

## **Cognitive Systems 2**

### **Winter 2008/09**

#### **Exercise 5**

Pardis Alizadeh  
Yuting Chen  
Adeel Naveed  
Stefan Kreitmayer

# **CoSy makes better people**

## **Introduction**

Looking back on four months of working our way into cognitive modeling, it appears that at the end of the course we have arrived at the beginning of a new field of research. New for each of us individually - but also new in the sense of being a domain where pioneers are at work. It is fields like that where scientific accomplishment is a thing of the present, categories are yet to be formed, consensus about common notions wants to be found, and Wikipedia is of little help. Different paradigms and theoretical models have emerged independently, for different reasons and with different goals, yet nobody knows for sure how different they really are. Existing software implementations like ACT-R or SOAR have accompanied these theoretical models for decades and grown along with them. Getting our eyes on and hands dirty with these living scientific artifacts, we have tasted the smell of infinite imperfection that these practical attempts of understanding the human mind carry with them, even at their current state of highly developed complexity. We have become witnesses of what is currently going on the sector, even if what we have seen is merely some of the basic principles.

Entering this new territory was an experience of adventure, a great deal of fascination coming from an immediate bodily sensation of these models showing their origins in real life. Trying to imagine the huge amount of effort it must have taken a brain to originally come up with e.g. the visual attention module of ACT-R makes it even more stunning to become aware of this abstraction actually making sense every second of our waking life. Out of ten thousand steps we walk in a week, most of them completely unaware, there are now always around five or six that suddenly delight us with the idea of "How glad I am that a highly trained network of neurons is taking care of my everyday sensor and motion businesses, and that it is doing all that in parallel, without even bothering my central system with annoying explicit representations". Introspection rocks. It is fun. But also our social experiences can benefit from embodying ideas derived from this course in our consciousness. We are not the only agents in our environment, and whenever someone is getting on - what we call our nerves - we now have the ability to think, "never mind, he is just acting this way because the conditions for a few of his rules have been met. After all, he's been trained this way,

and who knows what exact input pattern ended up triggering this behaviour. If I declaratively think of him as a stinkypoo, then that is just a slot value which I can easily change. It takes no more than a buffer request."

Yes, we believe that CoSy makes better people. The deeper one gets involved in cognitive modeling, the higher he or she ascends on the ladder of moral enlightenment.

As obviously as the early endeavors in Artificial Intelligence have failed their goal of making computers smarter than humans, at least we can say today that Cognitive Science has been successful in making humans smarter than humans. Compared to the brutal, ignorant troublemakers we were in October, especially XYZ. Look at her now. In just a few months we have learned more than we could imagine possible, and we are happy looking forward to further deepening our insights throughout the coming months and years. Quite aware that it would take more than a lifetime to fully catch up and become an expert in all the various disciplines that this enticing field has produced so far (and is still producing), each of us will take our current knowledge as a basis to further specialize in one of the areas and apply aspects of these technologies in their present and future field of work. Given our variety of backgrounds and perspectives, we expect interesting, novel cross-links to and from other fields of computer science and digital media.

## Implementing David Hasselhoff

Recreating real people in the digital domain, be it as full featured androids or just simple embodied conversational agents, has always been tempting but difficult. Challenges lie not only in producing a convincing physical appearance, but mainly in the realistic reproduction of typical behaviour, speech peculiarities and swing of the hip.

So we decided to try our luck with an easy example. Our goal was to create a completely recognizable representation of David Hasselhoff without the means of digital photography - that would have been too simple - or any kind of visual or acoustic emulation. For financial and logistic reasons we decided to refrain from any mechanical solutions and resort to a three dimensional simulation. Our target was represented visually as a neutral wireframe model without any visual resemblance, and for the voice we chose a standard speech synthesizer in order to avoid any audible hints to the identity. David Hasselhoff should be recognized as himself only through his autonomous behaviour and interaction with his environment, e.g. running down a sandy beach in slow motion or jumping into a talking car.

