now on the implicit conditioning on the model structure, M. The ML parameters are obtained by maximizig the likelihood, or equivalently the log likelihood:

$$\mathcal{L}(\theta) = \sum_{t=1}^{T} log P(o_t|\theta) \tag{4.56}$$

If the observations include all the variables (nodes) in the graphical model, then each term in the log likelihood further factors as:

$$log P(O_t|\theta) = log \prod_j P(O_{t,j}|O_{t,pa(j)}, \theta_j) = \sum_j log P(O_{t,j}|O_{t,pa(j)}, \theta_j), \tag{4.57}$$

where j indexes over the nodes in the graphical model, $pa(j)$ is the set of parents of node j, and θ_j are the parameters that define the conditional probability of O_t given its parents. The likelihood therefore decouples into local terms involving only each node and its parents, greately simplifying the ML estimation problem. For example, if the O variables are discrete and θ_j is the conditional probability table (CPT) for O_j given its parents, then the ML estimate of θ_j is simply a normalized table containing counts of each setting of O_j given each setting of its parents in the set.

4.7.3 ML Estimation with Incomplete Data via the EM Algorithm

When there are hidden variables in the graphical structure –i.e. unobserved nodes, such as the hidden states in a HMM– the log likelihood cannot be decomposed as in equation 4.57. Instead we have to marginalize the log likelihood by summing over all the hidden nodes:

$$\mathcal{L}(\theta) = log P(O_t|\theta) = log \sum_{H_t} P(O_t, H_t|\theta) \tag{4.58}$$

with $1 \leq t \leq T$ and where H_t is the set of hidden variables, and $\sum_{H_t}$ is the sum –or integral in the continuous case– over H_t required to obtain the marginal probability of the data. Because equation 4.58 cannot be computed directly, we will lower bound it by a computationally feasible bound. This is the essence of the Expectation-Maximization EM algorithm ([57]). Using any distribution Q over the hidden nodes we obtain a lower bound for $\mathcal{L}$[2]:

$$\begin{aligned} log \sum_H P(O, H|\theta) &= log \sum_H Q(H) \frac{P(O, H|\theta)}{Q(H)} && (4.59) \\ &\leq \sum_H Q(H) log \frac{P(O, H|\theta)}{Q(H)} && (4.60) \\ &= \sum_H Q(H) log P(O, H|\theta) - \sum_H Q(H) log Q(H) = && (4.61) \end{aligned}$$

[2]Note that it is a variational bound. For more details see [114]

$$= \quad -\mathcal{F}(Q,\theta) \tag{4.62}$$

where the inequality is known as Jensen's inequality and can be proven using the concavity of the log function. If we define the *energy* $\mathcal{E}$ of a global configuration (H,O) to be $-logP(H,O|\theta)$, then $\mathcal{F}$ in equation 4.62 is what is known in statistical physics as the *free energy*: $\mathcal{F}(Q,\theta) =< \mathcal{E}(\mathcal{Q},\theta) >_p -\mathcal{H}(\mathcal{Q})$, i.e. the expected energy under Q minus the entropy of Q [162]. The EM algorithm alternates between maximizing $\mathcal{F}$ with respect to Q and θ respectively, while holding the other fixed. Starting from some initial parameters θ_0:

E step: Calculate the probabilities of the hidden variables H given the observed variables O. This is the inference problem.

$$Q_{k+1} \leftarrow \underset{Q}{\text{argmax}} \ \mathcal{F}(Q,\theta_k) \tag{4.63}$$

M step: Parameter estimation for a fully observed graph, assuming that the hidden variables H have the values estimated in the E step.

$$\theta_{k+1} \leftarrow \underset{\theta}{\text{argmax}} \ \mathcal{F}(Q_{k+1},\theta) \tag{4.64}$$

The maximum in the E step results when $Q_{k+1}(H) = P(H|O,\theta_k)$, at which points the bound becomes an equality: $\mathcal{F}(Q_{k+1},\theta_k) = \mathcal{L}(\theta_k)$. The maximum in the M step is obtained by maximizing the expected energy under Q in expression 4.61, because the entropy $H(Q)$ does not depend on the parameters θ. Therefore the M step can be written as:

M step:

$$\theta_{k+1} \leftarrow \underset{\theta}{\text{argmax}} \ \sum_H P(H|O,\theta_k) logP(H,O|\theta) \tag{4.65}$$

This is the expression most often associated with the EM algorithm. The EM algorithm performs a coordinate ascent in $\mathcal{F}$. Since $\mathcal{F} = \mathcal{L}$ at the beginning of each M step and because the E step does not change the parameters θ, we are guaranteed not to decrease the likelihood after each combined EM step. An important aspect of the application of the EM algorithm to parameter estimation in PINs is that the JLO algorithm can be used to perform the E step.

Usually it is not necessary to evaluate the posterior distribution $P(H|O,\theta_k)$. Since the joint distribution $P(H,O|\theta_k)$ contains both hidden and observed variables in the graph, it can be factored as before as the sum of log probabilities of each node given its parents. Consequently, the quantities required for the M step are the *expected values under the posterior distribution* $P(H|O,\theta_k)$ of the analogous quantities required for ML estimation in the complete data case.

EM for PINs Without loss of generality and for illustration purposes, let us consider the case when all variables are discrete. Let x^k and $pa(X)^j$ denote the kth state of variable X and the jth state of variables $pa(X)$, respectively. Let assume that we have a directed PIN model M with mutually independent parameters $\theta = \bigcup_{jk}\{\theta_{H_{jk}}, \theta_{O_{jk}}\}$, where $\theta_{H_{jk}} = p(h_i^k|pa(H_i)^j, M)$ and $\theta_{O_{jk}} = p(o_i^k|pa(O_i)^j, M)\ \forall i$. In addition, let assume that we have observed data $D = \{e_1, e_2, \ldots, e_T\}$, an (iid) random sample from the true distribution.

The EM algorithm finds a local maximum of the likelihood $p(D|\theta, M)$ by initializing the parameters θ (randomly, via some clustering algorithm or exploiting prior knowledge) and iterating between the E and the M steps.

E step: Compute the sufficient statistic for each of the parameters, given the data D and the current values of θ. Let $S_{H_{jk}}$ be the sufficient statistic for $\theta_{H_{jk}}$. The expected sufficient statistic $E(S_{H_{jk}}|D,\theta,M)$ is given by:

$$E(S_{H_{jk}}|D,\theta,M) = \sum_{l=1}^{S}\sum_{i} p(h_i^k, pa(H_i)^j|e_l,\theta,M) \tag{4.66}$$

Note that each term in the sum can be computed using the JLO algorithm. The expected sufficient statistic for $\theta_{O_{jk}}$, $S_{O_{jk}}$ can be computed similarly.

M step: We use the expected sufficient statistics as if they were the actual sufficient statistics, and set the new values of the parameters θ to those that maximize the likelihood of these statistics:

$$\theta_{H_{jk}} = \frac{E(S_{H_{jk}}|D,\theta,M)}{\sum_l E(S_{H_{jl}}|D,\theta,M)} \qquad \theta_{O_{jk}} = \frac{E(S_{O_{jk}}|D,\theta,M)}{\sum_l E(S_{O_{jl}}|D,\theta,M)} \tag{4.67}$$

I would like to come back at this point to the three basic problems commonly addressed in statistical machine learning: (1) the inference problem, (2) the MAP state identification

problem, and (3) the parameter estimation and model learning problems. In the following sections I will describe the problem of parameter estimation in state-space models, Kalman filters and HMMs.

4.7.4 Parameter Estimation in State-space Models

Using equation 4.7 the log-likelihood, $\mathcal{L}$, can be written as:

$$\log P(O_t, H_t) = \log P(H_1) + \sum_{t=2}^{T} \log P(H_t|H_{t-1}) + \sum_{t=1}^{T} \log P(O_t|H_t) \tag{4.68}$$

Because each of the probability densities is Gaussian the overall expression is a sum of quadratic forms. For example,

$$\log P(O_t|H_t) = -1/2(O_t - CH_t)'R^{-1}(O_t - CH_t) - 1/2|R| + const \tag{4.69}$$

where R is the covariance of the observation noise v_t; ' denotes the matrix transpose, and $|\cdot|$ is the matrix determinant.

Due to the hidden nodes, we cannot directly compute the ML parameters. Applying the EM algorithm, in the E step we compute the expected values of some quantity $f(H)$ with respect to the posterior $P(H|O, \theta_k)$:

$$< f(H) >= \int_H f(H)P(H|O, \theta_k)dH \tag{4.70}$$

Finally the M step maximizes 4.68 with respect to the parameters. For example, in the case of the matrix C, the ML estimate would be:

$$C \leftarrow (\sum_t O_t < H_t >')(\sum_t < H_t H_t' >)^{-1} \tag{4.71}$$

Similar M steps can be derived for all the other parameters by taking derivatives of the expected log probability [80]. The expected terms $< H_t >$, $< H_t, H_t' >$ and $< H_t, H_{t-1}' >$ can be computed by Kalman smoothing, described in the following section.

4.7.5 Kalman smoothing

In a similar derivation as the one for HMM(1,1), the Forward-Backward recursion of the traditional Kalman smoother can be interpreted as a particular case of the JLO inference algorithm. The Kalman smoother solves the problem of estimating the state at time t of a linear-Gaussian state-space model given the model parameters and a sequence of observations $O = O_1, O_2, \ldots, O_T$. It consists of two parts: the forward recursion, where the message passing flow goes from "left to right", i.e. from H_t to H_{t+1}. It is known as the *Kalman Filter* [117]; and the backward recursion, with the messages flowing from "right to left", i.e. from H_{t+1} to H_t [202].

The Gaussian marginal density of the hidden state nodes is completely specified by its mean and covariance matrix. We define $< H_t^\tau >$ and $V_t^\tau =< H_t^\tau, H_t{}^{\tau\prime} >$ as the mean and covariance of the hidden Gaussian state node H_t, respectively, given observations $O = O_1, O_2, \ldots, O_\tau$. The Kalman filter, thus, (forward or "left to right" recursion) consists of the following computations:

$$< H_t^{t-1} > = A < H_{t-1}^{t-1} > \tag{4.72}$$

$$V_t^{t-1} = AV_{t-1}^{t-1}A' + Q \tag{4.73}$$

$$K_t = V_t^{t-1}C'(CV_t^{t-1}C' + R)^{-1} \tag{4.74}$$

$$< H_t^t > = < H_t^{t-1} > + K_t(O_t - C < H_t^{t-1} >) \tag{4.75}$$

$$V_t^t = V_t^{t-1} - K_t C V_t^{t-1} \tag{4.76}$$

with $< H_1^0 >$ and V_1^0 the prior (initial) mean and covariance of the hidden state. Equations 4.72 and 4.73 define the forward recursion (collect step in the JLO algorithm) of the state mean and covariance before having entered evidence for the observed node at time t. The mean evolves according to the dynamics given by the matrix A –which is the equivalent to the transition probability matrix in HMM(1,1)–. In the case of the Kalman Filter, however, the state dynamics is assumed to be known. The matrix A also affects the variance, which increases with the variance Q of the state noise. The observation O_t shifts the mean by an amount proportional to the prediction error $O_t - C < H_t^{t-1} >$, where the proportionality term K_t is known as the Kalman Gain Matrix. Note that introducing the evidence at time t, O_t, has the effect of effectively reducing the hidden state variance V_t^t.

These equations are direct results of the JLO algorithm when the integral Gaussians are analytically evaluated in the message passing algorithm.

Similarly, once evidence is introduced on a "left to right" schedule, one obtains $< H_T^T >$ and V_T^T. The backward step (distribute step in the JLO algorithm) proceeds on a "right to left" manner, evaluating the influence of future observations on the estimates of the hidden states in the past:

$$J_t = V_t^t A'(V_{t+1}^t)^{-1} \tag{4.77}$$

$$< H_t^T > = < H_t^t > + J_t(< H_{t+1}^T > - A < H_t^t >) \tag{4.78}$$

$$V_t^T = V_t^T + J_t(V_{t+1}^T - V_{t+1}^t)J_t' \tag{4.79}$$

where J_t is the backward matrix gain, with a similar role to the Kalman gain matrix. Again the estimated hidden state mean $< H_t^T >$ is shifted by a quantity proportional to the backward prediction error $< H_{t+1}^T > - A < H_t^t >$.

The expectations required for applying the EM algorithm are given by:

$$< H_t > = < H_t^T > \tag{4.80}$$

$$< H_t H_t' > = < H_t^T >< H_t^{T'} > + V_t^T \tag{4.81}$$

$$< H_t H_{t-1}' > = < H_t^T >< H_{t-1}^{T'} > + V_{t,t-1}^T \tag{4.82}$$

where $V_{t,t-1}^T = V_t^t J_{t-1}' + J_t(V_{t+1,t}^T - AV_t^t)J_{t-1}'$.

4.7.6 Parameter Estimation in HMMs

The log likelihood in an HMM(1,1) is given by

$$\log P(O_t, H_t) = \log P(H_1) + \sum_{t=2}^{T} \log P(H_t|H_{t-1}) + \sum_{t=1}^{T} \log P(O_t|H_t) \tag{4.83}$$

The hidden state nodes are usually discrete K-valued variables. We can represent them as K-dimensional unit column vectors, such that the state at time t taking on the value $'3'$ is represented as $H_t = [001 \ldots 0]'$. Each of the terms in 4.83 can be decomposed into

summations over H. The transition probability is $P(H_t|H_{t-1})$, given by

$$P(H_t|H_{t-1}) = \prod_{i=1}^{K}\prod_{j=1}^{K}(P_{ij})^{H_{t,i}H_{t-1,j}} \tag{4.84}$$

where P_{ij} is the hidden state transition probability, i.e. the probability of transitioning from state j to state i, arranged in a $K \times K$ matrix P. Then the term $\log P(H_t|H_{t-1})$ becomes:

$$\begin{aligned} \log P(H_t|H_{t-1}) &= \prod_{i=1}^{K}\prod_{j=1}^{K} H_{t,i}H_{t-1,j}\log P_{ij} &(4.85)\\ &= H_t'(\log P)H_{t-1} &(4.86)\end{aligned}$$

using matrix notation. The probability of the hidden states at time $t = 1$ is usually given by a vector of initial –prior– state probabilities π, such that

$$\log P(H_1) = H_1' \log \pi \tag{4.87}$$

Finally the $P(O_t|H_t)$ or *emission probabilities* depend on the form of observation:

1. **Discrete observations:** If O_t is a discrete variable taking D values, we represent it by D-dimensional unit vectors to obtain:

$$logP(O_t|H_t) = O_t'(logE)H_t \tag{4.88}$$

   where E is a $D \times K$ emission probability matrix: each column contains the probability of each hidden state producing each of the possible discrete output values, i.e. $e_{ij} = P(o_t = i|h_t = j)$, with $1 \leq i \leq D$ and $1 \leq j \leq K$.

2. **Gaussian observations:** In this case O_t is a continuous variable drawn from a finite mixture of Gaussians:

$$P(O_t|H_t = j, \mu_j, \Sigma_j) = \sum_{m=1}^{M} c_{jm}\mathcal{N}(O, \mu_{jm}, \Sigma_{jm}) \tag{4.89}$$

   where $1 \leq j \leq K$, $\mathcal{N}(O, \mu_{jm}, \Sigma_{jm})$ is a Normal Gaussian distribution (or any other log concave or elliptically symmetric density) with mean vector μ_{jm} and covariance matrix Σ_{jm}, and c_{jm} is the mixture coefficient or weight for hidden state j. The

mixture coefficients satisfy the stochastic constraint:

$$\sum_{m=1}^{M} c_{jm} = 1 \qquad 1 \leq j \leq K \tag{4.90}$$

$$c_{jm} \leq 0 \qquad 1 \leq j \leq K,\ 1 \leq m \leq M \tag{4.91}$$

In this case we have to estimate the mixture parameters, i.e. the mixing coefficients, the means and the covariance matrices.

Since the state variables are not observed we cannot compute 4.83 directly. We have to "fill-in" the incomplete data using the JLO algorithm –equivalent, as we have seen, to Baum-Welch, Forward-Backward and E step in the EM algorithm–. It will compute the expectation of 4.83 under the posterior distribution of the hidden state nodes given the observations, i.e. $P(H_t|O_t)$. This expectation can be expressed as a function of $< H_t >$ and $< H_t, H'_{t+1} >$ $(1 \leq t \leq T)$. The first term $< H_t >$ is a vector where each of its elements is $P(h_t = j|O, \theta)$, i.e. the probability of being in state j at time t given the entire sequence of observations O and the model parameters θ. Traditionally this term corresponds to $\gamma_t(j)$ in the HMM literature. The second term, $< H_t, H'_{t+1} >$, is a matrix of the joint probability of successive states, i.e. $P(H_t, H_{t+1})$ given the observation sequence O and the model parameters θ. Each element of this matrix, thus, is $\xi_t(ij) = P(h_t = i, h_{t+1} = j|O, \lambda)$ and the matrix is traditionally denoted by ξ_t.

The JLO algorithm (Baum-Welch, Forward-Backward or E-step) yields $\alpha_t(j) = f^*_{j,\Phi_{1,t}}$ and $\beta_t(j) = \lambda^*_{\Phi_{t+1,T}}(h_t = s_j)$. The previous expectations can be expressed in terms of α and β as follows:

$$< H_t = j >=< H_{t,j} >= \gamma_t(j) \quad = \quad \frac{\alpha_t(j)\beta_t(j)}{\sum_k \alpha_t(k)\beta_t(k)} \tag{4.92}$$

$$< H_t = i, H'_{t+1} = j >= \xi_t(ij) \quad = \quad \frac{\alpha_t(i)P_{ij}P(O_{t+1}|H_t = i)\beta_{t+1}(j)}{\sum_{k,l} \alpha_t(k)P_{kl}P(O_{t+1}|H_t = l)\beta_{t+1}(l)} \tag{4.93}$$

Once the expectations are computed, the parameters are estimated by maximizing the log likelihood 4.83 (M step): take the derivative of the log likelihood with respect to the parameters θ, set to zero, and solve subject to the sum-to-one constraints that ensure stochastic transition, emission and initial state probability matrices.

Transition Probability Matrix: It is the expected number of transitions from state i

to state j, divided by the total expected number of transitions from state i, i.e.

$$P_{ij} = \frac{\sum_{t=1}^{T-1} < H_t = i, H'_{t+1} = j >}{\sum_{t=1}^{T-1} < H_{t+1} = j >} = \frac{\sum_{t=1}^{T-1} \xi_t(ij)}{\sum_{t=1}^{T-1} \gamma_t(j)} \tag{4.94}$$

Emission Probability in Discrete Case: It is the expected number of times in state j and observing output symbol k, divided by the total expected number of times in state j:

$$P(O_t = k | H_t = j) = \frac{\sum_{t=1,\ s.t.\ O_t=k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{4.95}$$

Emission Probability in Continuous Case: Similarly, the mixture sufficient statistics are given by:

$$< c_{jk} > = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{l=1}^{M} \gamma_t(j,l)} \tag{4.96}$$

$$< \mu_{jk} > = \frac{\sum_{t=1}^{T} \gamma_t(j,k) O_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{4.97}$$

$$< \Sigma_{jk} > = \frac{\sum_{t=1}^{T} \gamma_t(j,k)(O_t - \mu_{jk})(O_t - \mu_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{4.98}$$

where $\gamma_t(j,k)$ is the probability of being in state j at time t with the kth mixture component accounting for O_t, i.e.

$$\gamma_t(j,k) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{l=1}^{K} \alpha_t(l)\beta_t(l)} \frac{c_{jk}\mathcal{N}(O_t, \mu_{jk}\Sigma_{jk})}{\sum_{m=1}^{M} c_{jm}\mathcal{N}(O_t, \mu_{jm}\Sigma_{jm})} \tag{4.99}$$

The reestimation formula for c_{jk} is the ratio between the expected number of times the system is in state j using the kth mixture component and the expected number of times the system is in state j. The mean vector μ_{jk} is the expected value of the portion of the observation vector accounted for by the kth mixture component. Similarly the covariance matrix Σ_{jk} is the expected value of the portion of the covariance matrix accounted for by the kth mixture component.

Finally, the MAP state identification problem is the last problem that needs to be addressed from the three basic problems that are commonly treated in statistical machine learning. The next section describes an efficient algorithm for MAP state assignment in graphical models.

4.8 MAP State Assignment via the Viterbi Algorithm or Dawid's Propagation Algorithm

The MAP state identification problem consists of the determination of the most likely state of a set of unobserved variables, given observed variables and the probabilistic model. The goal, thus, is to calculate $\hat{f}(u^h, e) = max_{h_1,\ldots,h_N} p(h_1, \ldots, h_N, e)$, where one would like to identify a set of values of the N hidden variables H_i which achieve this maximum. This calculation can be achieved via a *local propagation algorithm* on the JT if one makes two modifications to the standard JLO inference algorithm. This modified version of the JLO algorithm is due to Dawid [56] and it contains the well known Viterbi algorithm as a special case. The modifications are:

1. First, during a message passing flow, the marginalization of the separator is replaced by the maximum:

$$\hat{b}_S(h_S) = \max_{C \setminus S} \; a_C(x_C) \tag{4.100}$$

   where C is the originating clique of the flow and $x_C = x_C^h, x_C^o$ are all the variables (observed and hidden) in clique C. The definition of $\lambda_S(h_S)$ is similarly changed.

2. Second, marginalization within a clique is replaced by maximization:

$$\hat{f}_C = \max_{u \setminus x_C} \; p(u) \tag{4.101}$$

Given these two changes it can be shown that if the same evidence propagation operations are carried out as described for the JLO algorithm, the resulting representation $\hat{K}_f$ at equilibrium is such that the potential function on each clique C is

$$\hat{f}(x_C) = \max_{u^h \setminus x_C} \; p(x_C^h, e, \{u^h \setminus x_C\}) \tag{4.102}$$

where x_C^h corresponds to a value assignment of the hidden variables H in clique C. Once the new representation is obtained one can locally identify the values X_C^h which maximize the full joint probability as

$$\hat{x}_C^h = arg_{x_C^h} \hat{f}(x_C) \tag{4.103}$$

This is known in the probabilistic expert systems literature as the "most probable explanation" (MPE), given the observed evidence.

The HMM(1,1) MAP problem consists of inferring

$$\max_{h_1,\ldots,h_N} p(h_1,\ldots,h_N,e) \tag{4.104}$$

given a set of values for the observable variables, $e = O_1 = o_1, O_2 = o_2, \ldots, O_k = o_K$; or, equivalently, inferring the set of arguments that achieve this maximum, i.e. the *single* best state sequence (path) that maximizes the posterior probability of the hidden states given the model and the evidence (or equivalently the joint probability). Dawid's algorithm can be applied to any junction tree and therefore can be applied to the HMM(1,1) junction tree. The Viterbi algorithm, based on dynamic programming methods, is the traditional procedure for solving the MAP problem in HMM(1,1). It can be easily shown that Dawid's algorithm is a generalization of the Viterbi algorithm (see figure 4-13, (a)). In particular, for an HMM(1,1), from a computational viewpoint, instead of searching through all N^T possible paths for an HMM(1,1) with N hidden states, the Viterbi algorithm provides a recursive $O(TN^2)$ procedure: given the most likely incomplete path $H_{j,t-1}|O$ leading up to each state j at time $t-1$, the most likely path leading to each state i at time t is

$$\hat{H}_t = i|O = \text{argmax } p(O_t|H_t = i) \cdot P_{ij} \cdot \hat{H}_{t-1} = j|O \tag{4.105}$$

Therefore each state elects to continue one of the N best paths from the previous time slice. The most likely full state sequence is then $\hat{H}|O = \text{argmax}_i(\hat{H}_T = i|O)$.

4.9 Discussion

So far I have shown the equivalence between the JLO algorithm and the Forward-Backward, Baum-Welch or EM algorithms for doing inference in graphical models. I have also shown the equivalence between Dawid's and the Viterbi algorithms. Both are direct applications of dynamic programming to the MAP problem. The most important conclusion is that graphical models are more general than the specific algorithms developed for HMM(1,1): (1) while special extensions of the Baum-Welch and Viterbi algorithms can be defined with no little effort, the JLO algorithm provides, by definition, a completely exact inference

method for any PIN. (2) One can use the graphical model algorithms for any other inference tasks beyond just calculating the likelihood or the evidence or the MAP solution. Graphical models let us, in a quite simple way, deal with missing or probabilistic evidence, simulating values from the model, or calculating partial solutions. The remaining of this chapter describes extensions of the traditional $HMM(1,1)$. I will first motivate the need for extensions of HMMs. Then I will present an overview of the most relevant extensions to $HMM(1,1)$ that have been proposed in the literature. Finally I will describe in detail the specific architecture used in this thesis: Coupled Hidden Markov Models (CHMMs) or $HMM(1,2)$.

4.10 PINs for extensions of HMM(1,1)

Although HMM(1,1) have provided an extremely useful framework for modeling time series (specially speech), it is also true that a single standard HMM(1,1) has strong limitations as a model of some real complex phenomenon or behavior. Among the major limitations of HMM(1,1) one finds:

- Many real signals are generated by coupled physical processes that are not well modeled by the unstructured state transition matrix of HMM(1,1).

- The first-order Markov properties of HMM(1,1) are not well suited to modeling many long term dependencies that occur in real life, such as co-articulation effects in speech that extend across several phonemes.

- The representation of context is limited to a single state variable. Therefore to accommodate multiple channels of data, an HMM(1,1) would have to be formulated with multivariate Normal distributions on the output variables. However, if there are multiple processes generating those signals (channels), one must hope that these processes evolve in a synchronized manner, since any variation between them would be modeled as noise. If the processes interact and therefore such variation carries information, HMM(1,1) might be an inappropriate graphical structure, because with context limited to a single variable, a distinct state must be reserved for each possible combination of signals on all the channels. It is easy to see that this Cartesian product solution rapidly becomes intractable. Other extensions of the simple HMM(1,1)

aimed at multiple channel modeling include using neural networks for outputs [15], input-output models [21] and per-state mixture models. However all of them are based on the traditional Markov formulation of *single process dynamics*.

Signals from systems with multiple processes are ubiquitous in the real world. Nearly any signal produced by human behavior can be usefully and meaningfully decomposed into a group of interacting processes. For such problems, it seems more appropriate to extend the single process traditional HMM(1,1) structure to handle multiple state variables. The naive solution, i.e. an HMM(1,1) with the Cartesian product of all possible process states is rarely satisfactory: the computational cost is prohibitive, a surfeit of parameters leads to overfitting, and there is often insufficient data for a large number of states, leading to undersampling and numerical errors. Reducing the state space ameliorates the problem somewhat but introduces new problems of underfitting where states that should be distinct are fused. In either case, the interactive processes are only implicitely represented in the model in an obscure manner. In consequence, HMM(1,1), even with the correct number of states and huge amounts of data, may train poorly because the data might be partitioned among the states too early and usually incorrectly during training. Because of the Markov independence assumption, the data is not shared among the states, thus reinforcing the mistakes in the initial partitioning. Moreover systems with multiple generating processes could produce "state perceptual aliasing": there might be states that share properties and therefore emit similar signals generating ambiguity in the state identification.

Models with compositional state representations would offer conceptual advantages of parsimony and clarity, with consequent computational benefits in efficiency and accuracy. Using graphical models one can construct various architectures for multi-HMM couplings offering compositional state under various assumptions of independence. The role of graphical modeling is key in at least two ways: (1) first, it provides a concise description of the probabilistic independence assumed by a particular model, and (2) second, it provides a general algorithm, the JLO algorithm, for doing inference and ML estimation.

4.10.1 Beyond Tractable Models

The rest of this chapter focusses on extensions of the traditional HMM(1,1) graph structure. For notation purposes, I will denote in the following by S_t –instead of H_t – the hidden state variable in the extended HMM structure and by O_t the observed variables.

The traditional, single HMM(1,1) framework can be extended in different ways. A basic taxonomy classifies the couplings depending on whether the outputs, the states or both are coupled.

1. **Coupling the outputs:** The first option is to *couple the outputs* of several independent hidden state Markov chains, i.e. the processes are nominally coupled at the output, superimposing their outputs in a single signal (see figure 4-15, (a)). These models are called *factorial hidden Markov models*, FHMMs by Gharahmani and Jordan ([82]). The hidden state, therefore, is factored into C distinct state variables $H_t = S_t = \{S_t^{(1)}, \ldots, S_t^{(c)}, \ldots, S_t^{(C)}\}$, each of which can take on $K^{(c)}$ values. For simplicity and without loss of generality I will assume that $K^{(c)} = K \ 1 \leq c \leq C$, i.e. all the Markov processes have the same number of states. The state space of this model is represented in a distributed manner and consists of the cross product of these state variables. FHMMs correspond to HMM(1,K) structure.

   With a distributed state space, FHMMs allow the state space to be decomposed into features that naturally decouple the dynamics of the process generating the time series. Moreover distributed state representations simplify the task of modeling time series generated by the interaction of multiple independent processes. For example, the traditional source identification problem, where signals with zero mutual information are overlaid in a single channel, e.g. a speech signal generated by the superposition of multiple simultaneous unrelated speakers.

   FHMMs are closely related to the work in unsupervised learning directed to discovering multiple independent causes or factors underlying the data [16], [93]. The motivation behind factorial learning algorithms is that many real world learning problems are best characterized by an interaction of multiple independent causes or factors. The goal of factorial learning is to invert the data generation process and discover a representation that will both parsimoniously describe the data and reflect its underlying causes.

   Given that the state space of FHMMs consists of all K^C combinations of the $S_t^{(c)}$ variables, placing no constraints on the state transition structure would result in a $K^C \times K^C$ transition matrix. Such an unconstrained system is uninteresting for several reasons: (1) It is equivalent to the cross –cartesian– product HMM with K^C states;

(2) it is unlikely to discover any interesting structure in the K state variables as all variables are allowed to interact arbitrarily; (3) both time and space (sample) complexity of the algorithm are exponential in K. FHMMs constrain the underlying state transitions by assuming that each state variable transitions according to its own dynamics, and is *a priori* uncoupled from the other state variables:

$$P(S_t|S_{t-1}) = \prod_{c=1}^{C} P(S_t^{(c)}|S_{t-1}^{(c)}) \tag{4.106}$$

Let O_t be the observation vector of dimensionality D. The observation model in a FHMM is given by

$$P(O_t|S_t) = |R|^{-1/2}(2\pi)^{-D/2}\exp\{-1/2(O_t - \mu_t)'R^{-1}(O_t - \mu_t)\} \tag{4.107}$$

where

$$\mu_t = \sum_{c=1}^{C} W^{(c)} S_t^{(c)} \tag{4.108}$$

Each $W^{(c)}$ matrix is a $D \times K$ whose columns are the contributions to the means for each of the settings of the hidden states in the $(c)th$ HMM, $S_t^{(c)}$, and R is a $D \times D$ covariance matrix.

The marginal distribution for O_t is obtained by summing over all possible states. There are K settings for each of the C state variables. Thus there are K^C possible mean vectors obtained by forming sums of C columns where one column is chosen from each of the $W^{(c)}$ matrices. The resulting marginal density of O_t is a Gaussian mixture model, with K^C Gaussian mixture components each having a constant covariance matrix R. This static mixture model, just considering one time slice and ignoring the Markov dynamics, is a factorial parameterization of the standard mixture of Gaussians model [79].

The transition structure for the FHMM model can be parametrized using C distinct $K \times K$ matrices. FHMM are tractable in space, taking KC states as opposed to K^C. However they present an inference problem equivalent to that of a combinatoric HMM. The naive exact algorithm, consisting of translating the FHMM into an equivalent HMM with K^M states and using the Baum-Welch algorithm, has time

complexity of $O(TK^{2C})$. Like in other models with multiple densely-connected hidden variables, this exponential time complexity makes exact learning and inference intractable. Thus, although the Markov property can be used to obtain Forward-Backward-like factorizations of the necessary expectations across time steps, the sum over *all possible configurations* of the other hidden state variables within each time step is unavoidable. This intractability is due to the mixture nature of the observations: the setting of one state variable only determines the mean of the observation if all other state variables are fixed.

Rather than computing the exact posterior probabilities, one can approximate them using a Markov Chain Monte Carlo sampling scheme, and therefore avoid the sum over exponentially many state patterns at some cost in accuracy. In [82] Ghahramani and Jordan introduce several approximations to the inference problem in FHMMs: Gibbs sampling and mean field theory from statistical physics [268], [177]. They also propose a "structured mean field" solution in which the mean field graph is partitioned into subgraphs that can be tractably estimated via exact methods (i.e. Forward-Backward analysis of the independent, decoupled HMMs). A free energy function is defined over the entire graph. To prevent exponential growth in the number of parameters, rather than specifying higher-order couplings through probability transition matrices, they introduce second-order interaction terms in the energy (log probability) function. Such terms effectively couple the chains in a more efficient way, with much fewer parameters than a full probability transition matrix would require. In the graphical model formalism these correspond to *symmetric undirected links*, making the model like a chain graph. The JLO algorithm can still be used to propagate evidence exactly in chain graphs. However such undirected links cause the normalization constant of the probability distribution –the *partition function*– to depend on the coupling parameters. Therefore, like in Boltzmann machines [3], both clamped and unclamped phase are required for learning, where the goal of the unclamped phase is to compute the derivative of the partition function with respect to the parameters [161]. The mean field approximation can be used to train virtually any elaboration on HMM structure in $O(TCK^2)$ time. However, if there are strong and varied interactions across the links that have been removed, the approximation will be quite poor.

2. **Coupling the states:** Conventional HMMs excel for processes that evolve in lockstep; FHMMs are meant for processes that evolve independently. However, many problems tend to lie between the two extremes. Two generating processes might interact without wholly determining each other. Each process has its own internal dynamic and its affected by what the other processes do, possibly in a casual manner. Moreover, the inter-process coupling might be stronger or weaker depending on each state. A variety of inference graph structures have been proposed to model these phenomena (see figure 4-15).

   Tree structured HMMs: MHDT An interesting generalization of FHMMs results if one conditions on an input O_t and orders the C state variables such that $S_t^{(c)}$ depends on $S_t^{(l)}$, for $1 \leq l \leq c$ (figure 4-15 (b)). The resulting architecture is known as hidden Markov decision tree (HMDT) [83]. This architecture can be interpreted as a probabilistic decision tree with Markovian dynamics linking the decision variables. HMDTs provide a useful starting point for modeling time series with both temporal and spatial structure at multiple resolutions [115].

   Linked HMMs: LHMMs In [216] two parallel Boltzmann chains are coupled by weights that connect their hidden state nodes. Saul and Jordan propose couplings between synchronous states for chains that evolve in lockstep at the same rate or with disparate time scales. We may call such inference graph Linked HMMs (LHMMs), depicted in figure 4-15 (c). The resulting network for 2 coupled chains is tractable. They present an $O(TK^3)$ exact algorithm for training an equivalent two-chain Boltzmann machine based on *decimation*, a method from statistical mechanics in which the marginal distributions of singly or doubly connected nodes are integrated out. A limited class of graph structures can be recursively decimated, obtaining correlations for any connected pair of nodes.

   Coupled HMMs: CHMMs Finally, the graphical model structure used in this thesis captures *causal –temporal–* influences of one chain on the other. To capture interprocess influences across time, the coupling must bridge time slices, as depicted in figure 4-15 (d). Intuitively and empirically (as the results of this thesis show), it is appropriate for processes that influence each other symmetrically and possibly causally. This architecture is known as Coupled HMMs, CHMMs.

The graphical structure of a C processes CHMM corresponds to a HMM(1,C). Inference in CHMMs has the same time complexity as the Cartesian product HMM. Therefore exact algorithms are not attractive. Given a HMM(1,2), i.e. a 2-chain CHMM, exact MAP inference is $O(TK^4)$ [108]. Figure 4-16 depicts the junction tree for a 2-chain CHMM. Note that the hidden state cliques are of size 4. Sampling methods improve over random sampling by discarding, weighting and/or varying sample state sequences according to their posteriors [92, 72, 119]. These algorithms are consistent, i.e. they converge asymptotically to the true distribution. However it is not known whether they are efficient, i.e. they produce the best estimates given the order of computation.

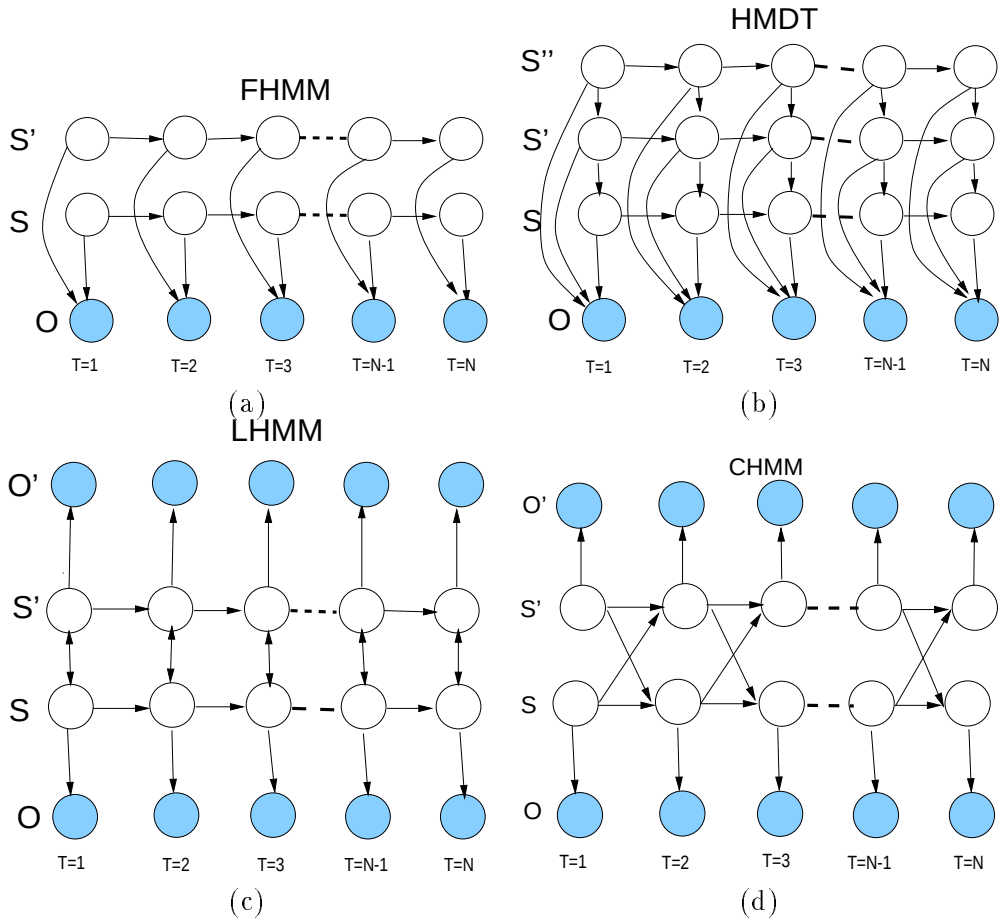


Figure 4-15: Variety of couplings for dependent processes: (a) FHMM, (b) HMDT, (c) LHMM and (d) CHMM

All these examples suggest that the graphical modeling framework provides a useful

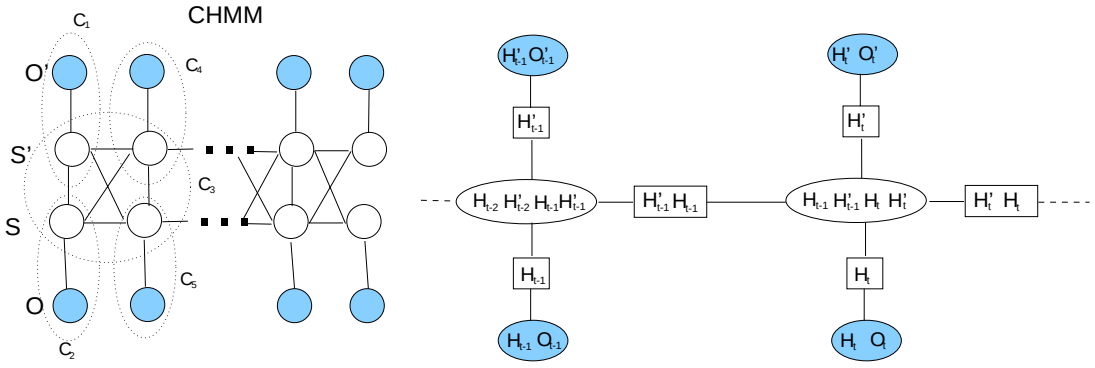


Figure 4-16: (a)Triangulated graph for a 2-chain CHMM with the cliques. (b)Junction Tree for a 2-chain CHMM. Note that the clique size for the hidden state variables is 4. Exact MAP inference in such junction tree is $O(TK^4)$

framework for exploring extensions of HMMs. The examples also make clear, however, that for many interesting graphical structures, the exact solution is computationally intractable. The K^C complexity of HMM(1,C) is prohibite for large C. In the next section I will develop the theory behind the new extension of HMM(1,1) used in thie thesis work, namely *Coupled Hidden Markov Models*, CHMMs or HMM(1,C).

