8
Conclusion

Our research has been very fruitful, overall, although those fruits are not yet ripe for the harvest.

The first test application (*Creepy Crawlies*) was capable of developing "nice" animations for a variety of creature designs. The results were successful in displaying realistic motion, although they were not always aesthetically-pleasing. The application of "style points" [Gritz and Hahn 1997] might be able to steer the evolution towards better-looking animations, but we do not yet know how to formulate style guidelines for arbitrarily-shaped creatures. This is clearly a good topic for future research.

The *Creepy Crawlies* example could be easily converted into an application to be used through the internet, as a casual game. In fact, some of the creatures fascinate users by displaying hilarious but plausible ways of locomotion.

One clear disadvantage of the proposed method is that the generated animations only apply to the environment where they were evolved. For instance, a walking animation that evolved on a flat floor is useless if the character is confronted with a staircase[1] In fact, during the evaluation phase, the animation is played for only ten seconds and the application cannot guarantee that it will be viable after that short period. During our test runs, an animation that caused the creature to stumble after the initial ten-second period was assigned a high fitness score; when that animation was selected for playback, the creature continued to blindly execute its programmed movements despite the fact that they were no longer carrying it forward (see Figure 8.1).

We hoped that genetic programming with expression trees would be able to circumvent this problem. Unfortunately, the second test application (*CC3D*) failed to produce viable animations at all. We suspect that this may be caused by an excessively "loose" formulation of the problem: The animation programs could be represented by mathematical equations that returned any real number, resulting in a search space that was substantially broader than

[1] Although the authors make no mention of it, we believe that an implementation of [Gritz and Hahn 1997] would suffer from the same problem.

the discrete commands (*stretch*, *relax*, and *contract*) available to the first test application. Although we have not thoroughly analyzed the results for this second test application, we decided to keep the references to it because we believe there is promising future research to be done on those results.

Due to lack of time, we could not experiment with all active components listed in Chapter 3. In the future, we hope to build more test applications with different combinations of active components, passive components, physics engines, and animation representations, as well as attempt to integrate "style points" into the evaluation phase.

It would also be possible to improve the applications by making the system flexible enough to introduce new op-codes and tree operators by the programmer. A simple code could be a variation of contracted/stretched states, such as "greatly contracted/stretched". Furthermore, this could raise interesting questions about the possibility of introducing fuzzy quantities into the process.

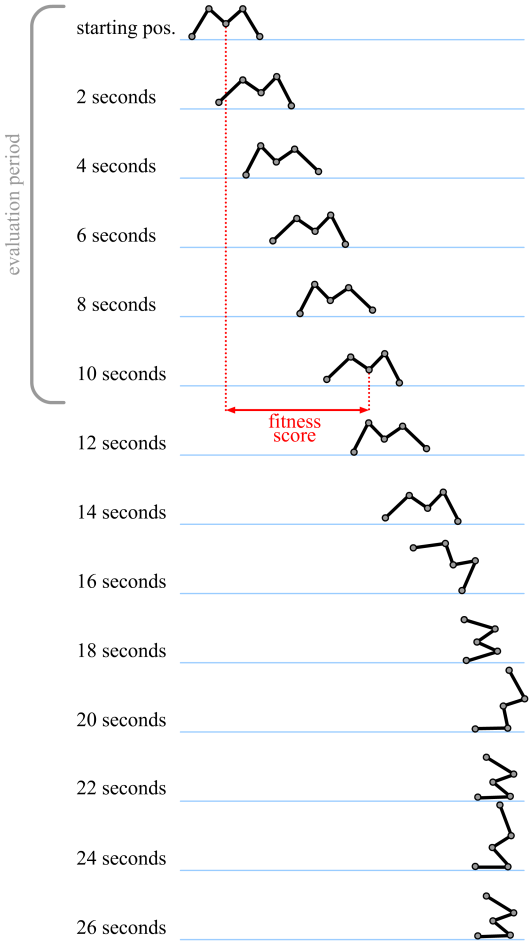

Figure 8.1: The animation that results from interpreting an op-code sequence is only evaluated for ten seconds. Later playback may reveal flaws in the animation that were not detected in the evaluation phase of the genetic algorithm.