ALife using Adaptive, Autonomous, and Individual Agent Control

Ovi Chris Rouly

Department of Computational Social Science
George Mason University
Fairfax, Virginia, USA
orouly@gmu.edu

Abstract. This is a review of three, agent control algorithm, replication experiments. In the 1948 essay, *Intelligent Machinery*, the English mathematician Alan Turing described an algorithm for constructing a machine that he claimed was capable of cybernetic (or steered feedback) self-organization. There are few, if any, references in either the historical or the technical literatures to instantiations made of this algorithm. Turing named the machine the P-Type Unorganized Machine. Considering the lack of replication evidence in the literature, three hypotheses motivated this review: 1) Turing did not describe the algorithm with sufficient detail so as to make it possible to instantiate, or that 2) if the algorithm could be instantiated it would not operate as Turing stated, or 3) both. The three replication experiments reviewed here proved the hypotheses qualitatively false when unique P-Type machines were instantiated. Each instantiation functioned as an adaptive, autonomous, and individual agent controller.

Keywords. ALife • Turing • self-organization • adaptation • learning • autonomy • multi-agent systems

1 Introduction

A few years ago, while attempting to better understand the P-Type machine algorithm, an opportunity to speak with the English mathematician, Dr. Jack Good, at Virginia Tech (University) in Blacksburg, Virginia presented itself at the suggestion of Dr. Jack Copeland from the University of Canterbury in New Zealand. Dr. Good had worked closely with Turing at Bletchley Park during World War II. Then, after the war, Dr. Good again worked with Turing but this time at Manchester University in England. It was thought that by talking with Dr. Good (someone who knew Turing well and had worked with him for many years) an understanding might be gained as to why this Turing algorithm had remained untested for so many decades. After reading the essay Dr. Good told this author that it seemed to him that the essay, *Intelligent Machinery* [16] was, "never peer-reviewed" (personal conversation with J. Good in Blacksburg, Virginia, 2004). This seemed reasonable if by nothing else but logical association and the following circumstances.

It is clear, Turing chose His Majesty's Stationery Office (HMSO) to become the first public printer of the *Intelligent Machinery* essay. And, by default, there was relatively little need for a peer-review process if Turing were merely reporting his work through an official government publisher. So, given these facts, it seemed less curious that the P-Type algorithm had been relatively unknown by the computer science community since almost the day of its first printing. Nevertheless, there was still a nagging question.

It had always seemed odd that within the *Intelligent Machinery* essay, Turing explicitly called for the instantiation of the P-Type algorithm in electronic computing hardware at a time in the future whenever, "some electronic machines are [were] in actual operation" (Section 11, p. 22). Moreover, Turing also commented that, "I made a start on the latter but found the work [building the paper and pencil machines by hand] altogether too laborious at present." Was this a case of Turing intentionally creating a contradiction by publishing an algorithm he wanted others to explore but doing so through a seemingly mundane government printing office?

We may never know the complete answer. But, what we do know is that implicit in the reporting of some Turing scholars [1,2,3,4], [18] the algorithm went largely unexploited for over fifty years. We also know that this invention predates all other machine intelligence work, in some cases by decades. Additionally we know that Turing himself demonstrated a paper and pencil instantiation of the algorithm complete with tests in his essay. Thus, whether it was because of its HMSO pedigree and its lack of critical review polish or some other reason, it was more than half a century after the algorithm was first printed (and it became available for general study) that anyone publicly reported the results of their efforts to instantiate the P-Type algorithm in electronic computing hardware.

The following pages contain a review of three qualitative replication experiments. Their contents will describe how three P-Type machines were instantiated and then demonstrated control of situated and embodied agents individually and in simulated small-group social settings. While one does not need to read the original Turing essay in order to understand the experiments described in this review, a comparison of the former with the latter may offer a more complete sense of the historical and subject matter context otherwise unavailable to this singular written forum. Additionally, a comparison of the original algorithm with that described here may assist the reader in adjudicating the fitness of the experimental method next described for its claims of replication. For reference, if one chooses to read the original work, then one should pay special attention to Sections 10 and 11 entitled "Experiments in Organizing Pleasure-Pain Systems" and "The P-Type Unorganized Machine," respectively.

2 Method & Results

Understanding how the P-Type algorithm was instantiated (the experimental method) and how it performed (the experimental result) is important for at least three reasons. First, this is a replication experiment. And, as a replication experiment, a close facsimile of the abstract functionality of the original Turing description will be required

in order to make any claim of satisfactory replication. Second, an operational instantiation of the algorithm is required in order to refute the stated hypotheses with an existence proof. Third, there are no known template-examples to which one can refer in the literature (outside of those describing these experiments) that can guide us in building and or evaluating a machine previously described only in sixty year old text. By comparison, even related but dissimilar Turing Unorganized Machines have recently received formal mathematical description [1].

One must also be reminded that the Turing essay was written in 1948. This was well before any standardized computer engineering vocabulary had been established and it was roughly coincident with the time when behavioral (behaviorism) psychology was just beginning to achieve its full dominance within the Social Sciences. One might be motivated to ask if Turing was influenced by, or at least was aware of, the latter because he appears to have anthropomorphized much of the operational characteristics of the essay machine. This comment regarding anthropomorphization is simply one of fact and not a claim that Turing had become a student of the Social Sciences. For example, when one reads in the essay the phrase, "it is intended the pain stimuli occur when the machine's behavior is wrong, pleasure stimuli when it is particularly right," (Section 10, p. 18) one can draw their own conclusion on this point.

