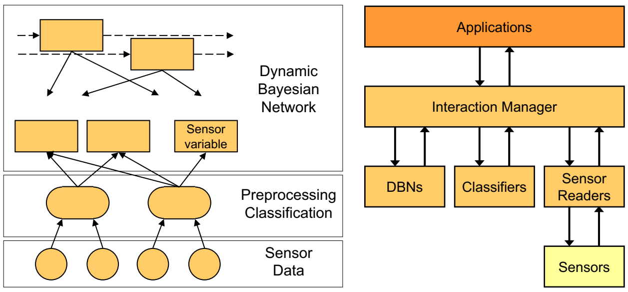
Presentation of a Modular Framework for Interpretation of Sensor Data With Dynamic Bayesian Networks on Mobile Devices

Boris Brandherm, Michael Schmitz
Saarland University, Department of Computer Science
P.O.Box 15 11 50, D-66041 Saarbrücken, Germany
{*brandherm, schmitz*}*@cs.uni-sb.de*

We show in this work an example of how physiological data of a user acquired by bio sensors and interpreted by dynamic Bayesian networks on a personal mobile device are used to adapt notifications to the user state. The described sample application is an instance of our framework that is currently under development in our research group, aiming towards a modular and cross-platform toolkit for sensor integration.

We will give a brief overview of the whole system with its components and conclude with an outlook on future work in this project.

At the conference we want to present the example application based on this framework and the single components as described in this work.



(a) Sensor Interpretation Hierarchy (b) Framework

Figure 1: Schematic overview of sensor data flow

System Architecture and Example Scenario

Figure 1(a) visualises the relationship between sensor data, their preprocessing and classification and the instantiation of the results as evidence in a *dynamic Bayesian network* (DBN) – a computational framework for the representation and the inference of uncertain knowledge that takes into account previous states as well as new evidence. In Figure 1(b) is shown how we plan to integrate the existing components into a framework where applications only query the layer of the Interaction Manager. This Interaction Manager will for instance be responsible of forwarding raw sensor data to its corresponding classifiers or DBNs and synchronizing DBN rollups with processing and reading sensors. It will also register new classifiers and sensors and handle their access. This will further separate development of applications from extending sensor, classifier and DBN libraries — which can still be used independently from all other modules or the framework itself.

All software components are developed in Java to provide cross-platform modules that can easily be ported to various devices as long as they have a Java Virtual Machine running.

To gather data about the state of the user we used in our example scenario a *Varioport*, a mobile device that allows for recording signals from environmental and physiological sensors. We applied in our example scenario an electromyogram sensor (EMG) at the forearm, an electrooculogram sensor (EOG) between the eyebrows and an acceleration sensor (ACC) at one thigh.

These data are readout from the Varioport over a serial interface by the *Javario* software, currently running on a laptop. After processing and structuring into a more convenient format, it will be sent over a wireless connection to a hand held - a Zaurus SL-5500G - where they will be instantiated as evidence in a DBN. The DBN computes an estimation about the user state and sends the results to the *Alarm Manager* application.

The Alarm Manager tries to inform the user by acoustic and visual notifications with increasing intensity until it is assumed that the user has responded to the message, depending on the values of the estimations received from the DBN.

Varioport and Javario API

The Varioport is a small and compact recorder for mobile acquisition of psycho physiological or environmental data with up to nine different sensors. The device itself already preprocesses incoming data (filtering, pre-amplification, digitalisation) whereby the frequency of retrieving and storing data locally is configurable.

For controlling and reading data from the device we developed the *Javario* library that manages the sensors over a serial port: It transmits a definition file to control and configure the recording, starts and stops the measurements by commands and observes the output of current values of the sensors (on line data control).

Currently the bandwidth of the serial line limits the on line transmission of sensor data to other devices such that we can only read out 3 sensors at the same time.

As a first preprocessing step of Javario, activation thresholds are computed for each sensor. The measurements of the accelerometer are furthermore interpreted to determine the horizontal position of the sensor. Steps are recognized as well by applying certain thresholds, such that the walking speed can be computed by the frequency of recognized steps.

Zaurus With Dynamic Bayesian Network

To model the time slice schemas of the DBN, we use our software *JavaDBN* (see Figure 2) that enables DBNs –as

