

## Ola: What Goes Up, Must Fall Down

Henrik Hautop Lund Jens Aage Arendt Jakob Fredslund Luigi Pagliarini

LEGO Lab  
InterMedia, Department of Computer Science  
University of Aarhus, Aabogade 34, 8200 Aarhus N., Denmark  
<http://www.daimi.aau.dk/~hhl>  
hhl@daimi.aau.dk

### Abstract

We have made a robot soccer model using LEGO Mindstorms robots, which was shown at RoboCup98 during the World Cup in soccer in France 1998. For the robot soccer model, we constructed a stadium out of LEGO pieces, including stadium light, rolling commercials, moving cameras projecting images to big screens, scoreboard and approximately 1500 small LEGO spectators who made the “Mexican wave” as known from soccer stadiums. These devices were controlled using the LEGO Dacta Control Lab system and the LEGO CodePilot system that allow programming motor reactions which can be based on sensor inputs. The wave of the LEGO spectators was made using the principle of *emergent behaviour*. There was no central control of the wave, but it emerges from the interaction between small units of spectators with a local feedback control.

### Introduction

World Cup and regional games (such as European Championship) in soccer always attract very big crowds of passionate fans. The passionate fans on the grandstands are part of what makes soccer an animated game. For instance, the fans are known to make big choreographic scenery such as the one on figure 1. The impressive image arises when each spectator holds up a coloured piece of paper that he/she has been handed out from an organising group of fans. We can interpret this organising group of fans as a central control that decides what colour should be handed out to the individual spectator. Without this central control, the choreographic scenery would fail. The “Mexican wave” that is made when spectators stand up and sit down does not have such a central control. The wave is initialised when a couple of spectators anywhere on the stadium decide to make the stand up + sit down movement, and some nearby spectators go with the others. There is no central control to tell the individual spectator to do a specific thing at a given time, rather it is an emergent behaviour.



Figure 1. Impressive choreographic scenery made by Lazio's tifosi in Curva Nord of the Olympic Stadium in Rome. The scenery is constructed when each spectator holds up a coloured piece of paper.

*Emergent behaviour* is an interesting phenomenon that can be observed in natural systems. We define emergent behaviour as being the behaviour of a system that is the product of interaction between smaller sub-systems. The emergent behaviour is of higher complexity than the sum of the behaviours of the smaller sub-systems. The reason that behaviour of higher complexity than the sum can emerge is the interaction between the sub-systems.

Emergent behaviour is known from flocks of birds, schools of fish and herds of land animals. When observing a flock of birds, we will notice that there is no apparent leader in the flock and there appears to be no central control of motion. The motion of the flock might seem complex and at times random, but on the other hand, it also appears synchronous. The motion of a flock of birds is an example of emergent behaviour and can be modelled as such. Reynolds [Reynolds 1987] has made an impressive study of the general motion of flocks, herds, and schools in a distributed behavioural model with the goal of using this to model flocking in computer graphics. Recently, similar models have been used in the Disney movie *Lion King* for a wild-beast stampede and to produce photo-realistic imagery of bat swarms in the feature motion pictures *Batman Returns* and *Cliffhanger*. Reynolds calls his simulated bird-like organisms *boids* (bird-oids). The boids are controlled by three primary rules:

1. Collision Avoidance: avoid collision with nearby boids
2. Velocity Matching: attempt to match velocity with nearby boids
3. Flock Centering: attempt to stay close to nearby boids

The three rules are local in the sense that a boid only has knowledge about nearby boids and there is no global knowledge like size or centre of the flock. For instance, Flock Centering is achieved by having boids to perceive the centroid of nearby boids only. This actually gives the advantage of allowing for bifurcation: the flock can split around an obstacle in the moving direction, since the boids only tend to stay close to nearby flock-mates.

In general, the phenomenon of emergent behaviour is fundamental in a number of artificial life systems. Artificial life tries to synthesise life with a bottom-up approach by using small building blocks that emerge to a complex system by their interaction. For instance, emergence is used as the basic principle in cellular automata, and in a sense, emergence is also one of the basic principles behind the success of artificial neural networks. Artificial neural networks are built from units (neurons) and connections between units. Each unit has a simple processing capability and the connections have propagating abilities. But the interaction between many units with simple processing capability results in a more complex behaviour than just the sum of the processing capabilities. In fact, this was used to refute the criticism of neural networks put forward by Minsky and Papert [Minsky and Papert, 1968] when Rumelhart, Hinton, and Williams showed that neural networks can indeed solve the XOR problem [Rumelhart et al., 1986].

