Cooperating Robots
Lecture Notes
Behavior

Peter Johansen

21. februar 2001

The present note outlines the principles for building the robot control. The basic inspiration has come from Tu and Terzopoulos' work simulating fish [2] and from Reynolds' work on what he has coined *boids* [1]. Both works have been inspired by/are implementations of Brooks' subsumption architecture.

The architecture consists of a control, called the *intention generator* and a set of *primitive behaviors*. In rough terms the intention generator is a finite state automaton (FSA) with inputs from the sensors and with outputs that select primitive behaviors. An *emergent* behavior is a global behavior property that has not explicitly been programmed for. An example is bees building a 2D beehive. Assume initially two neighbor hexagons. Let the bees build a new hexagon from the primitive behavior: build a hexagon at a location which requires a minimum of additional material. The bees will select the location of a new hexagonal cell in such a way that the resulting beehive is approximately circular. This property was not explicitly programmed in the bees primitive behavior.

There are two difficulties. One is that the amount of possible different sensor inputs is much to vast to tabulate. Another is that the primitive behaviors can not always be independently executed. An input may interrupt a behavior and trigger a higher priority one. Examples are

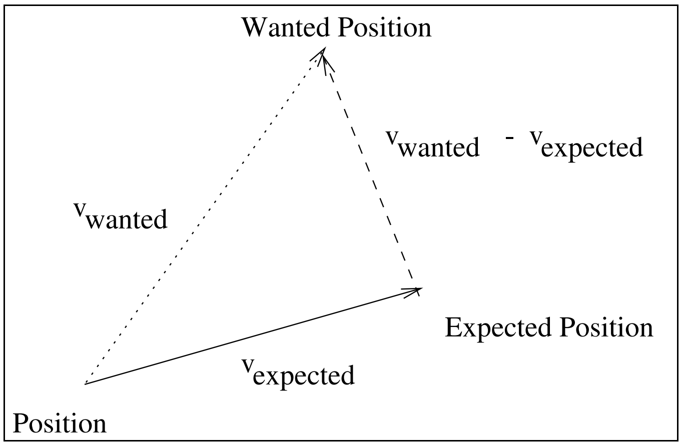
Reynolds' boids combine two behaviors: Searching towards the center of the neighboring boids and simultaneously trying to avoid colliding with the nearest neighbor. He computes a velocity vector for each behavior and combines them by vector addition of the two velocities.

A fish in Tu and Terzopoulos [2] interrupts its feeding behavior when in danger to invoke fleeing behavior. When no more in danger it returns to the interrupted behavior.

Primitive Behaviors

Reynolds mentions several examples of primitive behavior, among them are: **seek**, **flee**, **pursuit**, **evasion**, **arrival**, **wander**, **separation** (figure 2), **cohe-**

sion (figure 3), **alignment** (figure 4).