Our first intuitive approach was to implement David Hasselhoff in ACT-R, which failed already in the first phase of our experiment, the beach run. The realistic orchestration of muscle contractions was not the problem. A mere thousand productions were enough to make sure our wireframe model would not lose balance, but instead create a recognizable impression of David Hasselhoff running in a straight line. The real time accuracy was breathtaking, and going from natural to slow motion merely required changing two variables in our ACT-R model. At first, the belly would wobble a little more than in the original, that was hard to debug, because we could not find out which production was the cause of the problem. In the end, we contributed a specially written belly contraction module to the source code of ACT-R and successfully worked around the issue. What gave us a real headache was the fact that we had to implement all the sand particles in LISP, which turned out to be more computationally threatening than expected. Reducing the total number of sand grains on the beach to five gave us good real time results on a 1.2 GHz G4 while hardly

influencing the interaction with the toe tips. Pardis had the brilliant idea to simply have the sand stick to the feet, so there would not be any unnecessary computation of sand just lying around without moving. So, without the water it all worked perfectly. But as soon as the waves came in, David Hasselhoff would have difficulty following a straight line along the beach because his visual attention got fixed on the surf going back and forth. There was no way we could stop him from getting dizzy and eventually losing balance. Without the water it would not have been a beach, and without a proper beach run it would not have been David Hasselhoff, so we gave up on the ACT-R approach in the end.

SOAR seemed to be a perfect solution at first, due to its more simplistic design, but seemed to have problems with the real time accuracy - and anyway it kept crashing, so we dumped it pretty soon.

Maybe, we concluded, explicit representations are not the way to go. So we set up a neural network with 10,000 virtual brain cells and a simple back-propagation learning scheme. We did this in JavaNNS, and the beach experiment ran beautifully. All we needed to do is show our agent the beginning of Baywatch a few million times, and after a week David Hasselhoff had perfectly adopted all the body motions by imitating the picture from TV. As a side effect, he had also grown large breasts, along with the habit of attacking sharks with his teeth. We are not sure at which part of the training phase this must have happened, but with absolute certainty we can not locate the bug inside the trained network. That is because Neural Networks have the characteristic that they do not represent their acquired knowledge in any human-readable form, but rather in the form of a large number of numbers distributed all over the entire network. So we decided to live with the result we had got, particularly as it had no negative influences on the show's public acceptance. Neither had the increasing predictability of the plot due to our David Hasselhoff's strictly deterministic behaviour. On the contrary, the television audience received it with great pleasure that our newly created David Hasselhoff could run down the beach, smile and wave at the same time. We called this innovation parallelism.

Also, we had good success implementing D.H. in the form of a mobot, as described by Rodney Brooks. We particularly liked this approach for its 80s flair and also the superiority of our agent when it comes to object avoidance. Since a crowded beach can be quite a messy environment nowadays, this version excelled with its ability to gracefully jump over towels and sandcastles lying in its way. The previous, neural network based agent had just stomped on these things without notice, owing to its high error tolerance. The mobot's reflexes, however, were incredibly quick, it seemed like the eyes had been directly connected to the legs, which was indeed what we had done in this one. Whatever he saw lying in its way, the agent would just jump over it without asking what it is. A wonderful thing to watch. It even jumped over shadows on the ground. The only downside of the mobot version was that it had to stop running in order to breathe or wave at somebody.

