Behavior-based Control

We are looking at this in the context of robot control, but many of these same principles apply with any program interacting with an environment.

Three primary levels of robot control, although in practice they may be mixed:

(1) Reactive control.
(2) Behavior-based control.
(3) Hierarchical control.

They are increasingly "smarter."

(1) In reactive control, the stimuli (or sensor inputs) are connected fairly directly to the actuators or effectors (the devices that do some action). The Braitenberg vehicles are examples of reactive control.

This is a very simple kind of control, but it can look surprisingly smart. Braitenberg observed that these vehicles, especially when placed in a complex environment (to which they were responding) could exhibit complex and apparently purposeful behavior.

(2) Behavior-based control: This based on dividing an agent's overall behavior into a number of more-or-less independent sub-behaviors (composed of a stimulus-action pair).

(3) Hierarchical control: Based on "thinking": includes cognition, reasoning, maps and other internal representations of the world, memory, planning, etc. This tends to be slow and sequential and it has proved too slow for controlling many robotic

processes.

Let's return to behavior-based control.
We break down the overall complex behavior of an agent into a number of independent behaviors. Each such behavior is tightly couple stimulus-action pair serving a well-defined purpose.
Examples: if see a seed, peck at it.
if you see an approaching cat, fly away.

The stimulus can be used in two different ways (often simultaneously):

Releaser: When a stimulus is used as a releaser, it triggers the corresponding action (or more correctly, releases the action). For example, the sight of the predator, might release the fleeing behavior.

Guide: the stimulus guides the behavior once it is released. For example, once the the fleeing behavior has been released, other stimuli will guide the action, for example, to fly away from the cat, or to fly to some safe place.

Sometimes the same stimulus is both releaser and guide, for example, the seed that the bird pecks.

Conceptually they serve different purposes. For any action, you should consider what you need to release the action, and what you need to guide the action.

One advantage of this approach, is that we don't have to understand everything in our environment. Each behavior needs only to be able to detect or use relevant affordances in the environment.

In BBC we break a complex behavior down into simpler behaviors. But then that raises the problem, how do we put them back together again? Often several different behaviors will be released at the same time. For example, the bird may see a seed to peck at the same time it sees an approaching cat.

(1) One way to combine behaviors is by some notion of priority. For example, fleeing probably has higher priority than eating.

(2) You can do both simultaneously, if that makes sense.

(3) You can blend the actions sometimes, accelerating and turning at the same time.

These are three typical ways of combining behaviors. The point is, you have think about what is the best way to combine them, and then implement it with some sort of arbitrator, which is a piece of software that combines actions according to some specific strategy.

One valuable aspect of BBC is that each behavior can be tested separately. This is called unit testing.

Formatted Input/Output:

Often we want to control the format of information we print on the console or write to a file. We can do this with printf().

```
printf ( <format string>, <expr 1>, <expr 2>, … , <expr N>);
```

The format string tells how to print the information. Generally characters in the string are printed as themselves, except for special format codes (beginning with "%"), which tell how to print each one of the <expr>s.