4.11 Coupled Hidden Markov Model: CHMMs or HMM(1,C)

As it has been presented in the previous section, there are some major limitations in the standard single process HMM, HMM(1,1), when dealing with real signals: (1) the unstructured state transition matrix of HMM(1,1) does not seem to be appropriate for modeling signals generated by coupled physical processes; and (2) the representation of context is limited to a single state variable. Various extensions of the simple HMM(1,1) have been proposed in the literature. In this thesis, I use and experimentally validate a new structure, namely Coupled Hidden Markov Models, HMM(1,C) or CHMMs, for causally coupling two or more HMMs.

In this section I describe a deterministic $O(TK^2)$ algorithm for maximum entropy approximations to state and parameter values in CHMMs. Part of the material presented here can be found in [28].

The algorithm is not limited to 2-chain CHMMs (HMM(1,2)) but arbitrary C-chain

CHMMs (HMM(1,C)). However, I will describe in detail the algorithm for 2-chain CHMMs, being this the structure employed in the validation experiments described in chapter 5.

Observing the graph structure depicted in figure 4-15 (c), the posterior hidden state probability for a 2-chain CHMM is given by:

$$P(S|O) = \frac{P_{s_1} p(o_1|s_1) P_{s'_1} p(o'_1|s'_1)}{P(O)} \prod_{t=2}^{T} P_{s_t|s_{t-1}} P_{s'_t|s'_{t-1}} P_{s'_t|s_{t-1}} P_{s_t|s'_{t-1}} p(o_t|s_t) p(o'_t|s'_t)$$

where $P_{s_t|s_{t-1}}$, $P_{s'_t|s'_{t-1}}$ are the "intra-state" transition probability matrices, $P_{s_t|s'_{t-1}}$, $P_{s'_t|s_{t-1}}$ are the "inter-state" or coupling transition probability matrices and $p(o_t|s_t)$, $p(o'_1|s'_1)$ are the observation probabilities.

4.11.1 N-heads dynamic programming

As we have seen in section 4.6 the Forward step in the Forward-Backward algorithm of traditional HMMs (or the collect step of the JLO algorithm) exploits dynamic programming to efficiently do inference (model likelihood) and estimate the most likely hidden state sequence (Viterbi analysis). Dynamic programming provides a method for collecting statistics on an exponential number of possible paths through an HMM state trellis in polynomial time, because evidence from all paths that share a state at time t may be combined without loss of information. Therefore, instead of having to track K^T paths in an HMM with K hidden states and T time steps, only K path "heads" are tracked. Every state and every transition at every time slice needs to be visited to collect statistics for estimation. Therefore dynamic programming in a state trellis of length T and K states requires $O(TK^2)$ time.

In the case of a CHMM with C chains, the joint state trellis has K^C states. The associated dynamic programming problem is $O(TK^{2C})$. In this section an algorithm is presented that relaxes the assumption that every transition needs to be visited. The resulting algorithm is $O(T(CK)^2)$ while closely approximating the full combinatoric result.

The goal is to formulate a policy for sampling a small and representative subset of the K^{CT} state sequences while obtaining as much information as possible. The posterior probability mass of an HMM is not evenly distributed among all possible state sequences. It is, by definition, concentrated in state sequences that are close to the "true" or MAP state sequence. Low-probability sequences carry relatively little information for estimation

problems.

Let O be a sequence of observed variables. The posterior of a model M, $P(M|O)$ is the sum of the posteriors of all possible state paths $S^{\{\}}$ through the model. Let $< P(M|O) >_{S^{\{\}}}$ be the expectation the posterior in each time slice with respect to all possible state sequence paths. If only a subset, $S^{\{Q\}}$, of all possible paths is explored, the question is to find the optimal path selection policy such that the least information is lost. This problem can be cast as a minimization of the cross entropy between the true posterior and the simplified posterior:

$$\begin{aligned} D(S^{\{\}} \| S^{\{Q\}}) &= \sum_O \langle P(M|O) \rangle_{S^{\{\}}} \log \frac{\langle P(M|O) \rangle_{S^{\{\}}}}{\langle P(M|O) \rangle_{S^{\{Q\}}}} && (4.109) \\ &= \sum_O \langle P(M|O) \rangle_{S^{\{\}}} \log \langle P(M|O) \rangle_{S^{\{\}}} && (4.110) \\ &\quad - \sum_O \langle P(M|O) \rangle_{S^{\{\}}} \log \langle P(M|O) \rangle_{S^{\{Q\}}} && (4.111) \\ &= f(H(O)) - \sum_O \langle P(M|O) \rangle_{S^{\{\}}} \log \langle P(M|O) \rangle_{S^{\{Q\}}} && (4.112) \end{aligned}$$

The first term is a monotonic function of the full posterior entropy and thus constant with respect to the "reduced" posterior, $\langle P(M|O) \rangle_{S^{\{Q\}}}$. Assuming that all observed sequences O are equally probable, the first expectation in equation 4.112 can be also treated as a constant and be pulled out of the sum:

$$D(S^{\{\}} || S^{\{Q\}}) = f(H(O)) - c \sum_O \log \langle P(M|O) \rangle_{S^{\{Q\}}} \quad (4.113)$$

Therefore, minimizing the cross entropy 4.113 is equivalent to maximizing the expectation, $\log \langle P(M|O) \rangle_{S^{\{Q\}}} \leq \langle \log P(M|O) \rangle_{S^{\{Q\}}}$, by virtue of Jensen's inequality. This expectation will be maximum when $S^{\{Q\}}$ is the set of paths with the greatest probability mass. Instead of directly maximizing $\log \langle P(M|O) \rangle_{S^{\{Q\}}}$, I will use the upper bound $\langle \log P(M|O) \rangle_{S^{\{Q\}}}$. In particular, for a 2-chain CHMM, equation 4.113 yields:

$$\begin{aligned} Q &= \underset{s \in S^{\{\}}}{\operatorname{argmax}}^{N} \sum_O \langle \log P(M|O) \rangle_s && (4.114) \\ &= \underset{s \in S^{\{\}}}{\operatorname{argmax}}^{N} \sum_O \left\langle \log P_{s_1} P_{s'_1} p(o_1|s_1) p(o'_1|s'_1) \prod_{t=2}^{T} P_{s_t|s_{t-1}} P_{s'_t|s'_{t-1}} P_{s_t|s'_{t-1}} P_{s'_t|s_{t-1}} p(o_t|s_t) p(o'_t|s'_t) \right\rangle_s \end{aligned}$$

$$= \underset{s \in S^{\{\}}}{\overset{N}{\operatorname{argmax}}} \sum_{O} \left\langle \begin{pmatrix} \log P_{s_1} + & \log P_{s'_1} + \\ \log p(o_1|s_1) + & \log p(o'_1|s'_1) \end{pmatrix} + \sum_{t=2}^{T} \begin{pmatrix} \log P_{s_t|s_{t-1}} + \log P_{s'_t|s'_{t-1}} + \\ \log P_{s_t|s'_{t-1}} + \log P_{s'_t|s_{t-1}} + \\ \log p(o_t|s_t) + \log p(o'_t|s'_t) \end{pmatrix} \right\rangle_{s} \quad (4.115)$$

Some desired properties of the procedure for finding these $S^{\{Q\}}$ paths in $O(TK^2C)$ time are: (1) No more than $O(K * C)$ path heads should be tracked, i.e. as many heads as the total number of states there are in all the Markov chains; (2) every component state should be visited so that statistics may be collected by re-estimating transition and output probabilities.

The policy can be visualized on the state trellises for the component HMMs in a 2-chain CHMM. Each state sequence i through the trellis is double tracked, having a head in one HMM $h_t(i)$ and an associated "sidekick" $k'_t(j)$ in the opposite HMM (see figure 4-17). In this case, a *path* is defined by the pair $\{h_t(i), k'_t(j)\}$. The *antecedent path* is the subsequence leading to a particular *path* at time t.

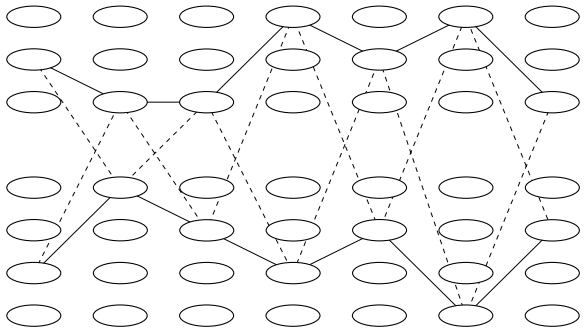


Figure 4-17: State trellises for a 2-chain 4-state CHMM

Every state i of each component chain c needs to be the head of some path to guarantee the visiting criterion, i.e. $h_t^c(i)$, $1 \leq i \leq K^c$, $1 \leq t \leq T$ and $1 \leq c \leq C$, in a C-chain CHMM of time length T and K^c states. Therefore coupling 2 HMMs with K and K' states takes $K + K'$ heads, each one with its own sidekick, $k'_t(j)$. The question is how to choose the sidekick for each head. Note that, for each component chain c, the left-hand column of each parenthesized expression in equation 4.115 is constant, because every state i in the chain has to be a head for some path. For example, for the first chain:

$$\text{constant} = \underset{s \in S\{\}}{\overset{N}{\text{argmax}}} \sum_{O} \left\langle \begin{pmatrix} \log P_{s_1} + \\ \log p(o_1|s_1) \end{pmatrix} + \sum_{t=2}^{T} \begin{pmatrix} \log P_{s_t|s_{t-1}} + \\ \log P_{s_t|s'_{t-1}} + \\ \log p(o_t|s_t) \end{pmatrix} \right\rangle_{s} \tag{4.116}$$

Consequently we only need to maximize the right-hand columns, which amounts to choosing the MAP sidekicks:

$$Q_1 = \underset{s \in S'\{\}}{\overset{N}{\text{argmax}}} \sum_{O} \left\langle \begin{pmatrix} \log P_{s'_1} + \\ \log p(o'_1|s'_1) \end{pmatrix} + \sum_{t=2}^{T} \begin{pmatrix} \log P_{s'_t|s'_{t-1}} + \\ \log P_{s'_t|s_{t-1}} + \\ \log p(o'_t|s'_t) \end{pmatrix} \right\rangle_{s} \tag{4.117}$$

By symmetry, the converse applies for the heads in the opposite chain. This policy defines a $O(K)$ algorithm for finding the MAP tuples in a 2-chain CHMM in $O(K^2)$ time by first associating MAP sidekicks to each antecedent path and then associating antecedent paths to each new head. This leads to approximate Forward-Backward and Viterbi algorithms described in the following sections.

4.11.2 Forward-Backward Algorithm for CHMMs

In each step of the forward analysis the MAP mass $\{h_t(i), k'_t(i)\}$ pair is needed given all antecedent paths. Every head $h_t(i)$, $1 \leq i \leq K$ sums over the same set of antecedent paths and therefore shares the same sidekick $k'_t(i) = k'_t$, $1 \leq i \leq K'$. As we have seen in section 4.11.1 the sidekick is chosen to maximize the marginal posterior given all antecedent paths. A two-step procedure for doing so without marginalizing is as follows: (1) *Sidekick selection:* in each chain, choose the MAP state given all antecedent paths. (2) *Compute path posterior:* for each head, $h_t(i)$, calculate the new path posterior given all antecedent paths and the previously chosen sidekick.

The approximate Forward-Backward algorithm for a 2-chain CHMM is described in the following. I denote heads and sidekick indices in each time slice t by $h_t(i), k'_t(j)$; $\alpha^*_t(i)$ is the probability mass associated with each head, $h_t(i)$; $q_t(i)$ is the partial posterior probability

(in the absence of sidekicks) of a state i given all $\alpha^*_{t-1}(j)$, $1 \leq j \leq K$ and the output at t. The maximizing policy selects the sidekick that maximizes $\langle q_t(k'_t(i))|\alpha^*_{t-1}(j)\rangle_i$.

1. Calculate all partial posteriors:

$$q_t(i) = p(o_t|i) \sum_j P_{i|h_{t-1}(j)} P_{i|k'_{t-1}(j)} \alpha^*_{t-1}(j) \tag{4.118}$$

2. Choose the sidekick from each chain:

$$k_t(i) = \operatorname*{argmax}_{i'} q_t(i') \tag{4.119}$$

3. Each state is its own head: $h_t(i) = i$.

4. Calculate full posteriors for each path:

$$\alpha^*_t(i) = p(o_t|i)p(o_t|k'_t(i)) \sum_j P_{i|h_{t-1}(j)} P_{i|k'_{t-1}(j)} P_{k'_t(i)|h_{t-1}(j)} P_{k'_t(i)|k'_{t-1}(j)} \alpha^*_{t-1}(j) \tag{4.120}$$

5. The forward variables in each chain are obtained by maginalizing out each head (over all possible sidekicks)

$$\begin{aligned} \alpha_t(i) &= p(o_t|i) \sum_j P_{i|h_{t-1}(j)} P_{i|k'_{t-1}(j)} \sum_g p(o_t|k'_t(g)) P_{k'_t(g)|h_{t-1}(j)} P_{k'_t(g)|k'_{t-1}(j)} \alpha^*_{t-1}(j) \\ &= p(o_t|i) \sum_j P_{i|h_{t-1}(j)} P_{i|k'_{t-1}(j)} \sum_g q_t(k'_t(g)) \end{aligned} \tag{4.121}$$

Therefore and in contrast to the conventional Forward-Backward procedure described in section 4.6, we have two different kinds of forward variables: α^* variables for propagating probabilities, and marginalized α variables for re-estimating parameters. Similarly, the backward variables

$$\beta^*_t(i) = \sum_j P_{h_{t+1}(j)|i} P_{k'_{t+1}(j)|i} P_{h_{t+1}(j)|k'_t(i)} P_{k_{t+1}(j)|k'_t(i)} p(o_{t+1}|i) p(o_{t+1}|k'_{t+1}(i)) \beta^*_{t+1}(j) \tag{4.122}$$

are computed using the sidekicks found in the forward analysis, and similarly marginalized to obtain the $\beta_t(i)$ variables.

Without increasing complexity, we can improve slightly over this greedy cross-entropy approximation by conditioning the choice of sidekicks in t on α^*_{t-1} *and* on sidekicks chosen in $t+1$, $k'_{t+1}(j)$. In practice, this causes the forward calculation to occasionally backtrack one time slice. Similarly, one may recalculate sidekicks in all t given α^*_{t-1} and β^*_{t+1}, then recalculate forward and backward variables. Although both schemes obtain slightly higher posteriors by expanding the temporal scale of the greedy method, neither has substantial impact on parameter estimation. Therefore, I used the basic algorithm in all the experiments carried out in this thesis.

4.11.3 Scaling

To prevent numerical overflow a scaling procedure is necessary, because the joint probabilities quickly become vanishingly small. Typically, a scaling variable $c_t = \sum_i \alpha^*_t(i)$ is used to normalize the forward variables ($\alpha^*_t(i) \leftarrow \alpha^*_t(i)/c_t$) on each iteration. The backward variables are rescaled using the same values ($\beta^*_t(i) \leftarrow \beta^*_t(i)c_t$). Scaling must preserve the stochastic nature of the posterior probabilites of all states, i.e. the posterior probabilities of all the states in a chain must sum to one in each time slice ($\sum_i \gamma_t(i) = \sum_i \alpha_t(i)\beta_t(i) = 1$). In a conventional HMM, the Forward-Backward algorithm –via dynamic programming– exhaustively samples all possible state sequences. In consequence this invariant is automatically obtained. In the N-heads dynamic programming version only a small fraction of state sequences are sampled, so that $\sum_i \gamma_t(i) < 1$. Noting that $\alpha_t(i) = p(o_t|i)\sum_j \alpha_{t-1}(j)P_{i|j}$, we obtain a simple procedure for rescaling the backward variables that implicitly restores the invariant:

$$\gamma_t(i) \leftarrow \frac{\alpha_t(i)\beta_t(i)}{\sum_i \alpha_t(i)\beta_t(i)} \tag{4.123}$$

$$\beta_t(i) \leftarrow \frac{\gamma_t(i)}{\alpha_t(i)} = \frac{\gamma_t(i)}{p(o_t|i)\sum_j \alpha_{t-1}(j)P_{i|j}} \tag{4.124}$$

4.11.4 MAP Estimation of the State Sequence: Viterbi

In each step of the Viterbi algorithm we seek the MAP density $\{h_t(i), k'_t(i)\}$ pairs given all antecedent paths. In conventional HMMs, for each head –state– i at time t, $h_t(i)$, the Viterbi algorithm selects the most likely antecent path in $t-1$. In N-heads Viterbi,

a sidekick at time t, $k'_t(i)$ needs to be chosen too. However, note that, in contrast to the Forward-Backward analysis, each head might have a different sidekick. These choices can be made in a two-step procedure: (1) for each antecedent path in $t-1$, $\{h_{t-1}(i), k'_{t-1}(i)\}$ select MAP sidekicks in t, $k'_t(j)$; (2) for each head in t, $h_t(i)$, select the antecedent path in $t-1$, $\{h_{t-1}(i), k'_{t-1}(i)\}$ and associated sidekick $k'_t(i)$ that maximizes the new head's posterior. Algorithmically, this N-heads Viterbi algorithm is obtained by taking the maximum rather than the sum in steps 1–4 (equations 4.118 to 4.121).

4.12 Synthetic Data as Priors

Bayesian inference is an approach to statistics in which all forms of uncertainty are expressed in terms of probability.

A Bayesian approach to a problem starts with the formulation of a model that we hope is adequate to describe the situation of interest. We then formulate a prior distribution over the unknown parameters of the model, which is meant to capture our beliefs about the situation before seeing the data. After observing some data, we apply Bayes' Rule to obtain a posterior distribution for these unknowns, which incorporates both the prior and the data. From this posterior distribution we can compute predictive distributions for future observations.

This theoretically simple process can be justified as the proper approach to uncertain inference by various arguments involving consistency with clear principles of rationality. Despite this, since Bayes (1763) and more importantly since Fisher (1922) the scope and merit of Bayesian inference have been debated. Critics have been claiming that the choice of the priors is too arbitrary and subjective to be acceptable. It is indeed subjective, but for this very reason it is not arbitrary. There is (in theory) just one correct prior, the one that captures your (subjective) prior beliefs. In contrast, other statistical methods are truly arbitrary, in that there are usually many methods that are equally good according to non-Bayesian criteria of goodness, with no principled way of choosing between them.

On the other hand, proponents are attracted to the logical consistency, simplicity and flexibility of the Bayesian approach such that they tend to view the selection of a prior as an important but manageable technical detail. There have been serious efforts on finding structural rules that determine priors, specially during the 1960s and 1970s and again in

the past several years.

The fundamental ideas originate with Jeffreys, who believed in the existence of an "initial state of knowledge", and thought that it was important to be able to make inferences based on data collected at this stage. In the case of a particular hypothesis being considered, he described this stage as one at which an investigator has "no opinion" about whether the hypothesis is true or not [107]. This has lead to what are called "non-informative" or "reference" priors. Many methods have been proposed for constructing reference priors. Among them, Laplace and the principle of insufficient reason, invariance, data-translated likelihoods, maximum entropy, minimum entropy, the Berger-Bernardo method, geometry-based methods, coverage matching methods, Zellner's method, decision-theoretic methods, and Rissanen's method. I direct the interested reader to [121] for a good survey of the proposed methods.

Two interpretations have been given to reference priors. The first interpretation asserts that they are formal representations of ignorance. Ignorance is desirable because the statistical analysis is often required to appear objective and "neutral". Noninformative priors are intended not to bias the posterior towards any direction. The second asserts that there is no objective, unique prior that represents ignorance; instead reference priors are chosen by public agreement, because of their convenience. Historically, the first interpretation was at one time the dominant interpretation and much effort was spent trying to justify one prior or another as being noninformative. In the last years, the trend has been to shift to the second interpretation. There are many situations where reference priors lead to posteriors with very undesirable properties. These include incoherence, inadmissibility of Bayes estimators, marginalization paradoxes, sample space dependence, improperty, and unsuspected marginal effects in high-dimensional problems.

Kass and Wasserman [121] highlight some open questions about prior selection, such as the computation of Jeffrey's prior and the verification that it leads to a proper posterior in nonnormal hierarchical problems. In particular, the authors emphasize the importance of the priors of small sample problems. In many real situations –as it is the case of this thesis– there is a very limited amount of data. In these occasions, the prior can overwhelm the data. In Kass and Wasserman's words 'when sample sizes are small (relative to the number of parameters being estimated), it is dangerous to put any faith in any "default" solution' [121].

I propose in this thesis a novel method for prior specification by means of models generated from synthetic data. For example, in the visual surveillance task I have developed a framework for building and training models of the behaviors of interest using *synthetic agents* [174, 208]. Simulation with the agents yields synthetic data that is used to train *prior models*. These prior models are then used recursively in a Bayesian framework to fit real behavioral data. This approach provides a rather straightforward and flexible technique to the design of priors, one that does not require strong analytical assumptions to be made about the form of the priors[3]. Moreover, because the prior models are generated using synthetic data that mimics real human behaviors, I would claim that they capture quite well our subjective prior beliefs about the problem. And this is the definition of a Bayesian prios.

Experimental data (see section 5.4) of human-to-human interactions have shown that by combining such synthetic priors with limited real data one can easily achieve very high accuracies of recognition of different interactions. Thus, the system is robust to cases in which there are only a few examples of a certain behavior (such as in interaction type 2 described in section 5.4.3) or even no examples except synthetically-generated ones. Figure 4-18 illustrates the two-step training procedure when using this synthetic priors.

This is a powerful yet simple new approach for the selection and design of priors in Bayesian approaches to inference.

4.13 Hierarchical PINs

Having already presented the theory behind the dynamic graphical models (DynPINs), at this point I would like to briefly remind the reader that I have used a two-layer hierarchy of DynPINs for modeling and recognizing human behaviors. The structure of the proposed computational model of human behavior depicted in figure 1-3 can be seen as a two-layer hierarchical dynamic graphical model.

The proposed hierarchy consists of two levels, depicted in figure 4-19: (1) small-scale, short-term structure of human behavior, described by linear dynamic models (Kalman Filters), thus incorporating constraints such smoothness and continuity; (2) long-term, large-

[3]Note that in most of the cases the priors could have the same form as the posteriors, namely they are graphical models.

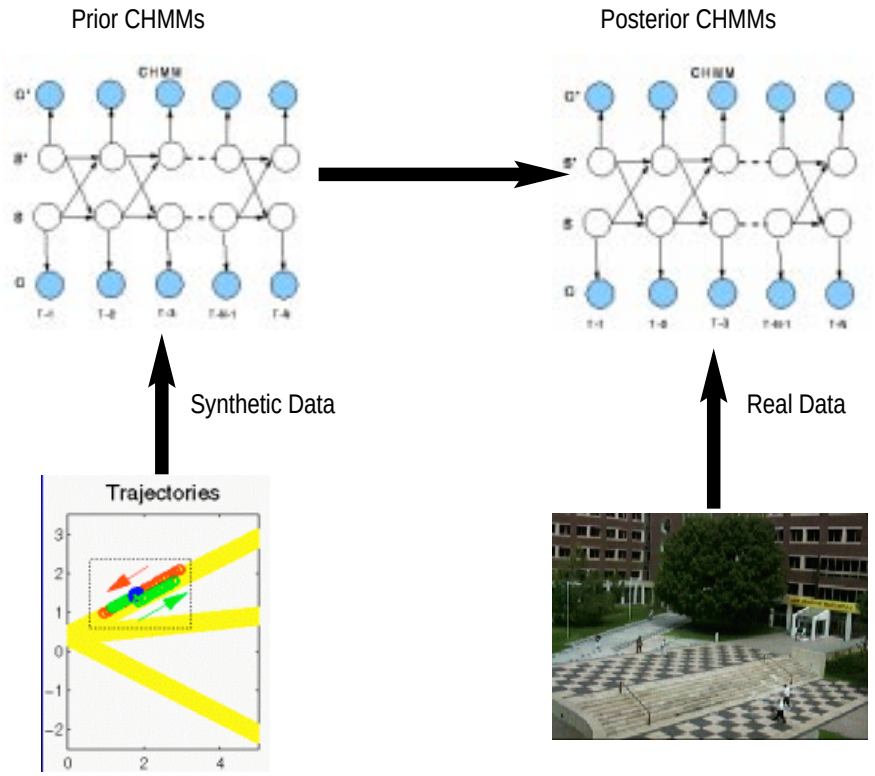


Figure 4-18: Training procedure when using synthetically generated priors

scale behaviors, modeled by HMM(1,1) and CHMM (HMM(1,2)). A similar framework was proposed by Pentland and Liu ([182], [184]) in the domain of driving, and it is related to research in robot control [150] and computer vision [31], in which elements from dynamic modeling or control theory are combined with stochastic transitions. These related efforts have been successful in tracking human motion and recognizing atomic actions such as running. My approach, similarly as in [182], goes beyond simple actions, to describe and classify more extended and elaborate behaviors, such as following a person or passing a vehicle while driving. These behaviors consist of several atomic actions chained together in a particular sequence via a HMM or CHMM. Therefore, in this thesis work, the observations that feed the DynPINs are not raw observations, but the state variables of a Kalman Filter.

Once the basic theoretical framework has been formulated, I will proceed to describe the experimental testbeds that I have developed in this thesis to validate the just-described theory. They are aimed to recognize and predict increasingly more complex individual and interactive behaviors in real situations.

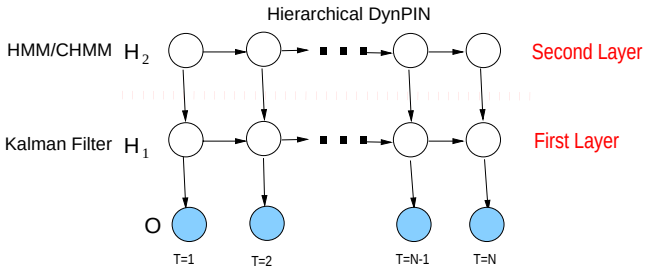


Figure 4-19: Hierarchical DynPIN used in this thesis

Chapter 5

Experiments and Applications

In this chaper I describe the experiments that test the computational model of human behavior proposed in this thesis. The theoretical background has already been presented in chapters 3 and 4. The proposed model consists of two layers: (1) at the bottom, the perceptual system –see chapter 3– senses the world (user's face, body, hands, car's internal signals, and surrounding traffic) and models its short-term dynamics using Kalman filters; (2) at the top, the behavior models (HMMs and CHMMs) –see chapter 4– recognize and predict the human behaviors of interest (facial expressions, two-hand gestures, pedestrian interactions and driver maneuvers).

I have developed four major testbeds for recognizing and predicting human behavior in real situations. These systems are evaluated by their recognition accuracy on testing data. In the case of interacting behaviors, the performance of the CHMMs –see section 4.11– will be compared to that of HMMs, a state-of-the-art competitive learning architecture. Some applications of the systems are also presented.

The four systems are: (1) LAFTER, a real-time face detection and tracking system with facial expression recognition; (2) two-hand gesture recognition in TaiChi; (3) a visual surveillance system for automatic detection and recognition of pedestrian interactions; and (4) a SmartCar for acquisition and automatic recognition of driver behavior.

The perceptual input (described in chapter 3) in these systems ranges from computer vision for face and mouth tracking in LAFTER –described in section 3.2–, 3-D hands and body tracking in the TaiChi gesture recognition system, and body tracking in the pedestrian surveillance system –described in section 3.2– to a sophisticated, multimodal

data acquisition system in the SmartCar, with four video signals and five car internal signals (brake, acceleration throttle, gear, steering wheel angle and speed) –described in section 3.3.

The behavior models in all four systems are dynamic graphical models or DynPINs (HMMs and CHMMs). As we have seen in chapter 4, PINs present several important advantages that are relevant to the problem of modeling human individual and interactive behaviors: they can handle incomplete data as well as uncertainty; they are trainable and easy to avoid overfitting; they encode causality in a natural way; there are algorithms for both doing prediction and probabilistic inference; they offer a framework for combining prior knowledge and data; and finally they are modular and parallelizable. From a psychological viewpoint, there is interesting connections between DynPINs and some of the models of human behavior that have been proposed in Psychology and Philosophy (see chapter 2).

As it has already been stated in section 4.12, I pursue in this thesis a Bayesian approach to modeling that includes both *prior* knowledge and *evidence* from data. One of the hypothesis of this thesis work is that the Bayesian approach provides the best framework for coping with small data sets and eventually novel behaviors. Dynamical graphical models or DynPINs (see chapter 4) [37], Hidden Markov Models (HMMs) [196] and Coupled Hidden Markov Models (CHMMs) [30, 28, 172], seem most appropriate for modeling and classifying human behaviors because they offer dynamic time warping, a well-understood training algorithm, and a clear Bayesian semantics for both individual (HMMs) and interacting or coupled (CHMMs) generative processes. These models address some of the major problems that psychologists and philosophers have traditionally been facing. Most of the psychological and philosophical models are manually built, relatively ad-hoc, not learnt from human behaviors in real situations and not predictive. Finally there is a lack of rigorous mechanisms for evaluating the performance of the models. As a consequence, there are very few systems able to perceive and understand aspects of human behavior. The human behavior modeling framework proposed in this thesis solves most of these problems: the model parameters are automatically learnt from real data, collected in real situations. The models are generative, predictive, and are evaluated according to their recognition accuracy on test data.

From a practical point of view, a major problem with a data-driven statistical approach, specially when modeling rare or anomalous behaviors, is the limited number of examples

of those behaviors for training the models. This is another important contribution of this thesis. I therefore, emphasize efficient Bayesian integration of both prior knowledge (by the use of prior models learned from synthetic data) with evidence from real data (by situation-specific parameter tuning). In this sense the goal is to be able to successfully apply the system to normal real-life situations without additional training.

To specify the priors in the learning framework developed in the thesis, I propose to use models generated from synthetic data. For example, in the visual surveillance task I have developed a framework for building and training models of the behaviors of interest using *synthetic agents* [174, 208]. Simulation with the agents yields synthetic data that is used to train *prior models*. These prior models are then used recursively in a Bayesian framework to fit real behavioral data. This approach provides a rather straightforward and flexible technique to the design of priors, one that does not require strong analytical assumptions to be made about the form of the priors[1]. Experimental data (see section 5.4) of human-to-human interactions have shown that by combining such synthetic priors with limited real data one can easily achieve very high accuracies of recognition of different interactions. Thus, the system is robust to cases in which there are only a few examples of a certain behavior (such as in interaction type 2 described in section 5.4.3) or even no examples except synthetically-generated ones.

5.1 Isolated Single User Behaviors in LAFTER: Continuous Real-time HMMs for Facial Expression Recognition

The first and simplest system that I have developed for modeling human behavior is LAFTER. As it has been described in chapter 3, LAFTER extends previous efforts to real-time analysis of the human face using our blob tracking methodology. The main key elements of LAFTER are mixture-of-Gaussians blob model as a representation for computer vision applications, batch and on-line Expectation-Maximization for blob parameter estimation, real-time active camera tracking by means of a PD controller, and continuous, real-time HMM classification method for mouth shape recognition. I will describe in this section the mouth shape recognition experiments and results obtained with LAFTER. In

[1] Note that in most of the cases the priors could have the same form as the posteriors, namely they are graphical models.

LAFTER the temporal interpretation of facial expressions is performed by use of Hidden Markov Models (HMMs) [197] to recognize different patterns of mouth shape. HMMs are one of the basic probabilistic tools used for time series modeling and one of the simplest dynamic graphical models or DynPINs (see section 4.4.2 in chapter 4). A HMM is essentially a mixture model where all the information about the past of the time series is summarized in a single discrete variable, the hidden state. This hidden state is assumed to satisfy a *first order Markov condition*: any information about the history of the process needed for future inferences must be reflected in the current state.

Before proceeding with a detailed description of LAFTER, I will review the most relevant previous work in facial expression recognition using computer vision.

5.1.1 Previous Work in Facial Expression Recognition

In recent years, much research has been done on machine recognition of human facial expressions. Feature points [12], physical skin and muscle activation models [145, 258, 212], optical flow models [64], feature based models using manually selected features [186], local parametrized optical flow [22], deformable contours [143, 157], combined with optical flow [274] as well as deformable templates [120, 278, 91, 32] among several other techniques have been used for facial feature analysis.

Even though there are numerous face detection, tracking and facial features analysis systems, there are relatively few facial expression recognition systems. Among them and to the best of my knowledge, none of them performs robustly in real-time. I will overview in the following the recognition performance of some of them. The approach proposed by Matsuno et al [145] performs extremely well on training data (98.4% accuracy) but more poorly on testing data, with 80% accuracy. They build models of facial expressions from deformation patterns on a potential net computed on training images and subsequent projection in the so called *Emotion Space*. Expressions of new subjects are recognized by projecting the image net onto the Emotion Space. Black et al [22] report an overall average recognition of 92% for 6 different facial expressions (happiness, surprise, anger, disgust, fear and sadness) in 40 different subjects. Their system combines deformation and motion parameters to derive mid- and high-level descriptions of facial actions. The descriptions depend on a number of thresholds and a set of rules that need to be tuned for each expression and/or subject. The system described in [135] has a recognition rate of about 74% when using 118 testing

images of the seven psychologically recognized categories across several subjects. They use flexible models for representing appearance variations of faces. Essa et al [65] report 98% accuracy in recognizing 5 different facial expressions using both peak-muscle activations and spatio-temporal motion energy templates from a database of 52 sequences. An accuracy of 98.7% is reported by Yael Moses et al [157] on real-time facial expression recognition. Their system detects and tracks the user's mouth, by representing it by a valley contour based between the lips. A simple classification algorithm is then used to discriminate between 5 different mouth shapes. They consider only confusions but not false negatives (confusions of any expression to neutral) on two independent samples of about 1000 frames each and of a predetermined sequence of 5 different expressions plus the neutral face. Padgett et al [176] report 86% accuracy on emotion recognition on novel individuals using neural networks for classification. The recognized emotions are happy, sad, fear, anger, surprise, disgust or neutral across 12 individuals. Finally the method adopted by Lien et al [141] is the most similar to LAFTER in the sense of the recognition approach, because they also use HMMs. The expression information is extracted by use of facial feature point tracking (for the lower face –mouth–) or by pixel-wise flow tracking (for the upper face –forehead and eyebrows–) followed by PCA to compress the data. Their system has an average recognition rate for the lower face of 93% and for the upper face of 91% using FACS.

LAFTER extends these previous efforts to real-time analysis of the human face using our blob tracking methodology. This extension required development of a new mixture-of-Gaussians blob model, an incremental Expectation Maximization method, an active camera control by means of a PD controller, and a continuous, real-time HMM classification method suitable for classification of shape data.

As it has been described in chapter 4, HMMs are a particular case of DynPINs, dynamic probabilistic inference networks or graphical models. HMMs offer dynamic time warping, an efficient learning algorithm and clear Bayesian semantics. From a practical viewpoint, HMMs have been prominently and successfully used in speech recognition and, more recently, in handwriting recognition. However, their application to visual recognition purposes is more recent [276], [269], [270], [233].

As we have seen in chapter 3 (equation 4.15), the posterior state sequence probability

in a HMM is given by

$$P(S|O) = P_{s_1} p_{s_1}(o_1) \prod_{t=2}^{T} p_{s_t}(o_t) P_{s_t|s_{t-1}} \tag{5.1}$$

where $S = \{a_1, \ldots, a_N\}$ is the set of discrete states, $s_t \in S$ corresponds to the state at time t. $P_{i|j} \doteq P_{s_t=a_i|s_{t-1}=a_j}$ is the state-to-state transition probability (i.e. probability of being in state a_i at time t given that the system was in state a_j at time $t-1$). In the following they will be written as $P_{s_t|s_{t-1}}$. The prior probabilities for the initial state are expressed as $P_i \doteq P_{s_1=a_i} = P_{s_1}$. Finally $p_i(o_t) \doteq p_{s_t=a_i}(o_t) = p_{s_t}(o_t)$ are the output probabilities for each state[2]. The Viterbi algorithm provides a formal technique for finding the most likely state sequence associated with a given observation sequence. To adjust the model parameters (transition probabilities $P(s_{t-1}|s_t)$, output probabilities parameters $p_{s_t}(o_t)$ and prior state probabilities P_{s_1}) such that they maximize the probability of the observation given the model an iterative procedure – such as the JLO 4.5 or *Baum-Welch* algorithm– is needed.

In LAFTER a continuous real-time HMM system computes the maximum likelihood of the input sequence with respect to all the models during the testing or recognition phase. Note that the observations that feed the HMMs are not the raw signals coming from the computer vision modules, but the predictions provided by a Kalman Filter that tracks each of the blobs –or features of– of interest. LAFTER runs on an SGI Indy, with the low-level vision processing occurring on a separate Indy or Pentium PC, and communications occurring via a socket interface.

5.1.2 Mouth Feature Vector Extraction

The mouth shape is characterized by its area, its spatial eigenvalues (e.g., width and height) and its bounding box. Figure 5-1 depicts the extracted mouth feature vector. The use of this feature vector to classify facial expressions has been suggested by psychological experiments [275, 156], which examined the most important discriminative features for expression classification.

Rotation invariance is achieved by computing the face's image-plane rotation angle and rotating the region of interest with the negative of this angle. Therefore even though the

[2]The output probability is the probability of observing o_t given state a_i at time t

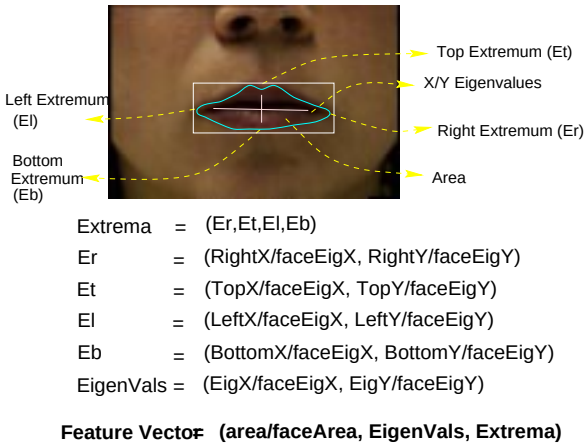


Figure 5-1: Mouth feature vector extraction

user might turn the head the mouth always appears nearly horizontal, as figure 3-6 in chapter 3 illustrates.

Using the mouth shape feature vector described above, 5 different HMMs were trained for each of the following mouth configurations (illustrated in figure 5-2): neutral or default mouth position, extended/smile mouth, sad mouth, open mouth and extended+open mouth (such as in laughing).

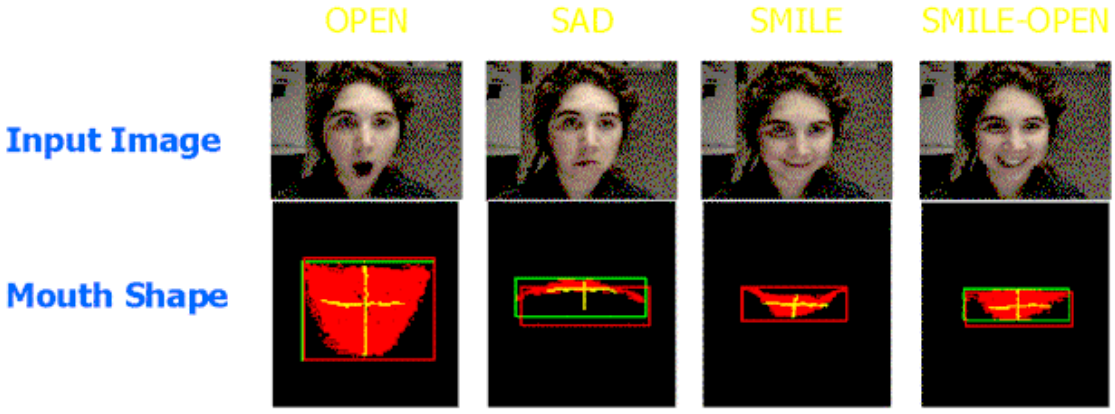


Figure 5-2: Open, sad, smile and smile-open recognized expressions.

The neutral mouth acted to separate the various expressions, much as a silence model acts in speech recognition. The final HMM structure was determined using 10-fold cross-validation (described in section 4.7.1) derived for the non-neutral mouth configurations consisted of 4-state forward HMMs. The neutral mouth was modeled by a 3-state forward HMM.

