



## Behavior-Monitoring Machines

**A new breed of computer software keeps tabs on how people and objects act. To really work, though, these programs have to learn and adapt.**

By Erik Sherman  
November 12, 2003

He sees you when you're sleeping, he knows when you're awake. Ever wonder how Santa has time to keep tabs on the world while building toys? Maybe he's using behaviorometric software—applications designed to understand the behavior of people and things and to react appropriately in order to provide security, facilitate health care, and enhance productivity.

In a limited sense, computers have always had to understand highly select behavior. A word processor, for instance, must know when people want to copy text blocks and when they want to delete them. But these user actions are highly constrained, with people only allowed to perform expected actions. The genius of behaviorometric software is that it tackles more unpredictable actions, whether performed by people, animals, machines, or a combination of all three. What developers must do is measure and describe behavior in manageable ways while enabling computers to learn more about the targeted behavior over time through observation.

Researchers have known for many years that humans can extract essential characteristics of behavior quickly and easily. "In the 1960s, the psychologist Gunnar Johansson performed a series of famous experiments in which he attached lights to people's limbs and recorded videos of them performing different activities, such as walking, running, and dancing," says M. Alex O. Vasilescu, a researcher at New York University's Media Research Lab in New York and a doctoral candidate at the University of Toronto. People were asked to watch videos where only the lights were visible and then classify the activity. Usually, Vasilescu says, observers were able to "perform this task with ease, and they could sometimes determine gender and even recognize specific individuals." The difficulty scientists have is in figuring out how to view the task in a simpler and more essential form.

The process of paring away the unnecessary can take decades. Take, for example, a project at the Technion-Israel Institute of Technology in Israel, in which a pair of students wrote an application that can identify people with near 100 percent accuracy by their typing styles. "People have studied problems of trying to match keystroke sequence to a typist for almost 20 years," says Ran El-Yaniv, a Technion professor of computer science and expert in machine learning who was one of the supervising faculty. When the project started, El-Yaniv, says, software could make highly accurate identifications this way only if the person was required to type a predetermined sequence of characters. Move outside that keyboarding script, and the programs couldn't keep up.

The Technion project found a way to ignore the actual content of the typing by concentrating on the number of milliseconds it took people to type a character. The team wrote software that monitors how quickly a typist pressed and released a key. Individual times are grouped to take slight variations into account. Researchers have also found that pressing a given key would affect the typing of the next letter. Because there were a finite number of key combinations, researchers could use existing algorithms to create a probabilistic model for an individual user. Someone whose typing did not fit the model was likely not the person. The tricky part was determining whether there was a fit with the model. It is called a single-class identification problem—like designing a system to know what a dog looks like, then showing it a cat and asking if that animal belongs to the class.

Researchers in behavior-detecting software find that they must walk a tight rope: seeking to represent the essentials of behavior a problem in the simplest possible way that still carries enough statistical information to allow the computer to place an action into a category and make an appropriate decision. And the single biggest obstacle is having too much information. Data filtering—passing on only the cats and dogs and tossing out rocks and trees—is vital. Without it, the fastest computer would grind to a halt under the weight of detail. "We look at it as a data reduction problem," says Glenn McGonnigle, CEO of VistaScape Security Systems, an Atlanta-based company makes software that examines video surveillance tapes for such high-risk facilities as shipping ports, airports, and petrochemical plants. Using generic image templates of objects—people, boats, birds, and so forth—VistaScape's software matches activity against rules provided by client. One rule, for instance, might forbid people to climb a fence and enter a sensitive area.

To alert security personnel to potential problem situations, the software must sift through a bewilderingly large number of object behaviors: water undulating in waves, boats moving against the horizon, birds darting and soaring through the air, people strolling on a beach. From the cradle, people learn to instantly make these distinctions, using experience to increase their powers of discernment; machines lack that advantage.

But whether something is important or superfluous depends completely on what the scientists are trying to detect. For McGonnigle's purposes, vital details might include object size and speed, while color might be irrelevant. Direction of motion would matter in some cases—someone walking one way through an airline security check would be normal, while travel in the opposite direction could indicate a potential problem. Delivery trucks arriving on a Tuesday or Thursday might be expected, though a Wednesday appearance might warrant an investigation.

Since what is important in one context is of no interest in another, each behaviometric system enlists subject experts to help define what behavior will be important. New York City-based Living Independently recently began shipping a system that keeps tabs on elderly people living at home, alerting family or health professionals if behavior suddenly changes. Living Independently had to know what actions might shine some light on the state of someone's health. "Working with gerontologists, we identified the behaviors that would be most valuable to monitor," says George Boyajian, executive vice president for strategy, research, and development.

As it turned out, the list of considerations could be pared to a bare minimum. Infrared sensors note when someone gets up, nears a food preparation area, opens a medicine cabinet, and enters the bathroom. Data travels wirelessly to a base station, which relays the information over an ordinary phone line to the company. Proprietary algorithms compare the actions to what the system considers "normal" for the person based on previously collected data. Sudden changes in behavioral patterns—more frequent night time trips to the bathroom, for example, or a sudden cessation of opening the medicine cabinet to obtain pills—can be early signs of trouble, or might indicate an immediate problem. The system then sends alerts to family members or appropriate healthcare professionals.

Normal is, of course, relative, and can change over time. That is why behaviometric software must also incorporate machine learning, so that decisions are not made based on some behavioral standard that is no longer applicable. Living Independently creates a baseline of behavior that changes over time. Each person is observed, with software abstracting statistically "typical" behavior on a roughly ten day rolling time frame, shifting as someone's habits change. Thus, the software avoids raising alarms unnecessarily.

Behaviometric software is not perfect. The Living Independently application, for example, can only tell if people approach food preparation areas—not if they eat. A ship nearing an offshore oil rig might be only a lost pleasure craft, and not terrorists planning an assault. Still, an approximation is sometimes all that's needed to answer a question. So the name time you are wondering how Aunt Sadie is doing—or who is at her keyboard—ask her computer. It might just tell you.