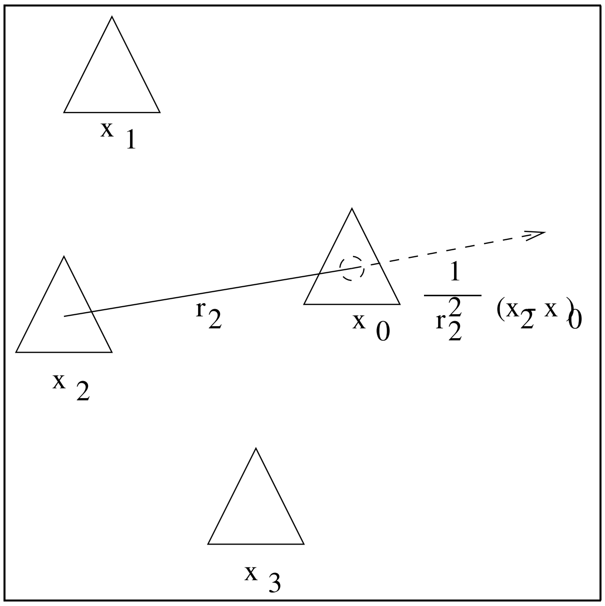
Figur 1: Primitive behavior

Combination of Behaviors

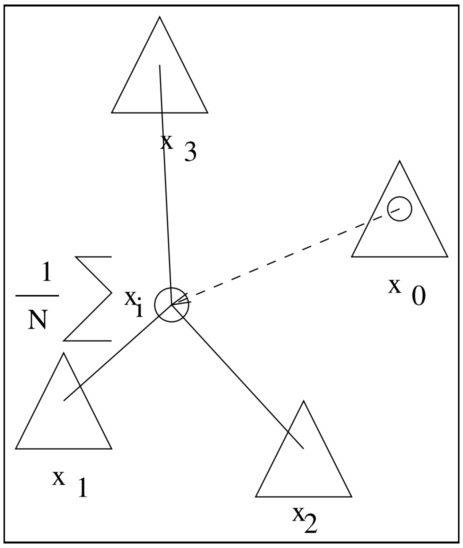
Reynolds' flocking behavior is obtained by superimposing the behaviors separation (figure 2), cohesion (figure 3) and alignment (figure 4) by simply adding the three vectors. This composite behavior is the primitive schooling action in the diagram in figure 5.

Tu and Terzopoulos control their fish by an intention generator. Rather than superimposing the different behaviors, they shift between different primitive behaviors. The intention generator decides which behavior to execute based upon the state of the environment. It can also schedule the behaviors if interrupts are necessary. One may design the primitive behaviors as uninterruptable atomic actions. In that case no interrups scheduling is necessary. If, however, the primitive actions extend over longer periods of time, an interruption system may be necessary. This control design is similar to the job scheduling design in a computers' operating system. Figure 5 shows a diagram of a simple behavior control. Notice that wandering is not interruted by approach of a predator. This is due to the assumption that the wandering behavior only takes very little time.

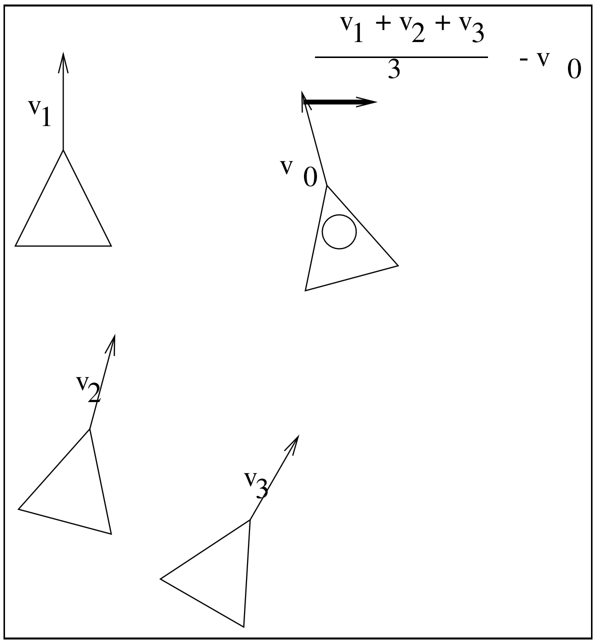
Reynolds also mentions *prioritized dithering* in which primitive behaviors are selected according to their priority in relation to the environmental conditions, but each with a certain probability conditioned upon the outcome of a test. With reference to figure 5, control at danger of collision is transfered by choosing "Yes" and "No" with a probabilities which depend on the outcome of the collision test.



Figur 2: **Separation** $v_{wanted} - v_{expected} = -\sum_i \frac{1}{r_i^2}(x_i - x_0)$



Figur 3: **Cohesion** $v_{wanted} - v_{expected} = \left[\frac{1}{N}\sum_i x_i\right] - x_0$



Figur 4: **Alignment** $v_{wanted} - v_{expected} = \left[\frac{1}{N}\sum_i v_i\right] - v_0$