Recognition results for eight different users making over 2000 expressions are summa-

rized in table 5.1. The data were divided into different sets for training and testing purposes. The first line of the recognition results shown in table 5.1 corresponds to training and testing with all eight users. The total number of examples is denoted by N, having a total N=2058 instances of the mouth expressions (N=750 for training and N=1308 for testing). The second line of the same table corresponds to person-specific training and testing. As can be seen, accurate classification was achieved in each case.

	TEST ON:	
TRAIN ON:	training	testing
All users	97.73	95.95
Single user	100.00	100.00

Table 5.1: Facial expression recognition results on training and testing data

5.1.3 Applications

In the following I will briefly describe a number of applications of LAFTER. Many of them have been extensively tested on naive users during the Media Lab's open houses and scientific conferences demonstration sessions.

Automatic Camera Man

The static nature of current video communication systems induces extra articulatory tasks that interfere with real world activity. For example, users must keep their head (or an object of interest) within the field of the camera (or of the microphone) in order to be perceived by distant parties. As a result, the user ends up being more attentive to the way how to using the interface than to the conversation itself. The communication is therefore degraded instead of enriched.

In this sense, LAFTER, with its active camera face tracking acts as an "automatic camera man" that is continuously looking at the user while he/she moves around or gestures in a video-conference session. In informal teleconferencing testing, users have confirmed that this capability significantly improves the usability of the teleconferencing system.

Experiences with a virtual window system

Some of the limitations of traditional media spaces -with respect to the visual information- are [76]: restricted field of view on remote sites by the video, limited video resolution, spatial discontinuity, medium anisotropy and very restricted movement with respect to remote spaces. Each of these negatively affects the communication in a media space, with movement one of the most influential, as Gibson emphasized in [85]. Motion allows us to increase our field of view, can compensate for low resolution, provides information about the three-dimensional layout and allow people to compensate for the discontinuities and anisotropies of current media spaces, among other factors. Therefore, not only allowing movement in local media spaces is a key element for desktop mediated communication and video-conference systems -as I have previously emphasized-, but also the ability of navigating and exploring the remote site.

The Virtual Window proposed by Gaver [77] illustrates an alternative approach: as the user moves in front of his local camera, the distant motorized camera is moved accordingly: exploring a remote site by using head movements opens a broad spectrum of possibilities for systems design that allow an enriched access to remote partners. Figure 5-3 depicts an example of a virtual window system.

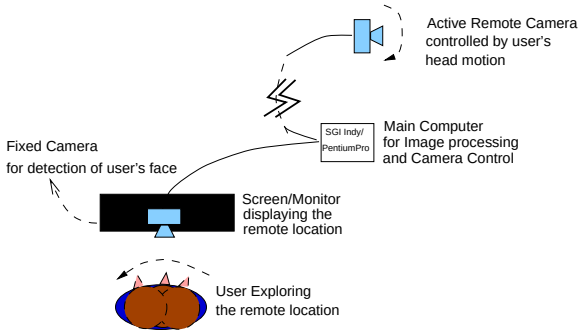


Figure 5-3: The virtual window: Local head positions are detected by the active tracking camera and used to control a moving camera in the remote site. The effect is that the image on the local monitor changes as if it were a window. The second image illustrates the virtual window system in use.

One of the main problems that Gaver recognized in his virtual window system was that its vision controller was too sensitive to lighting conditions and to moving objects. Consequently, the tracking was unstable; users were frustrated and missed the real purpose of the system when experiencing it.

I found that by incorporating LAFTER's face tracker into a Virtual Window system,

users could successfully obtain the effect of a window onto another space. To the best of our knowledge this is the first real-time robust implementation of the virtual window. In informal tests, users reported that the LAFTER-based virtual window system gives a good sense of the distant space.

Real-time computer graphics animation

Because LAFTER continuously tracks face location, image-plane face rotation angle, and mouth shape, it is a simple matter to use this information to obtain real-time animation of a computer graphics character. This character can, in its simplest version, constantly mimic what the user does (as if it where a virtual mirror) or, in a more complex system, understand (recognize) what the user is doing and react to it. A "virtual mirror" version of this system — using the character named Waldorf shown in figure 5-4 — was exhibited in the Digital Bayou section of SIGGRAPH'96 in New Orleans.

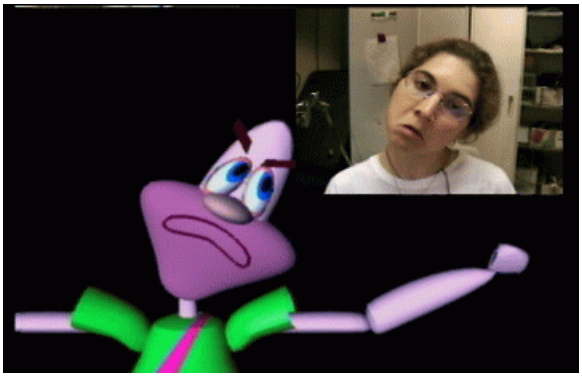


Figure 5-4: Real time computer graphics animation

Responsive Portraits

A responsive portrait [230] consists of a multiplicity of views whose dynamic presentation results from the interaction between the viewer and the image. The viewer's proximity to the image, head movements, and facial expressions elicit dynamic responses from the portrait, driven by the portrait's own set of autonomous behaviors [229]. Figure 5-5 illustrates one example of the interaction between the user and the portrait. This type of interaction reproduces an encounter between two people: the viewer and the character portrayed. The perceptual system of a responsive portrait is implemented using LAFTER. Figure 5-6 depicts the system architecture of a responsive portrait.

Figure 5-5: Responsive Portrait typical interaction

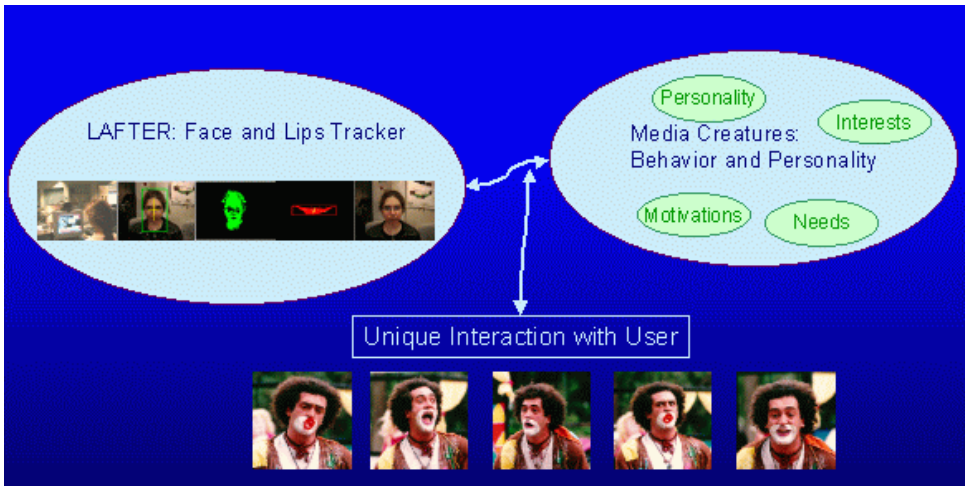


Figure 5-6: Responsive Portrait system architecture

The experience of an individual viewer with the portrait is unique, because it is based on the dynamics of the encounter rather than on the existance of a unique, ideal portrait of the subject. A responsive portrait, thus, challenges our notion of the photographic portrait as a unique image that captures the essence of the subject. In a responsive portrait the whole notion of "who is watching who" is reversed: the object becomes the subject, the subject is observed. By layering a multiplicity of images of the portrayed person on the same interactive display and offering a natural interactive interface and mapping modalities, an extended set of expressive communication abilities is available to the artist photographer. Also, through this artwork, new venues are described for the design of interactive photo exhibitions for galleries and museums.

Preferential Coding

Finally, LAFTER can be used as the front-end to a *preferential image coding system*. It is well-known that people are most sensitive to coding errors in facial features. Thus it makes sense to use a more accurate (and more expensive) coding algorithm for the facial features, and a less accurate (and cheaper) algorithm for the remaining image data [63, 170, 5]. Because the location of these features is detected by our system, this coding scheme can be used. The improvement obtained by such system is illustrated in figure 5-7.

Figure 5-7: Preferential coding: the first image is the JPEG flat encoded image (File size of 14.1Kb); the second is a very low resolution JPEG encoded image using flat coding (File size of 7.1Kb); the third one is a preferential coding encoded image with high resolution JPEG for the eyes and mouth but very low resolution JPEG coding for the face and background (File size of 7.1Kb).

5.2 Interaction Models via CHMMs

As we have seen in section 4.10 even though HMMs are a popular probabilistic framework for modeling processes that have structure in time, many interesting systems are composed of multiple interacting processes, and thus merit a compositional representation of two or more variables. This is typically the case for systems that have structure both in time and

space. With a single state variable, Markov models are ill-suited to these problems. In order to model these interactions a more complex architecture is needed.

Extensions to the basic Markov model generally increase the memory of the system (durational modeling), providing it with compositional state in time. I am interested in systems that have compositional state in *space*, e.g., more than one simultaneous state variable. It is well known that the exact solution of extensions of the basic HMM to 3 or more chains is intractable. In those cases approximation techniques are needed ([216, 83] [228, 268]). However, it is also known that there exists an exact solution for the case of 2 interacting chains, as it is our case [216, 28].

In this thesis I use and validate with real data two Coupled Hidden Markov Models (CHMMs) for modeling two interacting processes: hands, individual humans or cars. As it has been described in detail in section 4.11 of chapter 4, in this architecture state chains are coupled via matrices of conditional probabilities modeling causal (temporal) influences between their hidden state variables. The graphical representation of CHMMs is shown in figure 4-15 (d). From the graph it can be seen that for each chain, the state at time t depends on the state at time $t-1$ in both chains. The influence of one chain on the other is through a causal link.

I have develop two testbeds that validate the suitability of CHMMs for recognizing and predicting interactive behaviors. First, two-hand gestures in Tai-Chi and second pedestrian interactions in a visual surveillance task.

5.3 CHMMs for Tai-Chi Gesture Recognition

The first experiment to validate how appropriate CHMMs are for modeling and recognizing interactive behaviors is a Tai-Chi gesture recognition system [30]. Tai-Chi ch'uan is a Chinese martial art and meditative exercise, consisting of stylized full-body and upper-body gestures. Many gestures, indeed, most signals generated by human activity are the result of multiple interacting processes. In gesture, the arms are neither independent nor wholly mutually determined; some form of interactional modeling is appropriate.

Using a real-time self-calibrating, 3-D stereo blob tracker [13], I obtained 3D hand tracking data for three Tai-Chi gestures involving two, semi-independent arm motions: the left single whip, the left cobra, and the left brush knee. Figure 5.3 illustrates one example of

each of the gestures and the blob-tracking. A detailed description of this set of *Tai-Chi* experimental results can be found in [30] and viewed at `http://nuria.www.media.mit.edu/~nuria/chmm/taichi.html`. An Extended Kalman filter (EKF) tracks the blobs' location, coarse shape, color pattern, and velocity. This information is represented as a low-dimensional, parametric probability distribution function (*pdf*) composed of a mixture of Gaussians, whose parameters (sufficient statistics and mixing weights for each of the components) are estimated using Expectation Maximization (EM) (described in section 4.7.3).

The visual input, thus, detects and tracks the user's hands and head in 3D and outputs a feature vector describing the position and motion. These output feature vectors constitute the temporally ordered stream of data input to the stochastic state-based behavior models. Both HMMs and CHMMs, with varying structures depending on the complexity of the behavior, were used for classifying the observed behaviors.

Data collection

A total of 52 sequences, roughly 17 of each gesture, were collected. The extracted feature vector consisted of the 3D (x, y, z) centroid (mean position) of each of the blobs that characterize the hands. All the gestures were performed by the same person, seated in a swivel chair and moving her upper body and hands. Each gesture began with both hands in a rest or neutral position and ended with the hands in a gesture-specific final position or returning to neutral position. The experiments were oriented to a *single word* recognition task; the extension to continuous gesture trains is the same as with conventional HMMs. The main sources of noise were blob instabilities, variations in the performance of each gesture, and variations in initial body rotation and position from sequence to sequence. The extracted feature vector, being simple (x, y, z) positions, reflects this noise directly.

The frame rate of the vision system varied from 15-30 Hz. The data was resampled using timestamped frames and cubic spline interpolation to produce a 30Hz signal, then low-pass filtered with a 3Hz cutoff. Similar preprocessing is used by Campbell *et al.* [40], who converted the feature vector to head-centered cylindrical coordinates derivatives $(dr, d\theta, dz)$ for rotation and shift invariance. In the experiments reported in this section raw 3D (x, y, z) coordinates were used.

Tai-Chi gesture models

The best trained HMMs and CHMMs –using 10-fold crossvalidation (explained in section 4.7.1)– were used to classify the full data set of 52 gestures. The Viterbi algorithm was used to find the maximum likelihood model for HMMs and CHMMs. Two-thirds of the testing data had not been seen in training, including gestures performed at varying speeds and from slightly different views.

It can be seen from the classification accuracies, shown in table 5.4, that the CHMMs outperform the HMMs. Note that this difference is not purely due to intrinsic modeling power, however; from earlier experiments [40] we know that when a large number of training samples is available then HMMs could potentially reach similar accuracies. One can conclude thus that for data where there are two partially-independent processes (e.g., coordinated but not exactly linked), the CHMM method requires much less training to achieve a high classification accuracy.

Recognition Results on Tai-Chi Gestures		
	Single HMMs	**Coupled HMMs (CHMMs)**
Accuracy	69.2% (25+30+180)	100% (27+18+54)

Table 5.2: Recognition accuracies for HMMs and CHMMs on Tai-Chi gestures. The expressions between parenthesis correspond to the number of parameters of the largest best-scoring model.

Table 5.4 illustrates the source of this training advantage. The numbers between parenthesis correspond to the number of degrees of freedom in the largest best-scoring model: state-to-state probabilities + output means + output covariances. The conventional HMM has a large number of covariance parameters because it has a 6-D output variable; whereas the CHMM architecture has two 3-D output variables. In consequence, due to their larger dimensionality HMMs need much more training data than equivalent CHMMs before yielding good generalization results.

Sensitivity analysis

HMMs are notoriously sensitive to the random values assigned to parameters at initialization of training. To test the sensitivity of final model likelihoods to initial conditions, I randomly initialized each architecture, trained it on 5 examples of a gesture taken randomly,

and tested it on all sequences of that gesture. This was repeated 50 times per gesture and architecture. The likelihoods of the testing sets conditioned on recovered models was computed and mean and variance statistics were computed for each gesture and model. The resulting Gaussian distributions are depicted in figure 5-10, which shows the probability distribution of the per gesture likelihood for coupled (CHMMs), linked (LHMMs) (described in section 4.10.1) and single HMMs.

As may be expected, conventional HMMs were extremly sensitive to the initial values of the parameters. Linked HMMs were generally less sensitive, with a sensitivity (variance) that appears to depend on the structure of the gesture. Finally, coupled HMMs were most robust with respect to the initial conditions, and on average produced the best models –even in the case of the single whip, in which one hand is mostly stationary. In sum, CHMMs reliably produce better models– a highly desirable feature for a trained classifier.

These results also show why the HMMs performed as well as they did in the classification test. In choosing the best-of-50, I took models from the right (optimal) end of the distribution. Had typical models been picked (the mean), the HMMs would have done quite a bit worse than their already mediocre performance.

5.4 CHMMs for Pedestrian Interaction Recognition in a Visual Surveillance Task

The goal of the third testbed is to develop a framework for detecting, classifying and learning generic models of behavior in a visual surveillance situation. It is important that the models be generic, applicable to many different situations, rather than being tuned to the particular viewing or site. This was one of the main motivations for developing the virtual agent environment –described in section 5.4.3– for modeling behaviors. If the synthetic agents are "similar" enough in their behavior to humans, then the same models that were trained with synthetic data should be directly applicable to human data. This section describes the experiments I that have performed analyzing real pedestrian data using both synthetic and site-specific models (models trained on data from the site being monitored).

5.4.1 Previous Work in Visual Surveillance

There is extensive previous work on building visual surveillance systems, mostly for security applications. However, very few of these systems are able to automatically interpret the video sequences. In this section I will enumerate few of the most remarkable visual surveillance systems that incorporate some kind of recognition or interpretation of the video scenes. The system developed by Buxton and Gong [39] is one of the first visual surveillance systems that interpreted the image sequences by means of Bayesian Networks (BNs) and Dynamic Bayesian Networks (DBNs). Courtney ([48]) developed a system, which allows for detection activities in a closed environment. The activities include person leaving an object in a room, or taking it out of the room. Perhaps the most complete general solution is described in Brill at al. ([35]), who are working on an Autonomous Video Surveillance system. Brand ([29])) showed the results of detecting manipulations in video using a non-probabilistic grammar. This technique is non-probabilistic and requires relatively high quality low-level detectors. Davis and collaborators have developed a number of systems for real-time detection and tracking of multiple people, with some interpretation of the visual scene [88]. Their W4 system is a PC based real-time visual surveillance system for tracking people and their body parts, and monitoring their activities in monochromatic imagery. It operates on grayscale video imagery, or on video imagery from an infrared camera. Unlike other systems for tracking people, their system makes no use of color cue. Instead W4 employs a combination of shape analysis, robust tracking techniques, and a silhouette based body model to locate and track the people and understand the interaction between people and objects - e.g., people exchanging objects, leaving objects in the scene. In [112] and [234] different methods for classifying the trajectories of the tracked objects are used. None of the systems, however, is intended to provide an interpretation of the image sequence. Finally, in [102] a monitoring system is described as an example of an end-to-end implementation, which is adaptive to the physical features of the monitored environment and exhibits certain contextual awareness. Contextual labeling is performed by a stochastic parser, which is derived from that developed by Stolcke in [235]. The authors extended standard Stochastic Context-Free Grammar (SCFG) parsing to include (1) uncertain input symbols, and (2) temporal interval primitives that need to be parsed in a temporally consistent manner. The system is capable of maintaining concurrent interpretations when multiple activities are taking place simultaneously. Furthermore, the system allows for interpretation of activities

involving multiple objects, such as interactions between cars and people during PICK-UP and DROP-OFF. Their experimental results are remarkable. However in the current system the rule probabilities (grammar structure) need to be specified by hand as opposed to being automatically learnt from data, as it is the case of the models built in this thesis.

5.4.2 Interaction Models

In this visual surveillance situation the perceived behaviors are generated by pedestrians walking in an open outdoor environment. The goal from the behavior modeling viewpoint is to develop a generic, compositional analysis of the observed behaviors in terms of states and transitions between states over time in such a manner that (1) the states correspond to our common sense notions of human behaviors, and (2) they are immediately applicable to a wide range of sites and viewing situations. Figure 5-11 (left) shows a typical image for the pedestrian scenario.

I use two CHMMs for modeling two interacting processes, that, in this case, correspond to individual humans. In this pedestrian surveillance task the performance of HMMs and CHMMs is compared for maximum *a posteriori* (MAP) state estimation. The most likely sequence of states $\hat{S}$ within a model given the observation sequence $O = \{o_1, \ldots, o_n\}$ is obtained by $\hat{S} = \text{argmax}_S \, P(S|O)$.

The posterior state sequence probability $P(S|O)$ for a HMM is given by equation 5.1.

As it has been stated in chapter 4, in the case of CHMMs it is necessary to introduce another set of probabilities (see section 4.11 for a detailed description), $P_{s_t|s'_{t-1}}$, which correspond to the probability of state s_t at time t in one chain given that the other chain — denoted hereafter by superscript $'$ — was in state s'_{t-1} at time $t-1$. These new probabilities express the causal influence (coupling) of one chain to the other. The posterior state probability for CHMMs is therefore given by equation 5.2:

$$P(S|O) \;=\; \frac{P_{s_1} p(o_1|s_1) P_{s'_1} p(o'_1|s'_1)}{P(O)} \prod_{t=2}^{T} P_{s_t|s_{t-1}} P_{s'_t|s'_{t-1}} P_{s'_t|s_{t-1}} P_{s_t|s'_{t-1}} p(o_t|s_t) p(o'_t|s'_t)$$

where $s_t, s'_t; o_t, o'_t$ denote states and observations for each of the Markov chains that compose the CHMMs.

Coming back to the problem of modeling human behaviors, two persons (each modeled as a generative process) may interact without wholly determining each others' behavior.

Instead, each of them has its own internal dynamics and is influenced (either weakly or strongly) by others. The probabilities $P_{s_t|s'_{t-1}}$ and $P_{s'_t|s_{t-1}}$ describe this kind of interactions and CHMMs are intended to model them in as efficient a manner as is possible.

5.4.3 Prior Models via Synthetic Behavioral Agents

One of the key contributions of this thesis is the use of a new way for designing priors, as section 4.12 describes. For this particular pedestrian surveillance application, I have developed a framework for creating synthetic agents that mimic human behavior in a virtual environment [174, 208]. The agents can be assigned different behaviors and they can interact with each other as well. Currently they can generate 5 different interacting behaviors and various kinds of individual behaviors (with no interaction). The parameters of this virtual environment are modeled on the basis of a real pedestrian scene from which measurements of typical pedestrian movement were obtained.

One of the main motivations for constructing such synthetic agents is the ability to generate synthetic data which allows to determine which Markov model architecture will be best for recognizing a new behavior (since it is difficult to collect real examples of rare behaviors). By designing the synthetic agents models such that they have the best generalization and invariance properties possible, one can obtain flexible prior models that are transferable to real human behaviors with little or no need of additional training. The use of synthetic agents to generate robust behavior models from very few real behavior examples is of special importance in a visual surveillance task, where typically the behaviors of greatest interest are also the most rare.

Agent Architecture

The dynamic multi-agent system consists of some number of agents that perform some specific behavior from a set of possible behaviors. The system starts at time 0, moving discretely forward to time T or until the agents disappear from the scene.

The agents can follow three different paths with two possible directions, as illustrated in figures 5-12 and 5-13 by the yellow paths [3]. They walk with random speeds within an

[3] The three paths were obtained by statistical analysis of the most frequent paths that the pedestrians in the observed plaza followed. Note however that the performance of neither the computer vision nor the tracking modules is limited to these three paths.

interval, and they appear at random instances of time. They can slow down, speed up, stop or change direction independently from the other agents on the scene. Their velocity is normally distributed around a mean that increases or decreases when they slow down or speed up. When certain preconditions are satisfied a specific interaction between two agents takes place. Each agent has perfect knowledge of the world, including the position of the other agents.

In the following I will describe, without loss of generality, the two-agent system that was used for generating prior models and synthetic data of agents interactions. Each agent makes its own decisions depending on the type of interaction, its location and the location of the other agent on the scene. There is no scripted behavior or a priori knowledge of what kind of interaction, if any, is going to take place. The agents' behavior is determined by the perceived contextual information: current position, relative position of the other agent, speeds, paths they are in, directions of walk, etc., as well as by its own repertoire of possible behaviors and triggering events. The agents incorporate elements of "situation awareness" in their behavior. For example, if one agent decides to "follow" the other agent, it will proceed on its own path increasing its speed progressively until reaching the other agent, that will also be walking on the same path. Once the agent has been reached, they will adapt their mutual speeds in order to keep together and continue advancing together until exiting the scene.

For each agent the position, orientation and velocity is measured, and from this data a feature vector is constructed which consists of: $\dot{d_{12}}$, the derivative of the relative distance between two agents; $\alpha_{1,2} = sign(< v_1, v_2 >)$, or degree of alignment of the agents, and $v_i = \sqrt{\dot{x}^2 + \dot{y}^2}, i = 1, 2$, the magnitude of their velocities. Note that such feature vector is invariant to the absolute position and direction of the agents and the particular environment they are in.

Agent Behaviors

The agent behavioral system is structured in a hierarchical way. There are *primitive or simple behaviors* and *complex interactive behaviors* to simulate the human interactions.

In the experiments reported in this section five different interacting behaviors were considered. They appear illustrated in figures 5-12 and 5-13:

1. Follow, reach and walk together (inter1): The two agents happen to be on the same

path walking in the same direction. The agent behind decides that it wants to reach the other. Therefore it speeds up in order to reach the other agent. When this happens it slows down such that they keep walking together with the same speed.

2. Approach, meet and go on separately (inter2): The agents are on the same path but in opposite direction. When they are close enough, if they realize that they "know" each other, they slow down and finally stop to chat. After talking they go on separately, becoming independent again.

3. Approach, meet and go on together (inter3): In this case, the agents behave like in "inter2", but now after talking they decide to continue together. One agent changes therefore its direction to follow the other.

4. Change direction in order to meet, approach, meet and continue together (inter4): The agents start on different paths. When they are close enough they can see each other and decide to interact. One agent waits for the other to reach it. The other changes direction in order to go toward the waiting agent. Then they meet, chat for some time and decide to go on together.

5. Change direction in order to meet, approach, meet and go on separately (inter5): This interaction is the same as "inter4" except that when they decide to go on after talking, they separate becoming independent.

Proper design of the interactive behaviors requires the agents to have knowledge about the position of each other as well as synchronization between the successive individual behaviors activated in each of the agents. Figure 5-14 illustrates the timeline and synchronization of the simple behaviors and events that constitute the interactions.

These interactions can happen at any moment in time and at any location, provided only that the preconditititions for the interactions are satisfied. The speeds they walk at, the duration of their chats, the changes of direction, the starting and ending of the actions vary highly. This high variance in the quantitative aspects of the interactions confers robustness to the learned models that tend to capture only the invariant parts of the interactions. The invariance reflects the nature of their interactions and the environment.

5.4.4 Performance Comparison of CHMM and HMM architectures with Synthetic Agent Data

Both CHMM and HMM models of the five previously described synthetic agent interactions were built. In the case of CHMMs 2 or 3 states per chain were used, and 3 to 5 states in the case of HMMs (accordingly to the complexity of the various interactions). The optimal number of training examples, of states for each interaction as well as the optimal model parameters were obtained by a 10-fold cross-validation process (for a description of cross-validation see section 4.7.1). Because the same amount of data was used for training both architectures, I tried keeping the number of parameters to estimate roughly the same. For example, a 3 state ($N = 3$) per chain CHMM with 3 dimensional ($d = 3$) Gaussian observations has $(CN)^2 + N * (d + d!) = (2 * 3)^2 + 3 * (3 + 6) = 36 + 27 = 63$ parameters. A 5 state ($N = 5$) HMM with 6 dimensional ($d = 6$) Gaussian observations has $N^2 + N * (d + d!) = 5^2 + 5 * (3 + 6) = 25 + 45 = 70$ parameters to estimate.

Each of these architectures corresponds to a different physical hypothesis: CHMMs encode a spatial coupling in time between two agents (e.g., a non-stationary process) whereas HMMs model the data as an isolated, stationary process. From 11 to 75 sequences were used for training each of the models, depending on their complexity, such that overfitting was avoided. In all cases, the models were set up with a full state-to-state connection topology, so that the training algorithm was responsible for determining an appropriate state structure for the training data. The feature vector was 6-dimensional in the case of HMMs, whereas in the case of CHMMs each agent was modeled by a different chain, each of them with a 3-dimensional feature vector, as previously described.

To compare the performance of the two previously described architectures I used the best trained models to classify 20 unseen new sequences. In order to find the most likely model, the Viterbi algorithm was used for HMMs and the N-heads dynamic programming forward-backward propagation algorithm for CHMMs.

Table 5.3 illustrates the accuracy for each of the two different architectures and interactions. Note the superiority of CHMMs versus HMMs for classifying the different interactions and, more significantly, identifying the case in which there were no interactions present in the testing data.

Complexity in time and space is an important issue when modeling dynamic time series. The number of degrees of freedom (state-to-state probabilities+output means+output

Accuracy on synthetic test data (%)		
	HMMs	CHMMs
No inter	68.7	90.9
Inter1	87.5	100
Inter2	85.4	100
Inter3	91.6	100
Inter4	77	100
Inter5	97.9	100

Table 5.3: Accuracy for HMMs and CHMMs on synthetic data. Accuracy at recognizing when no interaction occurs ("No inter"), and accuracy at classifying each type of interaction: "Inter1" is follow, reach and walk together; "Inter2" is approach, meet and go on; "Inter3" is approach, meet and continue together; "Inter4" is change direction to meet, approach, meet and go together and "Inter5" is change direction to meet, approach, meet and go on separately

covariances) in the largest best-scoring model was 85 for HMMs and 54 for CHMMs.

An analysis of the accuracies of the models and architectures with respect to the number of sequences used for training was also performed. Efficiency in terms of training data is specially important in the case of on-line real-time learning systems -such as ours would ultimately be- and/or in domains in which collecting clean labeled data may be difficult.

The cross-product HMMs that result from incorporating both generative processes into the same joint-product state space usually requires many more sequences for training because of the larger number of parameters. In our case, this appears to result in a accuracy ceiling of around 80% for any amount of training that was evaluated, whereas CHMMs were able to reach approximately 100% accuracy with only a small amount of training. From this result it seems that the CHMMs architecture, with two coupled generative processes, is more suited to the problem of modeling the behavior of interacting agents than a generative process encoded by a single HMM.

In a visual surveillance system the *false alarm* rate is often as important as the classification accuracy. In an ideal automatic surveillance system, all the targeted behaviors should be detected with a close-to-zero false alarm rate, so that one could reasonably alert a human operator to examine them further. To analyze this aspect of our system's performance, the system's ROC curve was calculated. The left-most graph in figure 5-16 shows that it is quite possible to achieve very low false alarm rates while still maintaining good classification accuracy.

5.4.5 Pedestrian Behavior Recognition

Data Collection and Preprocessing

Using the person detection and tracking system described in section 3.2 2D blob features for each person in several hours of video were obtained. Up to 20 examples of *following* and various types of *meeting* behaviors were detected and processed.

The feature vector $\bar{x}$ coming from the computer vision processing module consisted of the 2D (x, y) centroid (mean position) of each person's blob, the Kalman Filter state for each instant of time, consisting of $(\hat{x}, \dot{\hat{x}}, \hat{y}, \dot{\hat{y}})$, where $\hat{}$ represents the filter estimation, and the (r, g, b) components of the mean of the Gaussian fitted to each blob in color space. The frame-rate of the vision system was of about 20-30 Hz on an SGI R10000 O2 computer. I low-pass filtered the data with a 3Hz cutoff filter and computed for every pair of nearby persons a feature vector consisting of: $\dot{d_{12}}$, derivative of the relative distance between two persons, $|v_i|, i = 1, 2$, norm of the velocity vector for each person, $\alpha = sign(< v_1, v_2 >)$, or degree of alignment of the trajectories of each person. Typical trajectories and feature vectors for an "approach, meet and continue separately" behavior (interaction 2) are shown in figure 5-15. This is the same type of behavior as "inter2" displayed in figure 5-12 for the synthetic agents. Note the similarity of the feature vectors in both cases.

Behavior Models

CHMMs were used for modeling three different behaviors: meet and continue together (interaction 3); meet and split (interaction 2) and follow (interaction 1). In addition, an *interaction* versus *no interaction* detection test was also performed. HMMs performed much worse than CHMMs and therefore I omit reporting their results.

Models trained with two types of data were used:

1. Prior-only (synthetic data) models: that is, the behavior models learned in our synthetic agent environment and then directly applied to the real data with *no additional training or tuning of the parameters.*

2. Posterior (synthetic-plus-real data) models: new behavior models trained by using as starting points the synthetic best models. In this case, 8 examples of each interaction data from the specific site were used.

Recognition accuracies for both these "prior" and "posterior" CHMMs are summarized in table 5.4. It is noteworthy that with only 8 training examples, the recognition accuracy on the real data could be raised to 100%. This results demonstrates the ability to accomplish extremely rapid refinement of the behavior models from the initial prior models.

Accuracy on real pedestrian test data (%)		
	Prior CHMMs	Posterior CHMMs
No-inter	90.9	100
Inter1	93.7	100
Inter2	100	100
Inter3	100	100

Table 5.4: Accuracy for both untuned, a priori models and site-specific CHMMs tested on real pedestrian data. The first entry in each column is the interaction vs no-interaction accuracy, the remaining entries are classification accuracies between the different interacting behaviors. Interactions are: "Inter1" follow, reach and walk together; "Inter2" approach, meet and go on; "Inter3" approach, meet and continue together.

Finally the ROC curve for the posterior CHMMs is displayed in figure 5-16.

One of the most interesting results from these experiments is the high accuracy obtained when testing the a priori models obtained from synthetic agent simulations. The fact that a priori models transfer so well to real data demonstrates the robustness of the approach. It shows that with the proposed synthetic agent training system, one can develop models of many different types of behavior — avoiding thus the problem of limited amount of training data — and apply these models to real human behaviors without additional parameter tuning or training.

Parameter Sensitivity In order to evaluate the sensitivity of the classification accuracy to variations in the model parameters, a set of models was trained, where different parameters of the agents' dynamics were changed by factors of 2.5 and 5. The performance of these altered models turned out to be virtually the same in every case except for the "inter1" (follow) interaction, which seems to be sensitive to people's relative rates of movement.

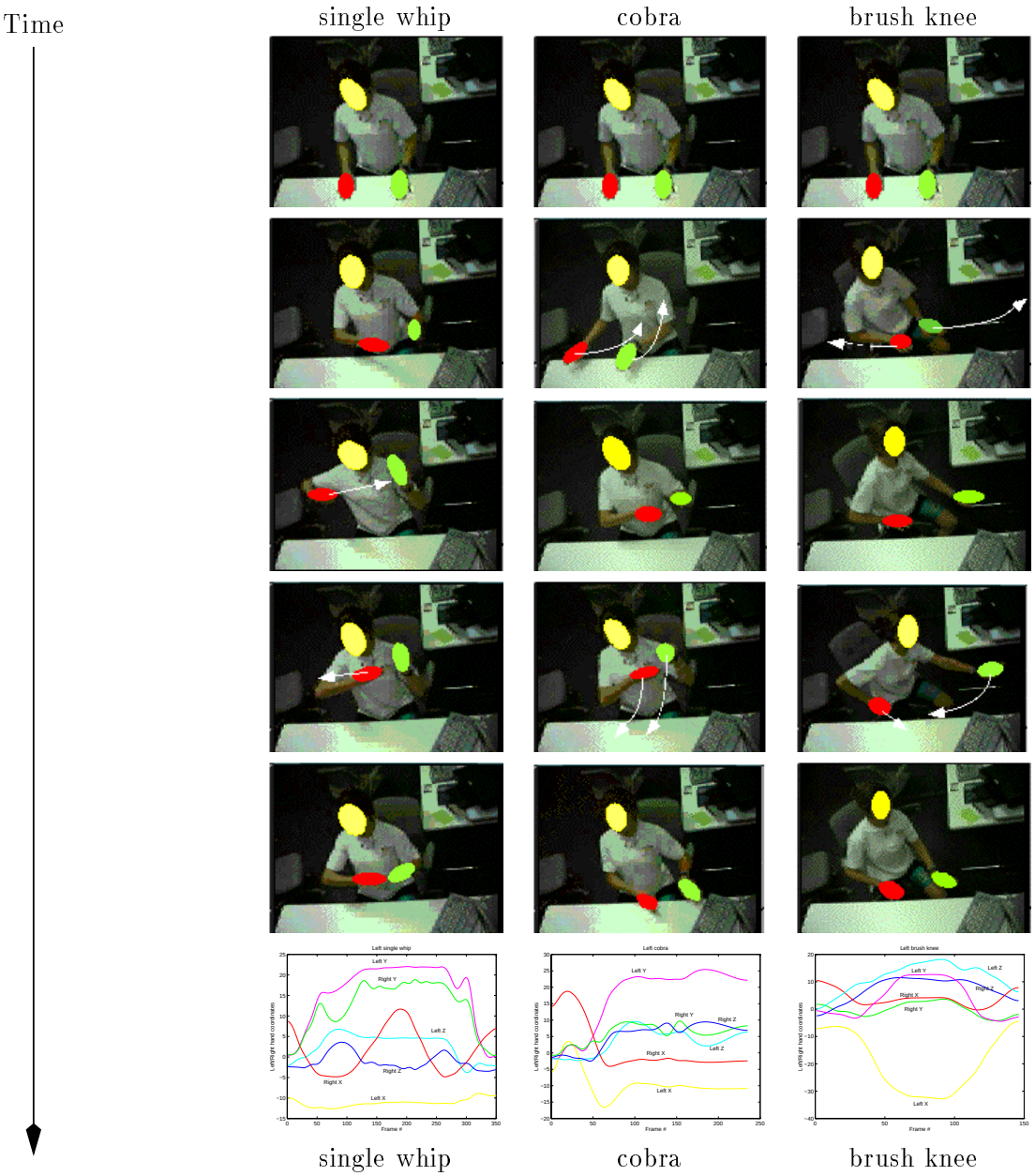
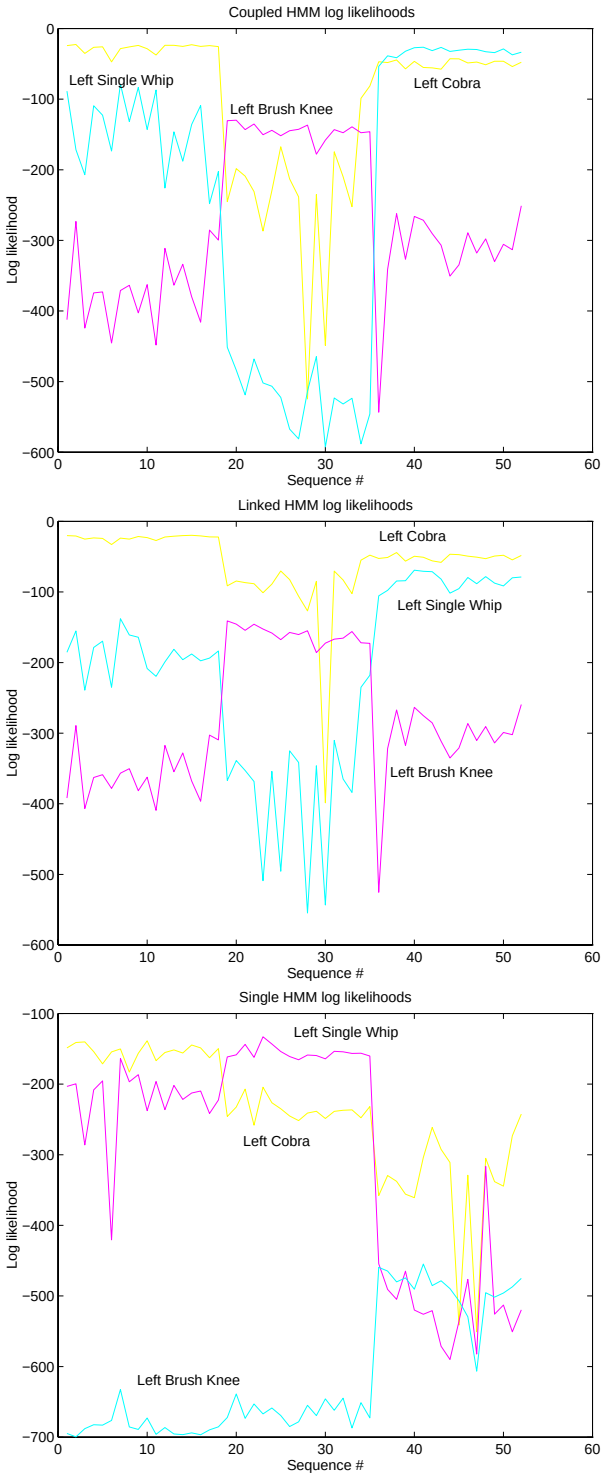


Figure 5-8: Hand tracking of three Tai-Chi gestures: selected frames overlaid with hand blobs from vision. The bottom-most graph shows the evolution of the feature vector over time

| 1–17: single whip | 18–34: brush knee | 19–52: cobra |



| 1–17: single whip | 18–34: brush knee | 19–52: cobra |

Figure 5-9: Classification by the CHMM, LHMM, and HMM, showing per-sequence normalized log likelihood. Only the CHMM attains the right discrimination structure

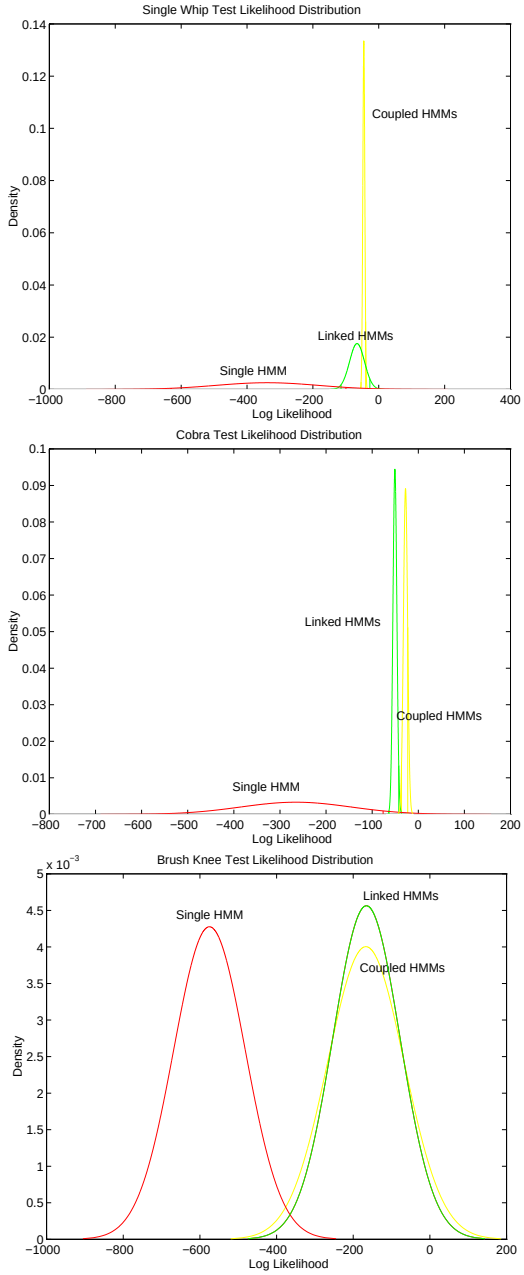
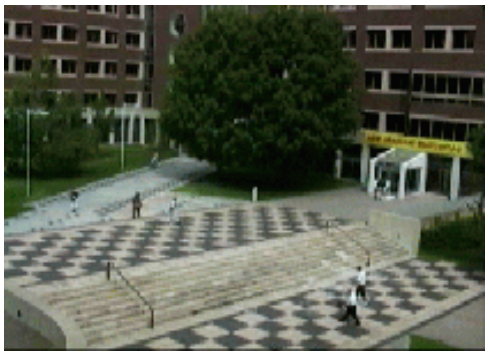
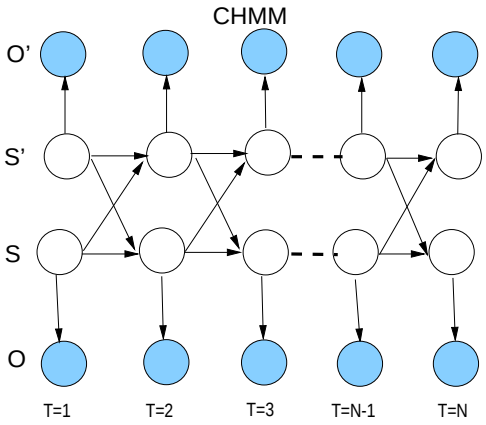


Figure 5-10: Likelihood probability distribution for each HMM type, learning single whip, cobra, and brush knee gestures, respectively. The CHMM consistently has the highest likelihood and the tightest distribution.

