

# Teaching Contexts to Applications

**Kristof Van Laerhoven**  
Starlab Research NV/SA  
St-Michielslaan 47  
1040 Brussels  
[kristof@starlab.net](mailto:kristof@starlab.net)

## Abstract

Enhancing applications by adding one or more sensors is not new. Incorporating machine-learning techniques to fuse the data from the sensors into a high-level context description is less obvious. This paper describes an architecture that combines a hierarchy of self-organizing networks and a Markov chain to enable on-line context recognition. Depending on both user and application, the user can teach a context description to the system whenever he or she likes to, as long as the behavior of the sensors is different enough. Finally, consequences and complications of this new approach are discussed.

## 1 INTRODUCTION

### 1.1 CONTEXTS AND APPLICATIONS

Some applications enhance their user interface by adding a sensor and using the sensors' value in some simple rule. A typical example is connecting a light sensor to a screen-based device and adjusting the contrast and brightness of the screen according to the value of the light sensor.

Other applications can change their behavior only when the user explicitly tells them to. It is also possible to use user-defined profiles that describe the devices' behavior. Profiles in mobile phones, for example, can be set to make the phone ring very loud outside or on the train, but only vibrate in a meeting. This approach leads to a lot of user-involvement, though: the user first needs to program these profiles, and then these profiles must be set in every context ('in a meeting', 'in the train', ...).

The combination of all of the approaches mentioned earlier leads to an *automated profiles selection*: context recognition based on simple sensors sets the behavior of the device (see [1] and [5]). Knowing the context usually leads to being able to improve the application and

particularly enhancing the interaction with the user. This approach is far from simple, however: how can a device, equipped with sensors, recognize a context?

### 1.2 CONTEXT

The notion of context is very broad and incorporates lots of information, not just about the current location, but also about the current activity, or even the inner state of the person describing it. As a consequence, multiple people can describe their contexts in different ways, even if they are in the same location doing the same thing. Someone familiar with a building might know a room as 'classroom 402B', while a visitor would probably describe it as just 'a classroom'.

In addition, the application defines the description of the context as well. Some applications require more location-based contexts, while others need contexts that give more information about the user.

### 1.3 CONTEXT DESCRIPTION

The simplest method for giving a context description would be to sum up all the values from the sensors to a formatted description, like for instance "movement: (87%, 29%), light: 78%, humidity: 69%, temperature: 50%, ...". A simple, rule-based architecture could be used to enhance this description into "moving slowly in a cold, humid, well-lit room".

The architecture described here works the opposite way: the system merges the output from the sensors and maps them to a description given by the user. The description could then be something like "walking in the basement". This way, we deal with the fact that context description depends on both the individual describing it and the application using context perception.

## 2 ONLINE ADAPTIVE CONTEXT AWARENESS

This section will describe the architecture of the algorithm that does on-line training and recognition.

Instead of just using the raw sensor values as input for the next layer, small pre-processing routines were chosen to enhance the future clustering. For example, instead of just looking at the brightness of the light, it is also possible to look at its frequency, which results in easier distinguishing of several types of artificial light. Taking the standard deviation of the accelerometer values can also give more qualitative information. Other sensors like microphones and infrared sensors have similar mini-transformations from the raw sensor data to one, but usually multiple, values, which are usually called cues or features.

Another advantage of the cues is that that they are sent less frequently to the next layer. The light sensor, for instance, is read a few hundred times per second. The cues from this sensor (light level and frequency) are sent every second. Cues are very significant for a fast, but accurate context recognition system. However, using cues results in a large input dimension, which makes the mapping-algorithm very slow in learning. This difficulty arises when many irrelevant inputs are present and is usually referred to as *the curse of dimensionality* (see [4]).

## 2.1 SELF-ORGANIZATION

When a rat has learned its location in a labyrinth, certain braincells on the hippocampal cortex respond only when it is in a particular location. Self-organization of neuronal functions seems to exist on very abstract levels (like geographic environments) in the brain. The Kohonen Self-Organizing Map (SOM) has a similar principle: neurons (artificial, this time) are recruited topologically for tasks depending on the (sensory) input. The SOM is also known to handle noisy data relatively well, which makes it a sensible choice for clustering the inputs.