In this review, modern computer engineering vocabulary will be used as much as possible. Additionally the words controller or engine will often be used as a synonym for the phrase machine-intelligence. Later in the review, the concept of a controller agency will be introduced when the controllers (the machine-intelligence) is associated with increasingly more sophisticated, situated-interaction experimentation. Also, when the word host appears it will usually serve as a surrogate phrase for the embodiment and/or the vehicle through which and with which a controller comes into interface with its environment. For example, the host embodiments described in these experiments are those real or simulated objects ("rodents" and "bugs" respectively) within which the Turing algorithm acted as a controller (or machine-intelligence). More exactly: The instantiated Turing engine was the source of all control and the hosts were the vehicles from within which the Turing engines engaged their various environments. Technically speaking these are not formal definitions but rather experimental ones. Finally, it is believed necessary that vocabulary from the domains of psychology and also animal ethology occasionally be used to explain some controller-host-environment functional interaction activities.

2.1 Method

The reason the machine is called a P-Type is that Turing designed it to learn (or acquire) adaptive behaviors as a consequence of receiving differential stimulus reinforcement (*Pleasure* or *Pain*) at its machine inputs. Turing explicitly proposed that this reinforcement might come from some external steering signal like a teacher. Intuitively, however, it seems reasonable to assume that any bi-valued exogenous signal should be able to re-direct the trajectory of the learning capability of the device. Using this assumption, it turns out, will allow us to instantiate the machine as a controller

and to embody it within a host situated in an artificial ecology. In fact, by expanding our intuition only slightly, we will discover that we can construct host embodiments controlled by more than one P-Type machine harnessed in a parallel-controller configuration. Moreover, we will see such instantiations can respond to an array of external steering signals. But, we are getting ahead of our evidence. Simply, Turing referred to this type of learning by means of differential reinforcement and adaptive steering as guided or "interference" learning (Sections 10 & 11). In the essay, he instantiated a machine using paper and pencil and he claimed he showed it could be taught. Today a psychologist would refer to this form of stimulus-response behavior modification [19] and the overall process as operant conditioning [13],[15].

General. The P-Type machines instantiated for this review were tightly coupled, three-piece, computing machines composed of a Behavior Library, a Transducer, and a Finite State Machine (FSM). The Behavior Library Turing wrote about was just a set of random numbers over {0, 1} each one being a possible output behavior. The Transducer is our word for the component whose function was described by Turing (generally) as a table, memory, and or storage device. He recommended using a "tape" in larger systems and he referenced this memory component using figures and descriptive text in the essay. The FSM instantiated here provided the logic and automatic functions needed to manipulate the transition table entries in the Transducer. Additionally, per the Turing essay, there were no hidden memory units used in any experiment, nor were "next situation" (next state) selections made by sense stimuli. The next state was always chosen by *Pleasure-Pain* steering on the machine input and then was directed through the Transducer transition table.

Fig. 1 is a block diagram illustration depicting the internal functions of a P-Type machine as they were instantiated here. One or more of these engines was instantiated in each of the experiments reviewed.

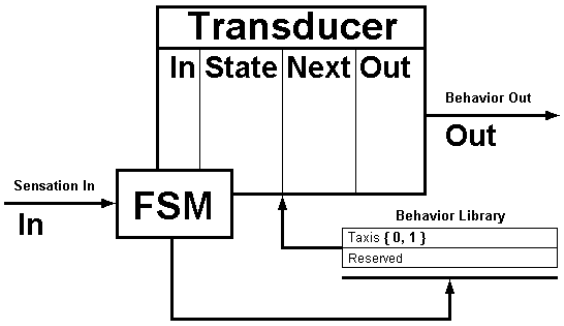
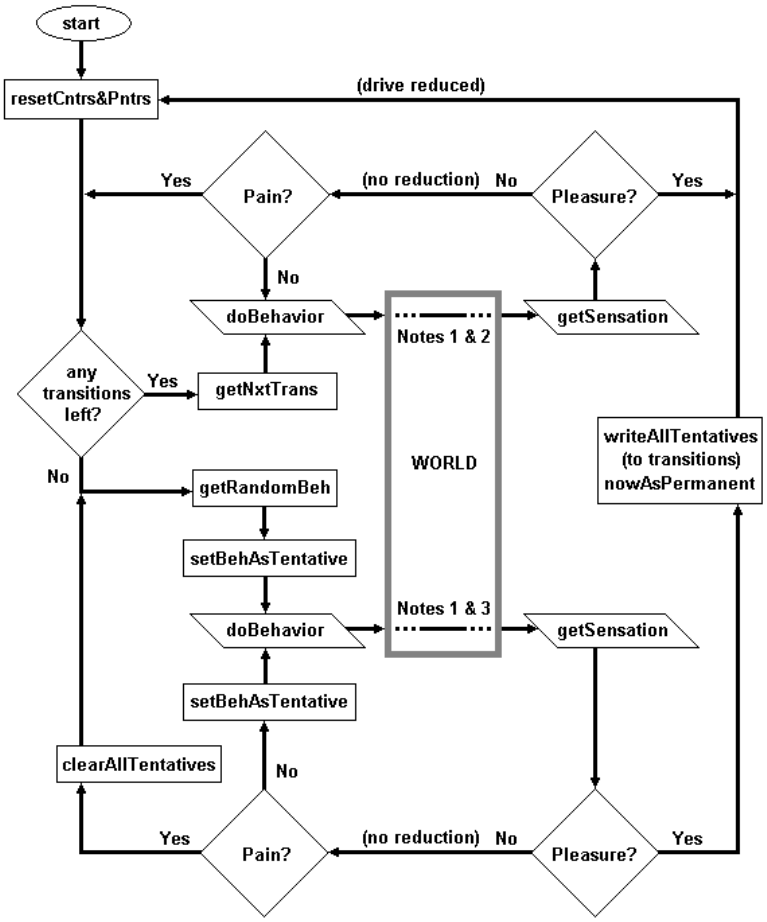