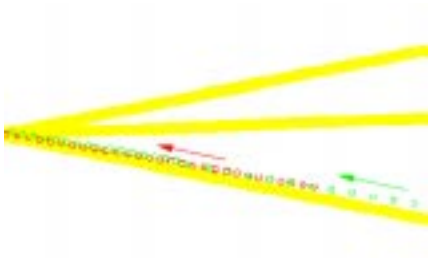


A typical image of a pedestrian plaza

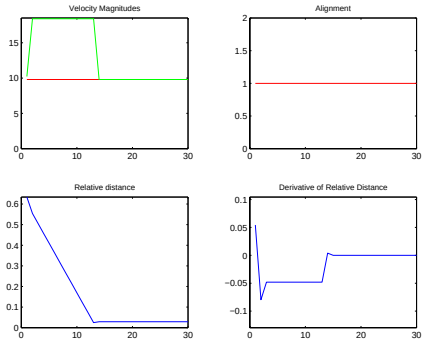


Graphical Representation of CHMMs

Figure 5-11: Visual surveillance system

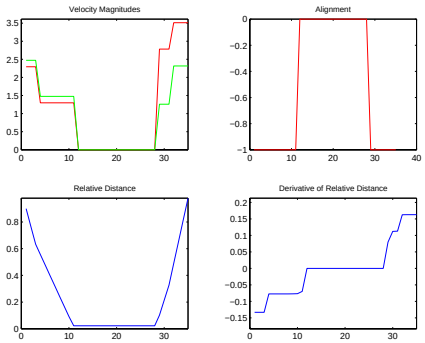


Agent trajectories for inter1

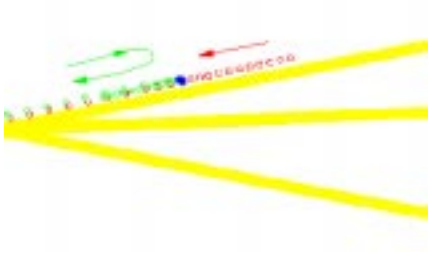


Feature Vector for inter1

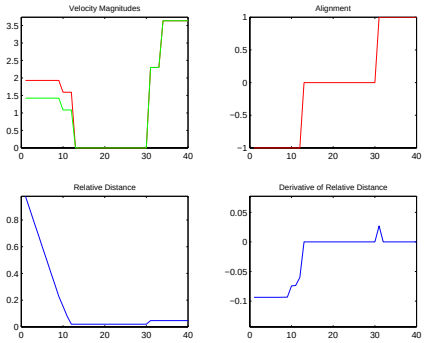
Agent trajectories for inter2



Feature Vector for inter2

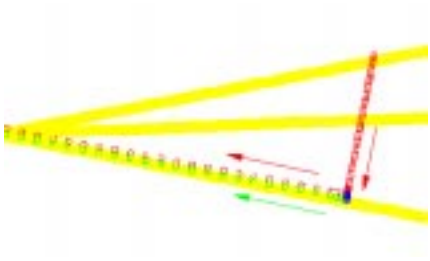


Agent trajectories for inter3

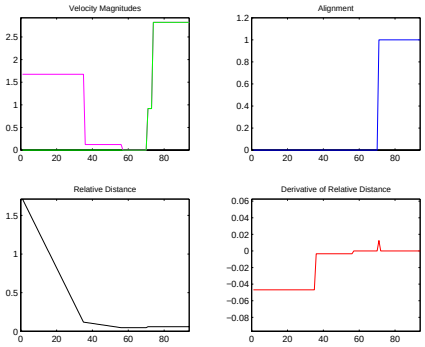


Feature Vector for inter3

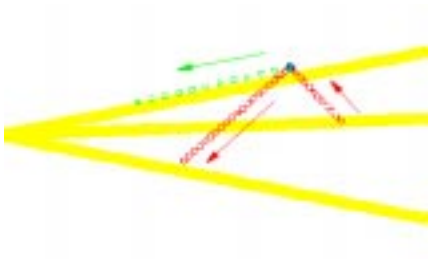
Figure 5-12: Example trajectories and feature vector for the interactions: follow, approach+meet+continue separately, and approach+meet+continue together



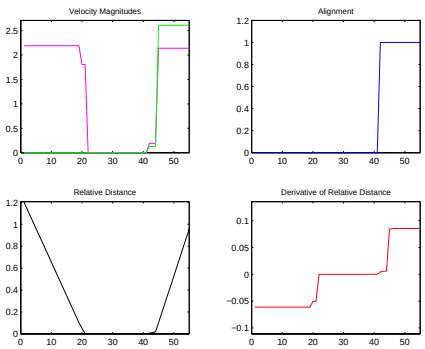
Agent trajectories for inter4



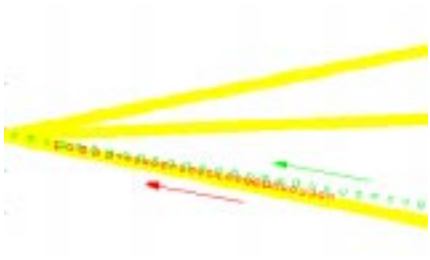
Feature Vector for inter4



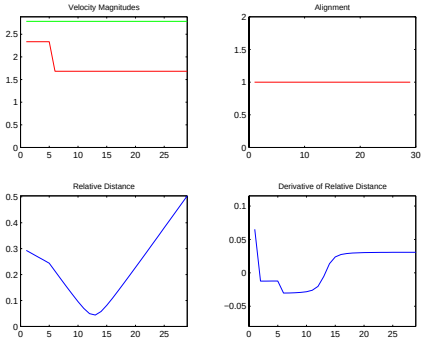
Agent trajectories for inter5



Feature Vector for inter5



Agent trajectories for no interaction



Feature Vector for no interaction

Figure 5-13: Example trajectories and feature vector for the interactions: change direction+approach+meet+continue separately, change direction+approach+meet+continue together, and no interacting behavior

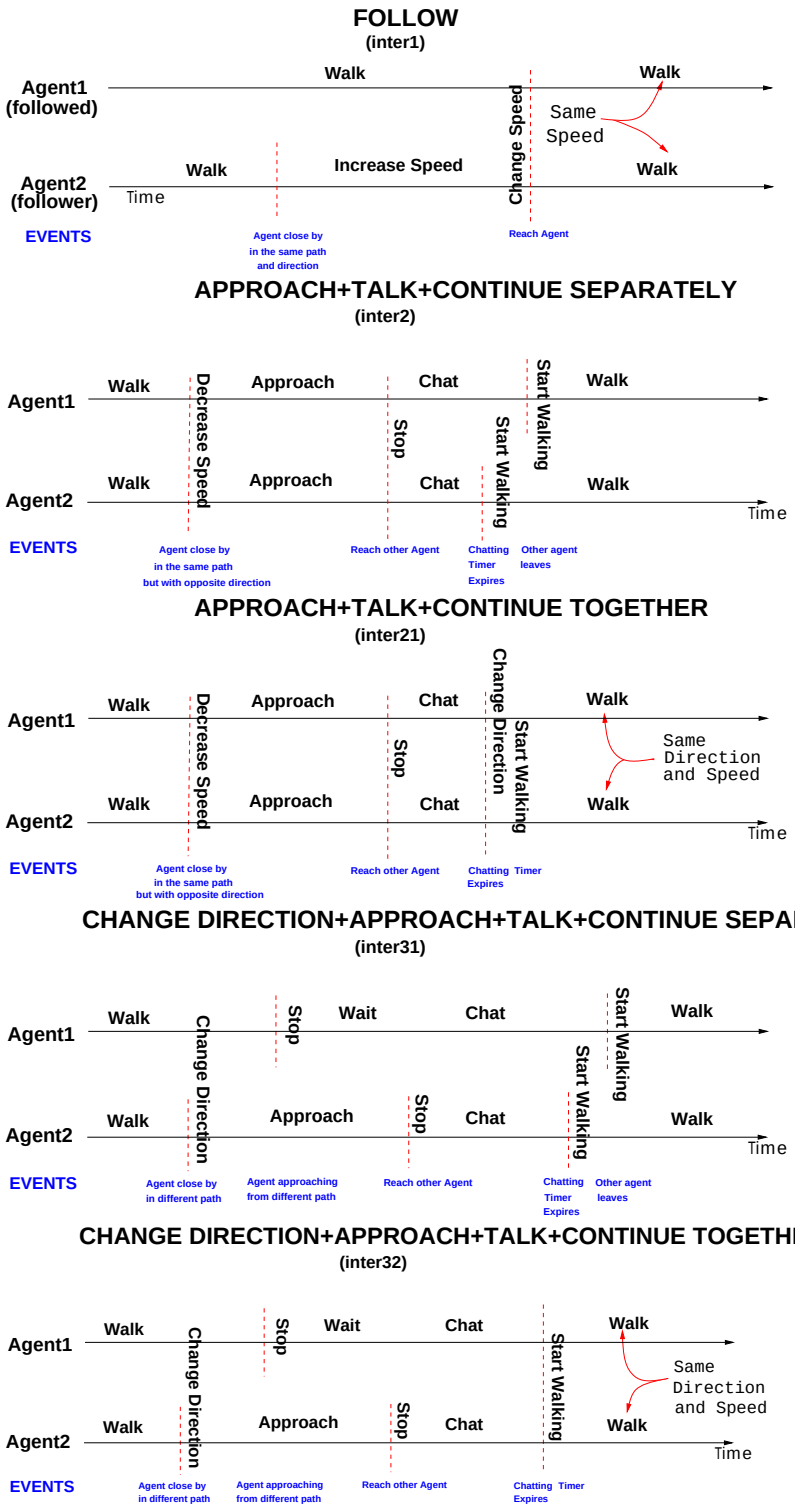


Figure 5-14: Timeline of the five complex behaviors in terms of events and simple behaviors

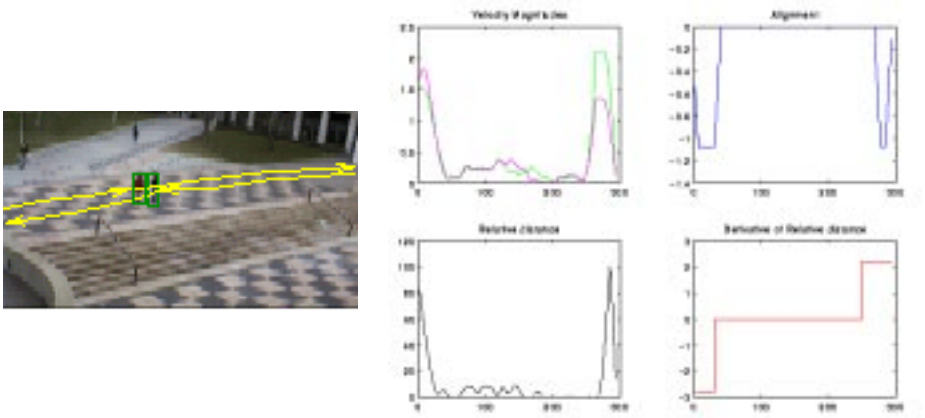


Figure 5-15: Example trajectories and feature vector for interaction 2, or approach, meet and continue separately behavior.

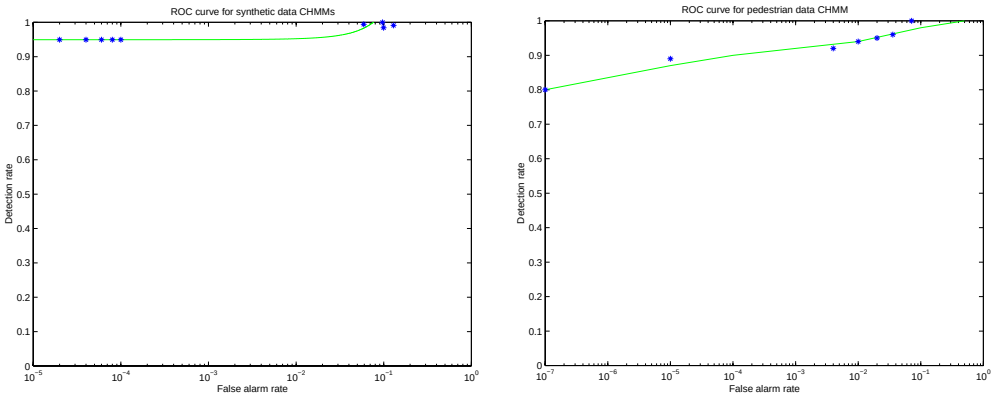


Figure 5-16: *First figure:* ROC curve on synthetic data. *Second Figure:* ROC curve on real human data.

5.5 Dynamic Graphical Models for Driver Behavior Recognition and Prediction in a SmartCar

5.5.1 Motivation

To collect driving data in a real car, I have developed a platform for driver maneuver recording and recognition in a real car. By modeling driver behavior I mean building machine models of typical driver maneuvers, such as changing lanes or passing another car. The goal is that the car, by assisting –as opposed to replacing– the driver, would make of driving a safer and easier experience.

Before the invention of the automobile, most forms of human transportation involved some kind of biological intelligence. For example, a rider could always rely on the horse's self-preservation instincts to avoid obstacles, or its sense of direction to find the way home [237]. Currently, the automobile possesses neither: a moment's inattention on the driver could cause the car to leave its lane, or crash into a nearby vehicle. Government studies attribute 96.2% of accidents in United States to driver error [244]. A large fraction of these deaths could be prevented by the introduction of intelligent systems with the ability of understanding the driver's behaviors and contextual situation. Such systems could, for example, warn the driver or automatically adjust some control parameters in the vehicle to improve safety. In consequence, these "smart vehicles" could somehow recapture the lost intelligence of the first transportation systems. Therefore an important motivation for developing machine models of driver behavior is to improve human driver performance.

Moreover driving is an important, natural-feeling, familiar and culturally assimilated type of human behavior that exhibits complex patterns lasting for several seconds. From an experimental viewpoint, it is important that the number of distinct driving behaviors is limited by the heavily engineered nature of the road system and driving rules. Furthermore, it is feasible to instrument a car to do data acquisition. These characteristics make driving a very suited and interesting testbed for modeling human behaviors.

The fact that humans learn how to drive and improve with experience suggests that a smart car should be able to acquire better and better understanding of driving by being exposed to different traffic situations. Although machine learning has been successfully applied to operational-level tasks, such as lane-tracking [190, 189] and trailer-truck docking [266], there are remarkably very few systems that have even attempted to apply machine

learning to tactical-level driving [266, 184, 70], and even fewer, if any, that have modeled driver behavior in real driving situations beyond simulators. This is an important contribution of this thesis.

One critical issue in machine-human interface systems are the transitions between manual and automated operation. They should be as seemless and smooth as possible. Such transitions would occur, for example, when the system encounters non-supported situations, when it fails, returning to manual mode; or when initiated by the driver. In any case, it is very important not to interfere with the driver's intended maneuver, specially in emergency situations, and to avoid discontinuities in the system, inducing feelings of incongruity while driving. Therefore, developing systems for predicting the driver's next maneuver or inferring driver's intentions is imperative to facilitate smooth and appropriate control mode transitions.

Building effective driver behavior recognition methods requires a thorough understanding of driver behavior and the construction of a model capable of both generating and explaining the drivers' behavioral characteristics. The task of driving has traditionally been characterized as consisting in three different levels: strategic, tactical and operational [103]. At the highest (strategic) level, a route is planned and goals are determined; at the intermediate (tactical) level, maneuvers are selected to achieve short-term objectives –such as deciding whether to pass a blocking vehicle–; and at the lowest (operational) level, those maneuvers are translated into control operations. In this thesis I focus on recognizing driving maneuvers at a tactical level. Namely, I have built models of passing, changing lanes right and left, turning right and left, starting, and stopping.

Previous studies in psychology have found that driver behavior can be characterized as a sequence of basic actions each associated with a particular state of the driver-vehicle-environment system and characterized by a set of observable features [90]. This studies support the computational model proposed in this thesis for human behavior modeling in general and driver maneuver recognition and prediction in particular.

5.5.2 Previous Work

Human driver modeling is an interdisciplinary endeavor involving a number of fields including robotics, psychology, control theory and statistics. Driving in a real-life traffic situation is a very difficult task because good decisions need to be made given only incomplete in-

formation in real time. Traditional AI techniques such as search-based planning [68] are infeasible for at least two reasons: most of these methods cannot function under noisy, uncertain conditions, and the state-space is extremely large if realistic maneuvers such as aborted lane changes are taken into account.

The tasks and subtasks involved in human driving could be comprehensively listed as, for example, McKnight and Adams do in [149]. Unfortunately, since their work is not directed towards computer implementations, many tasks are difficult to express computationally. Some recommended actions such as '[Driver] selects lane relative to car's speed, maneuvers and traffic flow' are too vague, while others are contradictory. Although human drivers may be able to understand such tasks easily, the issue of conflict resolution is not addressed. Despite these shortcomings, the heuristics may provide good starting points for rule-based modules. Imprecise rules such as 'Traffic behind should be checked about every five seconds when there are vehicles also ahead' [262, 149] may be robustly captured by a fuzzy [127] formulation.

Most of the projects developed in the Intelligent Vehicles community are directed towards automatic navigation of a vehicle. In the majority of the cases the previous work in tactical-level driving has concentrated on expressing driving knowledge in the form of rules [167, 130], hand-crafted decision trees [203, 239], finite state machines [50], Dynamic Bayesian Networks (DBNs) [70] or HMMs [184]. Expert systems have been used in other related fields with relatively promising results [193, 210]. In [70] a new approach for autonomous vehicle driving in normal traffic is proposed. The authors describe the problem as a decision-theoretic architecture using dynamic probabilistic networks (DBNs) to represent and update the belief state. The decision making process is modeled following three approaches: Partially Observable Markov Decision Process (POMDP), dynamic decision networks and decision trees. Furthermore there has been relatively substantial research in the autonomous agents community for building autonomous intelligent vehicles in simulated environments. One of the most sophisticated systems is the one presented in [240], where Sukthankar et al propose a distributed reasoning system (PolySAPIENT) with a novel evolutionary optimization strategy (PBIL) for the tactical level of driving. However most of these systems have only explored greatly simplified aspects of the driving task and none of them has focused on modeling (learning) explicitly the interactions between the driver and the surrounding traffic.

Driving Taxonomy

The task of driving has traditionally been characterized as consisting in three different levels: strategic, tactical and operational [103]. At the highest (strategic) level, a route is planned and goals are determined; at the intermediate (tactical) level, maneuvers are selected to achieve short-term objectives –such as deciding whether to pass a blocking vehicle–; and at the lowest (operational) level, those maneuvers are translated into control operations. This driving taxonomy intersects with the proposed general taxonomy presented in chapter 1, section 1 as follows: strategic driving corresponds to the complex intentional behaviors with substantial extent on time; tactical driving corresponds to the so called "communicative behaviors" that involve the interactions with other possible agents; finally, driving at a operational level corresponds to simple, short actions.

Mobile robot research has successfully addressed the three levels to different degrees. Strategic-level planners [205, 254] have advanced from research projects to commercial products. The operational level has been investigated for many decades, resulting in systems that range from semi-autonomous vehicle control [144, 74] to autonomous driving in a variety of situations [59, 191, 238]. Substantial progress in autonomous navigation in simulated domains has also been reported in recent years [50, 203, 199]. However the decisions required at the tactical level are difficult and a general solution remains elusive.

Tactical-level driving is characterized by the constant battle between long-term goals and real-time constraints. Drivers must select appropriate maneuvers such as lane changing, accelerating, and car following given very little knowledge of the intentions of other drivers in their environment. In this complex and dynamic problem space, optimal solutions are rarely to be found, but the penalties for bad decisions are clear and severe. Unfortunately safety cannot be guaranteed, even by conservative driving. Tactical driving thus forces a careful balance between competition and cooperation: aggressive maneuvering is successful but not when it results in a crash. Because of these reasons, the driver behavior modeling framework proposed in this thesis seems quite suitable and appropriate

In the following I will briefly review some models that have been proposed for explaining driver behavior at a tactical level.

Task Models Task models define the broad tasks involved in driving (e.g. car following) and decompose these tasks into detailed subtasks (e.g. headway maintenance). In [149]

a comprehensive treatment of the situations and actions involved in driving is presented. However since their report was targeted towards human driver education, most of their descriptions are too vague to be directly used in tactical computer systems.

A second difficulty with most task models is that the recommendations are often contradictory. As Reece [203] notes, the McKnight and Adams task list includes two subtasks that instruct drivers to "observe pedestrians and playing children" and to "ignore activity on the sidewalk that has no impact on driving" without providing any insights as to which sidewalk activities have no impact on driving. Since these discriminating between these situations requires "common sense" encoding this knowledge in the form of driving rules for a reasoning system is challenging.

Task models are nevertheless useful for two reasons. First, they highlight aspects of the tactical driving task that need to be addressed by a smart vehicle. Second, they provide insights about mapping observable phenomena into specific conditions (e.g. the driver should initiate an overtaking maneuver in response to a slower vehicle ahead).

Risk Models Risk models for driving have emerged from psychological research in the area of perceived risk. By combining the decision theoretic notions of expected utility and the willingness of humans to take "acceptable risks", these models attempt to explain commonly observed phenomena such as speeding, aggressive driving styles and intoxicated driving. Intelligent systems exhibiting some degree of situation/contextual awareness may require sophisticated models of human drivers. Utility functions based on perceived risk (such as time-to-impact measures) can also be used by reasoning systems to select tactical-level maneuvers. A representative example of a risk model is *Wilde's Risk Homeostasis Theory*. Risk Homeostasis Theory maintains that, in any activity, people accept a certain level of subjectively estimated risk to their health, safety, and other things they value, in exchange for the benefits they hope to receive from that activity [267].

Counterintuitively risk homeostasis theory predicts that humans adjust their behavior so as to maintain –rather than minimize– their perceived risk at a constant *set-point risk* level: The degree of risk-taking behavior and the magnitude of loss ... are maintained over time, unless there is a change in the target level of a risk [267].

A case study, known as the *Munich Taxicab Experiment* [9] was conducted to test the implications of risk homeostasis theory under controlled situations. Some vehicles in a taxi

fleet were equipped with an anti-lock braking system (ABS) that allowed drivers to maintain steering control during hard braking on slippery roads. Conventional wisdom predicted that the ABS-equipped vehicles would be safer than unequipped vehicles. Surprisingly the results [9, 267] showed that:

- Among the accidents involving the company's taxis, there was no statistically significant difference between the involvement rate of the two vehicle types (in fact, the ABS vehicles were involved in slightly more accidents).
- Accident severity was independent of the presence or absence of ABS in the taxi.
- Accelerometers installed in the taxis measured more extreme decelerations (associated with hard braking) in vehicles equipped with ABS.
- Drivers in ABS cabs made sharper turns in curves, were less accurate in lane-keeping behavior, maintained shorter headway distances, made poorer merge maneuvers and created more "traffic conflicts". All these differences were statistically significant.

Therefore risk homeostasis theory, as supported by those experiments, has pessimistic predictions for any attempt to improve highway safety solely through technology. However in the context of tactical-level reasoning, risk homeostasis theory provides support for utility-based approaches to situation awareness.

Information Processing Models Information processing models focus on process control application domains, where the operators have to control dynamic processes of considerable complexity. To do this they must rely on, and successfully interpret, large quantities of complex information about the process state. The consequences of failing to attend to, and correctly interpret, such information can be as costly as the life, such as in driving. Providing the right "quality" of information for operators has therefore become a key design goal for interface designers. I will describe some of the information processing models proposed that are relevant to the driving task:

1. The Operator Functional Model [113] allows the functions of the operator, which are modelled as a transition graph at the highest level, to be naturally decomposed into sub-functions, tasks and actions in lower level graph representations. Nodes in the graphs are functions and tasks etc., whilst the arcs represent events which trigger

transitions between the nodes. The ability to represent such events makes it more suitable for dynamic, non-deterministic environments.

2. In the Problem Behavior Graph [165], nodes represent states of knowledge or system messages and the connecting arcs are operator tasks and actions. This model was found to be unsuitable for dynamic environments since it has no mechanism for modelling unexpected events such as failures. Each time a new problem context arises, a whole new sub-graph must be constructed.

3. The Decision Ladder [201] represents decision episodes in a natural manner in separate diagrams although the transitions between different decision episodes are not modelled. The Decision Ladder was developed from analyses of process operators and is therefore suited to this domain. However, it does not provide a formalism sufficient, in itself, for system design.

4. The Goals-Means Network [96] approach differs from the other models in that it models the structure of complete systems rather than just the human operator. It is particularly useful for representing the relationships between the system objectives and the actions required by the operator to achieve the goals. It is less suited for representing changes in goals and means.

Operator models have usually been developed for the purpose of design. The Decision Ladder is based around the classification of information processing into three types: skill-based, rule-based and knowledge-based ([201]). The premise for this theory is that humans are equipped to control their environment according to abstract objectives and there are three layers of processing which help them in achieving this.

At the bottom of the hierarchy are the sensory and motor skills (*skill-based behavior*). These acts require no conscious control, and function independently of central processing and working memory. Skill based behavior is often exhibited as people learn to master a task involving sensimotor processing. Legge and Barber ([140]) describe several theories about the nature and acquisition of motor skills. In summary, it is believed that our physical abilities are controlled in two ways. At the most basic level there are direct connections between stimulus and response. Tracking an object round a screen with a joystick driven cursor is an example of such behavior. Each time an action is taken (change direction or speed) the response must be observed and used to determine the next action. Thus it is a

stimulus-response chain which forms a closed loop with the environment. However, human beings are also capable of ordering their motor system to perform sequences of actions without relying on feedback between actions. Writing a signature is an example of this. Composite actions can be performed in the absence of feedback by executing what is called a motor program. Such programs (skills) can be built up with practice, and be controlled by higher levels of cognitive behavior.

In particular, a sequence of actions may be activated by a stored rule or procedure. This is rule-based behavior or "know-how". Rules and procedures may be derived through practice or learned from other people or instructional material. The rules which dominate are those which have been shown to be effective in reaching a goal. As Rasmussen ([201], p. 102) says; "The control evolves by the survival of the fittest rule". This selection process may not be conscious. Rules are, rather inexplicably, triggered automatically from given states of knowledge.

In contrast, the highest level - knowledge-based behavior - is invoked by the absence of previous experience of a situation. In order to handle such a situation, the goal must be stated explicitly, rather than being implicit in the chosen rule, as described above. The environment must be analysed and plans for controlling it must be considered. This process is thought to be aided by a mental model of the environment that is being controlled.

Boer et al. [26], [27] have proposed an integrated driver model (IDM) which borrows from Rasmussen's and Michon's [103] model and incorporates the concept of the dynamic aspects of driver behavior as well as an important role of driver needs. Incorporating the idea of attention management, this model focuses on the switching of intra- or inter- process levels. It can explain the selection of maneuvers in manual driving but also the operation of mode transitions in driving assistance systems. An understanding of attention management or the characteristics of each process level is closely related to an understanding of the driver's intentions.

Rasmussen's taxonomy provides a framework for the understanding of information processes of human perception and cognition. It also broadly maps behavioral types on to the Decision Ladder. In the initial and final phases of the decision task, skills are required. Direct stimulus-response behavior occurs from the attention module to the execution module. If the problem is relatively well known, a stored rule will be triggered from which shortcuts can be used to move from observations to direct task selection. Rule based behavior

resides in the middle. The knowledge based domain is at the top where goals are explicit, the environment is considered in an abstract way, and plans are evaluated. However, the way in which the types of behavior are used in conjunction with each other is complex. Although diverse problem-solving strategies can be used to reach a goal ([87]), it is believed that people tend to use lower levels of behavior if at all possible ([201]). Essentially human beings minimise resource usage, and will consequently choose the path of least resistance.

Another point worth noting is that skill-based behavior is developed through practice. Learning a motor program requires a clear statement of the objective. By repeatedly evaluating the outcome against the objective and hypothesising what actions are needed to match the two, such programs can develop. Thus knowledge-based behavior is employed to develop skill-based behavior. Early learning can be aided by procedural instructions (rule-based) such as those received from a driving instructor on how to execute a smooth gear-change, but real proficiency requires practice as well. An interesting fact is that motor skills, such as riding a bike or driving a car, can be continuously improved after we have become proficient and therefore do so without conscious attention. This second phase of learning takes little mental effort and seems to involve matching of stimuli against an ideal pattern at a subconscious (skill-based) level.

The problems associated with determining the information processes underlying observable human behavior are numerous. Rasmussen's work is, to a large extent, based on verbal protocol analysis of different professional problem solvers ranging from radio technicians to nuclear power plant operators. The studies have been conducted over many years and constitute a substantial body of knowledge in this field. However, it is acknowledged that verbal protocol analysis may not always reveal the underlying processes. People are good at articulating rules and procedures, whereas knowledge based behavior is less easy to verbalise.

Perceptual and Motivational Models Perception models have been used to describe driver behavior in accidents [251], suggest methods for safer driving [262, 264] and motivate new collision warning devices [10]. In the tactical driving domain, perceptual models are particularly relevant in two areas: sensor modeling and driver intentions.

Sensor modeling at the operational level is primarily concerned with tracking objects and segmentation (low-level actions which humans typically take for granted). At the tactical

level, the focus shifts to reasoning about object-to-lane mapping, blind spots and occlusions –task which human drivers perform more consciously–. Unsurprisingly novice drivers are likely to forget about vehicles which are not currently visible [264]. Perceptual models also lead to heuristics for safer driving which can be exploited by both humans and intelligent vehicles (e.g. "At night do not over-drive the range of your headlights").

A smart car must be sensitive to its driver's intentions. Perceptual models can be used to gain some insights into this area. Recent research [182], [214] shows that drivers' eye fixation patterns are strongly correlated with their current mental state. Previous studies have found that driver behavior can be characterized as a sequence of basic actions, each associated with a particular state of the driver-vehicle-environment system, and characterized by a set of observable features [90]. In [184], Pentland and Liu researched the modeling of human action taking into account this observation. Therefore they modeled driver behavior as a transition of states internal to the driver. They claimed that only driving actions can be observed and proposed a driver intention and detection method using a four-state Hidden Markov Model (HMM). This model was intended to capture the sequential nature of these unobservable internal states that are each associated with a set of observable variables. Once the HMM had been trained the system was able to predict when the driver is about to brake or turn. This knowledge may then be used by the smart car to optimize its behavior for the expected maneuver –in some sense, the situation awareness is shared over the vehicle-driver system. In this thesis I extend this framework to include the influence of surrounding vehicles (larger and more complex context).

The notion of driver's internal state is fundamental to motivational models. In this framework, perceptual information is integrated with discrete mental states in an attempt to predict the actions that the human driver would take in that given situation [245]. This description of human cognitive activity can also involve aspects from utility theory –generally in the form of a perceived risk factor–. However since they do not describe how driving knowledge is represented, a large gap exists between the motivational model and its successful implementation. Although some efforts have been made to specify motivational models in a symbolic programming language [2] no successful implementation currently exists.

Control Models Control models for drivers are primarily important when modeling operational level phenomena. For example, the well-known *"Two second rule"* for car following [163] is based on the observation that humans require approximately 1.75 seconds to identify and react to a potentially dangerous situation [149]. Lane-keeping and steering models such as pure-pursuit tracking [255] are valuable at the tactical-level. First, such models can help the intelligent vehicle predict the future likely positions of observed vehicles. Second, such models can allow the reasoning system to estimate the time needed to execute a given maneuver (such as a lane change). Control models can also be applied to plan recognition.

Other control models have been developed in the traffic simulation domain. The ones of most interest to tactical driving research are those which model lane-changing [259, 4], car following [277], and emergency maneuvers [6]. Since these models are computational, they could be directly incorporated into a tactical reasoning system.

Decision-theoretic models The most successful approaches to modeling tactical-level driving fall in the framework of decision theory under uncertainty (probability). Forbes et al [70] propose in the BATmobile project a decision-theoretic architecture using dynamic probabilistic networks. The architecture provides a sound solution to problems of sensor noise, sensor failure and uncertainty about the behavior of other vehicles and about the effects of one's own actions. The real-time decision making is implemented in an approximate fashion using three different approaches: (1) dynamic decision networks which incorporate action nodes and an explicit utility function; (2) hand-coded, explicit policy representations –such as decision trees– that take as input the joint probability distribution encoded in the DPN; and (3) supervised learning and reinforcement learning methods for solving a POMDP, in which they learn a policy representation, a utility function on belief states or an action-value function on belief-state/action pairs. The design of appropriate utility functions as well as the use in a real vehicle are two major deficiencies of their approach. In [239] two tactical-level reasoning systems are proposed, MonoSAPIENT and PolySAPIENT, to drive autonomously in simulated traffic. A very complex decision tree is utilized in MonoSAPIENT. However MonoSAPIENT tree's complexity is unmanageable. Therefore PolySAPIENT distributes this complicated tree in separate experts tied to relevant physical entities in the driving environment.

5.5.3 Summary

As with any other complex and interesting human behavior problem, there is a very broad and rich spectrum of proposed models of driver behavior. From all of them, the closest work to this thesis work is that of Pentland and Liu [142, 184], and that of Kuge at al [131]. In [184] Pentland and Liu develop a computational state-based model of driver behavior. They model the driver's internal state as a four-state Hidden Markov Model (HMM). Once the HMM has been trained the system is able to predict when the driver is about to brake or turn. This knowledge may then be used by a smart vehicle to optimize its behavior for the expected maneuver –in some sense, the situation awareness is shared over the vehicle-driver system. In a similar way, Kuge et al. present a HMM method that characterizes and detects lane changing maneuvers. The authors focus on information processing models of human driver behavior generation and utilize them to adopt a model based approach in the development of a lane change detection and recognition model. The primary components are skilled low level maneuvers whose initiation is managed by higher level decision making components. Perceptual models can be used to gain some insights into this area. Recent research [182] shows that drivers' eye fixation patterns are strongly correlated with their current mental state. Other more constrained but certainly important aspects of driver behavior were estimated by few early methods, such as, for example, lane change intention [160]. However, none of these methods was human model-based.

Pentland and Liu validated their model in an experiment conducted in a driving simulator. The objective of that validation test was to recognize different driving maneuvers at a tactical level, such as a right turn, a left turn or stopping. In order to apply such a model to a driver assistance system, it is necessary to assess to what degree the HMM based behavior recognition model also provides a plausible model for human behavior generation. This knowledge may not only offer better insight into selecting a particular HMM structure but also provide better insight into potential limitations of the characterization in situations that were not part of the training set used to fit the HMM parameters.

None of these previous systems, however, incorporates contextual information when modeling driver behavior. Nonetheless, knowledge of the context is necessary to properly make decisions in complex dynamic environments such as driving. Psychologists attribute this competence to a task-specific understanding of the situation, termed *situation awareness*. I this thesis I develop machine models of driver behavior that incorporate elements

of situation awareness for tactical driving.

There is today strong research efforts invested in developing partially or fully automated driver assistance systems. For example, headway distance control or lane keeping control systems, which make use of Intelligent Transportation System (ITS) technologies [66, 100]. To achieve such assistive systems, it is important to adopt approaches aimed at improving the performance of the driver-vehicle-context cooperative system by regarding driving as an interaction between the driver, the vehicle and the surrounding road information and traffic.

Finally, it has also been argued that laboratory research of SA should be conducted under conditions that afford as much realistic behavior as possible. Due to the simplicity of most car simulators, specially the lack of realism of the computer generated automated cars, the experiments carried out in this thesis took place in a real car while driving in the greater Boston area.

To summarize, the work of this thesis on driver behavior modeling extends Pentland and Liu's framework [142, 184] in several ways: (1) I model a larger number of maneuvers at a tactical lever –namely seven–; (2) I show that contextual information is critical for the accurate recognition of some maneuvers; (3) I use real data collected in an instrumented car, as opposed to using a car simulator.

5.5.4 Modeling Issues

From a computational viewpoint, much of the previous work done on intelligent vehicles has been from the perspective of robotics –specially in the case of autonomous vehicles–. Traditional approaches to robotics [68] structure the processing cycle in three stages. In the first stage, sensors gather information about the world and convert it to a symbolic internal representation, known as the *world model*. In the second stage, the world model is processed by AI algorithms (typically involving planning and search) to find a course of action for the robot to achieve its goals, known as a *plan*. In the final stage, this plan is executed by the robot as a series of *actions* (actuator commands).

This process suffers from several serious drawbacks. First, the approach implicitly underestimates the role played by perception [203]: due to sensing constraints and uncertainties the world model is likely to be both incomplete and partially incorrect. Second, the process assumes that it is possible to plan a complete path from the robot's initial state to the

desired final state. This is infeasible (particularly in real-time) in most complex domains due to an explosion in the number of searchable states, and the inability to perfectly predict the outcomes of actions. Third, the chosen plan cannot be guaranteed to execute perfectly –the robot may be forced to react immediately to unforeseen problems (very likely to take place, given an incomplete world model). Consequently current mobile robot architectures recognize that planning-heavy approaches to real-time problems in dynamic environments –such as driving in traffic– are infeasible [209].

The information coming from the SmartCar's sensors is both noisy and incomplete. Therefore one would need a modeling architecture that allows for incomplete, missing data. Dynamic Graphical Models are suited for that task. In this thesis work, however, I have not developed a sophisticated user interface that would let the SmartCar actively take the appropriate actions, depending on the current situation. However the information needed for building such an interface should be available from this thesis work. Therefore, even though it is not the subject of this thesis to develop such an interface, the contributions of this thesis will move forward towards more intelligent, personalized and pro-active user interfaces in cars.

Modeling the world Information extracted by the perception modules is assimilated into a representation of the world. As described in section 3.3, there are at least three different aspects relevant to tactical-level driving: (1) Smart car self-state, including physical and mental components; (2) road state, including road geometry and exit information; (3) traffic, speeds, relative positions and hypothesized intentions.

- **Physical State:** Consists of information sensed from the speedometer, steering wheel angle (rotary potentiometer), gear, brake pedal, and acceleration throttle. At the tactical level, the important elements include: current speed, current steering curvature, and current lateral displacement (distance from center of current lane).

