## pima744.1 fall2009 lecture 1

**today:**

- who am i?
- what is programming and where you can find it
- introduction to programming: an agent-based perspective



## about me

- undergrad: Barnard, CS major/English minor, BA 1985
- grad school: Brandeis University, MA 1997, PhD 2000 Computer Science
- 10 years of industry experience working as a scientific and business programmer (prior to grad school), mostly at MIT Lincoln Laboratory (`http://www.ll.mit.edu/`)
- previous teaching:
  - Monash University and University of Melbourne, Melbourne, Australia (Summer 2000)
  - Boston College, Massachusetts (Fall 2000–Spring 2001)
  - Columbia University (Fall 2001–Spring 2005)
  - Brooklyn College, Fall 2005– ...
- research interests:
  - multi-agent simulation, artificial intelligence (AI) and robotics
  - multi-media learning environments, including educational robotics
  - data visualization



## introduction to programming

- my pitch:
  - programming is ultimately about **control**!!!
  - learn how to control computers... and robots... and a surprisingly large number of devices and other seemingly non-technical components that you encounter in your everyday life!
  - today, technology is ubiquitious—learn how to control it before it takes control of you!
- but seriously:
  - introduction to computer programming from an agent-based, device-control perspective, for multimedia artists
  - i'll refer to *agents* and *robots* as a *context*



## how to learn a programming language

- YOU are responsible for your own learning!!!
- john and i will point you in the right direction...
- but YOU must PRACTICE, PRACTICE, PRACTICE...
- and PRACTICE some more!!!
- it's like learning to play the piano—you have to put your fingers on the keyboard!
- if you don't understand, then ASK for help!
- you need to be ready to THINK and learn how to figure things out.
- think "outside the box"!

## getting started

- programming is like solving puzzles
- think differently
- the world is now made up of...

  *agents*
  and
  *behaviors*

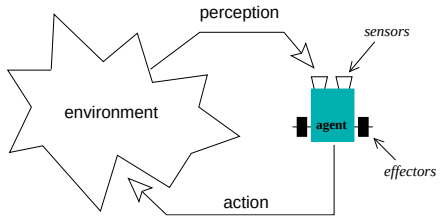
  or

  *objects*
  and
  *actions*



## what is an agent?

- an *autonomous* entity that exists and interacts in an *environment*
- contains *sensors* to perceive its own state and its surrounding environment
- possesses *effectors* (e.g., motors) which perform actions or *behaviors*
- has a *control program* which takes input from the sensors, makes *intelligent* decisions about actions to take, and outputs those actions by sending commands to effectors
- a *robot* is an *embodied agent* that interacts in a *physical environment*






## what is a control program?

- a *control program* is a set of instructions that tells the agent what to do
- a *programmer* is a person who writes those instructions
- there are many different *programming languages* that one can use to write programs—in this class, we will learn **MAX**
- the MAX system will translate your program from the high-level code that you write into low-level machine language that the computer, and the devices you control, can understand
- then the computer/robot/agent/device will *execute* those commands by reading the instructions in the low-level machine language that it understands



## commands

- a program/robot/agent/device follows commands
  *commands = series of instructions*
- you will learn how to *command* a computer/robot/agent/device
  *command = program = write instructions*
- you understand the commands,
  but does the computer? the robot? the agent? the device?
  that's a question of cognition...
  → Artificial Intelligence, Cognitive Science
- what do you think?
  something to ponder over and discuss over tea on a rainy day...

## back to agents...

different kinds of agents

- human "agent":

| | | |
|---|---|---|
| *environment*: | physical world | |
| *sensors*: | eyes, ears, ... | ← input |
| *effectors*: | hands, legs, ... | ← output |

- software agent:

| | | |
|---|---|---|
| *environment*: | e.g., UNIX operating system | |
| *sensors*: | `ls`, `ps`, ... | ← input |
| *effectors*: | `rm`, `chmod`, ... | ← output |

- internet agent:

| | | |
|---|---|---|
| *environment*: | the Internet | |
| *sensors*: | `http` requests | ← input |
| *effectors*: | `http` commands | ← output |

- embodied (robotic) agent:

| | | |
|---|---|---|
| *environment*: | physical world | |
| *sensors*: | light meters, bumpers, thermometers, ... | ← input |
| *effectors*: | motors attached to wheels, legs, grippers, ... | ← output |