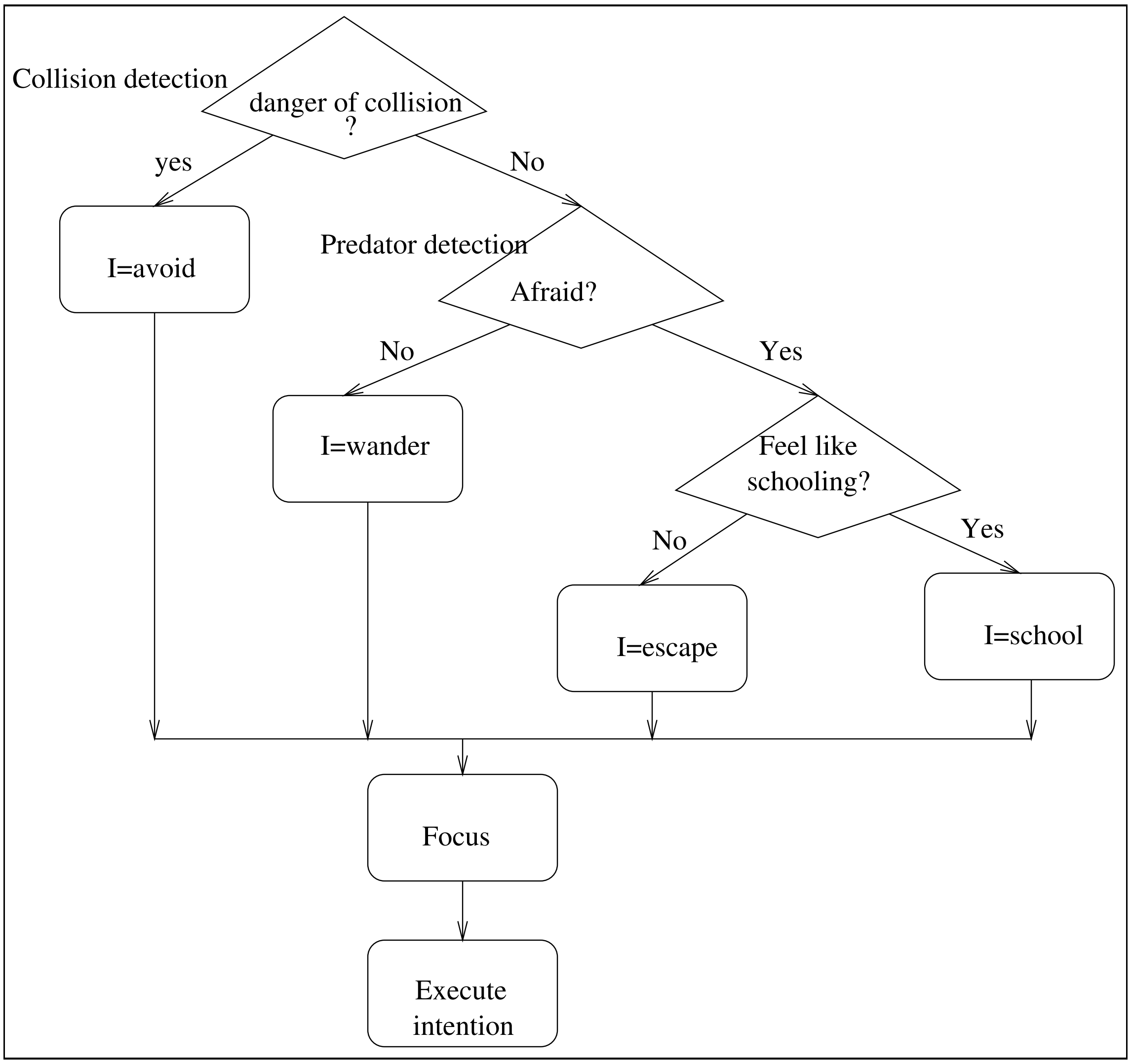
Conclusion

To me it is striking how closely the strategies to control virtual (fish and boids) and embodied agents (our robots) resemble control strategies in computer architecture. The programming facilities made available by Martin, Søren, and Stephan at http://www.diku.dk/research-groups/image/teaching/Robotlegestue/RoboCom/ makes it possible for you to program robots in the much same way as you would program a simulator of a digital computer.

Litteratur

[1] Craig W. Reynolds. Steering behavior for autonomous characters. In *http://www.red3d.com/cwr/steer/*, 1999.

[2] Tu and Terzopoulos. Artificial fishes: Physics, locomotion, perception, behavior. In *Proceedings of SIGGRAPH'94*. ACM Computer Graphics, 1994.



Figur 5: **Intention generator.**