- **Mental State:** In the experiments, eight different driver behaviors at the tactical level have been collected and modeled: overtake, change lane right/left, turn right/left, start, stop and merge. Each of these actions is decomposed in a number of finite subactions that correspond to the hidden states of the HMMs.

- **Driver's Head Pose and Facial Expressions:** The ELMO CCD camera mounted on the steering wheel provides real-time information of the driver's head pose and facial expressions. The driver's head pose, gaze and expressions convey information about his mental, emotional and physical states. For example, recent research [182] shows that drivers' eye fixation patterns are strongly correlated with their current mental state. Sleep researchers say that driver drowsiness accounts for as many highway accidents as drunkenness. Studies show that as many as one in 20 Americans have fallen asleep at the wheel. Drowsiness accounts for 30 % of fatal crashes, one study says equaling the number of fatal crashes blamed on alcohol intoxication. Sleepiness slows reaction time, decreases awareness and impairs judgment, just like drugs or alcohol. And just like alcohol and drugs, sleepiness can contribute to a collision. It has been accessed that accidents caused by drowsy driver are extremely severe because the vehicle collides with an object or other cars at full speed without the application of the brake. Many of these accidents could potentially be avoided with a warning sytem that would monitor the driver's eye movement patterns and head pose, detecting anomalous behaviors. Similarly an analysis the driver's facial expressions could detect dangerous extreme emotional states –such as anger– and have the car take some actions to correct them.

- **Road State:** Road state refers primarily to information about the road, such as the lane positions or eventually on-board navigation systems (digital maps combined with GPS). Aspects such as the limits of the road or the number of lanes constrain changing maneuvers while speed limit signals and closed curves constrain speed choices. The model should also have a record of eventual nearby exits or already known road obstacles.

- **Traffic State:** Modeling other vehicles is the most important aspect of tactical driving. While current speeds and relative positions of the surrounding vehicles can be ascertained in a relatively straightforward manner, their future behavior cannot. Therefore the SmartCar is forced to make some hypotheses about the other vehicles' intentions. Experienced human drivers can often predict the behavior of other drivers with surprisingly accuracy [149]. However this involves both a large database of world knowledge –equivalent to the driver's common sense and experience– as well as more

sophisticated perception than the state of the art in computer vision. All other previous systems that make the unrealistic assumption that vehicles will continue to drive in their current lanes at their current velocities over some prediction time interval. However in this thesis I propose a novel framework for modeling interactive behaviors that could be utilized in a driving situation. Using this framework one could predict the most likely actions that the car and surrounding traffic would do next, assuming a relatively simplified world and 'normal' (average) drivers.

Note that the models proposed in this thesis do not incorporate higher level variables such as driver's emotional state (frustration levels, tiredness, politeness, sleepiness).

5.5.5 SmartCar Experiments

To evaluate a system that models driving behaviors at a tactical level, quantitative measures of performance are desired. A major emphasis on this part of the thesis work is how context affects the driver's performance of a specific maneuver. To evaluate the model's performance, I carried out a large driving experiment in a self-instrumented Volvo with real traffic. The measure of performance has been the recognition accuracy of the driving maneuvers on labelled testing data. In the following, I will describe in detail the experiments and results on driver maneuver recognition and prediction.

Apparatus

An instrumented automatic Volvo V70XC (1998) was used to measure driver behavior data. The car sensors have been described in section 3.3, chapter 3.

Procedure

I carried out a set of experiments on the SmartCar platform in real traffic situations over a period of 2 months. The experiments took place on sessions of about 1.15 hours at four different times during the day (8 : 30am, 10 : 30am, 12 : 30pm, and 2 : 30pm). The task consisted of driving a circuit in the extended Boston area. The designed driving circuit includes both urban and highway sections. Figure 5-17 depicts the route followed in the experiments.

Over 70 drivers participated in the experiment. The drivers were asked to sign a consent

form (included in appendix 1) before starting the experiment. They were rewarded $20 for participating.

A driving instructor was with the driver throughout the experiment. The instructor set up the hardware and software for each of the experiments, gave directions to the driver about where to go and labelled the driving maneuvers as they took place using the laptop computer and the LabVIEW GUI, described in chapter 3, section 3.3. Because the focus was on predicting what is the most likely maneuver to take place next, the driver was requested to verbally report his/her **next intended action before** carrying it out. The four video signals were recorded for the entire route. The car signals, however, were only recorded when a maneuver was about to happen. A time window of 2 seconds was used, i.e. the car signals were recorded starting 2 seconds **before** the driver reported his/her intentionality to perform a maneuver. Both the video and car data was time stamped (the VCR and the laptop clocks were synchronized before every session). The driving maneuvers that I collected data for are: passing another car, turning right and left, changing lanes right and left, starting, stopping and merging.

Figures 5-18 and 5-19 show typical car and context signals in one example of a 'passing' and a 'turning' maneuvers collected in the experiments. Note how, in the case of passing, the car signals contain little information about the maneuver type, whereas the gaze and lane are much more relevant features.

After the driving task was completed, the drivers were asked to fill in a questionnaire with basic questions about their driving experience, skills and the experiment. A copy of the questionnaire is included in appendix 1.

Data Post-processing

To train the driving behavior graphical models, signals from different nature need to be synchronized and combined in the same feature vector. During the driving experiments, the laptop's and VCR clock's were synchronized to guarantee the temporal alignment of the car and video signals.

The contextual information was acquired via the video signals. I have developed a video processing graphical environment (GUI) that let's the user record, playback and annotate the video signals coming from the front, rear and face driver cameras. Contextual information –such as the driver's gaze, the relative position of the road lanes or the relative

position, velocities and direction of the surrounding traffic– was manually annotated (using the video annotation GUI) for each frame and maneuver. Table 5.5 summarizes the contextual information that was annotated.

	Front and rear traffic	Driver's face	Road lanes
Position	Right/Left/Same	Front/Rear-view mirror Right mirror/Left mirror Right/Left	Right/Left
Relative Speed	Slower/Same/Faster		
Relative Distance	Far/Medium/Close		
Direction	Same/Opposite		
Representation	Rectangle	Rectangle	line

Table 5.5: Information from the video annotation process

Due to the different sampling rate on the car and video signals, the car data was subsampled to match the video frame rate. The final sampling rate was of approximately 30 samples/s. All the continuous signals were low-pass filtered using Butterworth filters.

Driver Maneuver Recognition and Prediction

Using the car, driver's gaze and road lane data, HMMs for each of the maneuvers to be recognized were built. The performance on recognition (accuracy) of the best HMMs trained with different feature vectors was evaluated:

1. Only car signal data: brake, steering wheel angle, gear, and acceleration throttle.
2. Car data and lane position information (front and back lane positions).
3. Car data and driver gaze information.
4. Car data, lane and driver information.

The gaze was a discrete signal with 6 possible values: (1) front road, (2) rear window mirror, (3) right mirror, (4) left mirror, (5) right and (6) left.

In the case of the lanes, a single value was computed from the (x, y) image coordinates of the extrema (first (x_1, y_1) and last (x_2, y_2) points) of the road lanes:

$$lane_i = \text{atan2}(|y_2 - y_1|, |x_2 - x_1|) \quad (5.2)$$

$$i \in \{\text{front left (fl), front right (fr), back left (bl), back right (br)}\} \quad (5.3)$$

$$lane_{feat} = \frac{lane_{fr} + lane_{br} - (lane_{fl} + lane_{bl})}{4.0} \quad (5.4)$$

The best models (best number of states and feature vector) were selected using 10-fold cross-validation. The training data set was about 80% of the total amount of data. The testing data set consisted of the rest of the data that had not been used for training.

The number of examples collected in the driving experiments is summarized in table 5.6. The "car data" refers to the car signals from all the in-car annotated maneuvers. The "traffic data" refers to the contextual information that was manually annotated on the videos afterwards. Note that the number of "traffic annotated" examples is much smaller than the number of "car data" examples, because the former requires manual annotation of the videos. The table contains also the average length of each maneuver in number of samples and in seconds.

	Number of driving examples		Average Length #samples (s)	
	Car data	Traffic data	Car data	Traffic data
Passing	710	40	517 (17.2 s)	341 (11.6 s)
turning right	257	37	258 (8.6 s)	159 (5.3 s)
turning left	260	31	258 (8.6 s)	158 (5.3 s)
changing lane right	663	81	159 (5.3 s)	106 (3.5 s)
changing lane left	711	87	165 (5.5 s)	115 (3.8 s)
starting	401	30	174 (5.8 s)	103 (3.4 s)
stopping	404	26	199 (6.6 s)	123 (4.1 s)

Table 5.6: Number of driving examples and average length per maneuver in number of samples

The results on recognizing the previous driving maneuvers are depicted in table 5.7. Some interesting conclusions to be drawn from the experimental results are:

1. There is a plateau of accuracy that can be reached using car information only. Certain maneuvers –such as passing and changing lanes left– cannot be accurately distinguished using car information only.

2. The context is crucial for recognizing maneuvers such as turnings and lane changes.

3. As shown in [182] in a car simulator, the driver's gaze seems to be strongly correlated with the driver's mental state in real life driving. It is, thus, a relevant feature for driver maneuver prediction, specially in lane changes, passings and turnings.

Accuracy (%)			
	Car	Car and Lane	Car and Gaze
passing	100.0	100.0	100.0
turning right	71.4	85.7	85.7
turning left	0.0	33.3	66.7
changing lane right	0.0	12.5	6.3
changing lane left	29.4	17.6	23.5
starting	100.0	66.7	83.3
stopping	100.0	100.0	100.0

Table 5.7: Accuracy for HMMs car only, car and lane and car and gaze data

4. **Predictive Power**: The models are able to recognize the maneuver on average **1 second before** any significant (20% deviation) change in the car or contextual signals takes place. Table 5.8 contains the average prediction power for each of the maneuvers, and figure 5-20 illustrates through an example what this *predictive power* means. It depicts, frame by frame, the lane feature and the -log(likelihood) of the different models for a passing maneuver. There is no significant change in the lane position until frame 26. However, the models are able to recognize the passing from frame 4 on. In consequence, our driver behavior models are able to anticipate that the passing is going to take place about 2/3 seconds before any significant, perceivable change takes place. This is the so called predictive power.

Maneuver	Average Predictive Power in Frames (seconds)
passing	37.7 (1.26 s)
stopping	70.7 (2.4 s)
changing lane left	2.1 (.1 s)
turning left	23.0 (.8 s)
changing lane right	20.3 (.7 s)
turning right	15.1 (.5 s)
starting	41.7 (1.4 s)

Table 5.8: Predictive power of the models in frames and secods

The predictive power of this modeling framework is crucial in an automotive application, where there are tight time constraints. On average, the seven driving maneuvers can accurately be recognized **1** second **before** any significant change in the car signals takes place.

I would claim that these kind of models are essential to build more realistic automated cars in car simulators, to improve the human-machine interface in driver assistance systems, to prevent potential dangerous situations and to create more realistic automated cars in car simulators.

5.6 Summary

This chapter has described the experiments that I have carried out in four different testbeds to evaluate the recognition and prediction power of the human behavior models proposed in this thesis, namely dynamic graphical models (HMMs and CHMMs). The testbeds were intended to capture behaviors of increasing complexity. First, individual facial expressions in LAFTER using HMMs; second, two-hand gestures in TaiChi using CHMMs; third, CHMMs for recognizing pedestrian interactions; and finally HMMs with contextual information for driver maneuver recognition and prediction.

I have paid special attention on modeling interacive behaviors and on estimating how contextual information affects the performance of the behaviors. The recognition accuracy and prediction capability of the models has been reported. In the case of CHMMs, their performance has been compared to that of HMMs. In particular, the superiority of CHMMs versus HMMs for classifying interactive behaviors in two different domains (Tai-Chi gesture recognition and pedestrian interaction recognition) has been reported. In the pedestrian surveillance application (see section 5.4), CHMMs surpass HMMs in identifying the case in which there were no interactions present in the testing data. In a visual surveillance system the *false alarm* rate is often as important as the classification accuracy. In an ideal automatic surveillance system, all the targeted behaviors should be detected with a close-to-zero false alarm rate, so that one could reasonably alert a human operator to examine them further. The reported ROC curves for both HMMs and CHMMs (see figure 5-16) illustrate that it is quite possible to achieve with CHMMs very low false alarm rates while still maintaining good classification accuracy.

Section 5.4.3 describes and experimentally validates the proposed framework for designing prior models via synthetic data generated by agents. One of the main motivations for constructing such synthetic agents is the ability to generate synthetic data which allows to determine which Markov model architecture will be best for recognizing a new behavior

(since it is difficult to collect real examples of rare behaviors). By designing the synthetic agents models such that they have the best generalization and invariance properties possible, one can obtain flexible prior models that are transferable to real human behaviors with little or no need of additional training. The use of synthetic agents to generate robust behavior models from very few real behavior examples is of special importance in a visual surveillance task, where typically the behaviors of greatest interest are also the most rare.

Finally, in the driving domain (see section 5.5.5) HMMs have been used for the recognition and prediction of maneuvers at a tactical level. Context –via the driver's gaze and the relative position of the road lanes– has been shown to be critical for the accurate recognition of certain maneuvers, such as lane changes. Another powerful feature of the proposed models is their predictive power: on average, each of the seven driving maneuvers can accurately be recognized **1** second **before** any significant change in the car signals takes place. I believe that these models would be essential to build more realistic automated cars in car simulators, to improve the human-machine interface in driver assistance systems, to prevent potential dangerous situations and to create more realistic automated cars in car simulators.

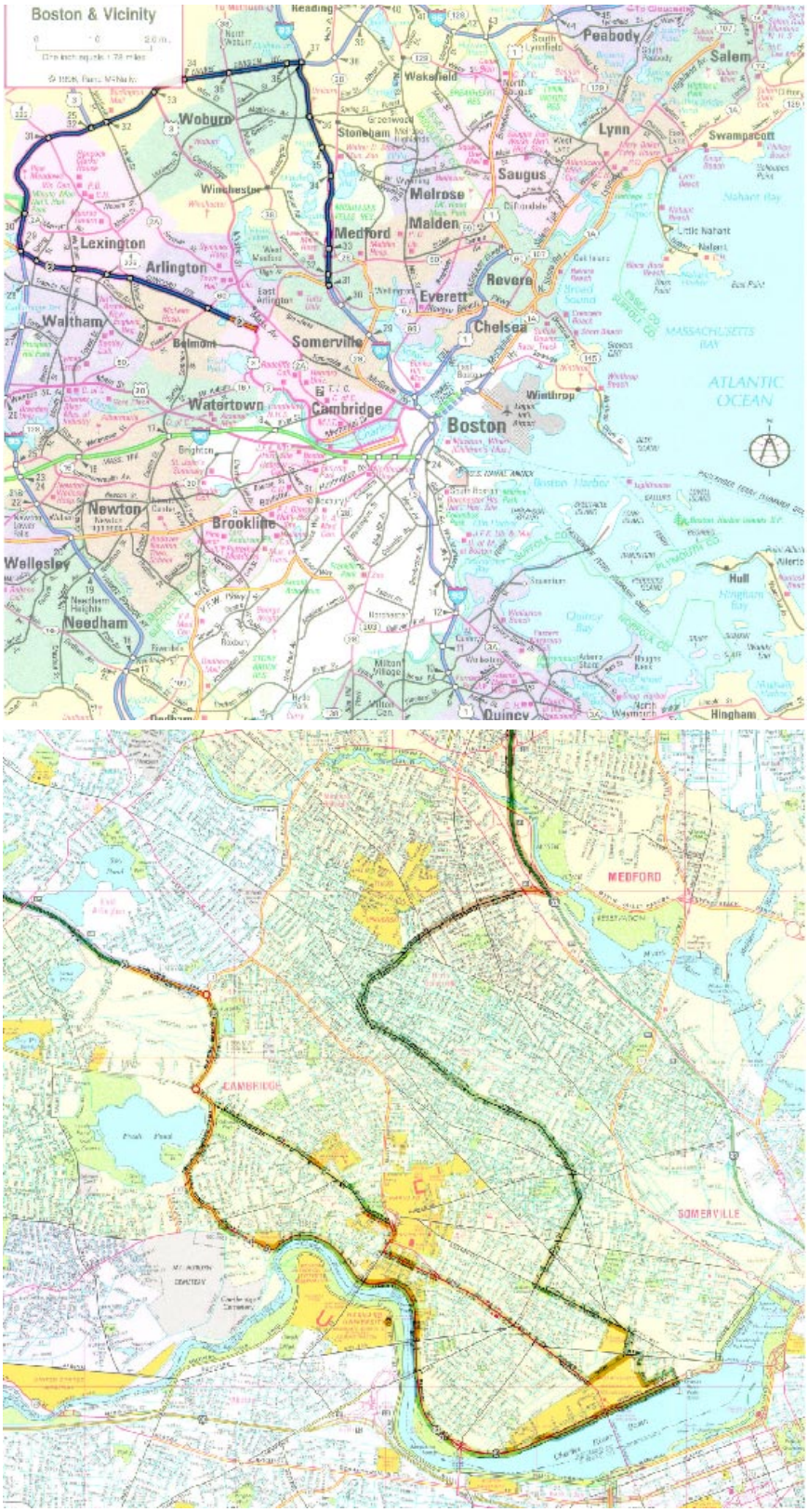
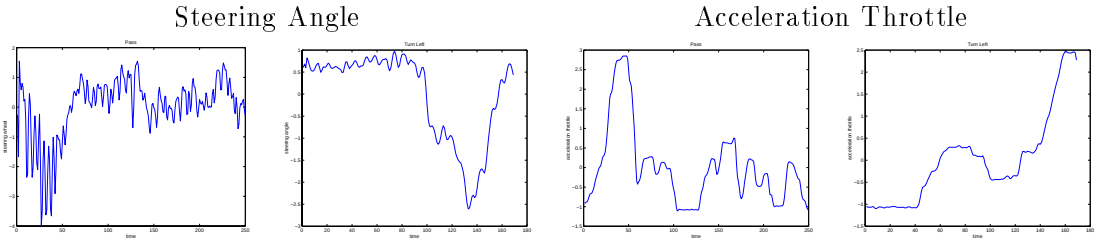
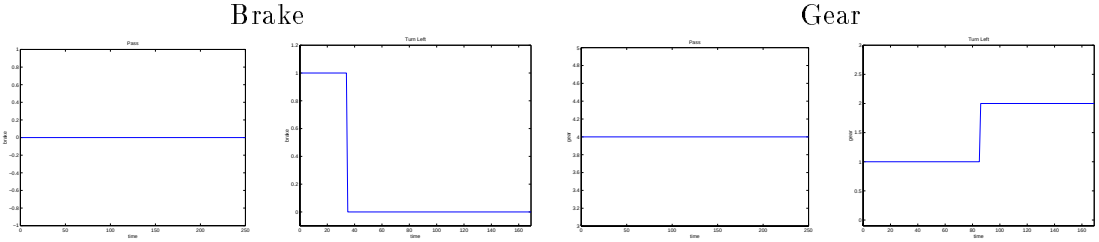


Figure 5-17: Route followed in the driving experiments: overview and city sections detail

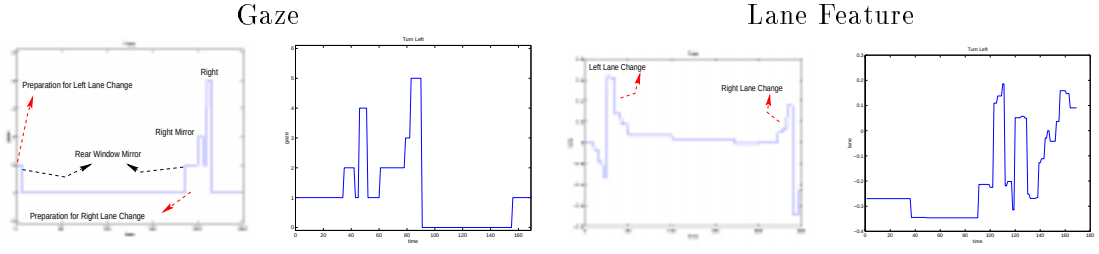


(a) Passing (b) Turning Left (a) Passing (b) Turning Left



(a) Passing (b) Turning Left (a) Passing (b) Turning Left

Figure 5-18: Typical car signals for passing and turning left maneuvers



(a) Passing (b) Turning Left (a) Passing (b) Turning Left

Figure 5-19: Typical contextual (gaze and lane) signals for a passing and turning left maneuvers

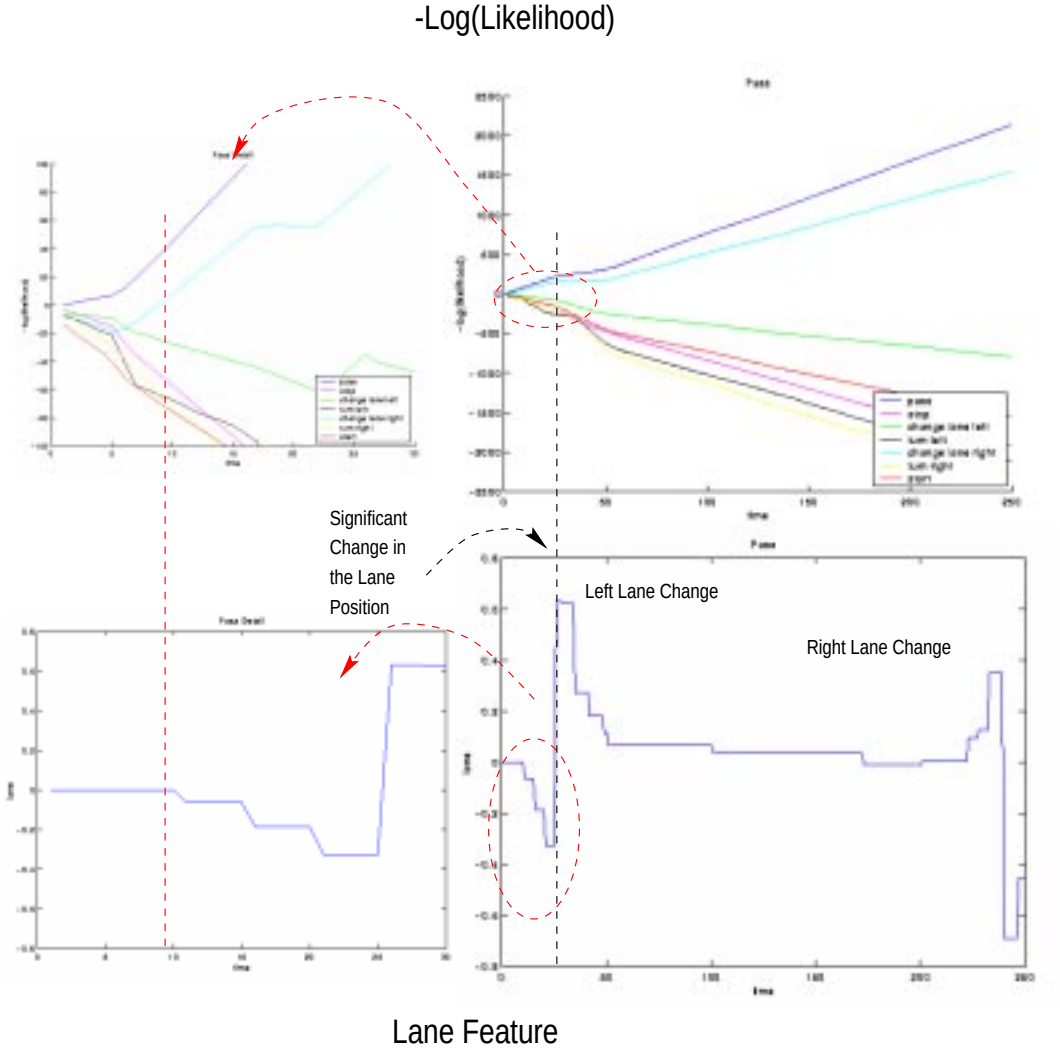


Figure 5-20: Prediction of a passing maneuver about 2/3 seconds before any significant lane change takes place.

Chapter 6

Contributions and Future Work

6.1 Contributions

In this thesis I have proposed a computational framework for the automatic recognition and prediction of different kinds of human behaviors from video cameras and other sensors, via *perceptually intelligent systems* that automatically sense and correctly classify real human behaviors, by means of *Machine Perception* and *Machine Learning* techniques. The proposed framework could be psychologically plausible at a general level, addresses many of the criticisms that current behavior theories suffer from and has been tested with experimental data of increasing behavioral complexity collected in four different domains:

1. Individual, isolated behaviors in the LAFTER [171] (Lips and Face TrackER) system: a real-time system for face detection, tracking and facial expression recognition (see figure 1-1)

2. Body gestures in a Tai-Chi real-time gesture recognition system

3. Human to human interactive behaviors in a visual surveillance system for detection and recognition of human-to-human interactions [173] (see figure 1-1)

4. Human behaviors when mediated by a machine (car) in the *SmartCar* testbed. More specifically models for recognizing driver's behaviors at a tactical level, with emphasis on how the context (road lanes, surrounding traffic) affects the driver's performance (see figure 1-2) and on the predictive power of the models.

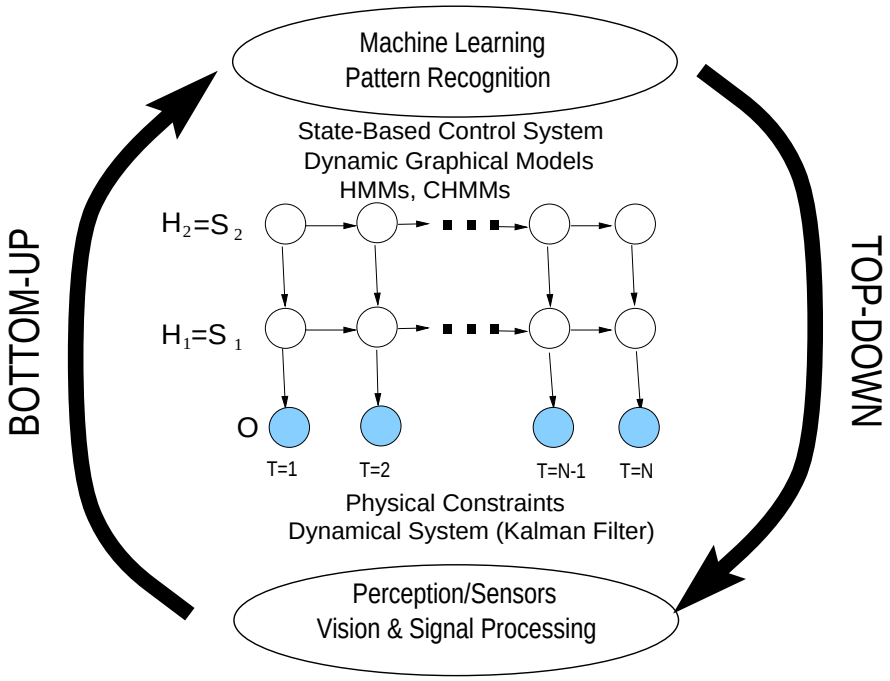


Figure 6-1: Proposed computational model for human behavior recognition and prediction

The proposed model's architecture (depicted in figure 6-1) is composed of a hierarchy of two layers. At the bottom (first layer) there is the Perceptual System, composed of cameras and other sensors. The signals captured by the sensors are typically the input to a Kalman Filter. Depending on the domain, different perceptual input modalities have been used: (1) In the case of facial expression recognition, an active camera looking at the user's face; (2) a stereo real-time head and hands tracking system is used in the Tai-Chi gesture recognition system; (3) in the framework of pedestrian interactions recognition, a static camera with wide field-of-view watching a dynamic outdoor scene; (4) finally, in the driver domain, multiple sensors of different nature are used: internal sensors of the car's internal state –acceleration, steering wheel angle, gear, speed and break pedal action–, and cameras for the visual context –front and rear traffic, driver's face and gaze, and driver's viewpoint. Some key elements of the computer vision algorithms developed in this thesis are: statistical (ML and MAP) appearance based segmentation of the objects of interest (face, mouth, hands and full body) using blobs, characterized by a mixture of Gaussians, off-line and on-line EM algorithms for adaptation to different users or changes in the environment, active camera control via a PD-controller, and pedestrian detection and tracking by means

of a novel eigenbackground subtraction technique.

At the top (second layer) there is the behavior models via Dynamic Graphical Models: HMMs and CHMMs. To recognize human interactive behaviors two Hidden Markov Models (HMMs) are coupled in a new architecture called CHMMs to capture the interactions between them. The algorithms for learning the parameters from data as well as for doing inference with those models have been developed and described. The Kalman filter estimations are the observations of the Dynamic Graphical Models (HMMs or CHMMs) at the second layer.

As it is depicted in figure 6-1, the proposed architecture includes a *bottom-up* stream of information provided by the various sensors, and a *top-down* information flow through the predictions provided by the behavior models. Consequently a Bayesian approach –such as the one followed– offers a mathematical framework for both combining the observations (bottom-up) with complex behavioral priors (top-down) to provide expectations that would be fed back to the perceptual system.

The four testbeds that I have built in this thesis capture human behaviors of different nature and increasing complexity: first, isolated, single-user facial expressions (LAFTER); second, 2-hand gestures (Tai-Chi); third, pedestrian interactions in a surveillance application (pedestrian surveillance), and finally potentially multi-agent interacting behaviors where human performance is mediated by a machine, more specifically, a car (SmartCar). In the SmartCar testbed, contextual information (road lanes position, driver's gaze and eventually surrounding traffic) has been shown to be critical towards accurate recognition of driver maneuvers at a tactical level. Moreover, the models are able to predict the maneuvers on average **1 second before** they take place. This predictive power is extremely important in a driving situation, where timeliness is critical.

The metric that I have used for quantifying the quality of the behavior models has been their accuracy: how well they are able to recognize the behaviors on testing data. Statistical machine learning usually suffers from lack of data for estimating all the parameters in the models. To alleviate this problem, a new framework for generating prior models has been proposed. In essence, synthetically generated data are used to bootstrap the models creating 'prior models' that are further trained using much less real data than otherwise it would be required. The Bayesian nature of the approach let us do so.

The predictive power of these models lets us categorize human actions very soon after the beginning of the action. Because of the generic nature of the typical behaviors of each of the implemented systems there is a reason to believe that this approach to modeling human behavior would generalize to other dynamic human-machine systems. This would allow us to recognize automatically people's intended action, and thus build control systems that dynamically adapt to suit the human's purposes better.

The main contributions of this thesis are consequence of the modeling approach proposed in my work on Perceptual Intelligence. Namely, the combination of Perceptual Computing with Statistical Machine Learning (dynamic graphical models or DynPINs) for recognizing human behaviors of increasing complexity in different domains. More specifically the proposed framework emphasizes the interactions between the agents and the importance of contextual information as an important element of behavior modeling. The domains explored in this thesis proceed along the "intentionality" axis (see the taxonomy described in chapter 1), with increasing complexity in the nature of their typical behaviors. Beyond the computational framework, some of the more specific key contributions are:

1. Real-time face expression recognition system using HMMs.
2. CHMMs for recognition of human-to-human interacting behaviors.
3. Flexible and interpretable prior behavior models by means of synthetic agents.
4. Dynamic Graphical Models architecture for the recognition and prediction of real driver behaviors at a tactical level.
5. SmartCar data acquisition and playback platform.

6.2 Future Work

Short Term In a short term future I am interested in understanding better the so valuable driver behavior database that I have created. I would like to build more sophisticated models of driver behavior, more specifically by modeling the interactions between the driver and the surrounding traffic.

In theory, one could think of a model where each car is represented by a separate HMM. Moreover, the HMMs do not evolve over time independently, but they are affected

by adjacent cars in a pairwise fashion. Figure 6-2 illustrates the proposed architecture: a lattice of CHMMs or LCHMM. Each HMM contains as observed nodes the corresponding sensory inputs –such as velocity, acceleration or relative position– and as hidden nodes the driver's mental states (intentions) –such as changing lane or slowing down–. Each HMM computes the probability distributions of its outputs based on its latest observations, its previous state estimate and the state estimate of the adjacent cars. It encodes the dynamics of the driving behavior at a tactical level. Each hidden state of the HMM can be interpreted as a sub-action whose temporal concatenation yields the entire tactical level action.

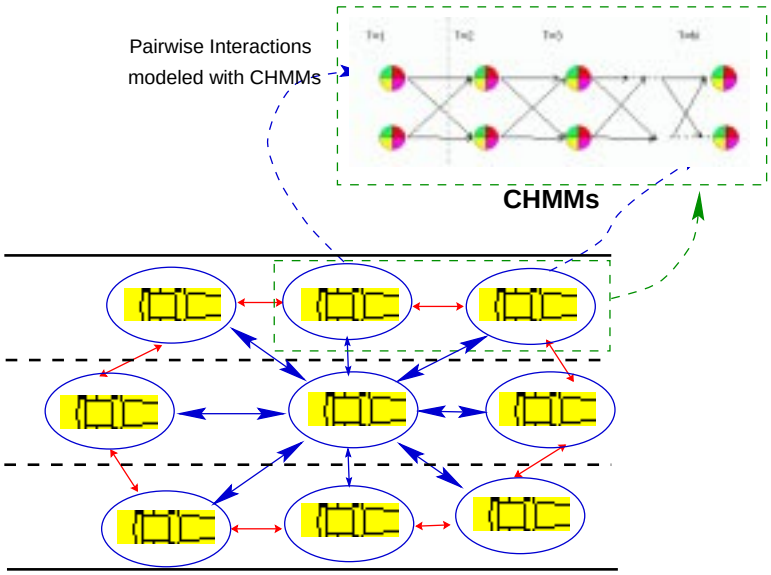


Figure 6-2: Representation of the Hidden Markov Models lattice for modeling car interactions

However, this original graphical architecture depicted in figure 6-2 needs to be modified to reflect the real driving behavior exhibited and collected in the driving experiments –described in section 5.5.5–: instead of a symmetric CHMM, it seems more reasonable an asymmetric CHMM (aCHMM) architecture, where the surrounding traffic affects the behavior of the driver, but not vice-versa. This is just an approximation to the more realistic situation of mutual interactions. The main justification of such an approximation comes from the fact that in our SmartCar experiments, the driver did indeed modify his/her behavior depending on the surrounding traffic, but not vice-versa. I will call this architecture as Lattice of Asymmetric CHMMs or LaCHMM.

At a perceptual level, I would also like to incorporate automatic algorithms for the detection and tracking of the surrounding cars and road lanes.

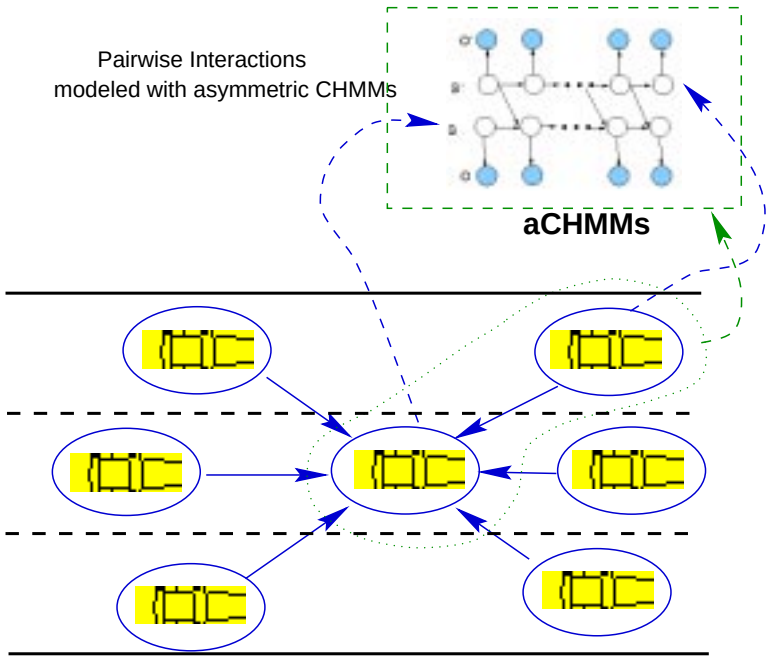


Figure 6-3: Representation of the asymmetric CHMMs lattice (LaCHMM) for modeling car interactions

Behavior Fusion In the architecture presented in figure 6-3, the main vehicle has attached a number of CHMMs that capture the pairwise interactions with the adjacent cars. At each instant of time, therefore, each of these CHMMs predicts an output. In consequence, multiple outputs are associated with the vehicle. However, the final output should be just one. I propose the use of Bayesian integration of the different outputs into a single one.

Long Term In a more distant future, I would like to build models of longer term behaviors, with higher order –not just first order– causal relations. I would also be interested in exploring hierarchical architectures, performing automatic identification of novel behaviors, learning the model structure from data, and exploring other domains where this framework might be appropriate.

Finally, a very important issue that I have barely addressed in this thesis is the human interaction aspects of Perceptually Intelligent Systems. Once we can build systems that sense and recognize human behaviors, what are they going to do with such information? How should the information be conveyed to the user? There is no single and simple answer to these questions. They certainly open new avenues of research topics that I will definitely would like to explore in the future.

Appendix 1: Driving Experiments Consent Form and Questionnaires

CONSENT FORM

You are being invited to participate in a research study. This form is designed to provide you with information about this study. The Principal Investigator, any of the Associated Investigators or representative will describe this study to you and answer any of your questions. Your participation in the following experiment is completely voluntary. You are free to withdraw this consent at any time, for any reason, and to request that any data collected be destroyed. If at any time you feel uncomfortable, or unsure that you wish your results to be part of the experiment, you may discontinue your participation with no repercussions.

In this experiment you will be asked to perform a driving task, fill out questionnaires, and answer some questions about the driving task. An experimenter will be with you throughout the experiment. During the driving task, the experimenter will be sitting on the copilot seat. The questionnaire will be shown to you before you are asked to sign this consent form. Please feel to talk to the experimenter if you have questions or feel uncomfortable at any time.

The purpose of this study is to build models of driver behavior at a tactical level. The focus is on the following maneuvers: passing, turning right/left, changing lanes right/left, stopping/starting after a stop, and following another car. We will record the following signals while driving: speed, brake, gear, acceleration throttle, steering wheel angle, front and rear road and traffic, and the driver's face. We will build statistical machine learning models of the previously mentioned maneuvers from the gathered data. The ultimate goal is to create an automatic system that will recognize the driver's maneuvers and predict which

will be the most likely action to be taken next. One important implication of such a system is safer driving by warning the user in potentially dangerous situations. For example, the car may predict that the driver is likely to change lanes and may warn the driver of the presence of a car in his/her blind-spot before trying to perform such a maneuver.

The driving task will take place in the greater Boston area. You will drive both in the city and in different highway sections. The car is an automatic 1998 Volvo V70 XC which has been instrumented with several sensors and video cameras. None of them affects the driving task in any sense. There are four cameras in the car: two Sony EVI-D30 cameras with wide field-of-view to record the traffic in front and behind the car; an ELMO CCD camera recording the driver's face and another ELMO CCD camera mounted on a pair of glasses to record the driver's viewpoint. There will be no audio processing involved in the experiment.

Because of the serious nature of the driving task, all results of the experiment should be considered secondary to safe driving practice. The risks associated with your participation in this experiment are the normal risks associated with a driving task in both urban and highway situations. If at any time you feel that any of the hardware components are distracting you from driving, please let the experimenter know immediately and the monitoring will be discontinued. In an emergency situation, your sole consideration should be safety. Do not be at all concerned if you move out of view of the camera or you realize that any of the hardware components gets detached.

Any responses that are collected during the experiment will be kept completely confidential. However, because of the video recordings and your name being requested in the questionnaire, anonymity can not be totally assured. In consequence, we will remove any part of the data that you may not wish us to use. You will not be asked any specific confidential question. The records of the driving experience will be archived on tapes labeled only with the subject ID. From this point forward, you will be referred to only as the ID number which appears on the upper right corner of this packet and never by your name.

If you have any questions, at any point during the experiment, the experimenter will gladly answer them.

In the unlikely event of physical injury resulting from participation in this research, I understand that medical treatment will be available from the MIT Medical Department, including first aid emergency treatment and follow-up care as needed, and that my insurance

carrier may be billed for the cost of such treatment. However, no compensation can be provided for medical care apart from the foregoing. I further understand that making such medical treatment available; or providing it, does not imply that such injury is the Investigator's fault. I also understand that by my participation in this study I am not waiving any of my legal rights.

I understand that I may also contact the Chairman of the Committee on the Use of Humans of Experimental Subjects, MIT 253-6787, if I feel I have been treated unfairly as a subject.

I hereby give consent for the data collected from the experiments in which I have participated to be used in research papers, presentations and demonstrations. Furthermore, if I wish to keep any part of the experiment from being made public, the experimenters will fulfill such requests. I understand that after the study is complete, the video data will be archived on CD-ROM so that our results may be recorded and verified. The data will only be used for the purposes of scientific research by researchers at the MIT Media Lab or their collaborators.