## **Emergent Behaviour in Reality**

When using emergent behaviour in real world models, there are a number of pitfalls that we have to be aware of. When we look at Reynolds' boid model, we notice that only local knowledge of neighbours is used, so the model might appear appropriate for control tasks for autonomous agents in the real world. However, it is not clear how to obtain even the local knowledge that is available for the simulated boids. For instance, we have to solve the question of how to measure *nearby* (distance and direction). This demands an advanced sensor that can measure distance and direction, and at the same time identify an object as being a neighbour (and, for instance, not an obstacle). The task is worsened further by the demand for doing this in real time with moving objects.

There are other significant differences between a simulation model and a real world implementation that we have to take into account. For instance, the actuators will produce friction and there will be a whole range of noise issues that makes it very difficult to transfer an idealised model from simulation to the real world. In some cases, it will be possible to transfer models from simulation to reality [Miglino et al., 1995; Lund and

Miglino, 1996; Jakobi, 1998]. This is done by very careful building of a simulator that models the important characteristics of the real device and the way that real world noise interferes with this device.

In our emergent behaviour model, we work directly in the real world, so we avoid the problems of difficulties in transfer from an idealised model to the real world. Our model therefore has to work with the noise, friction, etc. that exists in the real world.

## RoboCup and LEGO Robot Soccer

In the summer of 1998, we had to go to Paris during the World Cup in soccer to demonstrate LEGO Mindstorms robots playing soccer at RoboCup'98. RoboCup is an international initiative to promote artificial intelligence robotics and the task of robot soccer as a landmark project [Kitano et al., 1997]. As a landmark project, RoboCup differs from earlier artificial intelligence landmark problems, such as constructing an artificial chess player. One of the main differences is that robot soccer players have to play in the real world, where the chess play can be viewed as an idealised world, in which there is no need to address the problems of perception and noise in the real world. Essentially, the differences are similar to the differences between a simulated model of emergence and a real world model. In general, the differences can be summarised as shown in Table 1 (reprinted with permission from H. Kitano).

Table 1. Differences between the classical artificial intelligence landmark project of constructing an artificial chess player and the landmark project of constructing a team of robot soccer players [Kitano et al., 1997].

	Chess	Robot soccer
Environment	Static	Dynamic
State change	Turn taking	Real time
Information accessibility	Complete	Incomplete
Sensor readings	Symbolic	Non-symbolic
Control	Central	Distributed

We constructed team of LEGO Mindstorms robot soccer players to play a demonstration tournament during RoboCup'98. The LEGO Mindstorms robot soccer players are described elsewhere, so what follows will only be a short description of the physical set-up.



Figure 2. The LEGO robot soccer set-up. There is one goalkeeper and two field players on each team (one red and one blue team). The stadium has light towers, scanning cameras that project images to large monitors, scoreboard, rolling commercials, and almost 1500 small LEGO spectators that make the "Mexican wave". © H. H. Lund, 1998.

Each team consisted of one goalkeeper and two field players. The goalkeeper was controlled with a LEGO CodePilot, while the two field players were constructed around the LEGO Mindstorms RCX (Robot Control System). Each player had two independent motors to control two wheels to make the robot move around on the field, and one motor to control movement of the robot's mouth (so that it could "sing" the national anthem and "shout" when scoring a goal). A player had three angle sensors to detect the motion of wheels and mouth. All parts of the robots except for batteries and coloured hair to indicate the team were original LEGO elements (LEGO Dacta, LEGO Mindstorms, LEGO Technic).

In order to put the robot soccer play into a stimulating context, we built a whole LEGO stadium (see Figure 2). The stadium had light towers (with light) in each corner, and these towers also hold infra-red transmitters that could transmit information from a host computer to the RCXs. In one end, there was a scoreboard that could be updated when a goal was scored via an interface with the LEGO Dacta Control Lab (see Figure 3). Over each goal, there was a rolling commercial sign that held three commercials that were shown in approximately 30 seconds each before the sign would turn to the next commercial. The control of the two rolling commercial signs was made with the LEGO CodePilot. A camera-tower with a small b/w camera was placed in one corner. The camera (controlled from a CodePilot) could scan to the left and the right of the field, while displaying the image to the audience on a large monitor. Another camera was placed over the sideline on one side and should scan back and forth following the ball (see Figure 3). Also this camera image was displayed on a large monitors, and its control was made from LEGO Dacta Control Lab.