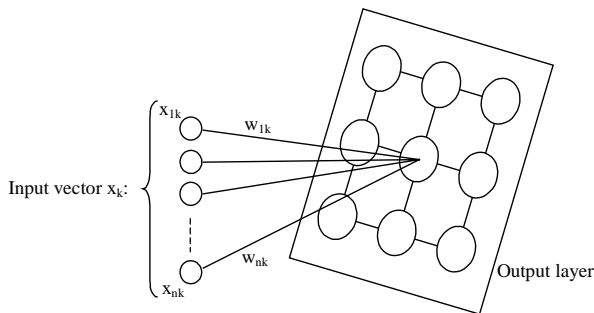


Figure 1: The Kohonen Self-Organizing Map

The Kohonen SOM (1982) [3] is based on earlier work of Willshaw and von der Malsburg [6] (1976), and starts with a competitive network, where basic units ‘compete’ for a particular kind of input. For every input, one unit is selected to be the winner and can adapt itself a bit more towards this input. More concrete, the winner can adjust

an internal weight vector (or codebook vector, or prototype vector) towards the input vector (See Figure 1). Therefore, different sensor inputs result in different neurons being activated on the SOM. It is possible to monitor the activation of the neurons and plot the resulting matrix as a landscape, where different hills ideally represent different contexts (see Figure 2).

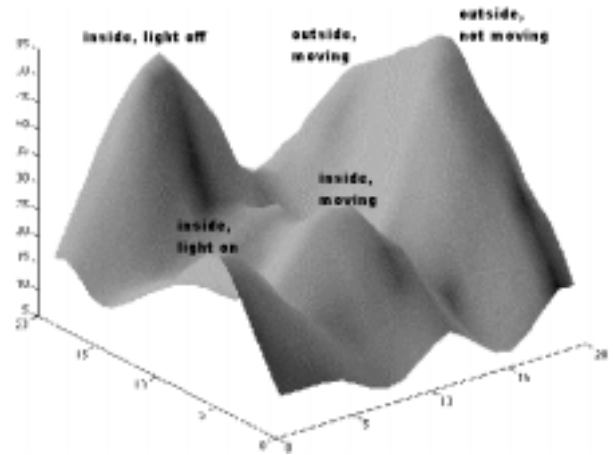


Figure 2: Example of an activity-plot of a SOM.

The traditional algorithm starts out highly adaptive (a large learning rate and huge neighborhood radius) and gradually becomes fixed. After this stage, it is not capable of learning anymore, which poses an obstacle if the system needs to remain adaptive. This is a problem also known as *the stability-plasticity dilemma*. It therefore is necessary to add some mechanism that controls the flexibility of the SOM locally, i.e. on the level of the neurons.

The fixed size of the map is a problem as well, since it leads to a limit of contexts that can be recognized. To deal with this problem, it is possible to use multiple (smaller) SOMs with each a subset of the original input vector instead of one big SOM. Since it is very likely that some contexts will persist much longer than others, the possibility exists that popular (long-lasting) contexts ‘overwrite’ others, i.e. undo the weight adaptations from the other contexts’ signals. It is therefore advantageous to pre-order the weights in a controlled environment before the real use instead of just giving them small random values as is customarily done with the SOM.

## 2.4 SUPERVISION AND USER BEHAVIOR

The next layer is primarily intended to supervise transitions from one context to another. It uses a probabilistic finite state machine architecture where each context is represented by a state, and transitions

are represented by edges between states. The model keeps a probability measure for each transition, so every time a transition occurs, the supervision model can check if this really is likely. If a transition is not really probable, the next state is not entered yet, but a buffer mechanism is initiated so that it does become more likely after several tries in a row. Each transition to a state is thus dependent on the previous state, which makes this model a first-order Markov model. Every state also keeps track of how much time was spend in a particular context, which controls the flexibility of the SOMs: the newer a context, the more flexible and adaptive the map should be.