Our final model was modeled on the work of Randall Beer who had previously been successful in teaching six-legged agents to walk. Now we wanted to teach our two-legged agent to run, a difference which we expected to be rather irrelevant, and so it turned out. Perfect running skills were the result of a good week of intense programming. Actually writing the code was fun, even though we had been spoiled by the neural network experience of letting the machine to all the dirty work for us in the same amount of time while we were out for swim. Teaching computers knowledge is such a waste of time, we concluded, when you can just tell the network to learn what it needs to learn by itself. All computer stuff should be like that. But on the other hand, we learned a few very essential ideas from Beer's approach. Namely, he claimed that "internal representation is not essential to genuine cognition." It is, however, to driving a car. In our second experiment David Hasselhoff performed a beautiful jump into the driver's seat, merely by comparing his current standing outside to his later sitting inside and figuring out the most aesthetically appealing interpolation between these two states. Dynamic System is what he called this technique. So far so

good, we thought, let's drive to the bakery. Since he knew where the bakery was, he stepped on the gas and took the shortest route. On his way he nearly ran over two grandmas and a cat - luckily the car had an autopilot and could hit the brake just in time. The autopilot was done in ACT-R by the way. We thought that the Dynamic Systems idea by Randall Beer was suited very well for doing things alone in a static environment. It seemed as though we would have to optimize this technique to a further level in order to make it suitable for busy street traffic. Or maybe it was indeed capable of reacting smartly to changes in its environment - and we had just made a mistake somewhere. Anyway, the police told us that Beer and driving don't go together and took away David Hasselhoff's driving license.

Luckily, we still had a few other David Hasselhoffs to try out. The mobot, for example, very successfully avoided crashing into anything, just because of his supernatural reflexes. However, driving as fast as he could, he spread fear and terror in the whole city, because obeying traffic rules and driving on the right side were not quite among his talents. Also, it took him eight hours to get to the bakery which was actually just around the corner. His sense of orientation could have been improved. But before doing that, we wanted to give the other David Hasselhoffs a shot.

The neurally trainable, connectionist David Hasselhoff did a fairly good job in learning how to drive after the autopilot had punished him often enough. He did learn from his own mistakes. Only that took a few more driving lessons than the original David Hasselhoff would have needed.

The best driving performance of all came from the David Hasselhoff that we had initially programmed in ACT-R. He entered the car, set the goal of the autopilot to "bakery", pressed the button and stepped out of the car as soon as the goal was reached.

## Conclusion

We have found out that Neural Networks can do pretty much everything if you give them enough time to learn what they are meant to perform.

We further understood that different cognitive tasks can be more or less 'representation-hungry', sometimes symbolic representations make sense, when sensor data alone can not guide behaviour and the system has to deal with "distal, non-existent or highly abstract properties" (Clark et al.).

As one example going beyond the tutorials and exercises, we would like to model a band playing music together, maybe in ACT-R or in SOAR. This will involve designing not only the cognitive agents (players) but also an appropriate environment in LISP (in the case of ACT-R) that interfaces with another process for sound generation and analysis.

In order to get even more deeply involved and feel the vibration of cognitive modeling not only on our fingers' skin but also in the bones, we would like to implement a simple, specialized cognitive model in Java or C++, in combination with a specialized environment. By further complicating this implementation we hope to soon come to a point where all the technical reasons for and against having a general model, separated from the environment, become obvious.

We would like to know how the winning teams in robot soccer are implemented in terms of cognition, how genetic algorithms tie in with all this, what technologies are applied and combined in what way in the most advanced android projects currently running.

How can we effectively embed features of ACT-R, SOAR, JavaNNS or similar systems into our own software projects which involve procedural / functional / visual programming?

With these questions we would like to open the discussion.

## Literature

István S. N. Berkeley Ph.D.: "What Is Connectionism?"

<http://www.ucs.louisiana.edu/~isb9112/dept/phil341/wisconn.html>

Andy Clark, Josefa Toribio: "DOING WITHOUT REPRESENTING"

[http://wexler.free.fr/library/files/clark%20\(1994\)%20doing%20without%20representing.pdf](http://wexler.free.fr/library/files/clark%20(1994)%20doing%20without%20representing.pdf)

Denise Peters: "JavaNNS - a practical introduction to neural networks"

<http://www.cosy.informatik.uni-bremen.de/teaching/cognitive-systems2/JavaNNS.pdf>