Genetic Algorithms
CP4: Fitness and Selection

Authors:
Kingdon Barrett
Andrew Bair

Evolutionary Camera Shots - Fitness and Selection

When speaking of the fitness algorithm regarding the camera engine as it has been defined, there are a number of fitnesses to be evaluated. In this document, three terms are differentiated: *camera* is used to refer to an individual camera angle, *camera population* refers to the evolutionary state of the whole population of camera angles as defined across the time and space of the simulation, and *camera shot* refers to the path of the (one or more) products of our evolutionary algorithm. A camera shot defines the position and orientation of the camera from the beginning of the simulation to the end. The evolutionary algorithm may produce more than one viable camera shot; this is a feature, not a bug.

Camera Fitness

An individual camera exists to frame a single shot. Given knowledge of all of the objects in the world, the camera should focus on the most interesting (largest cluster) of objects. As a secondary concern, the camera should exist in a position somewhere in the orbit of that largest cluster as to view as much of the action "off in the distance" as possible. Finally (this does not enter directly into the fitness of a camera, but is a variable nonetheless) the camera must exist at a particular moment in time. A given camera is more fit if it satisfies these requirements well. If not, the camera is invalid or unfit.

However, a population of highly fit cameras does not necessarily mean we have found a good shot. We may have found some partial good shots that occur in sequence, but with awkward transitions from partial shot to shot. Further, we may have discovered a number of camera shots that all capture the action equally well at the same moments, but with no way to quantify or decide between these known camera shots. To help with this decision-making process, we have introduced the notion of a population fitness.

Temporal Fitness of Camera Population

To produce a viable camera shot that spans from the beginning of the simulation to the end, there must be enough camera shots to transition in real time from one to the next without any significant gaps in the action. While a stationary camera may be the best option for as long as 10 seconds, it should be noted in the data model that this is intentional, and an effect of the algorithm. This could be achieved by making copies of the camera and placing them in the stationary space, separated by perhaps one second. This would prevent the camera breeding algorithm from filling up these "stationary" times with more cameras of lower fitness.

Similarly, some camera motions may map to a continuous function. Rather than adding a notion of functional behavior for a camera shot, which would complicate our breed and mutate operations among other things, the data model should be fully expressive of this sort of configuration, with data points for several locations along the path of the function.

In short, for every time, there should be at least one camera. For every time with multiple cameras, there must be another way to determine the best camera. (If our EA is well-designed, we will eventually eliminate the other cameras, or find a way to classify them into multiple camera shots. Our API should expose these choices as "camera styles" and allow the user or developer to choose between them.)

Camera Shot Fitness

The fitness of a camera shot is dependent on the fitness of individual cameras, and also the temporal fitness of a camera population. This is not the only determining factor; aside from stylistic variation, each camera shot should execute a single smooth and continuous motion from the beginning to the end of the simulation. This continuity is what is intended by the idea of camera shot fitness.

Given a temporally fit population, a path through the cameras is selected. The camera shot is born as an ordered bit string, where each bit corresponds to a camera in the sequence of all cameras ordered by time. Each Nth bit in the string corresponds to the the Nth camera in time; as such a camera shot may be mutated (or invalidated) by changes in the population of cameras due to further evolution.

The fitness of a camera shot is related to the summation of the fitness of each selected cameras in the camera shot. A camera is fit with respect to a camera shot if the camera falls roughly “between” the two camera shots on either side of it.

Summary

In plain English, the evolution of a camera shot is therefore the process of evolving the population of all cameras as a whole, and of selecting from the population a subset of all cameras which describes a fluid motion through the simulation. The code is the definitive guide to exactly how this works.