Fig. 1. The kernel engine at the heart of Turing prototype P-Type had few internal components.

Consider the left side of **Fig. 1**. The FSM receives a steering signal, as *Pleasure* or *Pain* affect sensation, as its input. The FSM then directs the behavior of the Transducer and may possibly negotiate with the Behavior Library in the lower right. Eventually, a selected behavior will emerge into the world from the Transducer based on

the Transducer transition table contents and the aforementioned FSM-Behavior Library negotiation. Once machine execution has started, it will continue to execute until halted by external means. By design, the primary drive of the machine is one involving the equilibration and acquisition of adaptive Transducer data structures. Other drives can be superimposed over the primary. A flow chart shown in **Fig. 2** may help explain the algorithm pictorially.



Notes:

1. There is no implied concurrency. Data passes straight through the world. See upper and the lower dotted lines.
2. The upper path is characterized as a path driven by the transitions in the Transducer.
3. The lower path is characterized as a path driven by random, trial-and-error learning.

Fig. 2. P-Type machine functional flow-chart

Specific. To start (or re-start) a full cycle a P-Type looks first into its memory (Transducer transition table) for any existing transitions. Turing started at the first entry in the list and sequentially executed every one until there were no more transitionable outputs left. If an output behavior resulted in a *Pleasurable* result (that is, the consequence of the behavior was reflected back from the world and the initial drive was reduced) then the full P-Type cycle was restarted. If an output behavior resulted in a *Painful* reflection from the world, the machine transitioned to the next table entry. If an output behavior resulted in a non-*Painful* result, the behavior (the transition) was simply repeated. When there were no stored transitions yet untested, his machine consulted the Behavior Library.

Turing pre-generated a pool of 0/1 random numbers (as in our Behavior Library) and made them ready for usage. In the instantiations reviewed here, the FSM consulted the Behavior Library for randomly generated zero or one behavior outputs (and more abstract behaviors like an orienting Taxis response) to be tested in the world.

At this point, it is understood no untested transitions remain in the Transducer. So, a pattern of random trials and behavior evaluation cycles can begin: a random behavior is taken from the Behavior Library and is output to the world. Remember, the machine is attempting to reduce its primary drive. When a *Pleasurable* result is encountered the "tentative" behaviors are transferred to the Transducer transition table as "permanent" entries and the entire (full) P-Type cycle is restarted. When *Pain* is encountered, all heretofore "tentative" behavior transitions are erased from scratch-pad memory and a new sequence of random trials will be restarted. If it is determined that the random behavior does not result in *Pleasure* but it also does not make things worse (it does not cause *Pain*) then that behavior is considered to be a "tentative" candidate for learning and it will be written to a scratch-pad memory. A sequence of such "tentative" behaviors can be tried and stored in scratch-pad memory any number of times. If a non-*Painful* result is encountered, the behavior is simply repeated. The goal is always to reduce the current drive.

The pattern of trying new behavior outputs continues until either the drive is reduced (*Pleasure* is encountered and the new sequence can be added to the bottom of any existing list of transitions in the Transducer), death occurs (execution ceases), or *Pain* is encountered and the entire random trails and behavior evaluation cycle can be restarted and all "tentative" results erased. An adaptive behavior sequence can be lengthy. Since [5], psychologists have come to refer to similar cognitive behaviors that involve sequential, patterned outcomes as serial learning.

The cycle of trial, evaluate, store/forget is the basis of the P-Type algorithm as it was designed by Turing. It appears to make the automata a serial "penultimate" learner. After any successful drive-reducing cycle, the machine will always return to the first transition in its Transducer memory and will be made ready to repeat the entire process.

2.2 Results

The first two experiments were complementary instantiations of the machine depicted in **Fig. 1.** The first experiment was a situated and embodied single agent simulation

based purely in software. The second experiment was a hybrid simulation constructed using an untethered mobile robot and a controller embedded in reconfigurable hardware, i.e., Field Programmable Gate Array (FPGA) technology. In each of those experiments, the P-Type served as an adaptive controller (an "agency") embodied by a mouse (an icon-mouse and a robotic rodent, respectively) situated in a simulated Hull enclosure similar to those developed by [6].