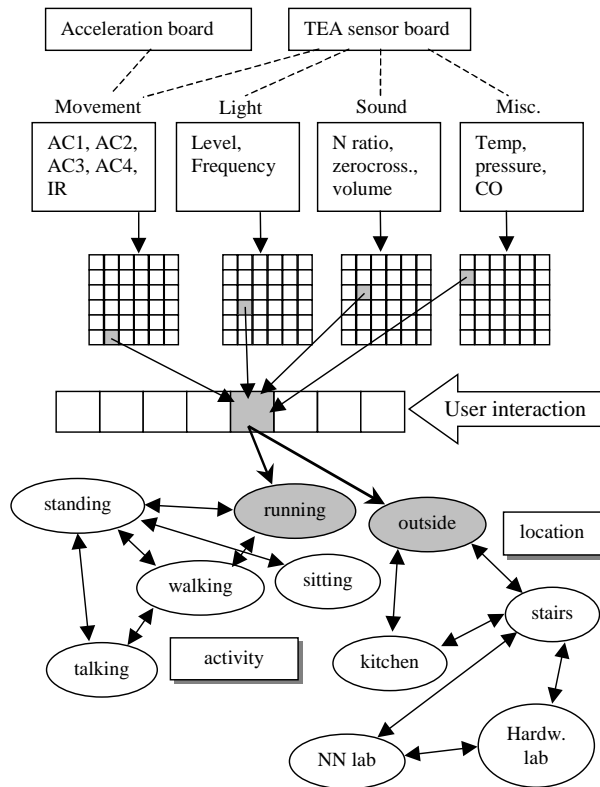


Figure 3: Overall architecture. User interaction is only necessary after the clustering.

The result is that after some time this model generates a graph depicting the behavior of a user with relation to the contexts visited. When the user tends to go from A to B rather than to C, then this will be reflected in the graph's connection strengths. Figure 3 depicts the typical layout of the final architecture.

### 3 CONCLUSIONS AND FUTURE WORK

But adaptive context perception not just solves problems; it introduces new ones too. For an application (using context-perception) to be effectively user-friendly, it is

necessary that the system gets feedback from the user whenever the *user* would like to. These constraints are both hard and challenging from a machine-learning point of view. The combination of unsupervised neural networks and a context model gives promising results, without creating a bulky overhead on the user-computer interaction. However, the performance must be boosted by improving both sensors and cues in both quality and quantity. The experiments up until now used 10 sensors, but we expect to increase this number significantly. Other important issues are placement of sensors, the grouping of sensors for the clustering, and redundancy of sensors to make the system truly robust.

### Acknowledgements

The framework of this paper was given by the TEA project [2], which is sponsored by the European Commissions 'Fourth Framework'. Thanks go out to the people from all project-partners: Starlab Research (Belgium), Nokia Mobile Phones (Finland), TecO (Germany) and Omega Generation (Italy).

### References

- [1] Chen, D., Schmidt, A. & Gellersen, H.-W. (1999) *An Architecture for Multi-Sensor Fusion in Mobile Environments*, In Proceedings International Conference on Information Fusion, Sunnyvale, CA, USA.
- [2] Esprit Project 26900. Technology for Enabling Awareness (TEA). <http://tea.starlab.net> or <http://www.omega.it/tea>, 1999.
- [3] Kohonen, T. (1997) *Self-Organizing Maps*, Springer Springer-Verlag Heidelberg,
- [4] Mitchell, T.M.(1997) *Machine Learning*, McGraw-Hill.
- [5] Schmidt, A., Aidoo, K.A., Takaluoma, A., Tuomela, U., Van Laerhoven, K. & Van de Velde, W. (1999) *Advanced Interaction in Context*, in H. Gellersen (Ed.) *Handheld and Ubiquitous Computing*, Lecture Notes in Computer Science No. 1707, Springer-Verlag Heidelberg, p. 89-101.
- [6] von der Malsburg, C. (1991) *Self-Organization of Orientation Sensitive Cells in the Striate Cortex*, in G.A. Carpenter and S. Grossberg (Eds.) *Pattern Recognition by Self-Organizing Neural Networks*, MIT Press, p. 179-205.