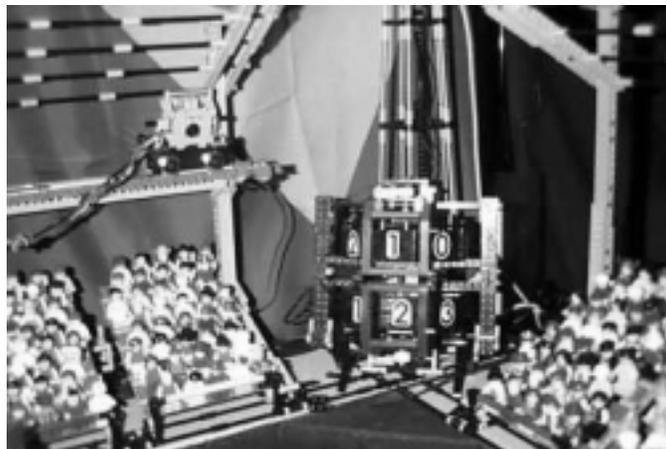


Figure 3. The scoreboard and one of the small cameras. The camera runs up along the sideline, while projecting the images to a large monitor. © H. H. Lund, 1998.

### **The LEGO spectator wave**

Apart from robot soccer players, cameras, rolling commercials, and scoreboard, we had placed almost 1500 small LEGO spectators on the grandstands. Our idea was to have all these spectators make the "Mexican wave" as we see soccer fans make at real World Cup matches. Our first intuition was to make a central control with one block running underneath the spectators and pushing the single spectator up when reaching him/her. Then, when the block had passed, the spectator would move down again. As the block moved around on the grandstands, it should therefore produce a wave when pushing spectators up and allowing them to move down again.

We made a prototype of this first model with a block running underneath the spectators. The prototype produced a wave of the spectators, but it had some deficiencies. First of all, the central timing control made the wave stop if the block became stuck for some time, and, secondly, the big LEGO spectators that we

were using would easily get wrapped up together. The second problem was a pure technical problem that we could soon solve using smaller LEGO figures and mounting them into sections. By doing this, the position of the LEGO figures would be fixed and they would not sway from one side to the other like the larger LEGO figures tended to do when they were moving up and down.

However, the first problem was of a more fundamental nature, and we would have to change the control perspective. The control approach was directly opposing what had been lectured about in the lectures, but this was apparently not realised until observing the implemented prototype. The control of the block was a classical example of timing control, or *open-loop control*, in which the reaction of the agent is dependent on an internal timing. The block would move forward for a pre-defined time and then move backward for the same pre-defined time. If something unexpected happened during the movement, the block would not be able to respond to this change of circumstances, since its actions depended on a timer rather than on environmental circumstances. The approach had been to idealise the world and assume a smooth movement of the block (essentially believing in a simulated model of the real world). On the other hand, the lectures had taught the students about *feedback control*, in which the behaviour of an agent is dependent on the feedback from the environment, so that the agent can react on changed environmental circumstances rather than having a fixed behaviour that depends on a timing.



Figure 4. The LEGO spectator wave. When the switch sensor is released, the next section of LEGO spectators will start to move upwards. © H. H. Lund, 1998.

The control of the wave was now changed to a feedback control, and the idea was to allow the wave to *emerge* from the interaction between the different sectors of spectators that each had their own, local feedback control. In this case, the implementation of a system with emergent behaviour should be possible in the real world, since there would be no demand of advanced sensing. In fact, a switch sensor for each section of LEGO spectators turned out to be enough (see Figure 5). The idea was that the movement of one section should be dependent on sensing what the adjacent section was doing. If a section was moving upwards, then the section to the right should sense this and start to move upwards itself. The section would fall down when reaching the top position (in this way, we used the principle that *what goes up, must fall down*). This was built by placing a switch sensor under each section and connecting this switch sensor to the control unit (a LEGO CodePilot) of the next section. In resting mode, the switch sensor would be pressed by the section of spectators above it, but when this section started to move upwards, the switch sensor would no longer be pressed. This triggered the simple control program in the next section to start moving the section of LEGO spectators upwards. In pseudo-code, the control program of each section could be as follows:

```
Section N control:  
  if (Switch(N-1)=false) then turn on motor  
  else turn off motor
```



Figure 5. The movement of a section is triggered by the switch sensor mounted underneath the adjacent section. The two arrows show the placement of the switch sensors. The left section of spectators has risen, so the switch sensor is no longer pressed. The control of the right section will notice this, and start to move upwards (immediately after this photo was taken). © H. H. Lund, 1998.