The third experiment involved a Multi-agent Systems (MAS) social simulation. This experiment explored the further extensibility of the Turing algorithm by harnessing the machine in a parallel configuration constructed of four independent P-Type machines coupled together as a single control "agency." That "agency," that controller, was then embodied in each respective host in the MAS experiment.

Replication Experiment One. The first experiment involved a simple icon-mouse embodied as a virtual agent situated in a simulated, 8-corner Hull enclosure. The experiment, depicted in **Fig. 3**, shows a screen-capture of the experiment in progress. **Fig. 4** shows some quantitative result products derived from that experiment (reproduced by permission [11]).

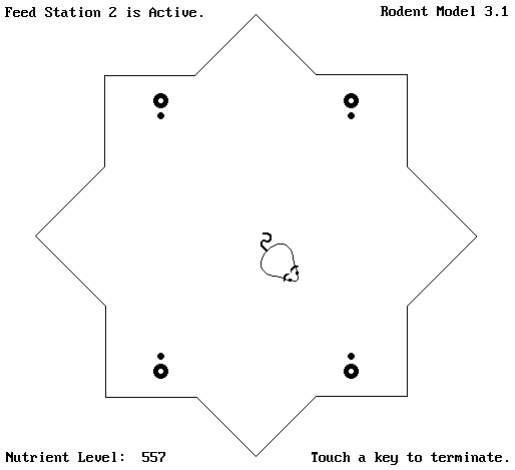


Fig. 3. The icon-mouse was in a simulated Hull enclosure. Here it is seen from overhead.

In this experiment a single P-Type machine was instantiated and embodied. The machine received a single steering signal input derived from host interoceptive and exteroceptive sensation. Those sensations included a simulated sense of satiety (a sense of being hungry, full, or in-between), a simulated haptic sense (the icon mouse could sense when it touched the walls of the enclosure), and a simulated olfactory sense (the icon mouse could sense a distance-modulated scent intensity coming from its food source). The goal of the experiment was (as it was for experiment number two) to see if a single P-Type machine, serving as a controller, could adapt to its surroundings using only a constrained set of behaviors from a species-specific ethology, i.e., a strong hunger drive to locate a food source using simulated olfaction, and a

drive to survive by eating a bit of simulated food while not colliding with the walls of a Hull enclosure. The code used in experiment one is available for download[1].

The test required the host, an icon-mouse, to find and take its replenishment from a feed station embedded in the floor of the Hull enclosure. At any given time only one of the four feed stations (the circles in the corners of the Hull in enclosure shown in **Fig. 3**) was active. Using its simulated haptic sense and its simulated olfactory sense, the icon-mouse had to follow a simulated olfactory gradient emanating from the food and then place the tip of its nose over the feed station in order to be fed. Between feedings, the icon-mouse was free to roam (at random) about the enclosure. During this "roaming" period the P-Type engine was disengaged and the icon-mouse could move about the enclosure without restraints other than the enclosure walls. No learning took place during "play time." The "mice" were blind and could not escape the enclosure. If a "mouse" could not find the food (its fuel source) when it became "hungry" and its controller was re-engaged then, it would expire.

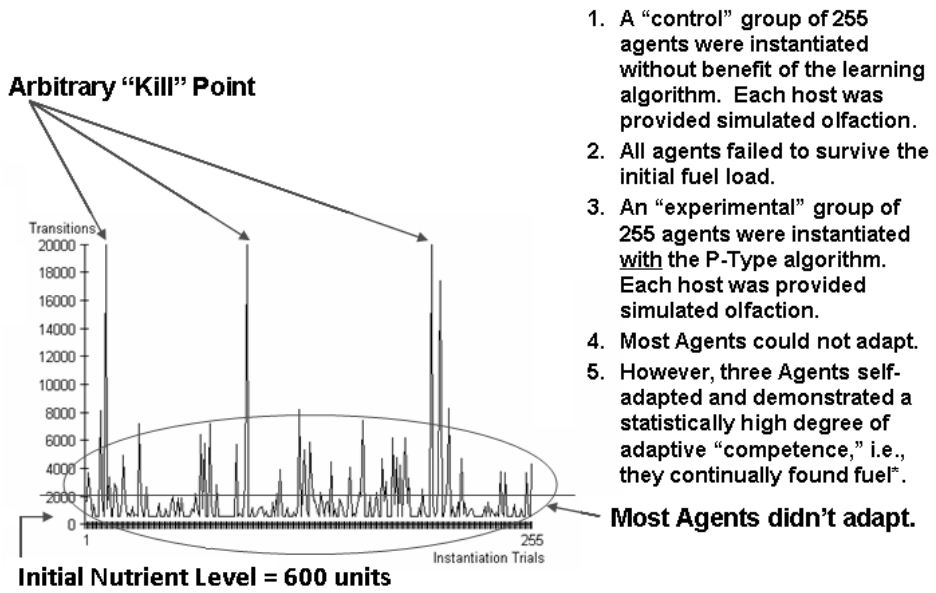


Fig. 4. Most agents perished but three learned how to survive. Reprinted by permission.