## agent decision-making: what to do?

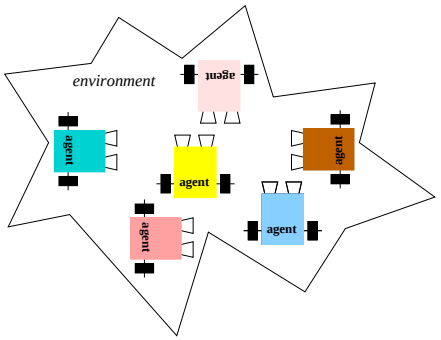
- need to know *what* to do in any given *state*
  - what = an *action* that the agent can take
  - state = a configuration of the agent and its environment
    for example: the position of all the pieces on a chess board, or the robots and the ball on a robot soccer field, or the position of a robot's gripper, or all the bids in an electronic market
- *autonomy*
  - a crucial concern for agents
  - run-time decisions are made by the agent *alone*—i.e., no human remote control
  - means behavior is based on *own* experience
  - implies *learning*, *adaptation*



## multiagent system

- a *multiagent system (MAS)* is...

  an environment in which many (well, two or more) agents exist and interact






## properties of multiagent systems

- individual agents are *self-interested*
  i.e., they have their own goals, even though there may be team rewards for a group of agents achieving a goal together
- cooperation is not governed—it is *emergent*
  (and is not necessarily a feature of every multiagent system)
- versus "distributed systems", where
  - goals are only group-based
  - cooperation is engineered to be inherent in the system

## artificial life

- *Artificial Life as a Tool for Biological Inquiry*,
  by Charles Taylor and David Jefferson (1995)
- "ALife" consists of four levels:
  1. molecular level — "wetware"
  2. cellular level — "software"
  3. organism level — "hardware" (e.g., one robot)
  4. population level — "multiagent systems" (e.g., multiple robots)

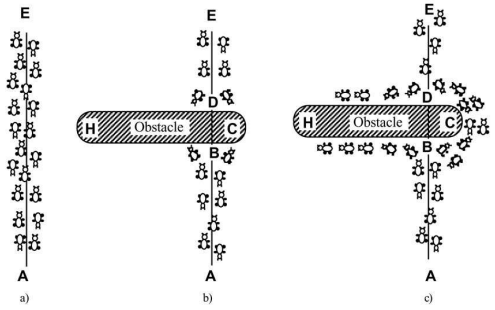


## ant systems

- *The Ant System: Optimization by a colony of cooperating agents*,
  by Marco Dorigo, Vittorio Maniezzo and Alberto Colorni (1996)
- the "ant system" as an approach to stochastic combinatorial optimization
- "ant" = simple agent which (sort of) mimics the behavior of real ants
  - real ants lay *pheromone* trails to mark their pathways
  - individual behavior is apparently random
  - collective behavior emerges as *autocatalytic* ($\Rightarrow$ positive feedback); i.e., the probability with which ants subsequently follow the same trail increases as more ants take that trail
- this is <u>not</u> about simulating ant colonies, but rather about taking advantage of abstract behavioral properties of ant colonies to address optimization problems
- major diversions from reality:
  - "ants" have memory
  - "ants" are not blind
  - "ants" live in a discretized environment (time and space)



- experimental example:






## multiagent-based simulation

- *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*,
  by Mitchel Resnick (1994)
- centralized versus decentralized models, ways of thinking
- the old way: *centralized* — "by lead or by seed"
- the new way: *decentralized*
- examples of decentralized computational models:
  - neural networks
  - subsumption architecture
  - cellular automata
- properties of decentralized models:
  - *emergent behavior*
  - *evolutionary learning*

## multiagent simulation for learning

- *Modeling Nature's Emergent Patterns with Multiagent Languages*, by Uri Wilensky (2002)
- decentralized tools for learning: *constructionism*
  - hands-on exploration
  - no recipe to follow
- **NetLogo**
  - "turtles (agents)
  - "patches" (environment)
- lessons for understanding decentralized thinking:
  1. positive feedback isn't always negative
  2. randomness can help create order
  3. a flock isn't a big bird
  4. a traffic jam isn't just a collection of cars
  5. the hills are alive