This very simple control allows the wave to *emerge* when one section is triggered from the external to move upwards. In the actual implementation in the LEGO CodePilot language, it was however necessary to use a timer, since the time slice of the CodePilot is so small, that the above pseudo-code program would result in the section barely moving upwards before the motor would be turned off. So when it was sensed that the switch was no longer pressed, the control would turn the motor one direction for 0.5 seconds and then the other direction for 0.5 seconds. It would have been more sensible to have a switch sensor on the top position that the section should reach, and then base the time of upward and downward movement on the feedback from the top and the bottom sensors. But since the LEGO CodePilot has only one input channel (see Figure 6), we opted for the timing solution. However, it must be noted that this timing is very different from the timing in the first prototype, since here, even though the timing of a section might have been wrong, the section would still lift itself for some time and therefore the next section would be triggered. In a sense, we are setting the time-slice to 0.5 seconds and use the pseudo-code program.

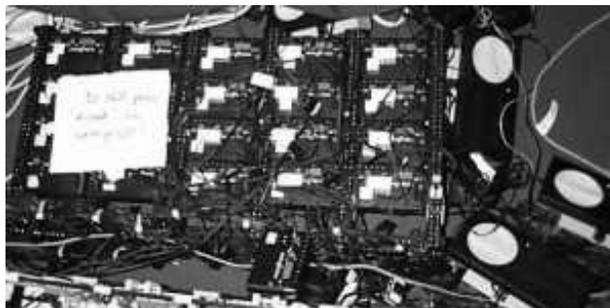


Figure 6. The mixer. 16 LEGO CodePilots were used to construct the dynamics of the “Mexican wave”. Each CodePilot had one switch sensor and one motor connected. © H. H. Lund, 1998.

The feedback control was used and the wave emerged from the interaction between the simple units. It was run numerous times daily in Paris for a week without any need for refinements apart from a couple of changes of physical aspects (once or twice, a section got stuck).

## Conclusion

We used emergent behaviour to construct a “Mexican wave” of LEGO spectators for the LEGO robot soccer demonstration. The wave emerged from the interaction between sections of LEGO spectators, each with its own simple control. The control was based on feedback from the local environment. Since sensing was straightforward with one switch sensor for each section, it was fairly easy to implement a real world emergent behaviour. Under other circumstances, it might be more difficult, since more advanced sensing might be necessary, and we cannot guarantee that the desired real world behaviour will emerge. Therefore, the study of real world emergent behaviour is important in order to identify the circumstances that will lead to successful results.

## Acknowledgements

Students from the LEGO Lab made a highly valuable work in the LEGO robot soccer RoboCup’98 project. Lars Nielsen from LEGO Media worked as a very helpful partner who gave numerous valuable ideas. Ole Caprani participated in a number of inspiring discussions. Thanks to H. Kitano for collaboration on the Paris RoboCup’98 event.

## References

- [Jakobi, 1998] N. Jakobi. The Minimal Simulation Approach to Evolutionary Robotics. In T. Gomi (ed.) *Proceedings of ER’98*, AAI Books, Kanata, 1998.
- [Kitano et al., 1997] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matsubara. RoboCup. A Challenge Problem for AI, *AI Magazine*, 73-85, spring 1997.
- [Lund and Miglino, 1996] H. H. Lund, and O. Miglino. From Simulated to Real Robots. In *Proceedings of IEEE 3<sup>rd</sup> International Conference on Evolutionary Computation*, IEEE Press, NJ, 1996.
- [Miglino et al., 1995] O. Miglino, H. H. Lund, and S. Nolfi. Evolving Mobile Robots in Simulated and Real Environments. *Artificial Life* 2(4), 417-434, 1995.
- [Minsky and Papert, 1968] M. Minsky and S. Papert. *Perceptrons*. MIT Press, Cambridge, MA, 1968.
- [Reynolds, 1987] C. W. Reynolds. Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics*, 21(4), 25-34, 1987.
- [Rumelhart et al., 1986] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Representations by Back-propagating Errors. *Nature*, 323, 533-536, 1986.