The graphic in **Fig. 4** reports the results of two, 255-sample, mass trials using the P-Type as a situated and embodied controller of icon-mice. The only difference between the two trials reported was that, in one case, the P-Type engine was turned-off permanently and, in the other, it was turned-on and completely available to the host. Note: The result when the engine was turned off is not shown here since every "mouse" agent in that experiment failed to survive its initial fuel (food) load, i.e., they all "died". Also, based on the results of the trials that did use the P-Type engine, it is obvious that convergence on a survivable solution was very slow even for "mice"

[1] http://css.gmu.edu/papers/ALifeAAIAgentControl.zip

having access to P-Type learning. Theirs was a pure discovery-learning process (without heuristics) in an unknown environment. Clearly, search by random trial-and-error learning was not an effective survival strategy. A better method will be suggested later in experiment three. But, on the other hand, the "mice" in this experiment were capable of being taught using the method of interference learning suggested by Turing. That is, during the experiment, special "training sessions" wherein spatial pre-positioning of the "mice" within the enclosure took place. Consistent with simple methods of behavior modification that use operant conditioning (mentioned earlier), the "mice" were taught how to find replenishment. That is they were taught how to avoid contacting the walls, home-in on the scent of "food" coming from a feed station, pursue it, and survive. Here "mice" learned quickly and survived well.

Another remarkable thing about the results of this experiment was that during the post-mortem analysis of the Transducer memories in the long-surviving random-search "mice," it was discovered that those "mice" that survived longest and were "sacrificed" had all acquired a similar set of survival strategies. These surviving "mice" eschewed wandering off and roaming about their enclosure when not feeding. Instead these surviving "mice" independently learned (or evolved) behaviors involving strategies to stay near the food source during "play time" and not to go far away from the food source, get lost, and ultimately expire. All "mice" were "sacrificed" when their total execution cycle count reached 20,000.

Replication Experiment Two. The second experiment instantiated a P-Type controller in a custom Field Programmable Gate Array (FPGA) logic circuit. The controller was effectively embodied within (exerted positive control over) a host: a rodent robot. The robot is shown in **Fig. 5**.

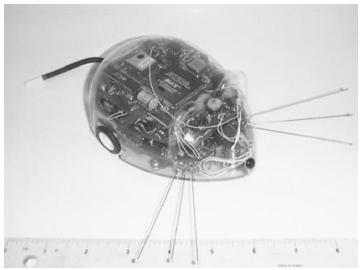


Fig. 5. A robotic rodent was the embodiment tested in a Hull enclosure in Experiment Two.

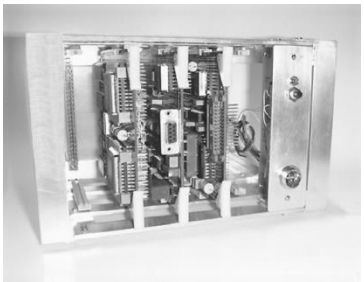


Fig. 6. A P-Type controller was on the circuit card to the left in the card cage.

The P-Type engine was physically located in a small card cage and communicated with the host embodiment via radio frequency/infra-red (RF/IR) telemetry. The controller and various systems support components were housed in the card cage shown in **Fig. 6**. The use of remote telemetry had *only* to do with concerns completely independent of the instantiation of the P-Type. Concerns involved the use of the high-current-draw FPGA circuits and static random access memory used for the Transducer component. The card cage handled these concerns easily while freeing up the host body for tasks associated with host embodiment as a mobile robot. The electrical schematics of all circuits and mechanical diagrams of the robot and the Hull enclosure are available for download[2].

During the experiment, the host (the robotic rodent) was situated in a physical Hull enclosure. The Hull enclosure was about a meter across and the robotic rodent several centimeters in length. In this experiment, the instantiation of the P-Type machine was similar to the one illustrated in Fig. 1 but differed in that the algorithm was instantiated using Very High Speed Integrated Circuit (VHSIC) Hardware Description Language (VHDL) and an Altera FPGA. The VHDL code may be available by request[3].

As with experiment one, experiment two attempted to see if the P-Type machine could adapt to its surroundings using only a constrained set of behaviors from a species-specific ethology, i.e., a strong hunger drive to locate a food source using simulated olfaction, and a drive to survive by eating a bit of simulated food while not colliding with the walls of the Hull enclosure. The P-Type machine was mounted on the circuit card located to the left in the card cage shown in **Fig. 6**.

Similar to experiment number one, the "rodent" had to find its own food (find a battery charging station in the enclosure) and keep its 9-volt battery charged. Using its simulated haptic sense and its simulated olfactory sense, the "rodent" had to follow a simulated olfactory food gradient (simulated by a 100 KHz RF beacon) and then place the tip of its nose over the feed station in order to be fed. Between feedings, the "rodent" was free to roam (at random) about the enclosure. The "rodent" was blind. If the "rodent" could not find the food when it became "hungry" (when its battery neared full discharge) it would expire. That is, the "rodent" would expire when its on-board 9-volt Ni-Cad battery failed.

The purpose of this work was to test the instantiability, extensibility and transportability of the basic algorithm to other platforms. No other results for experiment two are reported here.