I certify that I am a licensed driver and that I will obey the driving laws of the state of Massachusetts during this task. The car will be insured by MIT's policy with Liberty Mutual $\#AS2 - 111 - 060227$.

I understand that I will be paid \$10 per hour for my participation in the study, pro-rated for early withdrawal. I understand that the driving task will take approximately two hours to complete.

My participation in this driving experiment will be completed after the first session. However and in a totally voluntary manner, I will have the opportunity of coming back to other identical driving sessions at a later date.

Name: _____

Date: _____

Location: _____

Driving Experiment Subject Instructions

Sorry, you cannot participate if you do not have a valid and current driver's license *

Here is an outline of the course of the experiment:

The experimenter will take you to the East Campus Parking Garage behind building 68. You will get into the driver's seat of the 1998 Volvo V70 XC. The experimenter will be turning on the cameras and data acquisition hardware and computer system. One all the equipment has been set up and it's properly working you will start the driving task.

Your driving experience should include most of the following events:

1. A period of stationary monitoring
2. Exiting the garage
3. A period of city driving in Cambridge toward Route 93 North
4. Access to Route 93 North on exit 31
5. A period of highway driving out to North of Cambridge
6. Access to Route 95 West on exit 37 of Route 93
7. A period of highway driving on Route 95 South-West
8. Access to the Concord Turnpike on exit 20 of Route 95
9. A period of highway driving back to Cambridge
10. Access to the Cambridge exit
11. A period of city driving back to MIT (Memorial Drive/Massachusetts Avenue)
12. Parking in the East Garage

When we get back, you will be asked to fill out a questionnaire asking about your driving experience today and your driving habits and history. The whole process should take about 2 hours. Drive safely and remember to buckle up!

Sample Driving Questionnaire

Subject Name: _____

Subject Number: _____

Session Number: _____

Date: _____

Experimenter Name: _____

This questionnaire is designed to help us label the data and have more background information about you. You will be asked to rate how your driving was today with respect to a what you would consider a 'normal, neutral' driving day.

1. Background Questions

   Age:_____Sex:_____

   Height:_____Weight: lbs

   Profession:_____

   # hours worked/week: _____

   How long have you had your driver's license? months

   How often do you usually drive?

   (a) Every day

   (b) Few times a week

   (c) Few times a month

   (d) Few times a year

   (e) Never drive

   Do you own a car or have a car that use frequently?

   YES NO OTHER (Explain)

   If so, what kind of car is it?

   Do you feel comfortable in general driving a different car to the car that you normally

drive?

Are there any recent events in your life that you feel may have affected your driving experience today?

YES NO

If so, in which sense is your driving experience affected?

Rate yourself as a driver:

(a) Very experienced, good driver

(b) Moderate experience, fair driver

(c) Average driver

(d) Non-experience, under-average driver

(e) Bad driver

2. Today's Driving Experience

How would you rate today's driving experience compared to other days:

(a) More stressful than average

(b) Normal driving day

(c) Less stressful than average

Rate the following driving periods in terms of how 'neutral, average' your driving was during them according to the following 5 point scale listed below:

(a) Very comfortable

(b) Comfortable

(c) Normal driving day

(d) Uneasy

(e) Not comfortable at all

A. Stationary periods

B. City driving periods

C. Highway driving periods

D. Tolls (if applicable)

E. Merges and exits

Rate the following driving maneuvers in terms of how comfortable you felt performing them according to the following 5 point scale:

(a) Very comfortable

(b) Comfortable

(c) Average, normal

(d) Uneasy

(e) Not comfortable at all

A. Passing another car

B. Turning right

C. Changing lane right

D. Turning left

E. Changing lane left

F. Stopping

G. Following another car

H. Starting after a stop

On a scale of 1 to 7 with 1 representing the closest and 7 the furthest away to a normal driving day, please rate the 12 driving events.

(a) Period of stationary monitoring before driving

(b) Exiting the garage

(c) Period of city driving

(d) Access to Route 93 North

(e) Highway driving on Route 93 North

(f) Access to Route 95 South-West

(g) Highway driving on Route 95 West

(h) Access to Concord Turnpike back to Cambridge

(i) Highway driving on the Concord Turnpike

(j) Access to Cambridge exit

(k) City driving back to MIT

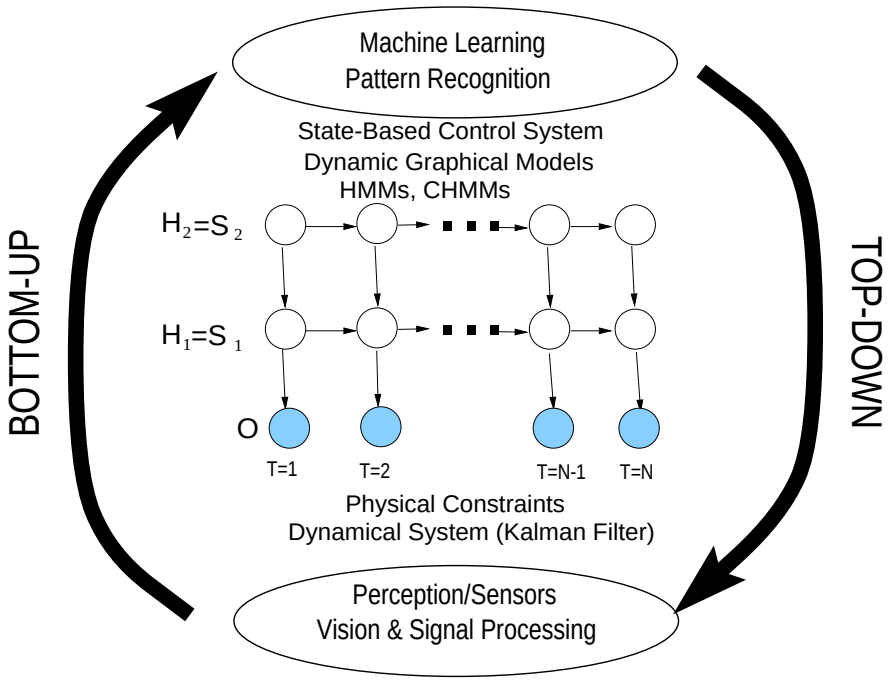
(l) Parking in the garage

Please, feel free to add any additional comments such as any highlights on your driving experience today that may be relevant and useful for the purpose of this study:

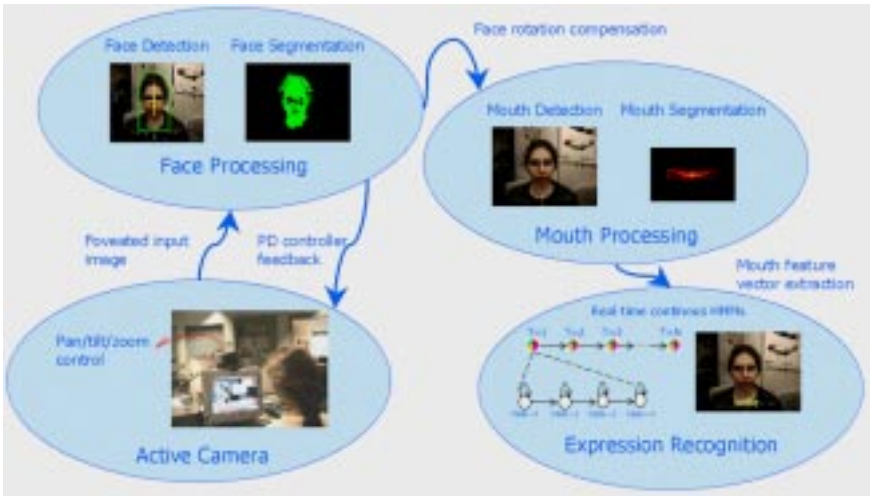
Thank you very much for participating in this experiment!! We hope that you enjoyed it!!

Appendix 2: Color Figures

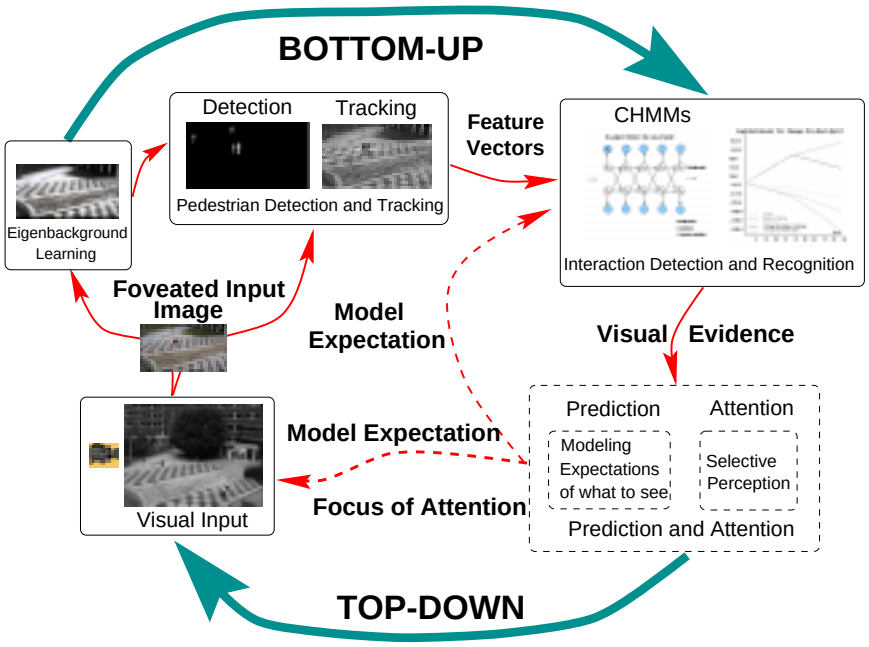
Chapter 1



Proposed computational model for human behavior recognition and prediction

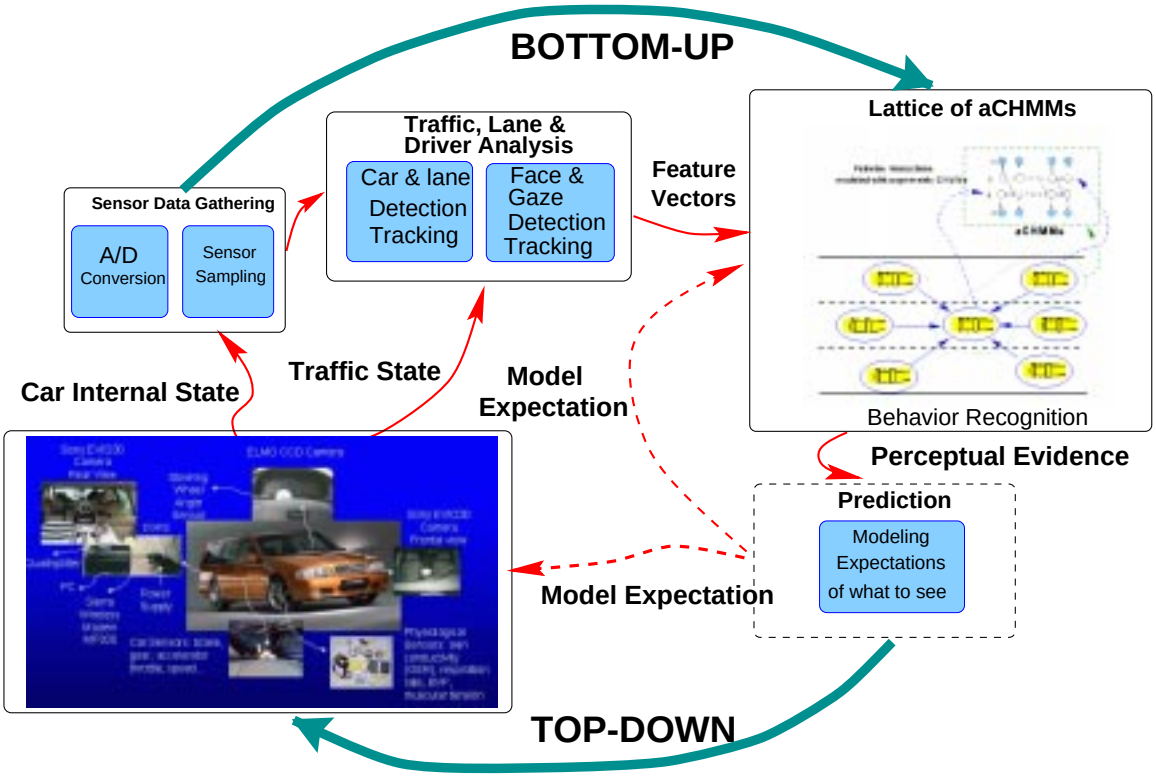


Architecture of LAFTER



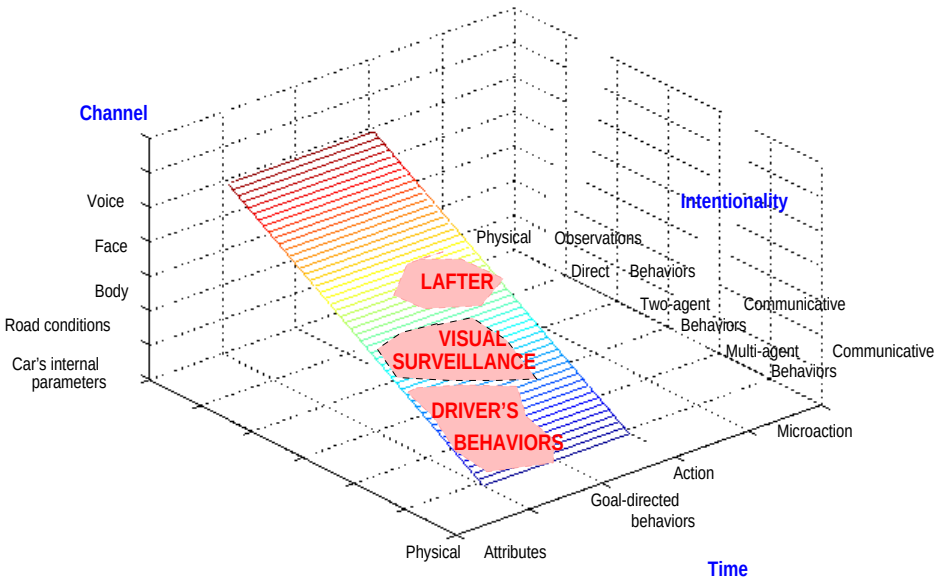
Architecture of the Visual Surveillance system

System architecture for LAFTER and the visual surveillance systems

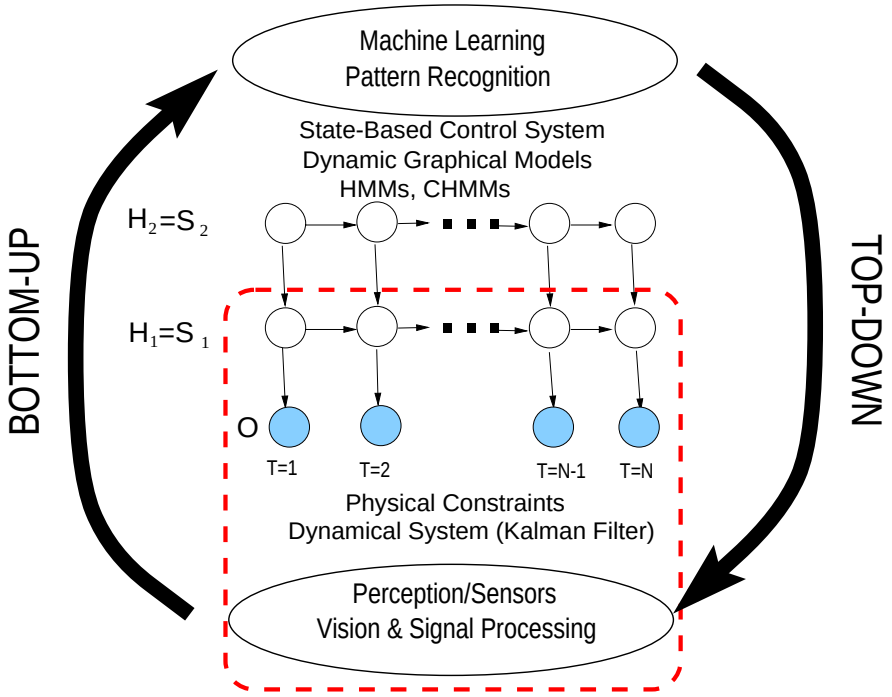


Architecture of the SmartCar testbed platform

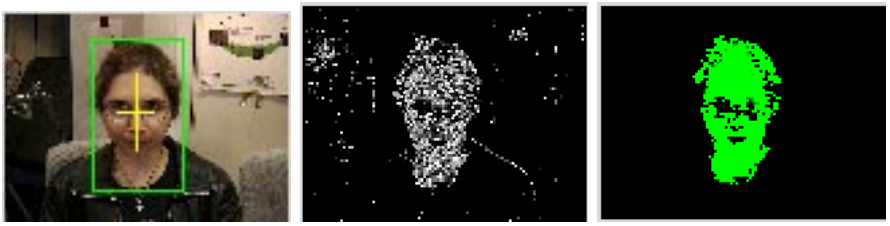
SmartCar System architecture


Channel
Voice
Face
Body
Road conditions
Car's internal parameters
Physical
Attributes
Intentionality
Physical
Observations
Direct
Behaviors
Two-agent
Behaviors
Multi-agent
Behaviors
Communicative
Communicative
LAFTER
VISUAL SURVEILLANCE
DRIVER'S BEHAVIORS
Microaction
Action
Goal-directed behaviors
Time

Taxonomy

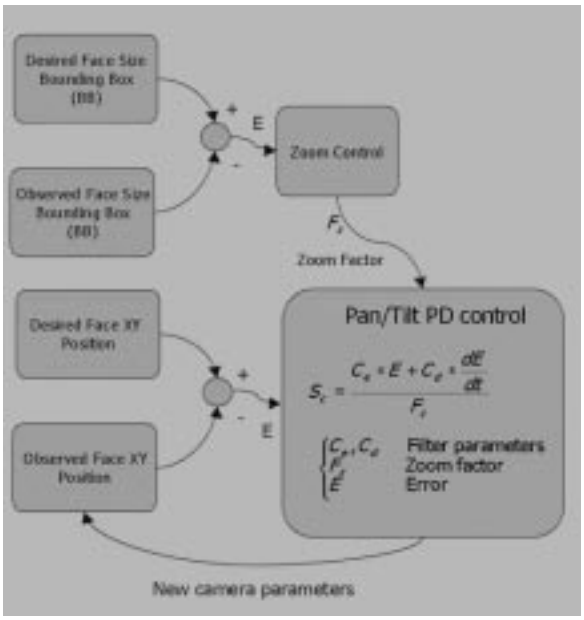


The perceptual system occupies the lowest level in the proposed model

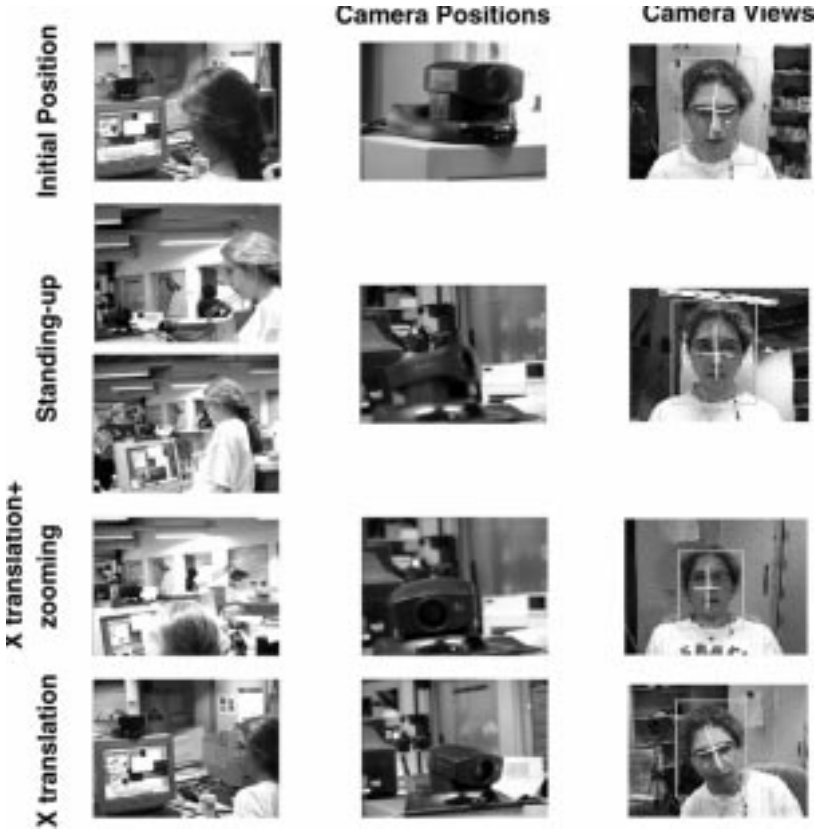


Face detection, per-pixel probability image computation and face blob growing

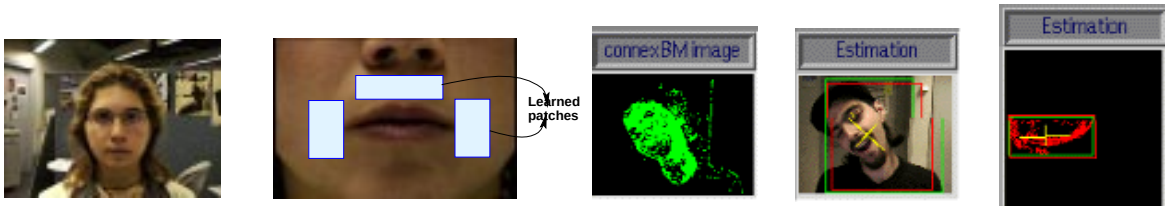
Background mean image, blob segmentation image and input image with blob bounding boxes



PD Controller

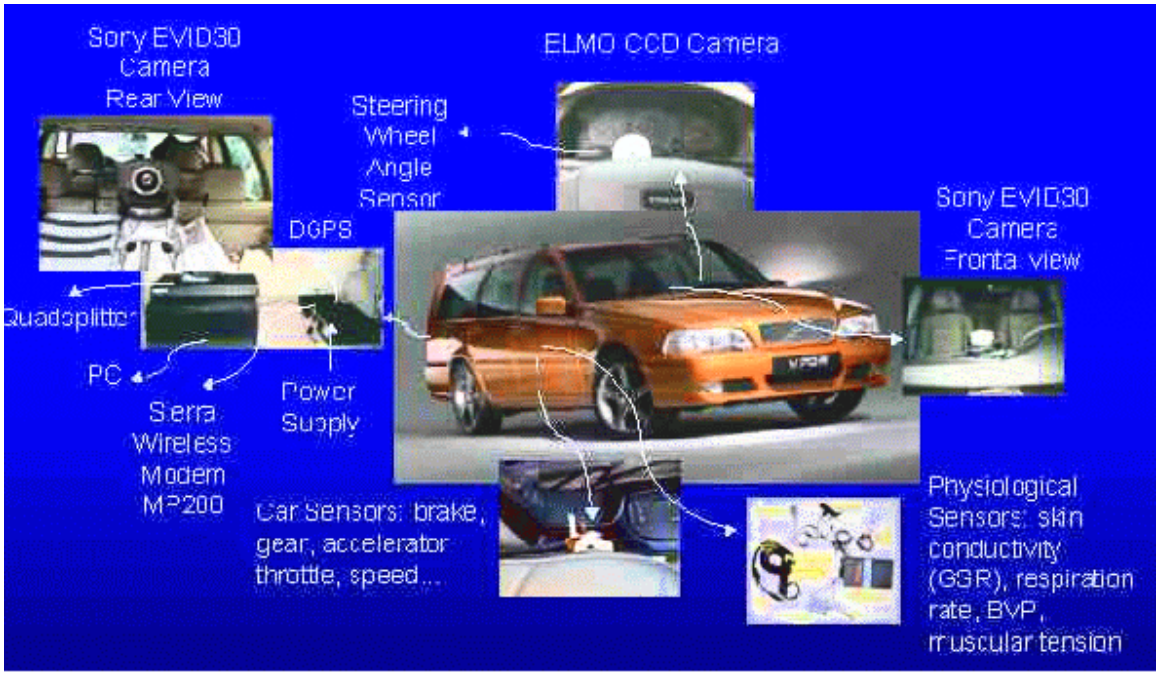


Active camera tracking

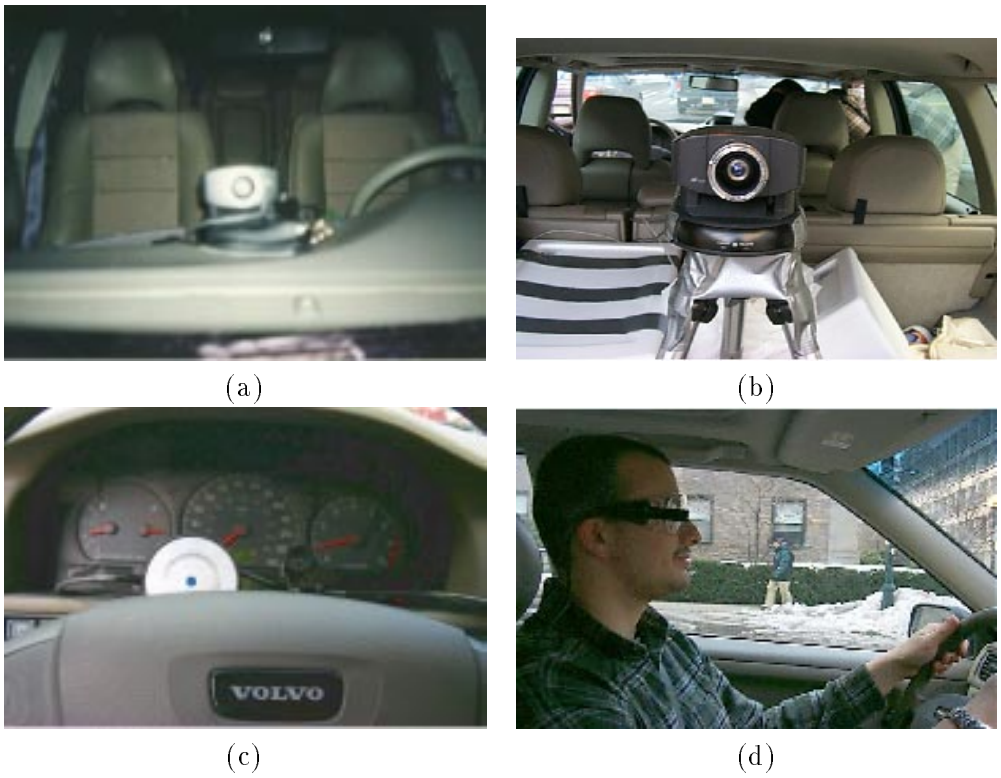


Multi-resolution and face skin patches learning

Multi-resolution mouth extraction, skin model learning. Head and mouth tracking with rotations and facial hair



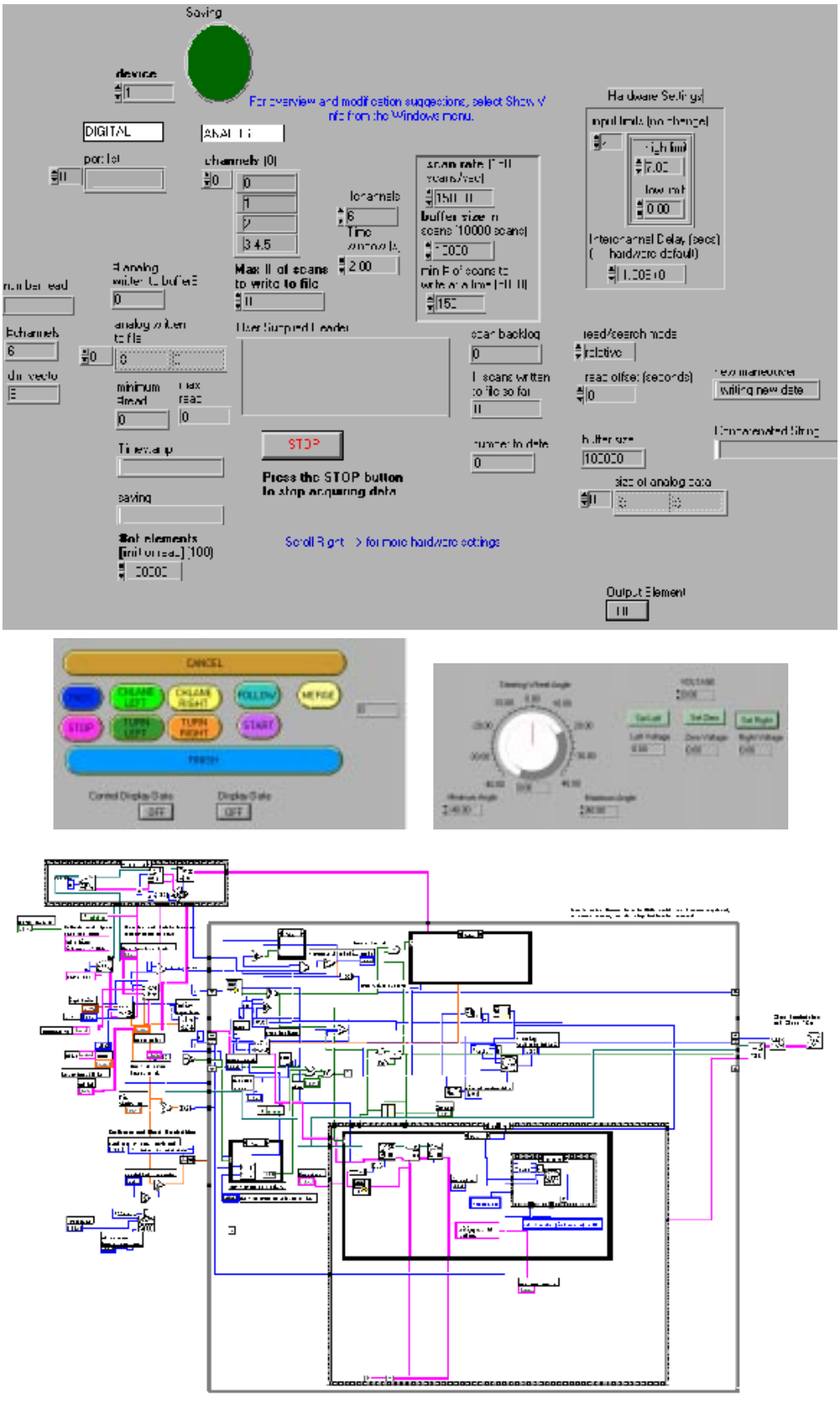
SmartCar (Volvo V70XC)



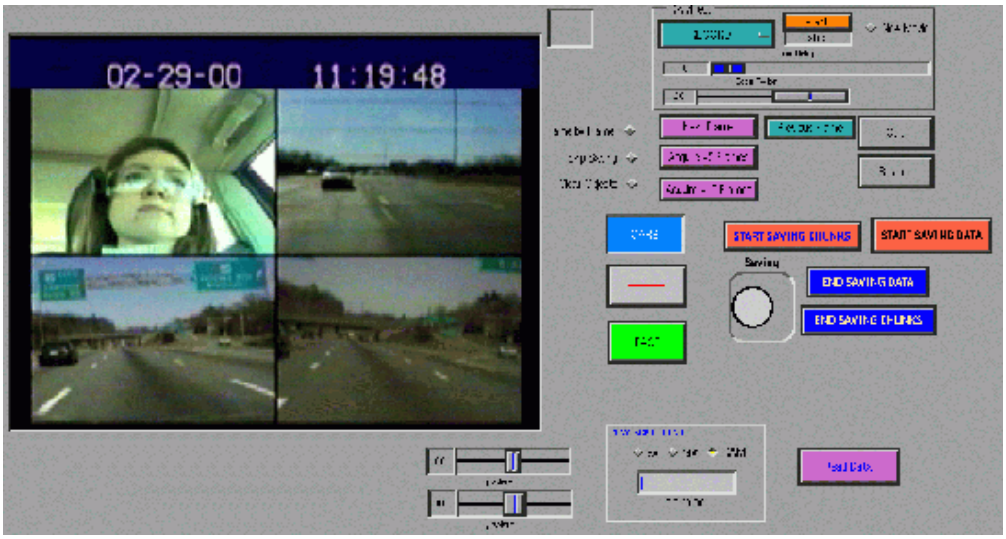
(a) (b)

(c) (d)

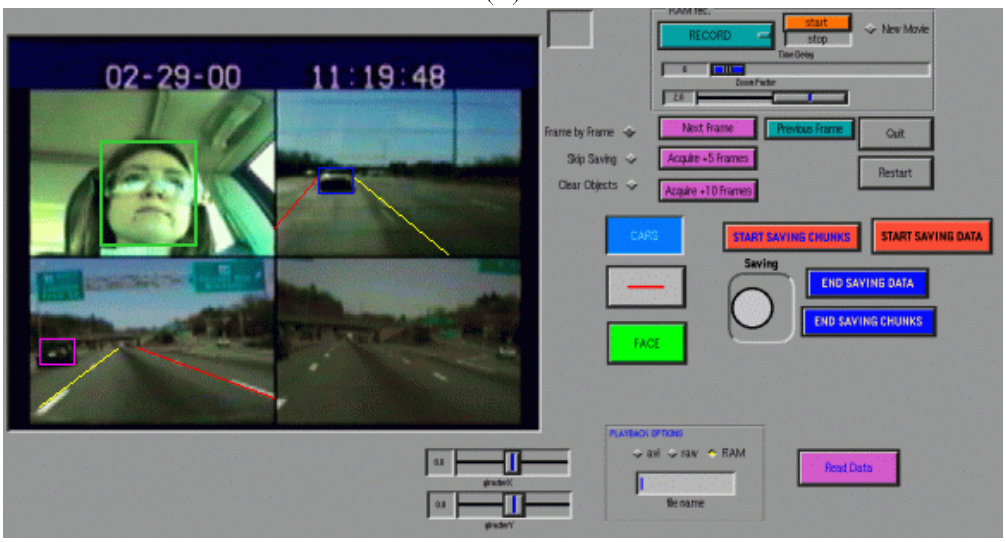
SmartCar sensors: (a) Front and rear wide-field-of-view cameras (b) Steering wheel sensor and driver's face camera (c) Driver's viewpoint camera



Example of LabVIEW graphical user interface and diagram.

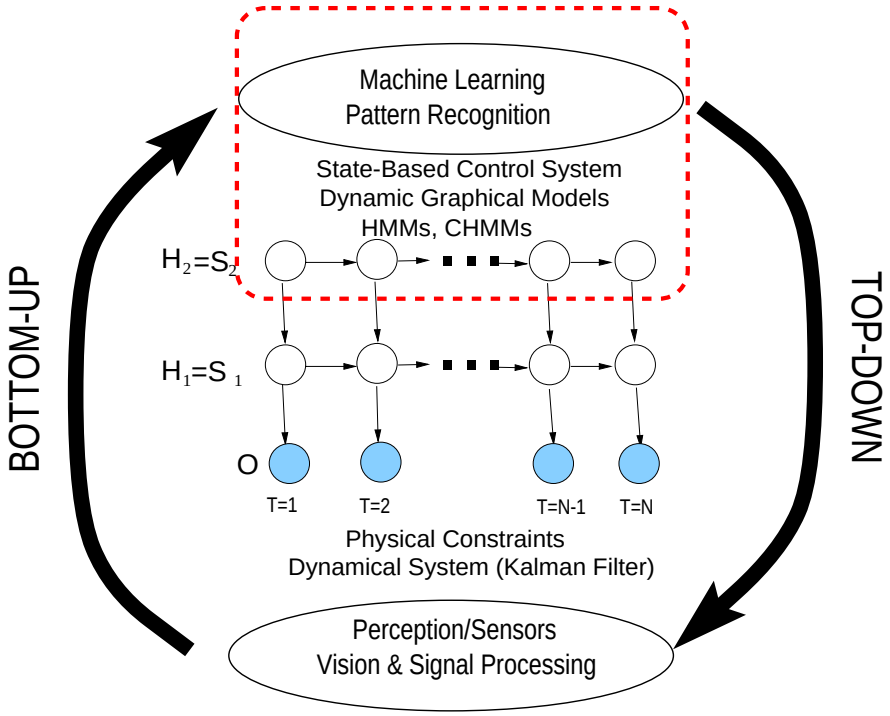


(a)

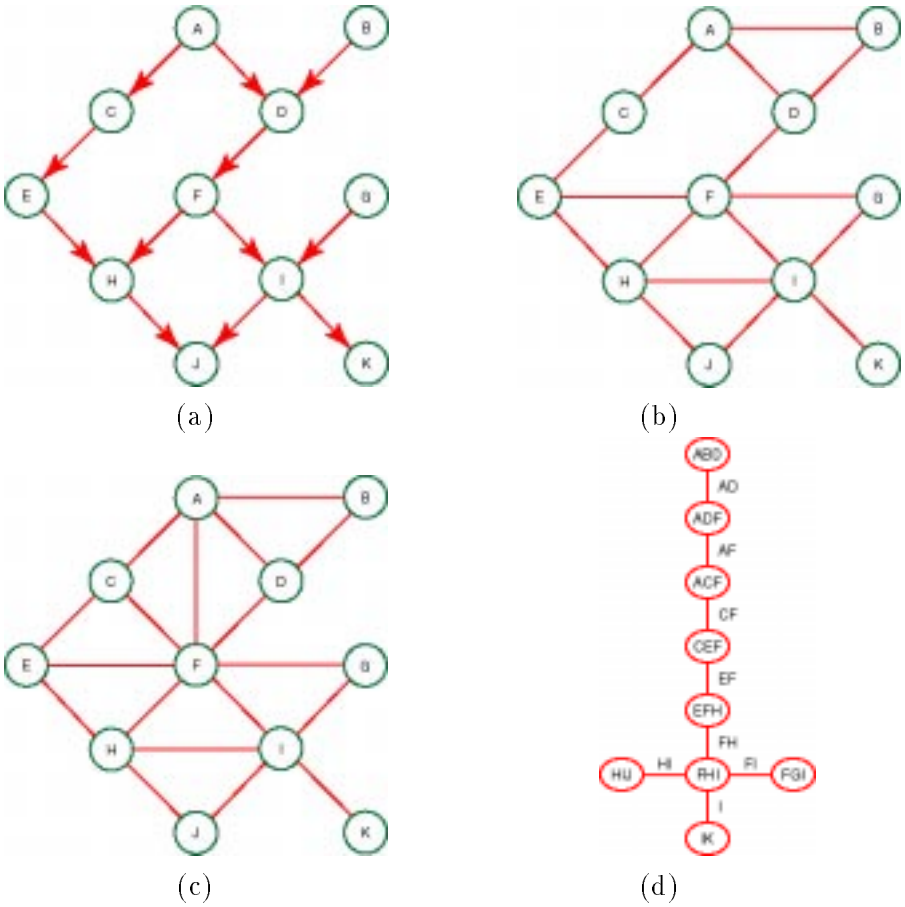


(b)

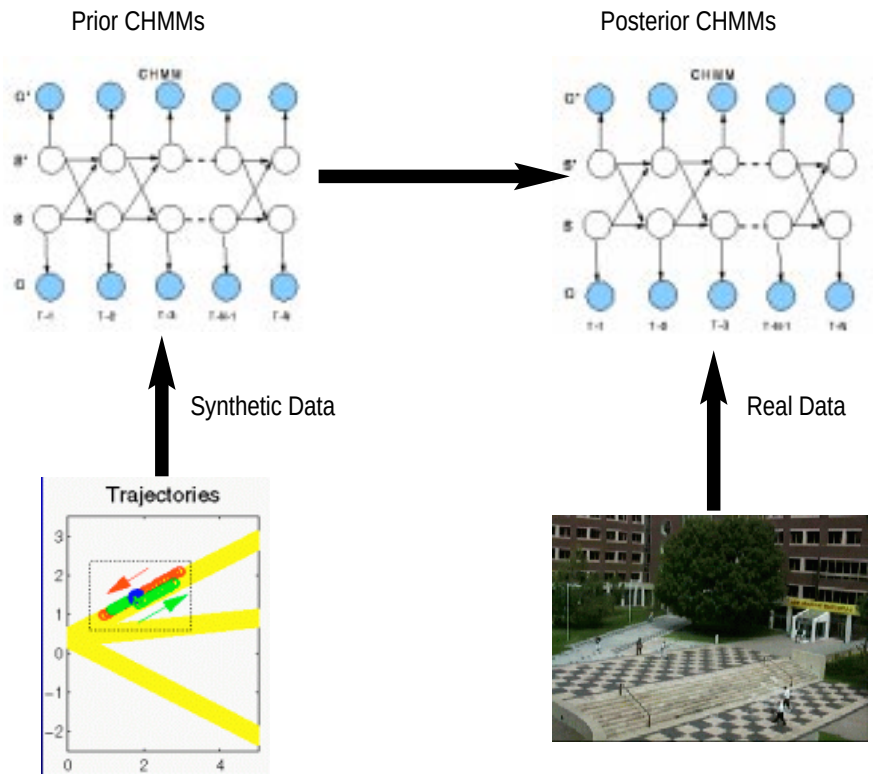
Graphical User Interface for video signals annotation: (a) Input image (b) Annotated image.