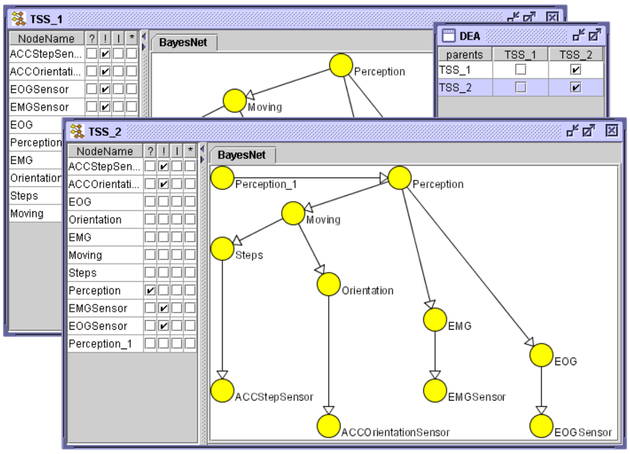


Figure 2: Screenshot of JavaDBN with directed graph (DEA) and time slice schemata (TSS_1 and TSS_2)

the first system to our knowledge– to run on mobile systems. A directed graph (DEA) determines in which orders the time slice schemas can be instantiated, furthermore the user can specify the query and evidence nodes of each time slice schema (see the tabular in each time slice schema window in Figure 2). Then JavaDBN generates java source code which represents this DBN and allows to inference in this DBN as well as to apply a non-approximative rollup method that cut off older time slices but incorporate their impact without loss of information on the remaining time slices of the DBN (see [Darwiche, 2003; Brandherm and Jameson, 2004] for the theoretical background). This java source code has the following additional features: All for the computation necessary variables are created on initialisation of the DBN. No variables are created or disposed at run-time to avoid the activation of garbage-collection. The inference in the DBN is optimized regarding the selected query and evidence nodes. Special functions to set evidence are provided. Since variables are freely accessible from outside the class, evidence of a node can also be set directly.

The DBN in our example scenario is only for demonstration purposes and is not based on an experiment. It consists of a time slice schema for instantiation of the initial time slice (TSS_1) and a time slice schema for instantiation of succeeding time slices (TSS_2). TSS_1 and TSS_2 are visualised in Figure 2. In TSS_2 are the nodes Perception_1, Perception, Moving, Steps, Orientation, EMG, EOG, and the sensor nodes ACCStepSensor, ACCOrientationSensor, EMGSensor, and EOGSensor that correspond to the sensor nodes of the DBN in Figure 1(a). These nodes will be instantiated with the results of the preprocessing (step, orientation, and activation threshold). The conditional probabilities of the nodes model the reliability of the sensors according to the real world situation. TSS_1 differs from TSS_2 that it does not contain a node Perception_1, which models the influence of the precedent time slice on the current one.

A new time slice is instantiated whenever data arrives, the estimation is calculated, followed by the rollup.

AlarmManager

The Alarm Manager is a simple example application to demonstrate the framework. It runs as a service that triggers different kinds of notifications/alarms in the instrumented environment, considering informations about the state of the user that are provided by estimation values of the DBN on the Zaurus. First step of the procedure is the start of the Alarm Manager, when an appointment or deadline is close - in our example the boarding time of a plane. It will now show a message with the boarding announcement on a handheld of the user and play a short audio notification on it to attract his or her attention. Now the arousal values computed by the DBN will be observed as long as the notification is active to see if the user reacted. If the arousal value remains below a certain threshold, it must be assumed that the user did not notice the message and the next notification step will be initiated: The Alarm Manager plays a general boarding announcement on the public speaker system in the user's vicinity - which is the spatial audio system in our lab [Schmitz, 2004]. Again the arousal level will be watched and if necessary the third and last notification step will be launched by displaying a virtual avatar that will address the user and request him to proceed to the gateway immediately.

Since this scenario is tailored to our instrumented environment, we exchanged the above presentations by appropriate audio/video messages that will run on a common laptop for mobile demonstrations.

Future Work

We are currently working on integrating the Javario and the JavaDBN software on a single handheld like the Zaurus in order to make the whole system more compact and lightweight. One problem so far was the communication over the serial interface between the Zaurus and the Varioport, but since the next generation of sensor equipment is based on bluetooth, we are confident that the hardware setup can easily be improved.

Experiments are in progress that deal with the recognition of time pressure and movement on the basis of sensor data, from which we hope to obtain more insights in dependencies between different sensor types. Furthermore classifiers are being investigated that interpret raw sensor data and provide more abstract representations of more complex sensor structures to provide a layer of higher level information to simplify development of affective applications.

Acknowledgments

This research is being supported by the German Science Foundation (Deutsche Forschungsgemeinschaft (DFG)) in its Collaborative Research Center on Resource-Adaptive Cognitive Processes, SFB 378, Project EM 4, Real, and Project EM 5, READY.

References

[Brandherm and Jameson, 2004] Boris Brandherm and Anthony Jameson. An extension of the differential approach for Bayesian network inference to dynamic Bayesian networks. *International Journal of Intelligent Systems*, 19(8):727–748, 2004.

[Darwiche, 2003] Adnan Darwiche. A differential approach to inference in Bayesian networks. *Journal of the Association for Computing Machinery*, 50(3):280–305, 2003.

[Schmitz, 2004] Michael Schmitz. Safir: Spatial audio framework for instrumented rooms. In *Proceedings of the Workshop on Intelligent and Transparent Interfaces, held at Advanced Visual Interfaces, Gallipoli*, 2004.