Replication Experiment Three. In this experiment, P-Type controllers were constructed so as to be explicitly ganged together in parallel and to cooperate with each other as a single cohesive control unit. The goal was to create an extensible controller having n-independent primary drives (n=4 in this experiment) that could cooperate with each other and provide a single control "agency" (a single, independent, machine-intelligence) capable of operating an embodied host in a highly social environ-

[2] http://www.maelzel.com/Programs/Circuits/Daryl/Circuits.htm
[3] http://www.maelzel.com/Contact/Moreinfo.php

ment. The four primary drives for these controllers were selected from a list of the lower-level physiological and higher-order socio-cognitive drives associated with the Maslow Hierarchy of Needs [8]. Thus, each four-way "agency" controlled one respective host icon-bug. Said another way, each "bug" had its own four-way control system. **Fig. 7** shows the icon hosts in their simulated maze at program start-up. In this ALife experiment, the large blue circles were predatory "bugs" and the smaller red "bugs" were the food (prey) species. (In grayscale red=dark gray, blue=light gray.) At program start, the investigator could select how many predator and or prey icon hosts would inhabit the enclosed maze. During the experimental trial shown here, a total of 30 "bugs" (mixed predators and prey) were simulated simultaneously.

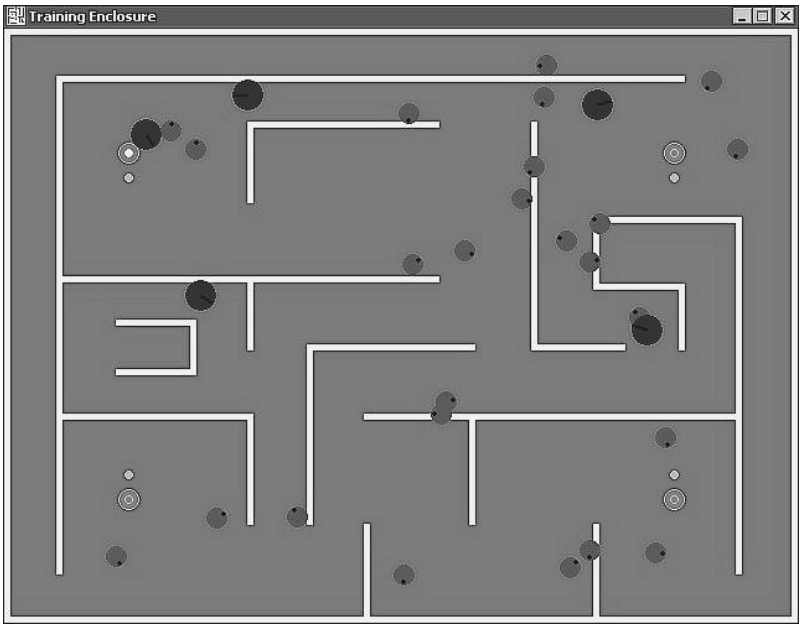


Fig. 7. Seen here from overhead at program start, the icon "bug" hosts were placed randomly. The white maze partitions were only permeable to simulated odor penetration and thus the bugs quickly fell to either side of the partitions once instantiated and the program began. The four gray circles in the floor of the training maze were prey feed-stations. These could emit a simulated odor (only one was active at a time). Predators and prey emitted unique scents of controllable intensity. The feed-station in the upper left corner is active. The icon bugs enjoyed a haptic modality, a simulated olfactory sense, a satiety sense, but no vision.

In operation, the icon hosts "were free to move about the maze until they were eaten, their energy levels were depleted, or they adapted, found available food, avoided their predators, and survived" [12]. An interesting result appeared during "bug" post-mortem Transducer table analysis. The transition tables revealed that emergent

sequences of lengthy concatenations of Taxis and Kinesis[4] motor behaviors were produced. These sequences were adaptive behavior patterns constructed from shorter movements and made navigating the long corridors of the maze possible. As was mentioned previously, it turned out this was consistent with the prediction made by Turing that the P-Type might incorporate old routines into new ([16] Section 11, p. 22) and it may suggest a rudimentary form of stimulus generalization[5] was taking place. Finally, the icon hosts had, differences in color, size, and markings, predators were distinguishable from prey as can be seen in the screen capture of the trial shown above.

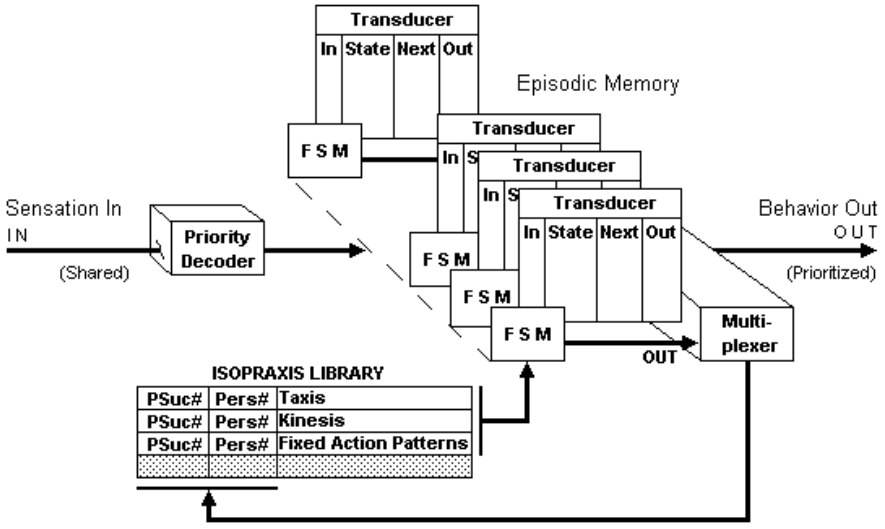


Fig. 8. In this experiment four P-Type machines were harnessed together to create a single control "agency."