The perceptual system occupies the lowest level in the proposed model

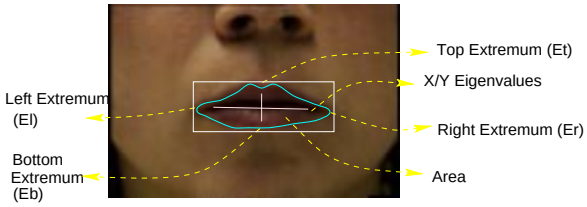


(a) A DPIN structure G^D. (b) The moral graph G^M for G^D, where the parents of every node have been linked. (c) The triangulated graph G^T where the nodes have been linked to satisfy the running intersection property. (d) The corresponding junction tree (JT).



Training procedure when using synthetically generated priors

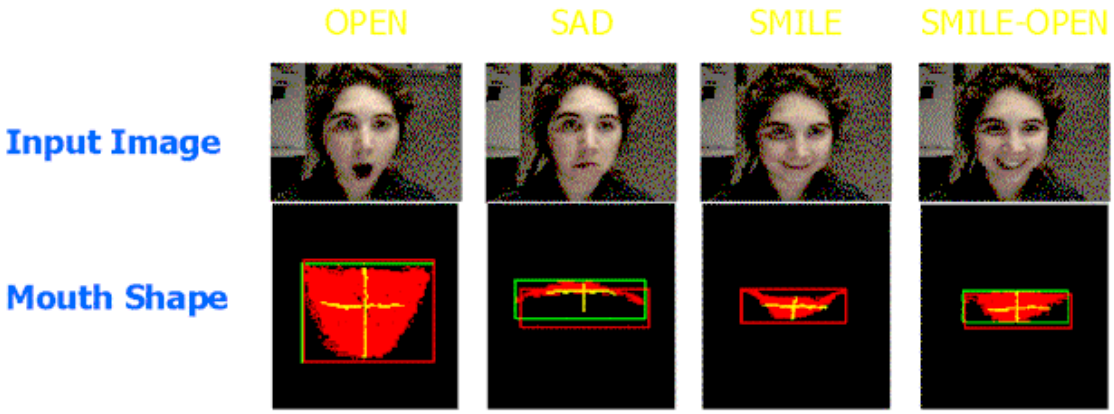
Chapter 5



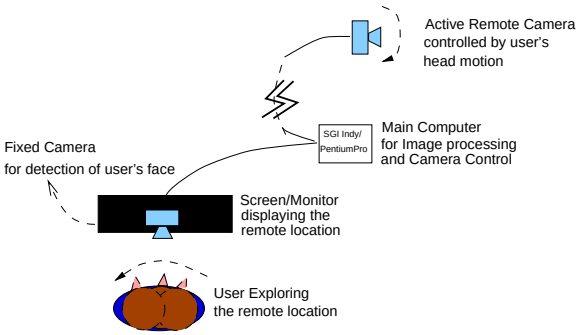
Extrema = (Er,Et,El,Eb)
Er = (RightX/faceEigX, RightY/faceEigY)
Et = (TopX/faceEigX, TopY/faceEigY)
El = (LeftX/faceEigX, LeftY/faceEigY)
Eb = (BottomX/faceEigX, BottomY/faceEigY)
EigenVals = (EigX/faceEigX, EigY/faceEigY)

Feature Vector = (area/faceArea, EigenVals, Extrema)

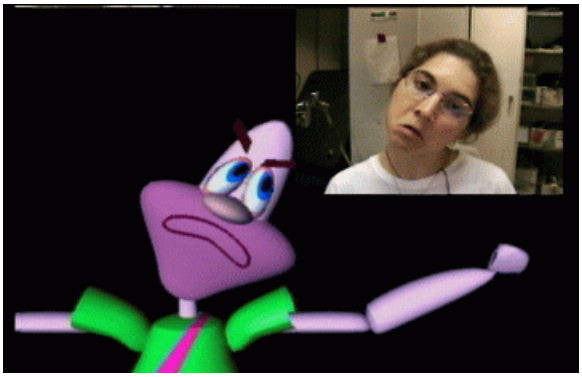
Mouth feature vector extraction



Open, sad, smile and smile-open recognized expressions.

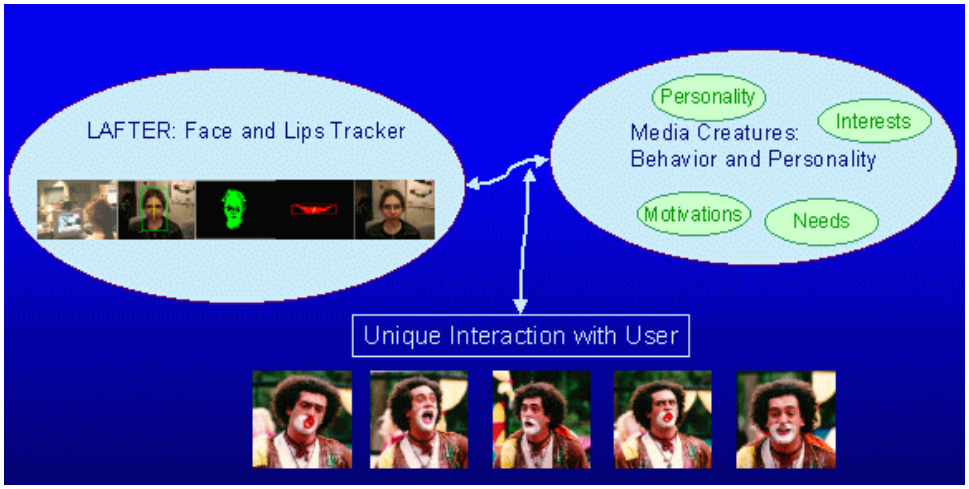


The virtual window: Local head positions are detected by the active tracking camera and used to control a moving camera in the remote site. The effect is that the image on the local monitor changes as if it were a window. The second image illustrates the virtual window system in use.



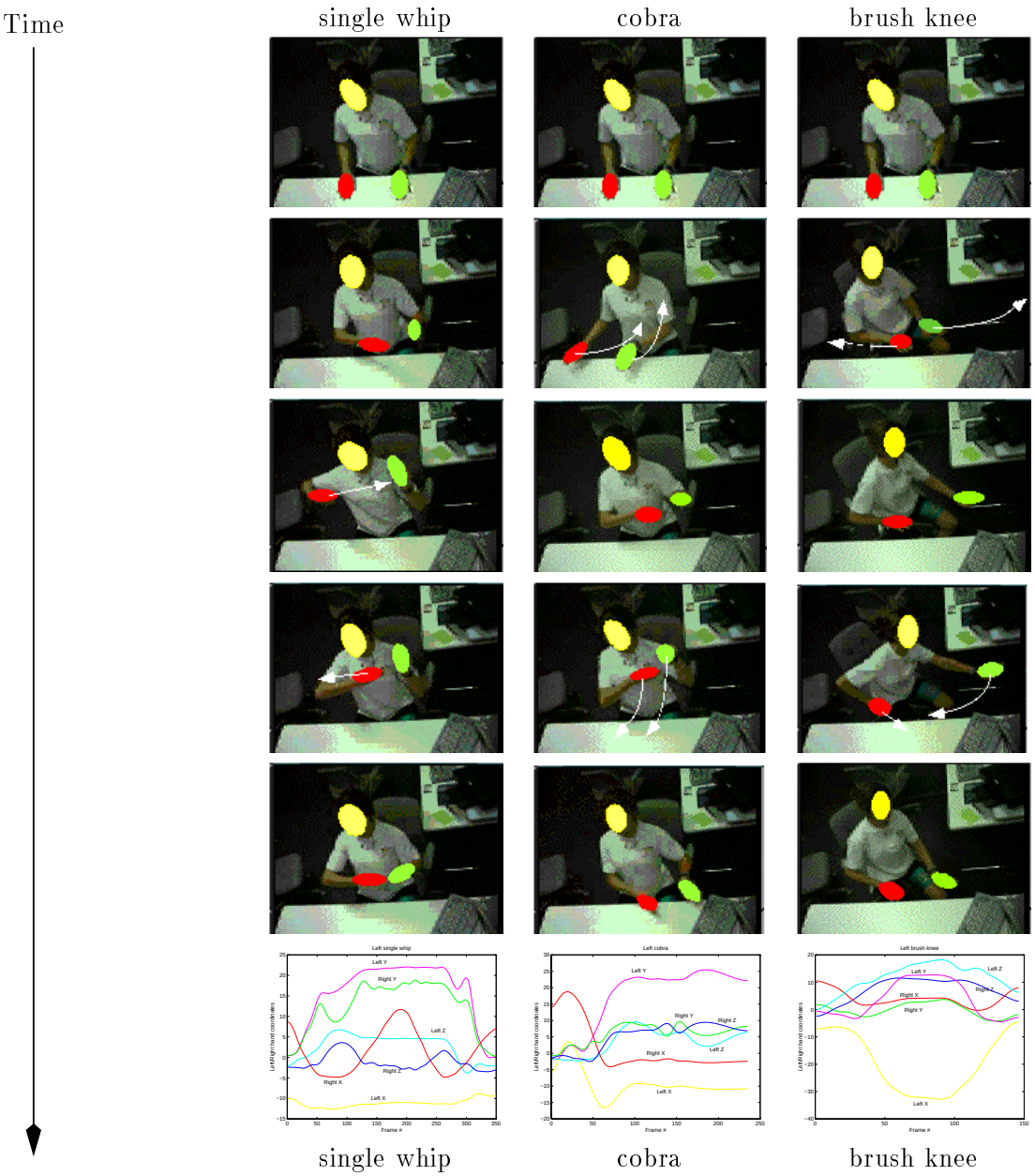
Real time computer graphics animation

Responsive Portrait typical interaction



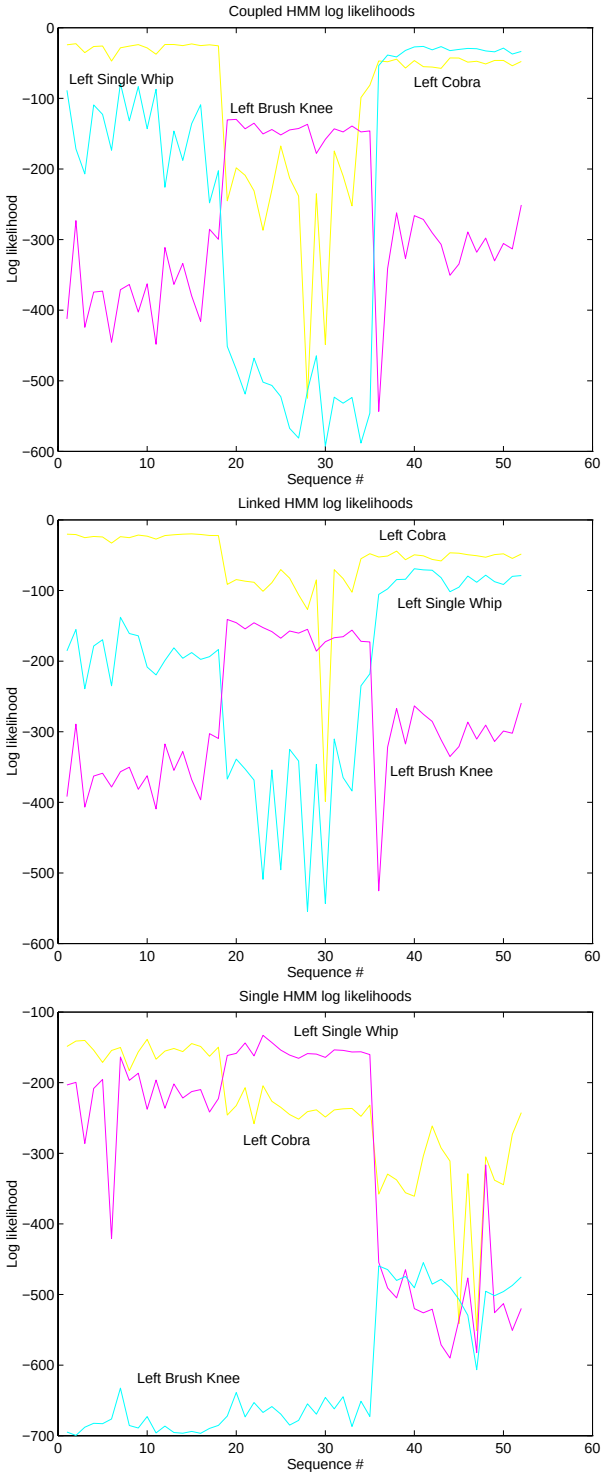
Responsive Portrait system architecture

Preferential coding: the first image is the JPEG flat encoded image (File size of 14.1Kb); the second is a very low resolution JPEG encoded image using flat coding (File size of 7.1Kb); the third one is a preferential coding encoded image with high resolution JPEG for the eyes and mouth but very low resolution JPEG coding for the face and background (File size of 7.1Kb).



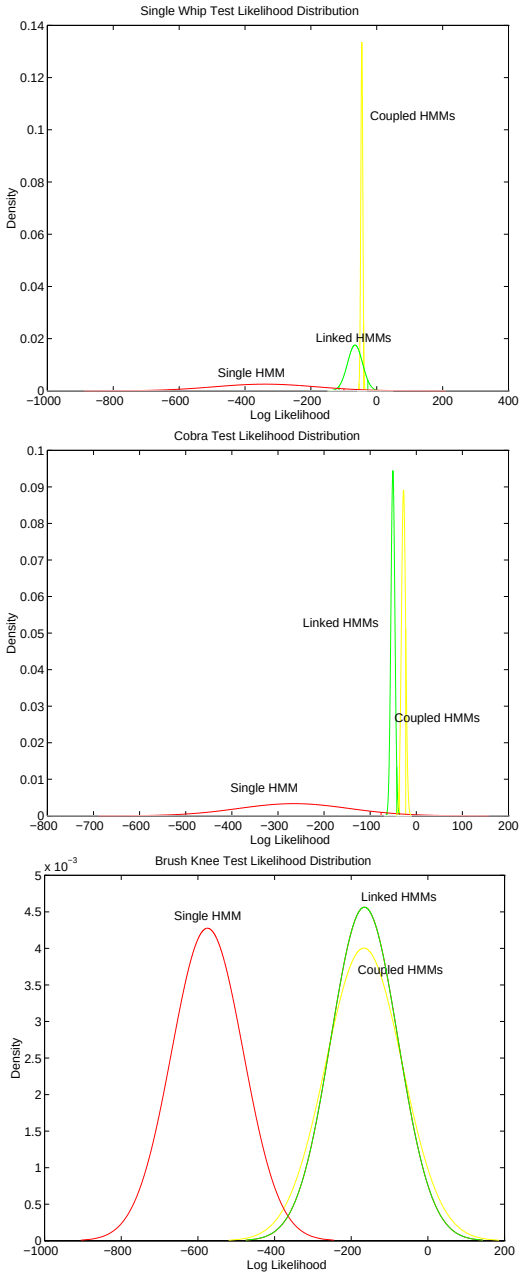
Hand tracking of three Tai-Chi gestures: selected frames overlaid with hand blobs from vision. The bottom-most graph shows the evolution of the feature vector over time

| 1–17: single whip | 18–34: brush knee | 19–52: cobra |



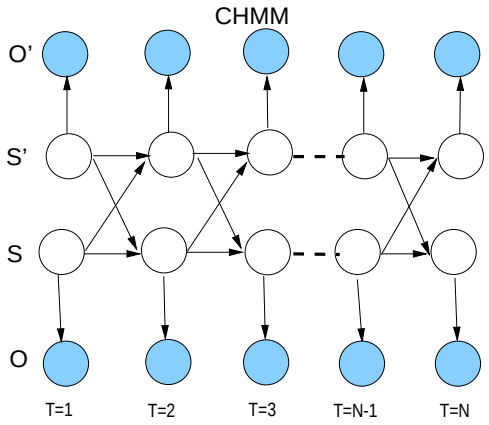
| 1–17: single whip | 18–34: brush knee | 19–52: cobra |

Classification by the CHMM, LHMM, and HMM, showing per-sequence normalized log likelihood. Only the CHMM attains the right discrimination structure

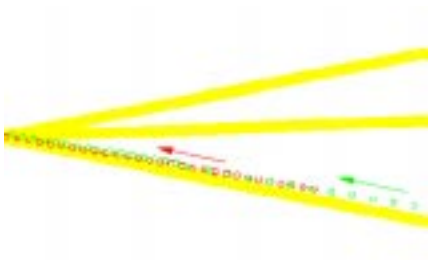


Likelihood probability distribution for each HMM type, learning single whip, cobra, and brush knee gestures, respectively. The CHMM consistently has the highest likelihood and the tightest distribution.

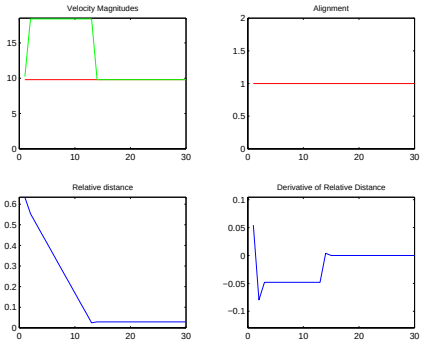
A typical image of a pedestrian plaza



Graphical Representation of CHMMs

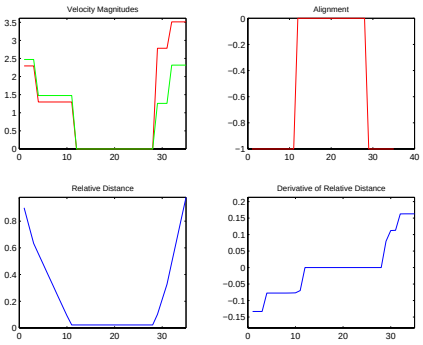


Agent trajectories for inter1

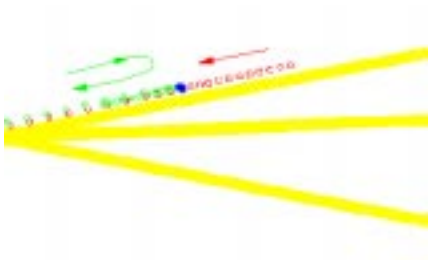


Feature Vector for inter1

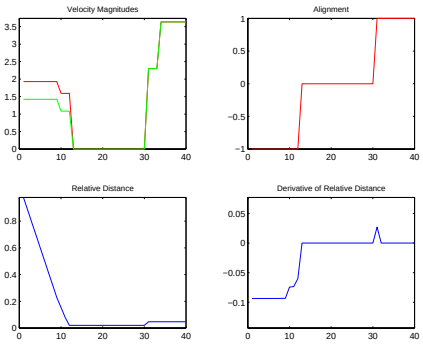
Agent trajectories for inter2



Feature Vector for inter2

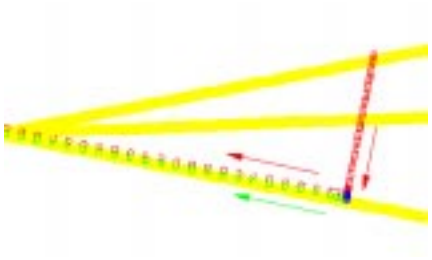


Agent trajectories for inter3

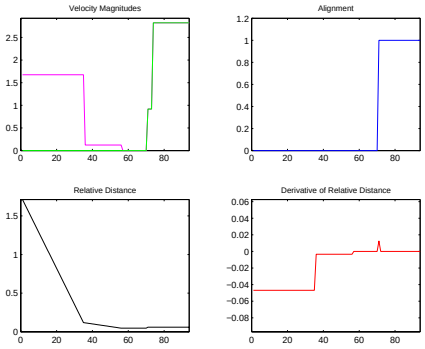


Feature Vector for inter3

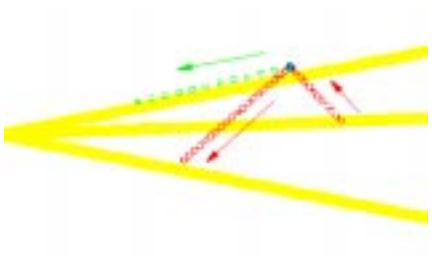
Example trajectories and feature vector for the interactions: follow, approach+meet+continue separately, and approach+meet+continue together



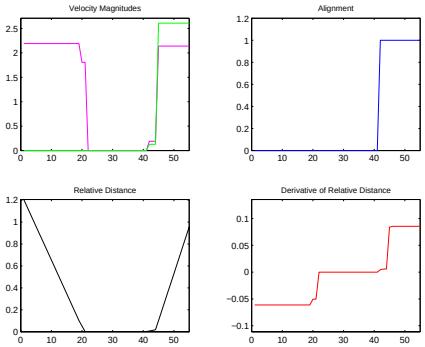
Agent trajectories for inter4



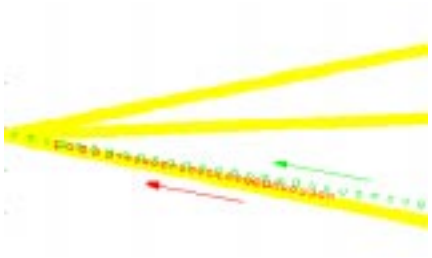
Feature Vector for inter4



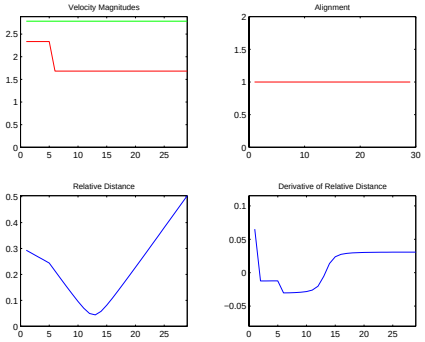
Agent trajectories for inter5



Feature Vector for inter5

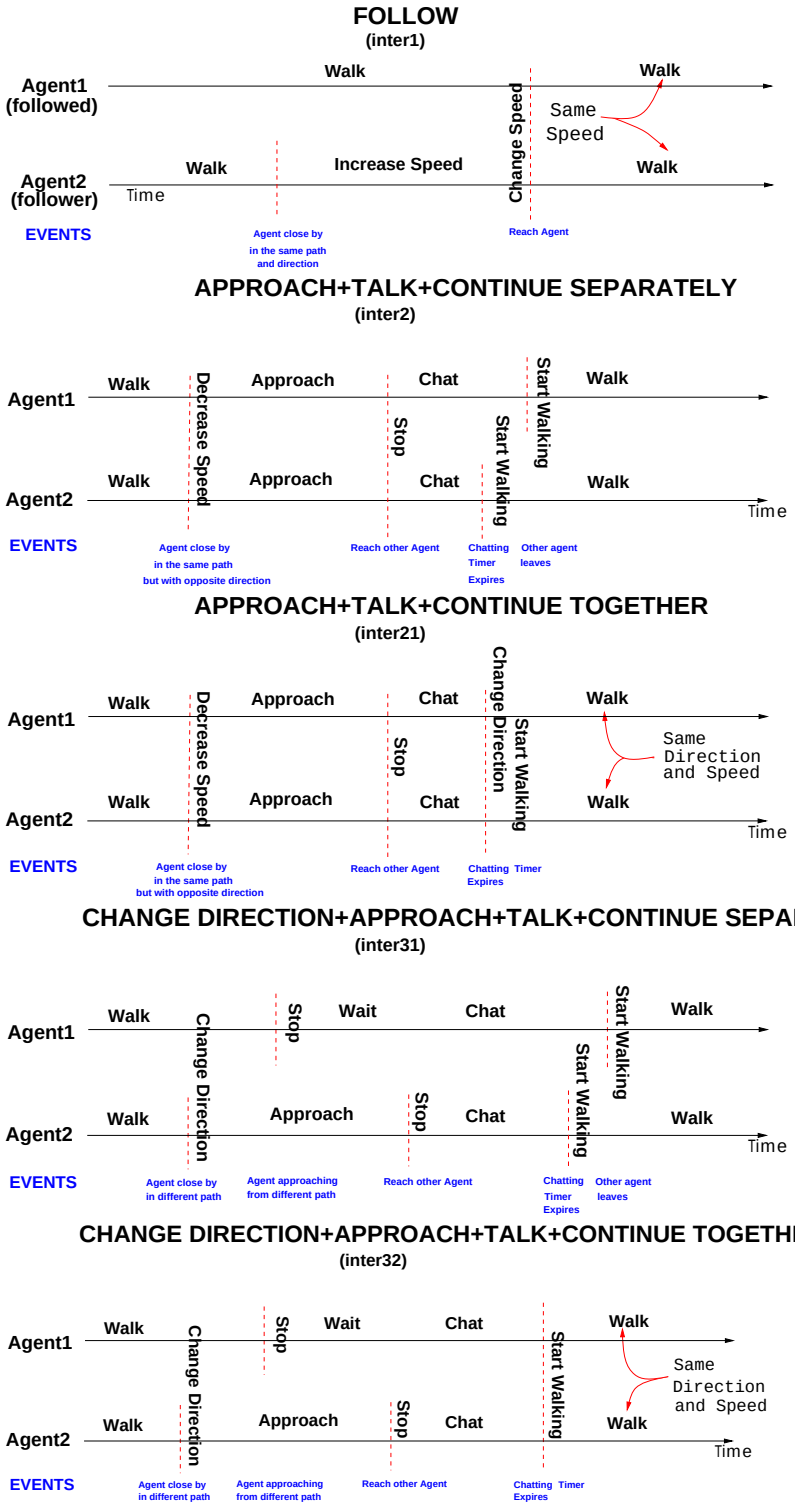


Agent trajectories for no interaction

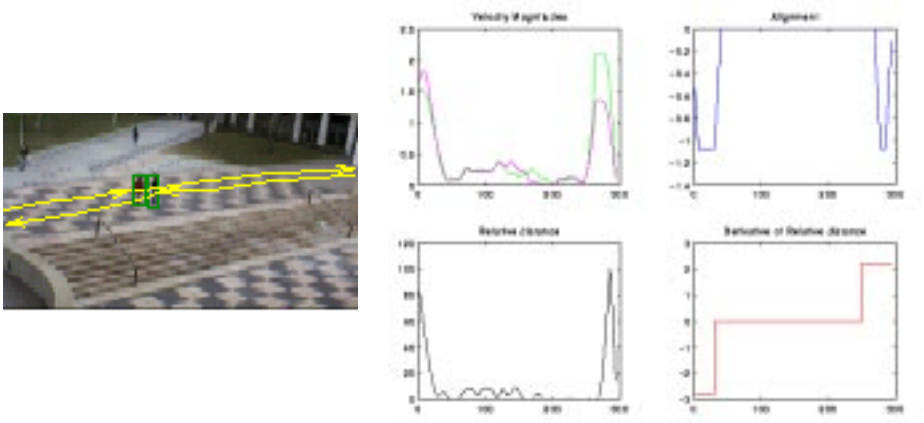


Feature Vector for no interaction

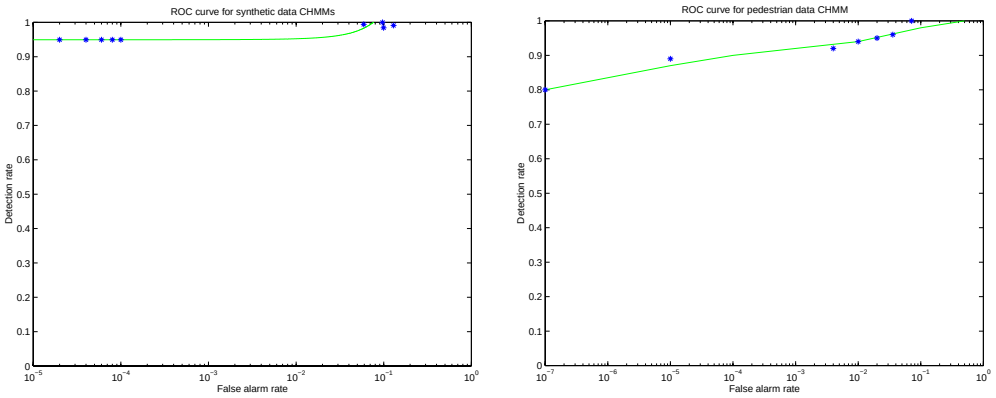
Example trajectories and feature vector for the interactions: change direction+approach+meet+continue separately, change direction+approach+meet+continue together, and no interacting behavior



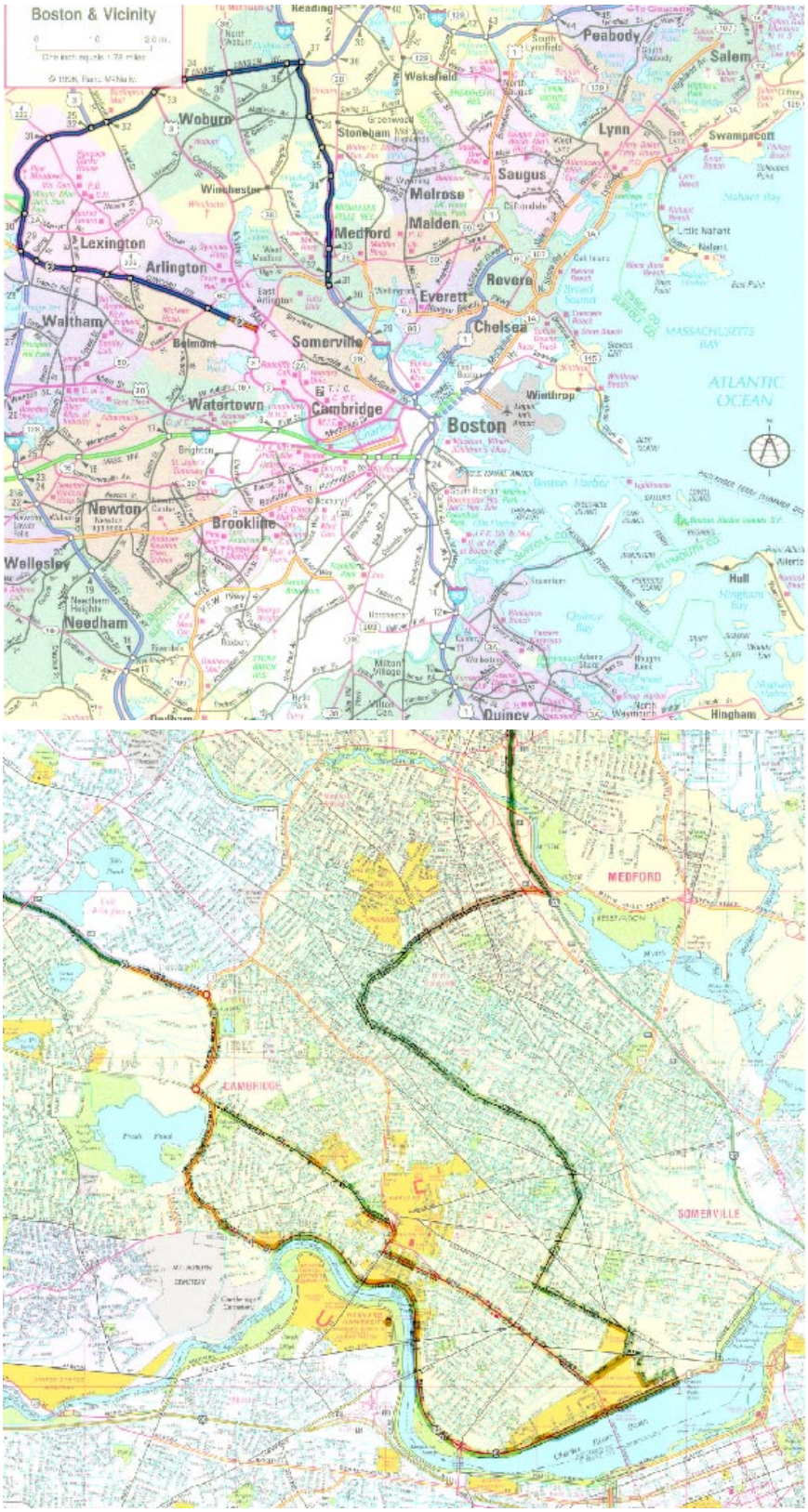
Timeline of the five complex behaviors in terms of events and simple behaviors



Example trajectories and feature vector for interaction 2, or approach, meet and continue separately behavior.

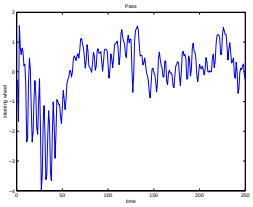


First figure: **ROC curve on synthetic data.** *Second Figure:* **ROC curve on real human data.**

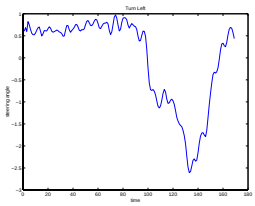


Route followed in the driving experiments: overview and city sections detail

Steering Angle

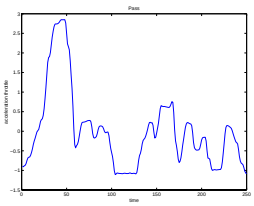


(a) Passing

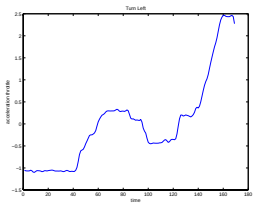


(b) Turning Left

Acceleration Throttle

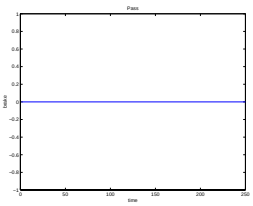


(a) Passing

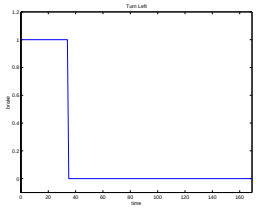


(b) Turning Left

Brake

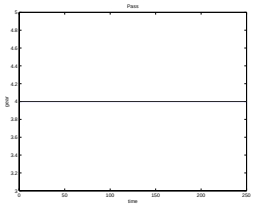


(a) Passing

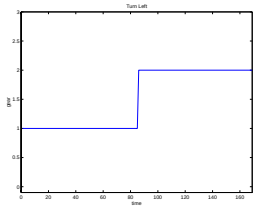


(b) Turning Left

Gear



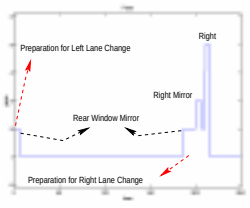
(a) Passing



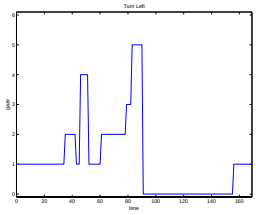
(b) Turning Left

Typical car signals for passing and turning left maneuvers

Gaze

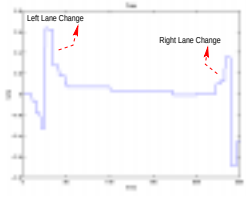


(a) Passing

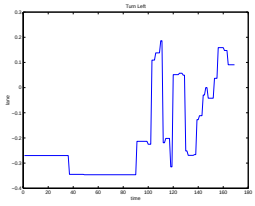


(b) Turning Left

Lane Feature

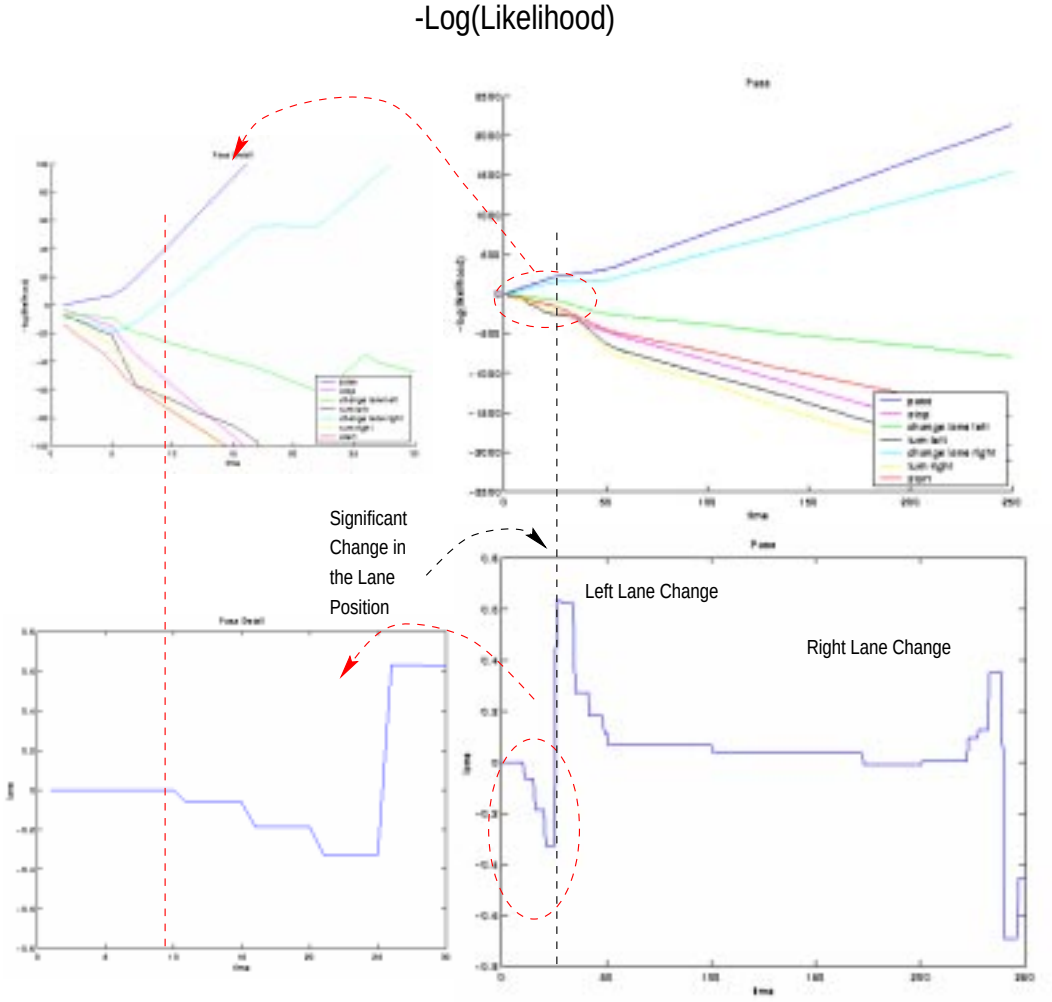


(a) Passing



(b) Turning Left

Typical contextual (gaze and lane) signals for a passing and turning left maneuvers



Prediction of a passing maneuver about 2/3 seconds before any significant lane change takes place.

Bibliography

[1]

[2] J. Aasman. *Road User Behavior: Theory and Practice*, chapter Implementations of car-driver behaviour and psychological models. Van Gorcum, Assen, 1987.

[3] D. Ackley, G. Hinton, and T. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9:147–169, 1985.

[4] K. Ahmed, E. Moshe, H. Koutsopoulos, and R. Mishalani. Models of freeway lane changing and gap acceptance behavior. In *Proceedings of 13th International Symposium on Transportation and Traffic Theory*, 1996.

[5] K. Aizawa and T.S. Huang. Model-based image-coding: Advanced video coding techniques for very-low bit-rate applications. *PIEEE*, 83(2):259–271, February 1995.

[6] R. Allen and T. Rosenthal. A computer simulation analysis of safety critical maneuvers for assessing ground vehicle dynamic stability. *Vehicle Dynamics and Simulation*, SAE SP-950, 1993.

[7] J.R. Anderson, C.F. Boyle, A.T. Corbett, and M.W. Lewis. Cognitive modeling and intelligent tutoring. *Artificial Intelligence*, 42:7–49, 1990.

[8] J.R. Anderson and C. Lebiere. *The atomic components of thought.* 1998.

[9] M. Aschenbrenner and B. Biehl. *Challenges to Accident Prevention: the Issue of Risk Compensation Behavior*, chapter Improved safety through improved technical measures? Empirical studies regarding risk compensation processes in relation to anti-lock braking systems. Styx Publications, 1994.

[10] H. Asher and B. Galler. Collision warning using neighboring vehicle information. In *Proceedings of ITS America*, 1996.

[11] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. In *ICCV95*.

[12] A. Azarbayejani and A. Pentland. Camera self-calibration from one point correspondence. Technical Report 341, MIT Media Lab Vision and Modeling Group, 1995. submitted IEEE Symposium on Computer Vision.

