Frictionless Vacuums

Physics students are often given homework with the note that they are to ignore air resistance, friction, wire resistance, and many other complicating factors unless explicitly specified. Invariably, applying naïve frictionless-vacuum solutions to the real world goes wrong, because the world is very much not frictionless and very full of air and other very complicated factors. Concepts break down outside of their context.

Jokes abound with what one should do should one ever find oneself with a particularly annoying physicist and a vacuum chamber.

On a similarly cruel note; the philosopher John Searle, discussing artificial intelligence, came up with the thought experiment of a Chinese room. A man, who does not speak a lick of Chinese, is locked in a room and given a stream of Chinese characters; he is equipped with a book listing appropriate responses, still in Chinese, which he does not understand, to various character streams; he then writes out the response stream and feeds it back out a slot in the room. To a human interacting with the room-man system, it would appear that it speaks fluent Chinese; the individual components of the room, however, are utterly incapable of doing so. Indeed only one of the three components, the man, could even be argued to be intelligent and even theoretically capable of understanding any Chinese language.

Searle's argument was that, as the man does not understand the words he writes, so too does an artificial intelligence not understand anything it says or does. Had he taken an interest in the theory of learning, Searle would have been a behaviorist.

Searle's focus is on the man, the central computational element, as being non-understanding, but the man is not the entire system. While the man does not understand what he writes, the system as a whole does. Just as a single neuron is incapable of complex thought, a single transistor is incapable of running Doom (1993), a single termite is incapable of building a mound, and a single man in a box is incapable of understanding a language he does not speak; however, a system of neurons is capable of complex thought, a system of transistors of running Doom (1993, or perhaps even 2016 in sufficient quantity), a system of termites of building a mound, and a box-man-book system of writing and understanding Chinese. The room is a gestalt, a greater whole, of its components; by analogy, a person is a gestalt of their memories and various cognitive processes.

There is another part of the system not yet mentioned, a part more immaterial than the rest: the flow of information from input slot to man to book to man to output slot. Devoid of the information stream, the gestalt degenerates back into a box, a man, and a book with strange characters in it. The system only functions when fed an input.

Compare also the case of a skilled woodworker building a cabinet; this system functions with an

input of wood and screws, which the woodworker transforms into an output of beautiful cabinetry. The woodworker, deprived of wood or screws, is unable to perform their function; similarly, the wood and screws, deprived of a woodworker, are unable to be transformed into beautiful cabinetry. The inputs, processes of transformation, and outputs, are all interdependent.

In both the aforementioned examples, the process of cognition - conversation in a foreign language or woodworking - is tied deeply to the act of performing it; it is not such a mental leap to imagine such circular dependencies in all processes of cognition. The process of knowing how to do something is inseparable from doing the thing itself, a concept called situated cognition.

Our interpretation of the room metaphor differs from Searle's; Searle's behaviorism-alike position cares little about the transformation of input to output, situated cognition does.

Situated cognition, as the latter half of the name implies, is all about about the process of cognition; to elaborate, situated cognition has the concept of a perception-action cycle, consisting of a being perceiving affordances and then possibly acting on them. The being bases their actions on their intent - goals - and on the current situation's (in)variance - how similar or different it is - from another situation they have been a part of. When a group of beings with highly invariant intent - similar goals - and act upon their affordances to reach their intent together, they are considered to have formed a "community of practice"; within the superset of all communities of practice are the notable subsets of the set of all professions and the set of all hobbies, but more fundamental communities of practice also exist, such as the community of all humans who are currently alive and want to stay that way.

As situated cognition focuses on learning through doing, a classroom implementing the theory must focus on teaching though doing. Situated cognition has the concept of "legitimate peripheral participation", which is the concept that a newcomer to a practice first participates in low-risk low-stakes tasks, moving up to higher-risk and higher-stakes tasks as they become more experienced and move closer to the center of the community of their practice. To teach, an instructor will introduce the practice and the community, an overview and pertinent details, and start the students on simple peripheral tasks.

We shall study the example case of an introductory computer programming course, prerequiring an elementary education in mathematics and basic computer skills.

In such a course, a student would be introduced to, in order, what it means to program a computer, primitive forms of computation, program flow, and variables. To note, the former two - what it means to program, and primitive computation - are far more foundational than the rest of the course and may be considered implicit in the course, its requirements, and modern life as a whole.

The students would first be taught about what it means to program a computer, to command it and make it do their bidding, and primitive computation, basic calculators and the like. A student would be already familiar with a four-function calculator, but the teacher would wrap their minds

around using a calculator being a form of programming a computer. In this same example, the students would also be attenuated to the variance between writing a full graphical application program, and the more fundamental operations that make up such a program. At this point in the student's learning, they may be unable to practically apply the lesson; optionally, they may fiddle with their calculator, or instruct their computer to print a photo, and thus marvel at their ability to program, but more knowledge is required for nontrivial peripheral participation.

The missing link is the concept of program flow, a concept that a program works by performing numerous operations in, at first, linear sequence. A teacher may give their students a simple program for them to execute, commonly programming the students to open and then walk through a door, or more abstractly to compute two numbers and then add a third number to the result. With the concept of program flow, a student may then be guided through creating their own abstract program examples, commonly creating a program for baking a cake. A delicious and rewarding follow-through would be to have a "processor" student follow the programmer student's instructions as strictly as possible, and hopefully getting an edible cake from it. In this follow-through, the students would have a visual representation for a program being executed, as well as, should anything have gone wrong, learned "the hard way" that a malformed program produces malformed results.

Implicit in the prior example of "compute two numbers and then a third" is the concept of a variable, a way to refer to a piece of data that changes; the common example is a cubbyhole or a locker containing some thing that may change, but the cubbyhole, the reference to that thing, not changing. The students would thus be exposed to the invariance between a concrete concept with which they are highly familiar and an abstract concept less familiar.

Through use and extension of situations a student is already familiar with, even a subject as abstract as computer programming may be taught to a complete newcomer. The field of computer science is stuffed full of opaque jargon, but grounding the jargon in its roots helps newcomers (and even, for that matter, old-timers) better understand the practice they are part of.

Returning to our room metaphor, it is not such a stretch to imagine the man inside responding faster as time goes on; improving at drawing characters, memorizing page numbers, memorizing short responses, and eventually becoming so intimately familiar with the book as to rarely need it at all. All the knowledge of the man, however, becomes quite useless when he is released from his prison; the knowledge of the system and the learning of the man are closely coupled to the situation which they are in.

All learning, all knowledge, is derived from context; put another way, all cognition is situational.

To think, do.