Fig. 8 is an illustration depicting the internal functions of the P-Type machines instantiated for the third experiment. A priority encoder provided the initial input steering and machine activation, per an (abbreviated) Maslow Hierarchy, to each of four parallel harnessed P-Type machine sensation inputs. A multiplexer encoded and steered the output behavior signals from the four machines into a single output stream. Input steering signal shaping used in this experiment was similar to that used

[4] From the study of animal ethology [9] and biology; a tropism (a taxis) is a concise, usually small, behavioral responses made with regard to a stimulus. Larger but less directed collections of motor behaviors in animals are kinesis. The latter are also often associated with intensity of movement but not necessarily direction.

[5] An early description of stimulus generalization appears in [17]. Stimulus generalization is that observable behavior pattern in a conditioned subject that occurs when a neutral, unconditioned stimulus evokes a previously conditioned response. When the conditioned response is generalized, appearing across more than one previously unconditioned (neutral) stimulus, the behavior syndrome is referred to as stimulus generalization.

in the first two experiments. Emulated sensations included a simulated haptic sense (the "bugs" could sense when they touched the walls of the maze), a simulated olfactory sense (the "bugs" could sense a distance-modulated scent intensity coming from their food sources, i.e., prey for predators and floor-mounted feed-stations for prey "bugs"), and a simulated sense of satiety (a sense of being hungry, full, or in-between).

This experiment explored not only the feasibility of a four-way P-Type "agency" but it also introduced a more complex Behavior Library. Based on a paradigm of species-specific behavior patterns characterized by [10], the Behavior Library became an Isopraxis Library and included host-appropriate Taxis, Kinesis, and Fixed Action Patterns as output behaviors. Additionally, numeric scores for the probability of behavior success (PSucc#) and persistence of utility (Pers#) were updated dynamically for each behavior in the library during runtime. The tags were used by the FSM to aid in its selection of individual behaviors during trial, evaluate, store/forget cycles. This stochastic steering scheme appeared to eliminate some of the learning convergence problems evident in experiment one caused by the inherent noise in a random trial-and-error discovery learning process. For comparison, one may want to consider the application and theory presented in [14].

3 Discussion

In the experiments reviewed here, the Transducer was a memory device that held the emergent configurations of the machine, i.e., the transitions. (Turing referred to his transition table data as the "configurations" of the machine.) In the experiments reviewed here, our instantiations also had ample memory available for scratch-pad functions during the P-Type "tentative" behavior evaluation cycles.

Turing referred to the possibility of creating a "dialing system" to manipulate system memory in systems where large amounts of memory were available. In the experiments reviewed here, a "dialing system" was incorporated into the FSM. That "dialing system" provided the logic to steer the selection of the addresses (rows) in the Transducer transition table. Additionally, our FSM contained logic capable of reading, writing, and erasing any of the transition table entries, and for manipulating the scratch-pad memory.

At start time, a P-Type memory (our Transducer transition table) was normally completely empty[6]. The final size of the Transducer transition table was constrained only by the availability of transition table memory and how (temporally) long the host device was allowed to operate. After a machine had operated for some time and (more importantly) if the machine had demonstrated any adaptive behaviors *in situ* then the transition table would have started to fill with transition information. At this point it could be said the machine had started to become "organized" or to learn and to adapt

[6] Technically, the table did not have to be empty. It was quite possible to transfer the acquired transitions of one machine to another and to start the latter machine pre-loaded with "knowledge" of the former. This type of transfer resembles the end result of Lamarkian evolution and was used successfully to support accelerated, inter-agent "shared" learning.

to its environment. In later decades researchers working in machine intelligence and reinforcement learning came to rely on a process that, in abstract, is actually somewhat similar to the P-type algorithm. Those researchers came to refer to the process as the accumulation of optimal policies. Consider, for example, the works in Reinforcement Learning described by [7].

In each of the experiments described here the machines were situated and embodied actors in unique environments. They were all subjected to "interference" stimulus during every cybernetic learning cycle. As the machines read their inputs they were forced to select output behaviors from individual, self-organizing behavior libraries, or in the case of the icon-mice they could engage in random "play." In turn, they received stimulus reinforcement (positive and negative from their environments and or their teachers) as a consequence of those previous output behavior choices.

While developing the very first P-Types for the first experiment it was discovered that the key to using the P-Type was to map it onto the targeted host's real physical, or simulated, "physiological" biomimetics and or "cognito-social" behavioral needs. This had to be done from the perspective of identifying (in advance) whatever the same or similarly functioning real mechanical and or operational properties existed in the targeted real host and needed to be controlled and or optimized by the P-Type. For example, when our final hosts had a need (a drive) to reduce hunger, or a need (drive) to avoid somatosensory contact with walls, or a need (drive) to follow a scent gradient toward food, or a need (drive) to avoid a scent gradient coming from a predator, etc., all of these had to be aligned with (or mapped onto) one or more P-Type engines in order to use the equilibratory properties intrinsic to the algorithm. Understating this somewhat, identifying this design requirement had real consequences for the experiments. Once it was identified and its requirements met, it meant a P-Type controller could be embodied easily and that it would be able to steer its host in the world. Before that, instantiating the algorithm was just a paper and pencil exercise. Now, however, the experiments have suggested that a controller based on the algorithm may be able to learn from either random discovery or teacher-delivered interference training. All just like Turing suggested.