[13] A. Azarbayejani and A. Pentland. Real-time self-calibrating stereo person-tracker using 3-D shape estimation from blob features. In *Proceedings, International Conference on Pattern Recognition*, Vienna, August 1996. IEEE.

[14] D. Bailey, J. Feldman, S. Narayanan, and G. Lakoff. Embodied lexical development. In NJ: Erlbaum Mahwah, editor, *Proceedings of the Nineteenth Annual Meeting of the Cognitive Science Society.*

[15] P. Baldi and Y. Chauvin. Hybrid modeling, hmm/nn architectures and protein applications. *Neural Computation*, 8(7):1541–1565, 1996.

[16] H. Barlow. Unsupervised learning. *Neural Computation*, 1:295–311, 1989.

[17] L.W. Barsalou. *Behavioral and Brain Sciences*, volume 22, chapter Perceptual symbol systems, pages 577–609. 1999.

[18] W. Bechtel and A Abrahamsen. *Connectionism and the Mind.* Oxford, 1991.

[19] R.D. Beer. A dynamical systems perspective on agent-environment interaction. *Artificial Intelligence*, 72:173–215, 1995.

[20] R.D. Beer. *Mind as Motion: Explorations in the Dynamics of Cognition.*, chapter Computational and Dynamical Languages for Autonomous Agents, pages 121–147. Cambridge MA: MIT Press, 1995.

[21] Y. Bengio and P. Frasconi. Input/output hmms for sequence processing. *IEEE Transactions on Neural Networks*, 7(5):1231–1249, 1996.

[22] M. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In *ICCV 95*, pages 374–381, 1995.

[23] N. Block. *Meaning and method: essays in honor of Hilary Putnam*, chapter Can the mind change the world?, pages 137–170. Cambridge: Cambridge, 1990.

[24] A. Bobick and R. Bolles. The representation space paradigm of concurrent evolving object descriptions. *PAMI*, 14(2):146–156, February 1992.

[25] A.F. Bobick. Computers seeing action. In *Proceedings of BMVC*, volume 1, pages 13–22, 1996.

[26] E.R. Boer, E.C. Hildreth, and M.A. Goodrich. A driver model of attention management and task scheduling: Satisficing decision making with dynamic mental models. In *Proceedings of the XVIIth European Annual Conference on Human Decision making and Manual Control*, 1998.

[27] E.R. Boer and M. Hoedemaeker. Modeling driver behavior with different degrees of automation: A hierarchical decision framework of interacting mental models. In *In Proceedings of the XVIIth European Annual Conference on Human Decision making and Manual Control*, 1998.

[28] M. Brand. Coupled hidden markov models for modeling interacting processes. *Submitted to Neural Computation*, November 1996.

[29] M. Brand. Understanding manipulation in video. In *AFGR96*, pages 94–99, 1996.

[30] Matthew Brand, Nuria Oliver, and Alex Pentland. Coupled hidden markov models for complex action recognition. In *submitted to CVPR97*, 1996.

[31] C. Bregler. Learning and recognizing human dynamics in video sequences. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 1997.

[32] Christoph Bregler and Stephen M. Omohundro. Surface learning with applications to lipreading. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 43–50. Morgan Kaufmann Publishers, Inc., 1994.

[33] L. Breiman. Bagging predictors. *Machine Learning*, 26(2):123–140, 1996.

[34] L. Breiman and P. Spector. Submodel selection and evaluation in regression: The x-random case. *International Statistical Review*, 60:291–319, 1992.

[35] F.Z. Brill, T.J. Olson, and C. Tserng. Event recognition and reliability improvements for the autonomous video surveillance system. In *Image Understanding Workshop*, Monterey, CA, 1998.

[36] R.A. Brooks. Intelligence without reason, computers and thought lecture. In *Proceedings of IJCAI*, Sidney, Australia, 1991.

[37] W.L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, 1994.

[38] W.L. Buntine. A guide to the literature on learning probabilistic networks from data. *IEEE Transactions on Knowledge and Data Engineering*, 1996.

[39] Hilary Buxton and Shaogang Gong. Advanced visual surveillance using bayesian networks. In *International Conference on Computer Vision*, Cambridge, Massachusetts, June 1995.

[40] Lee W. Campbell, David A. Becker, Ali Azarbayjani, Aaron F. Bobick, and Alex Pentland. Invariant features for 3-D gesture recognition. In *Proceedings, International Conference on Automatic Face and Gesture Recognition*, pages 157–162, Killington, VT, 1996. IEEE.

[41] Stuart K. Card, Thomas P. Moran, and Allen Newell. *The Psychology of Human-Computer Interaction*. 1983.

[42] C. Castel, L. Chaudron, and C. Tessier. What is going on? a high level interpretation of sequences of images. In *Proceedings of the workshop on conceptual descriptions from images, ECCV*, pages 13–27, 1996.

[43] D.J. Chalmers. On implementing a computation. *Minds and Machines*, 1994.

[44] D.J. Chalmers. *The Conscious Mind: In Search of a Fundamental Theory*. Oxford University Press, 1996.

[45] P.M. Churchland. *Scientific Realism and the Plasticity of Mind*. Cambridge University Press, 1979.

[46] P.S. Churchland and T. Sejnowski. *The computational brain*. Cambridge, MA, MIT Press, 1992.

[47] A. Clark. *Being there: Putting brain, body, and world together again*. Cambridge, MA: MIT Press, 1997.

[48] J.D. Courtney. Automatic video indexing via object motion analysis. *Pattern Recognition*, 30(4):607–625, 1997.

[49] T. Cover and J. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.

[50] J. Cremer, J. Kearney, Y. Papelis, and R. Romano. The software architecture for scenario control in the iowa driving simulator. In *Proceedings of the 4th Computer Generated Forces and Behavioral Representation*, 1994.

[51] J.L. Crowley and F. Berard. Multi-modal tracking of faces for video communications. In *CVPR97*, pages 640–645, 1997.

[52] A.R. Damasio. Time-locked multiregional retroactivation: A systems-level proposal for the neural substrates of recall and recognition. *Cognition*, 33:25–62, 1989.

[53] T. Darrell and A. Pentland. Active gesture recognition using partially observable markov decision processes. In *ICPR96*, page C9E.5, 1996.

[54] T. Darrell and A. Pentland. Cooperative robust estimation using layers of support. *PAMI*, pages 17(5):474–487, May 1995.

[55] T. Darrell, S. Sclaroff, and A. Pentland. Segmentation by minimal description. In *ICCV90*, pages pages 173–177, 1990.

[56] A.P. Dawid. Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, 2:25–36, 1992.

[57] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[58] A.P. Dempster, N.M. Laird, and D.B Rubin. Maximum likelihood from incomplete data via de *em* algorithm. *Journal of the Royal Statistical Society*, 39-B:1–38, 1977.

[59] E. Dickmanns and A. Zapp. A curvature-based scheme for improving road vehicle guidance by computer vision. In *Proceedings of the SPIE Conference on Mobile Robots*, 1986.

[60] B. Efron. The jackknife, the bootstrap and other resampling plans. *SIAM*, 1982.

[61] H.E. Egeth and D. Dagenbach. Parallel versus serial processing in visual search: Further evidence from subadditive effects of visual quality. *Journal of Experimental Psychology: Human Perception and Performance*, 17: 551-560, 1991.

[62] P. Ekman and W. Friesen. Facial action coding system. Consulting Psychology Press, Palo Alto, CA, 1978.

[63] A. Eleftheriadis and A. Jacquin. Model-assisted coding of video teleconferencing sequences at low bit rates. In *ISCAS*, May-June 1994.

[64] I. Essa and A. Pentland. Facial expression recognition using a dynamic model and motion energy. In *ICCV95*, pages 360–367, 1995.

[65] Irfan A. Essa. *"Analysis, Interpretation, and Synthesis of Facial Expressions"*. PhD thesis, MIT Department of Media Arts and Sciences, 1995.

[66] P.S. Fancher, R. D. Ervin, J. R. Sayer, M. Hagan, S. Bogard, Z. Bareket, M. L. Mefford, and J. Haugen. Intelligent cruise control field operational test. *Report No. UMTRI-97-4. The University of Michigan Transportation Research Institute, Ann Arbor, MI.*, 1997.

[67] J.H. Fernyhough, A.G. Cohn, and D.C. Hogg. Building qualitative event models automatically from visual input. In *ICCV98*, pages 350–355, 1998.

[68] R. Fikes, P. Hart, and N. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3(4), 1972.

[69] J. A. Fodor and Z. Pylyshyn. Connectionism and cog nitive architecture: a critical analysis. *Cognition*, 28, 3-71.

[70] J. Forbes, T. Huang, K. Kanazawa, and S. Russell. The batmobile: Towards a bayesian automated taxi. In *Proceedings of the Fourteenth Intl. Joint Conference on Artificial Intelligence*, 1995.

[71] W. J. Freeman and C. A. Skarda. *Brain Organization and Memory: Cells, Systems and Circuits*, chapter Representations: Who needs them?, pages 375–380. New York: Guildford Press, 1990.

[72] R. Fung and K.C. Chang. Weighting and integrating evidence for stochastic simulation in bayesian networks. In *Uncertainty in Artificial Intelligence*, volume 5. Morgan Kaufmann, 1989.

[73] R.M. Fung and S.L. Crawford. A system for induction of probabilistic models. In *Eighth National Conference on Artificial Intelligence*, pages 762–779. AAAI, 1990.

[74] K. Gardels. Automatic car controls for electronic highways. Technical Report GMR-276, General Motors Research Labs, 1960.

[75] S. Garlick and K. VanLehn. Cirrus: An automated protocol analysis tool. Technical Report Tech. Rep. 6, Dept. Psychology. Carnegie Mellon University, Pittsburgh, PA, 1987.

[76] W. Gaver. The affordances of media spaces for collaboration. CSCW, 1992.

[77] W.W. Gaver, G. Smets, and K. Overbeeke. A virtual window on media space. CHI, 1995.

[78] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[79] Z. Ghahramani. Factorial learning and the em algorithm. In MA MIT Press, Cambridge, editor, *Advances in Neural Information Processing Systems 7*, pages 617–624, 1995.

[80] Z. Ghahramani and G.E. Hinton. Parameter estimation for linear dynamical systems. Technical report, Dept. Computer Science, Univ. Toronto, 1996.

[81] Zoubin Ghahramani. Learning dynamic bayesian networks. *Adaptive Processing of Temporal Information*, 1997.

[82] Zoubin Ghahramani and Michael I. Jordan. Factorial hidden markov models. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 472–478. The MIT Press, 1996.

[83] Zoubin Ghahramani and Michael I. Jordan. Factorial hidden Markov models. In David S. Touretzky, Michael C. Mozer, and M.E. Hasselmo, editors, *NIPS*, volume 8, Cambridge, MA, 1996. MITP.

[84] Jr. Gibbs, R.W. *The poetics of mind: Figurative thought, language, and understanding.* New York: Cambridge University Press, 1994.

[85] J.J Gibson. *The ecological approach to visual perception.* Houghton Mifflin, New York, 1979.

[86] G.G. Globus. Toward a noncomputational cognitive neuroscience. *Journal of Cognitive Neuroscience*, 4(4):299–310, 1992.

[87] A.S. Grant and J.T. Mayes. *Human-Computer Interaction and Complex Systems*, chapter Cognitive Task Analysis? Academic Press Ltd., 1991.

[88] I. Haritaoglu, D. Harwood, and L. Davis. A real time system for detecting and tracking people. *Image and Vision Computing Journal*, 1999.

[89] David Heckerman. A tutorial on learning with bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, Redmond, Washington, 1995. Revised June 96.

[90] M. Helander. Applicability of driver's electrodermal response to the design of the traffic environment. *Journal of Applied Psychology*, 63:481–488, 1978.

[91] H. Hennecke, K. Venkatesh, and D. Stork. Using deformable templates to infer visual speech dynamics. Technical report, California Research Center, June 1994.

[92] M. Henrion. Propagation of uncertainty in bayesian networks by probabilistic logic sampling. In J.F. Lemmer and L.N. Kanal, editors, *Uncertainty in Artificial Intelligence*, volume 2. Elsevier/North-Holland, Amsterdam, 1988.

[93] G.E. Hinton, P. Dayan, B.J. Frey, and R.M. Neal. The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–1161, 1995.

[94] J.S.U. Hjorth. *Computer Intensive Statistical Methods Validation, Model Selection, and Bootstrap*. London: Chapman & Hall, 1994.

[95] T. Hobbes. *Leviathan*. Penguin Classics, 1651.

[96] E. Hollnagel and D.D. Woods. Cognitive systems engineering: New wine in new bottles. *International journal of Man-Machine Studies*, 18:583–600, 1983.

[97] T. Huang, D. Koller, J. Malik, G. Ogasawara, B. Rao, S. Russel, and J. Weber. Automatic symbolic traffic scene analysis using belief networks. pages 966–972. Proceedings 12th National Conference in AI, 1994.

[98] H. Hunke. Locating and tracking of human faces with neural networks. Technical report, Technical report, CMU, CMU, Pittsburgh PA, August 1994, 1994.

[99] C.M. Hurvich and C.-L. Tsai. Regression and time series model selection in small samples. *Biometrika*, 76:297–307, 1989.

[100] M. Ikegaya, N. Asanuma, S. Ishida, and S. Kondo. Development of a lane following assistance system. In *AVEC 98*, page 9837102, 1998.

[101] V. Isham. An introduction to spatial point processes and markov random fields. *International Statistical Review*, 49:21–43, 1981.

[102] Y.A. Ivanov and A.F. Bobick. Parsing multi-agent interactions. Technical Report 479, Media Lab, MIT, 1998.

[103] Michon J.A. *Human Behavior and Traffic Safety*, chapter A critical view of driver behavior models: what do we know, what should we do? Plenum, 1985.

[104] T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In *Neural Information Processing Systems 12 (NIPS'99)*, 1999.

[105] R. Jackendoff. On beyond zebra: The relation of linguistic and visual information. *Cognition*, 26:89–114, 1987.

[106] T.S. Jebara and A.P. Pentland. Parametrized structure from motion for 3d adaptive feedback tracking of faces. In *CVPR97*, pages 144–150, 1997.

[107] H. Jeffreys. *Theory of Probability*. London: Oxford University Press, 1961.

[108] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in recursive graphical models by local computations. *Computational Statistical Quarterly*, 4:269–282, 190.

[109] F.V. Jensen. *An Introduction to Bayesian Networks*. Springer-Verlag, London, 1996.

[110] B.E. John. Extensions of goms analyses to expert performance requiring perception of dynamic visual and auditory information. In New York ACM, editor, *Proceedings of CHI'90 (Seattle, Washington, April 30-May 4, 1990)*, pages 107–115, 1990.

[111] G. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of Machine Learning*, pages 121–129. Morgan Kaufmann, 1994.

[112] N. Johnson and D. C. Hogg. Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, 14:609–615, 1996.

[113] P.M. Jones, C.M. Mitchell, and K.S. Rubin. Validation of intent inferencing by a model-based operator's associate. *International Journal of Man-Machine Studies*, 33:177–202, 1990.

[114] M.I. Jordan, Z. Ghahramani, T.S. Jaakkola, and L.K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 1999.

[115] Michael I. Jordan, Zoubin Ghahramani, and Lawrence K. Saul. Hidden Markov decision trees. In David S. Touretzky, Michael C. Mozer, and M.E. Hasselmo, editors, *NIPS*, volume 8, Cambridge, MA, 1996. MITP.

[116] J.Rissanen. *Encyclopedia of Statistical Sciences*, volume 5, chapter Minimum-description-length principle, pages 523–527. Wiley:New York, 1987.

[117] R. Kalman and R.S. Bucy. New results in linear filtering and prediction theory. *Trans. ASME Ser. D., J. Basic Engineering*, 83:pp. 95–107, March 1961.

[118] R. E. Kalman. A new approach to linear filtering and prediction problems. *J. Basic Eng.*, 82:35–45, 1960.

[119] K. Kanazawa, D. Koller, and S. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In *Uncertainty in Artificial Intelligence*, volume 11. Morgan Kaufmann, 1995.

[120] M. Kass, A.P. Witkin, and D. Terzopoulos. Snakes: Active contour models. *IJCV*, 1(4):321–331, January 1988.

[121] R.E. Kass and L. Wasserman. The selection of prior distributions by formal rules. *Journal of the American Statistical Association*, (Vol. 91, N. 435):1343–1370, 1996.

[122] R. Kauth, A. Pentland, and G. Thomas. Blob: An unsupervised clustering approach to spatial preprocessing of mss imagery. In *11th Int'l Symp. on Remote Sensing of the Environment*, Ann Harbor MI, 1977.

[123] D.E. Kieras. *Handbook of Human-Computer Interaction*, chapter Towards a practical GOMS model methodology for user interface design, pages 135–158. Amsterdam: North-Holland Elsevier, 1988.

[124] D.E. Kieras and D.E. Meyer. A computational theory of executive cognitive processes and multiple-task performance: Part 1. basic mechanisms. *Psychological Review*, 104:3–65, 1997.

[125] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.

[126] D. Koller and M. Sahami. Towards optimal feature selection. In *Proceedings of the 13th International Conference on Machine Learning (ML)*, pages 284–292, 1996.

[127] B. Kosko. *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence*. Prentice-Hall, 1991.

[128] S.M. Kosslyn. *Image and mind*. Cambridge, MA: Harvard University Press, 1980.

[129] S.M. Kosslyn. *Image and brain: the resolution of the imagery debate*. Cambridge, MA: MIT Press, 1994.

[130] D. Kotchetkov. Creation of automobile automatic control system algorithms with use of computer simulation. 1996.

[131] N. Kuge, T. Yamamura, and O. Shimoyama. A driver behavior recognition method based on a driver model framework. *Society of Automotive Engineers Publication*, 1998.

[132] Y. Lallement. A hierarchical ensemble of decision trees applied to classifying data from psychological experiment. In AAAI Press, editor, *Proceedings of the Eleventh International FLAIRS Conference*, Sanibel Island, FL, 1998.

[133] R.W. Langacker. Constituency, dependency, and conceptual grouping. *Cognitive Linguistics*, 8:1–32, 1997.

[134] P. Langley and S. Ohlsson. Automated cognitive modeling. In CA: Morgan Kaufman Los Altos, editor, *Proceedings of the AAAI-89*, pages 193–197, 1989.

[135] A. Lanitis, C.J. Taylor, and T.F. Cootes. A unified approach for coding and interpreting face images. In *ICCV95*, pages 368–373, 1995.

[136] S.L. Lauritzen, A.P. Dawid, B.N. Larsen, and H.G. Leimer. Independence properties of directed markov fields. *Networks*, 20:491–505, 1990.

[137] S.L. Lauritzen and D.J. Speigelhalter. Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal Royal Statistical Society Ser.B*, 50:157–224, 1988.

[138] S.L. Lauritzen and D.J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal Royal Statistical Society Serie B*, 50:115–117, 1988.

[139] S.L. Lauritzen and N. Wermuth. Graphical models for associations between variables, some of which are qualitative and some quantitative. *Annals of Statistics*, 17:31–57, 1989.

[140] D. Legge and J. Barber. *Information and Skill.* London, Methuen, 1976.

[141] J.J. Lien, T. Kanade, J.F. Cohn, A. Zlochower, and C.C. Li. Automatically recognizing facial expressions in spatio-temporal domain using hidden markov models. In *Proceedings of the Workshop on Perceptual User Interfaces, PUI97*, Banff, Canada, 1997.

[142] A. Liu and A. Pentland. Towards real-time recognition of driver intentions. In *Proc. of the 1997 IEEE Intelligent Transportation Systems Conference*, 1997.

[143] R. Magnolfi and P. Nosi. Analysis and synthesis of facial motions. In IEEE, editor, *Intl. Workshop on Automatic Face and Gesture Recognition*, volume 1, pages 308–313, Zurich, 1995.

[144] I. Masaki. *Vision-based vehicle guidance.* Springer-Verlag, 1992.

[145] K. Matsuno and P. Nesi. Automatic recognition of human facial expressions. In IEEE, editor, *CVPR'95*, volume 1, pages 352–359, 1995.

[146] J. McCarthy. Circumscription: A form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[147] J. McCarthy. Epistemological problems of artificial intelligence. In B. L. Webber and N. J. Nilsson, editors, *Readings in Artificial Intelligence*, pages 459–465. Kaufmann, Los Altos, CA, 1981.

[148] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. L. Webber and N. J. Nilsson, editors, *Readings in Artificial Intelligence*, pages 431–450. Kaufmann, Los Altos, CA, 1981.

[149] J. McKnight and B. Adams. Driver education and task analysis volume 1: Task descriptions. Technical report, Department of Transportation, National Highway Safety Bureau, 1970.

[150] M. Meila and M.I. Jordan. "learning fine motion by markov mixtures of experts". In *Advances in Neural Information Processing Systems 8*, 1996.

[151] G. Miller and P.N Johnson-Laird. *Language and perception*. Harvard Univ. Press, 1976.

[152] J.O. Miller. Discrete and continuous models of human information processing: Theoretical distincitons and empirical results. *Acta Psychologica*, pages 67: 191–257, 1988.

[153] J. Modestino and J. Zhang. A markov random field model-based approach to image segmentation. *IEEE Trans. Patt. Anal. Mach. Int. PAMI*, pages 606–615, 1992.

[154] B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. In *ICCV95*, pages 786–793, 1995.

[155] P.C.M. Molenaar. Neural netowrk simulation of a discrete model of continuous effects of irrelevant stimuli. *Acta Psychologica*, pages 74: 237–258, 1990.

[156] S. Morishima. Emotion model. In *Intl. Workshop on Automatic Face and Gesture Recognition*, pages 284–289, Zurich, 1995.

[157] Y. Moses, D. Reynard, and A. Blake. Determining facial expressions in real time. In *ICCV 95*, pages 296–301, 1995.

[158] K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: an empirical study. In Laskey K.B. and Prade H. (editors), editors, *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers, San Francisco*, 1999.

[159] H.H. Nagel. From image sequences towards conceptual descriptions. *IVC*, 6(2):59–74, May 1988.

[160] S. Nakano, Katsutoshi, and Maeda. Development of an intelligent steering system for automated highway systems. In *Proceedings of FISITA 98*, 1998.

[161] R. Neal. Connectionist learning of belief networks. *Artificial Intelligence*, 56:71–113, 1992.

[162] R.M. Neal and G.E. Hinton. A new view of the em algorithm that justifies incremental and other variants. 1993.

[163] New Jersey Depart. of Law and Public Safety. *Motor Vehicle Services. New Jersey Driver Manual*, 1988.

[164] A. Newell. The knowledge level. *Artificial Intelligence*, 18:87–127, 1982.

[165] A. Newell and H.A. Simon. *Human Problem Solving*. Englewood Cliffs, 1972.

[166] Newell90. *Unified theories of cognition*. Harvard University Press, Cambridge, MA, 1990.

[167] A. Niehaus and R. Stengel. Probability-based decision making for automated highway driving. *IEEE Transactions on Vehicular Technology*, 43(3), 1994.

[168] D. C. Noelle and G. W. Cottrell. In search of articulated attractors. In Mahwah: Lawrence Erlbaum, editor, *Proceedings of the 18th Annual Conference of the Cognitive Science Society*, pages 329–334, 1996.

[169] S. Ohlsson. *Diagnostic Monitoring of Skill and Knowledge Acquisition*, chapter Trace analysis and spatial reasoning: an example of intensive cognitive diagnosis and its implications for testing, pages 251–296. In N. Frederiksen, R. Glaser, A. Lesgold, and M.G. Shafto (Eds.), 1990.

[170] K. Ohzeki, T. Saito, M. Kaneko, and H. Harashima. Interactive model-based coding of facial image sequence with a new motion detection algorithm. *IEICE*, E79B(10):1474–1483, October 1996.

[171] N. Oliver, F. Berard, and A. Pentland. Lafter: Lips and face tracking. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR97)*, S.Juan, Puerto Rico, June 1997.

[172] N. Oliver, B. Rosario, and A. Pentland. Graphical models for recognizing human interactions. In *To appear in Proceedings of NIPS98, Denver, Colorado,USA*, November 1998.

[173] N. Oliver, B. Rosario, and A. Pentland. Statistical modeling of human behaviors. In *Proceedings of CVPR98, Motion Understanding Workshop, Sta. Barbara, CA*, 1998.

[174] N. Oliver, B. Rosario, and A. Pentland. A synthetic agent system for modeling human interactions. Technical report, Vision and Modeling Technical Report, Media Lab, MIT, Cambridge, 1998. http://whitechapel.media.mit.edu/pub/tech-reports.

[175] N. Oliver, B. Schoelkopf, and A. Smola. Natural regularization in svms. In Springer Verlag, editor, *To Appear in Advances in Large Margin Classifiers*, 2000.

[176] C. Padgett and G. Cottrell. Representing face images for emotion classification. In *Neural Information Processing Systems, NIPS'96*, Denver, Colorado, USA, 1996.

[177] G. Parisi. *Statistical Field Theory*. Addison-Wesley, Redwood City, CA, 1988.

[178] V.L. Parsegian. *This Cybernetic World of Men, Machines and Earth Systems.* Garden City N.Y.: Doubleday & Co. Inc, 1973.

[179] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, California, 1988.

[180] A. Pentland. Looking at people: Sensing for ubiquitous and wearable computing. To appear in a speciall issue of PAMI.

[181] A. Pentland. Classification by clustering. In *IEEE Symp. on Machine Processing and Remotely Sensed Data*, Purdue, IN, 1976.

[182] A. Pentland and A. Liu. Towards augmented control systems. In *Proceedings of IEEE Intelligent Vehicles*, 1995.

[183] A. Pentland and A. Liu. Modeling and prediction of human behavior. In *DARPA97*, page 201 206, 1997.

[184] A. Pentland and A. Liu. Modeling and prediction of human behavior. *Neural Computation*, 11:229–242, 1999.

[185] J. Petitot. *Les Catastrophes de la Parole.* Paris: Maloine, 1985.

[186] I. Pilowsky, M. Thornton, and M. Stokes. *Aspects of face processing*, chapter Towards the quantification of facial expressions with the use ofa mathematics model of a face, pages 340–348. 1986.

[187] M. Plutowski, S. Sakata, and H. White. Cross-validation estimates imse. In J.D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 391–398. Morgan Kaufman, 1994.

[188] J. B. Pollack. Recursive distributed representations. *Artificial Intelligence*, 46:77–105, 1990.

[189] D. Pomerleau. *Neural Network Perception for Mobile Robot Guidance.* PhD thesis, Carnegie Mellon University, 1992.

[190] D. Pomerleau. Ralph:rapidly adapting lateral position handler. In *Proceedings of IEEE Intelligent Vehicles*, 1995.

[191] D. A. Pomerleau. Neural networks for intelligent vehicles. In *Proc. Intelligent Vehicles Symposium '93*, pages 19–24, Tokyo, 1993.

[192] A.M. Poritz. Hidden markov models: a guided tour. In New York: IEEE Press, editor, *Proceedings of the IEEE Intl. Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 7–13, 1988.

[193] R. Prerau. *Developing and managing expert systems.* Addison-Wesley, Reading, MA, 1990.

[194] C.E. Priebe. Adaptive mixtures. *Journal of the American Statistical Association*, 89(427):796–806, sep 1994.

[195] J.J. Prinz. *Perceptual cognition: An essay on the semantics of thought.* PhD thesis, University of Chicago, 1997.

[196] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *IEEE*, 77(2):257–286, February 1989.

[197] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, January:4–16, 1986.

[198] A. Raftery. *Sociological Methodology*, chapter Bayesian model selection in social research. Marsden P., 1995.

[199] A. Ram, R. Arkin, G. Boone, and M. Pearce. Using genetic algorithms to learn active control parameters for autonomous robotic navigation. *Adaptive Behavior*, 2(3):277–305, 1994.

[200] R.A.Redner and H.F.Walker. Mixture densities, maximum likelihood and the em algorithm. *SIAM Review*, 26(2):195–239, April 1984.

[201] J. Rasmussen. *Information Processing and Human-Machine Interaction: an approach to cognitive engineering.* New York: North-Holland, 1986.

[202] H.E. Rauch. Solutions fo the linear smoothing problem. *IEEE Transactions on Automatic Control*, 8:371-372, 1963.

[203] D. Reece. *Selective Perception for Robot Driving.* PhD thesis, Carnegie Mellon University, 1992.

[204] E.D. Reichle, A. Pollatsek, D.L. Fisher, and K. Rayner. Toward a model of eye movement control in reading. *Psychological Review*, 105:125–157, 1998.

[205] J. Rillings and R. Betsold. Advanced driver information systems. *IEEE Transactions on Vehicular Technology*, 40(1), 1991.

[206] F.E. Ritter. *A methodology and software environment for testing process models' sequential predictions with protocols.* PhD thesis, Dept. Psychology, Carnegie Mellon University, 1992.

[207] F.E. Ritter and J.H. Larkin. Developing process models as summaries of hci action sequences. *Human-Computer Interaction*, 9:345–383, 1994.

[208] B. Rosario, N. Oliver, and A. Pentland. A synthetic agent system for modeling human interactions. In *Proceedings of AA99*, Seattle, Washington, 1999.

[209] J. Rosenblatt. *DAMN: A Distributed Architecture for Mobile Navigation.* PhD thesis, Carnegie Mellon University, 1996.

[210] P. Rosenbloom, W. Johnson, R. Jones, F. Koss, J. Laird, J. Lehman, R. Rubinoff, K. Schwamb, and M. Tambe. Intelligent automated agents for tactical air simulation: a progress report. In *Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation*, 1994.

[211] M. Rosenstein and P. R. Cohen. Concepts from time series. In CA: AAAI Press Menlo Park, editor, *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 739–745, 1998.

[212] M. Rydfalk. *CANDIDE: A parametrized face.* PhD thesis, Linköpnik University, EE Dept., Oct 1987.

[213] G. Ryle. *The Concept of Mind (1949).* Chicago: University of Chicago Press, 1984.

[214] D.D. Salvucci. *Mapping eye movements to cognitive processes.* PhD thesis, Doctoral Dissertation, Department of Computer Science, Carnegie Mellon University, 1999. Tech. Rep. No. CMU-CS-99-131.

[215] P.M. Sanderson, J. Scott, T. Johnston, J. Mainzer, L. Watanabe, and J. James. Macshapa and the enterprise of exploratory sequential data analysis (esda). *International Journal of Human-Computer Studies*, 41:633–681, 1994.

[216] Lawrence K. Saul and Michael I. Jordan. Boltzmann chains and hidden Markov models. In Gary Tesauro, David S. Touretzky, and T.K. Leen, editors, *NIPS*, volume 7, Cambridge, MA, 1995. MITP.

[217] B. Schiele and A. Waibel. Gaze tracking based on face color. In *International Workshop on Automatic Face and Gesture Recognition*, pages pages 344–349, 1995.

[218] G. Schwarz. Estimating the dimensions of a model. *Annals of Statistics*, 6:461–464, 1978.

[219] R.D. Shachter, S.K. Anderson, and P. Szolovits. Global conditioning for probabilistic inference in belief networks. In Morgan Kaufmann, editor, *Proceedings of the Uncertainty in AI Conference*, pages 514–522, San Francisco, CA, 1994.

[220] J. Shao. Linear model selection by cross-validation. *J. of the American Statistical Association*, 88:486–494, 1993.

[221] J. Shao. An asymptotic theory for linear model selection. *Statistica Sinica*, 1995.

[222] J. Shao and D. Tu. *The Jackknife and Bootstrap.* Springer-Verlag, 1995.

[223] H. A. Simon. Allen newell: The entry into complex information processing. *Artificial Intelligence*, 59:251–259, 1993.

[224] J.B. Smith, D.K. Smith, and E. Kuptsas. Automated protocol analysis. *Human Computer Interaction*, 8:101–145, 1993.

[225] P. Smolensky. *Mathematical Perspectives on Neural Networks*, chapter Computational perspectives on neural networks, pages 17–39. Smolensky, P., Mozer, M. C., and Rumelhart, D. E. (Eds.), 1996.

[226] P. Smyth. Hidden markov models for fault detection in dynamic systems. *Pattern Recognition*, 27:149–164, 1994.

[227] P. Smyth, D. Heckerman, and M. Jordan. Probabilistic independence networks for hidden markov probability models. *Neural Computation*, 9(2):227–269, 1997.

[228] Padhraic Smyth, David Heckerman, and Michael Jordan. Probabilistic independence networks for hidden Markov probability models. AI memo 1565, MIT, Cambridge, MA, February 1996.

[229] F. Sparacino. Directive: Choreographing media for interactive virtual environments. Master's thesis, MIT Media Lab, 1996.

[230] F. Sparacino, N. Oliver, A. Pentland, and G. Davenport. Responsive portraits. In *The Eighth International Symposium on Electronic Art*, Sept. 1997.

[231] D.J. Spiegelhalter, A.P. Dawid, T.A. Hutchinson, and R.G. Cowell. Probabilistic expert systems and graphical modelling: a case study in drug safety. *Phil. Trans. Royal Society London.*, A. 337:387–405, 1991.

[232] T. Starner and A. Pentland. Real-time asl recognition from video using hmm's. Technical report, Technical Report 375, MIT, Media Laboratory, MIT, Media Laboratory, Cambridge, MA 02139, 1996.

[233] Thad Starner and Alex Pentland. Visual recognition of american sign language using hidden markov models. In *International Workshop on Automatic Face and Gesture Recognition*, Zurich, Switzerland, 1995.

[234] C. Stauffer. Automatic hierarchical classification using time-based co-occurrences. In *Computer Vision and Pattern Recognition*, Fort Collins, CO, 1999.

[235] A. Stolcke. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, (21(2)), 1995.

[236] M. Stone. Asymptotics for and against cross-validation. *Biometrika*, 64:29–35, 1977.

[237] M. Stoneridge. *Practical Horseman's Book of Riding, Training, and Showing Hunters and Jumpers*. Doubleday, 1989.

[238] R Sukthankar. Racoon: A real-time autonomous car chaser operating optimally at night. In *Proceedings of IEEE Intelligent Vehicles*, 1993.

[239] R. Sukthankar. *Situation Awareness for Tactical Driving.* PhD thesis, Robotics Institute, Carnegie Mellon University, 1997.

[240] R. Sukthankar, J. Hancock, and C. Thorpe. Tactical-level simulation for intelligent transportation systems. *Journal on Mathematical and Computer Modeling*, 27(9-11):228–42, 1998.

[241] P. Thagard. *Conceptual revolutions.* Princeton, Princeton University Press, 1992.

[242] E. Thelen and L. B. Smith. *A Dynamics Systems Approach to the Development of Cognition and Action.* Cambridge MA: MIT Press, 1993.

[243] D. M. Titternington. Recursive parameter estimation using incomplete data. *J. R. Statist. Soc. B*, 46:257–267, 1984.

[244] J. et al Treat. Tri-level study of the causes of traffic accidents: Final report, volume 1. Technical report, Federal Highway Administration, US DOT, 1979.

[245] H.H. van der Molen and A.T.M. Botticher. *Road Users and Traffic Safety*, chapter Risk models for traffic participants: A concerted effort for theoretical operationalizations. Van Gorcum, Assen, 1987.

[246] P. Van Geert. The dynamics of father brown. essay review of a dynamic ssytems approach to the development of cognition and action. *Human Development*, 39(1):57–66, 1996.

[247] T. J. Van Gelder and R. Port. *Symbol Processing and Connectionist Network Models in Artificial Intelligence and Cognitive Modelling: Steps Toward Principled Integration*, chapter Beyond symbolic: Towards a Kama-Sutra of compositionality, pages 107–25. In V. Honavar & L. Uhr ed., 1994.

[248] T. J. Van Gelder and R. Port. *Mind as Motion: Explorations in the Dynamics of Cognition.*, chapter It's About Time: An Overview of the Dynamical Approach to Cognition. Cambridge MA: MIT Press, 1995.

[249] T.J. Van Gelder. Compositionality: a connectionist variation on a classical theme. *Cognitive Science*, 14:355–384, 1990.

[250] D. Vandermeulen, R. Verbeeck, L. Berben, D. Delaere, P. Suetens, and G. Marchal. Continuous voxel classification by stochastic relaxation: theory and application to mr imaging and mr angiography. *Image and Vision Computing*, 12(9):559–572, 1994.

[251] R. Vanstrum and G. Caples. *Highway Research Record*, chapter Perception model for describing and dealing with driver involvement in highway accidents, pages 365:17–24. 1972.

[252] V. Vapnik. *Estimation of Dependences Based on Empirical Data [in Russian]*. Nauka, Moscow, 1979. (English translation: Springer Verlag, New York, 1982).

[253] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[254] R. von Tomkewitsch. Dynamic route guidance and interactive transport management with ali-scout. *IEEE Transactions on Vehicular Technology*, 40(1):45–50, 1991.

[255] R. Wallace, A. Stentz, C. Thorpe, H. Moravec, W. Whittaker, and T. Kanade. First results in robot road-following. In *Proceedings of the IJCAI*, 1985.

[256] J. Want and R. Knipling. Single vehicle roadway departure crashes: problem size assessment and statistical description. Technical Report Technical Report DTNH-22-91-C-03121, National Highway Traffic Safety Administration, 1993.

[257] D. A. Waterman and A. Newell. Pas-ii: An interactive task-free version of an automatic protocol analysis system. In *Proc. of the 3rd IJCAI*, pages 431–445, Stanford, MA, 1973.

[258] K. Waters. A muscle model for animating three-dimensional facial expression. In M. C. Stone, editor, *SIGGRAPH '87 Conference Proceedings (Anaheim, CA, July 27–31, 1987)*, pages 17–24. Computer Graphics, Volume 21, Number 4, July 1987.

[259] E. Weiss. *Untersuchung und Rekonstruktion von Ausweich und Fahrspurwechselvorgangen*. VDI-Verlag, 1987.

[260] S.M. Weiss and C.A. Kulikowski. *Computer Systems That Learn*. Morgan Kaufmann, 1991.

[261] Y. Weiss and W.T. Freeman. Correctness of belief propagation in gaussian graphical models of arbitrary topology. Technical report, UC Berkeley CS Department TR UCB–CSD-99-1046, 1999.

[262] D. Wharton. How to train your eyes for better driving. Readers' Digest, November 1958.

[263] William A. Wheeler, John D. Lee, Mireille Raby, Rhonda A. Kinghorn, Alvah C. Bittner, Jr., and Marvin C. McCallum. Predicting driver behavior using advanced traveler information systems. In *Proceedings of the Human Factors and Ergonomics Society 38th Annual Meeting*, volume 2 of *SYSTEM DEVELOPMENT: Developing Human Factors Guidelines for Intelligent Vehicle/Highway Systems: Third Progress Report [Symposium]*, pages 1057–1061, 1994.

[264] P. Wherrett. *Motoring Skills and Tactics*. Ure Smith, 1977.

[265] J. Whittaker. *Graphical Models in Applied Multivariate Statistics*. Chichester, UK, 1990.

[266] B. Widrow, D. Rumelhart, and M. Lehr. Neural networks: applications in industry, business and science. *Communications of the ACM*, 37(3), 1994.

[267] G. Wilde. Target risk. Technical report, PDE Publications, Toronto, 1994. Also on WWW as http://pavlov.psyc.queensu.ca/target.

[268] C. Williams and G. E. Hinton. Mean field networks that learn to discriminate temporally distorted strings. In *Proceedings, connectionist models summer school*, pages 18–22, San Mateo, CA, 1990. Morgan Kaufmann.

[269] A. Wilson and A. Bobick. Learning visual behavior for gesture analysis. In *IEEE International Symposium on Computer Vision*, 1995.

[270] A. Wilson and A. Bobick. Recognition and interpretation of parametric gesture. In *nternational Conference on Computer Vision, 1998*, 1998.

[271] L. Wittgenstein. *Philosophical investigations*. New York: Macmillan, 1953.

[272] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *In* Photonics East, SPIE, volume 2615, 1995. Bellingham, WA.

[273] C.R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, July 1997.

[274] Y. Yacoob and L.S. Davis. Recognizing human facial expressions from long image sequences using optical-flow. *PAMI*, 18(6):636–642, June 1996.

[275] H. Yamada. Dimensions of visual information for categorizing facial expressions. *Japanese Psychological Research*, 35(4):172–181, 1993.

[276] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden markov models. *Transactions of the Institute of Electronics, Information and Communication Engineers D-II*, vol.J76D-II(no.12):p. 2556–63, 1993.

[277] Q. Yang and H. Koutsopoulos. *Transportation Research*, chapter A microscopic traffic simulator for evaluation of dynamic traffic management systems. 1995.

[278] A.L. Yuille, P.W. Hallinan, and D.S. Cohen. Feature extraction from faces using deformable templates. *IJCV*, 8:99–111, 1992.