Genetic Walkers

Johan Andersson

Introduction

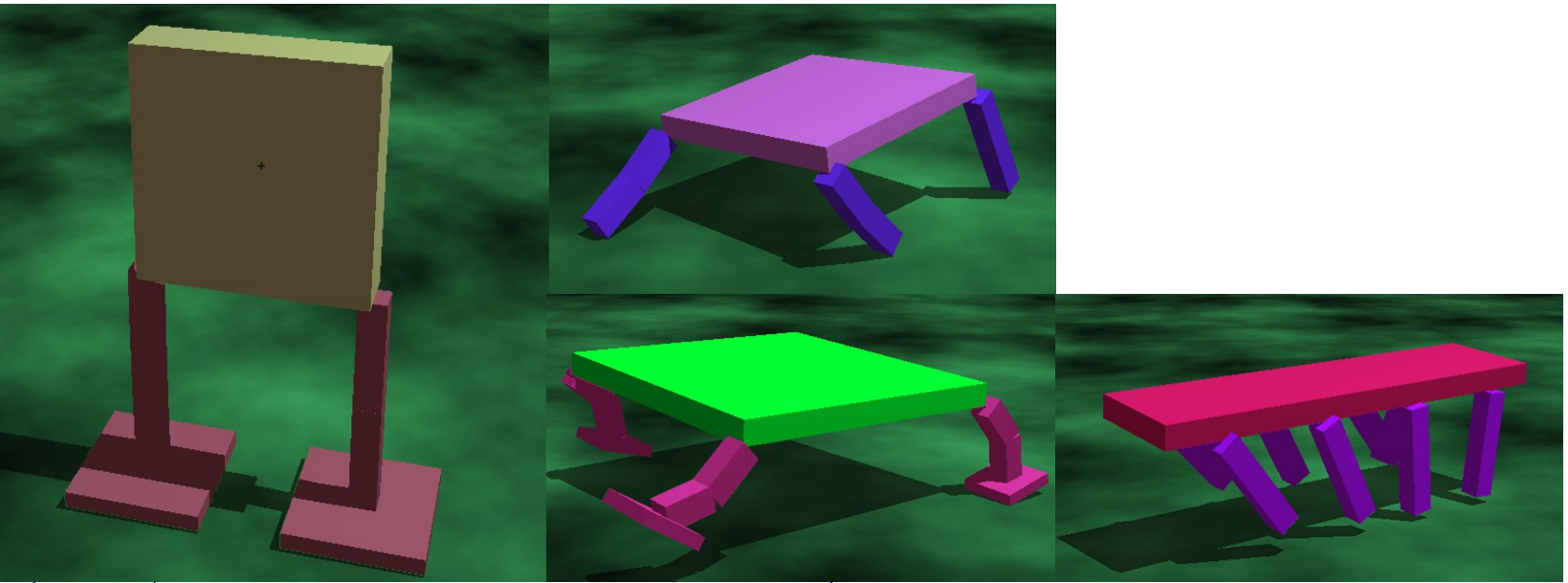
- Goal: Evolve movement of 3D 'creatures'
- How: Genetic algorithm
- Simulation environment: Breve (www.spiderland.org)

Why this project?

- Interesting

The Walkers

- Biped – Two legs and feet
- Quadruped – Four legs and feet
- Simple Quadruped – Four legs, no feet
- Octoped – Eight single segment legs, no feet



LUND UNIVERSITY

Movement

- Legs and feet are controlled by joints
- Joints move attached limbs with a certain velocity
- The velocities are calculated each iteration as:

```
def calculateJointVelocity( j, time ):

return amplitude * sin( angv * time + getDT(j) )
```

- Amplitude, angv and dt depend on the walker DNA

DNA

- List of values for the velocity function
- Example: Simple Quadruped
- 1 DT per joint + angV + amplitude = 10 values
- [dt0, dt1, … dt6, dt7, angv, amplitude]

DNA

- Quadruped
- Has an evolving shape
- 3 DT per leg + angV + max amplitude + body width + two leg lengths (upper and lower) + footWidth = 18 values
- Larger search space, harder to find a good solution!

Algorithm cycle

- Simulation
- Fitness evalutation
- Selection
- Breeding
- Repeat

The fitness functions

- Biped – Time until it falls over
- Quadruped – Distance from original position
- Simple Quadruped – Distance from original position
- Octoped –Distance from original position

Selection

- Elitism, the x best walkers get to live on
- Tournament selection:
- Two individuals are selected randomly, and the one with higher score get to be a parent.
- Repeat for another parent.

Breeding

- Two parents produce two children
- One point crossover
- Mutation

Mutation

- Mutation rate: around 5%
- Mutated values changed by 10%

Results

- [Video]

Possible improvements

- A new simulation environment
- Better movement function
- Optimization and multithreading

Questions?