4 Summary

In his essay, *Intelligent Machinery*, Alan Turing described a computing machine whose purpose was to learn by doing. Once embodied it became a cybernetic engine capable of recording the output sequence of its own situated behavior. Moreover, it did so while demonstrating that it could steer itself to reduce primitive drives and achieve adaptive success. Turing wanted the machine to receive its sensory feedback and external influences from a teacher. Once embodied it became possible to shape even complicated interoceptive and exteroceptive (sensory) feedback arriving at the machine turning them into simple *Pleasure* and *Pain* steering signals. In particular, Turing told us he built a paper and pencil machine and he showed us how his machine output behaviors that could be modified by its teacher. He challenged us to test his

algorithm, to expand it, and to try it. For whatever reason it has taken over half a century before anyone has picked up his challenge and reported their results.

In truth, the body of work in machine intelligence and machine learning has long overtaken this algorithm in scope, efficiency, and maybe even utility. That said, this author was tasked with only a simple challenge: review three machine intelligence (replication) experiments wherein P-Type Unorganized Machines were physically instantiated based on a best effort to understand the description of the machine in that original Turing essay. This last time, then, let us restate and consider the three original hypotheses so as to better frame a possible discussion of why this algorithm lay dormant for over half a century. The first hypothesis was that Turing did not describe the algorithm in sufficient detail so as to make it possible to instantiate. The second hypothesis was that if it were the case that it could be instantiated then, the algorithm would not operate as Turing had claimed. Finally, the third hypothesis was that both the first and the second were true. The foregoing three experimental embodiments of the algorithm seem to suggest that a constructive proof actually exists for the instantiability of the algorithm and for its operation. Thus, all of the hypotheses appear to be qualitatively false.

References

1. Burgin, M., Eberbach, E.: Evolutionary Turing in the Context of Evolutionary Machines. arXiv preprint arXiv:1304.3762. Cornell University Library (2013)
2. Copeland, B.J., Proudfoot, D.: On Alan Turing's Anticipation of Connectionism. In: Synthese, vol. 108(3), pp. 361–377. Kluwer Academic Publishers, Netherlands (1996)
3. Copeland, B.J., Proudfoot, D.: The Computer, Artificial Intelligence, and the Turing Test. In: Teuscher, C. (ed.) Alan Turing: Life and Legacy of a Great Thinker, pp. 317-351. Springer, Heidelberg (2004)
4. Eberbach, E., Goldin, D., Wegner, P.: Turing's Ideas and Models. In: Teuscher, C. (ed.) Alan Turing: Life and Legacy of a Great Thinker, pp. 159-196. Springer, Heidelberg (2004)
5. Ebbinghaus, H.: Memory: A contribution to experimental psychology. No. 3. Teachers College. Columbia University (1913)
6. Hull, C.L.: Principles of Behavior. Appleton-Century-Crofts, New York (1943)
7. Kaelbling, L., Littman, M., Moore, A.: Reinforcement Learning: A Survey. In: Journal of Artificial Intelligence Research 4, pp. 237-285. AI Access Foundation and Morgan Kaufmann (1996)
8. Maslow, A.: A Theory of Human Motivation. In: Psychological Review, vol. 50, pp. 370-396 (1943)
9. McFarland, D.D.: Animal Behaviour: Psychobiology. Ethology and Evolution. Longman 3 (1999)
10. McLean, P.D.: A Triangular Brief of the Evolution of Brain and Law. In: Gruter, M., Bohannan, P. (eds.) Law, Biology and Culture: The Evolution of Law. The Berkeley Electronic Press, Berkeley California (1982)

11. Rouly, O.C.: Cybernetic Intelligence: A Return to Complex Qualitative Feedback Theory. Unpublished thesis, New Mexico State University, Las Cruces (2000)
12. Rouly, O.C.: Learning Automata and Need-Based Drive Reduction. In: Ha, Q.P., Kwok, N.M. (eds.) Proceedings of the 8th International Conference on Intelligent Technologies, pp. 310-312. University of Technology Sydney, Sydney Australia (2007)
13. Skinner, B.F.: The behavior of organisms: An experimental analysis. (1938)
14. Sutton, R.S., and Barto, A.G.: Reinforcement learning. In: Journal of Cognitive Neuroscience 11.1, pp. 126-134 (1999)
15. Thorndike, E.L.: Animal intelligence: Experimental studies. Macmillan, New York (1911)
16. Turing, A.M.: Intelligent Machinery. In: National Physical Laboratory Report. HMSO, London (1948)
17. Watson, J.B. and Rayner, R.: Conditioned emotional reactions: Journal of Experimental Psychology. 3.1 pp. 1-14. (1920)
18. Webster, C.S.: Alan Turing's unorganized machines and artificial neural networks: his remarkable early work and future possibilities. In: Evolutionary Intelligence, vol. 5(1), pp.35-43. Springer-Verlag (2012)
19. Zirpoli, T.J. and Melloy, K.J.: Behavior management: Applications for teachers and parents. Merrill (1997)