Generating the Gopher's Grounds: Form, Function, Order, and Alignment

Jiayi Zhao[2][000-0003-2497-120X]*, Anshul Kamath[1][0000-0002-1641-4430]*, Nick Grisanti[1][0000-0002-6987-9194], and George D. Montañez[1][0000-0002-1333-4611]

[1] AMISTAD Lab, Dept. of Computer Science, Harvey Mudd College, Claremont, CA, USA
{akamath, ngrisanti, gmontanez}@hmc.edu
[2] Department of Computer Science, Pomona College, Claremont, CA, USA
jzae2019@mymail.pomona.edu

Abstract. Previous work has shown that artificial agents with the ability to discern function from structure (intention perception) in simple combinatorial machines possess a survival advantage over those that cannot. We seek to examine the strength of the relationship between structure and function in these cases. To do so, we use genetic algorithms to generate simple combinatorial machines (in this case, traps for artificial gophers). Specifically, we generate traps both with and without structure and function, and examine the correlation between trap coherence and lethality, the capacity of genetic algorithms to generate lethal and coherent traps, and the information resources necessary for genetic algorithms to create traps with specified traits. We then use the traps generated by the genetic algorithms to see if artificial agents with intention perception still possess a survival advantage over those that do not. Our findings are two-fold. First, we find that coherence (structure) is much harder to achieve than lethality (function) and that optimizing for one does not beget the other. Second, we find that agents with intention perception do not possess strong survival advantages when faced with traps generated by a genetic algorithm.

Keywords: Structure and Function · Agents· Genetic Algorithms.

1 INTRODUCTION

Imagine being placed in an unknown environment where you expect to encounter both treasures and dangers. Here you must rely on your instincts, observations, and ability to identify potential dangers. In nature, both humans and animals are equipped with the ability to perceive signals of risk, weigh their safety against possible rewards, and decide what actions to take; risk assessment is an essential aspect of survival. Recent work has revealed the potential advantage of *intention perception*, a particular kind of risk assessment ability to detect the intention of other agents, in artificial agents under a variety of adversarial situations [14,15,10].

In one such study, Hom et al. created a framework of simulated gopher agents surviving against a series of simple combinatoric traps, and they found that artificial gophers possessing the ability to perceive the environment as intentionally designed (to

*denotes equal contribution.

harm the gopher) had significantly higher survival rates than gophers without intention perception [10]. Specifically, they assumed that traps intentionally constructed by humans were more likely to be "coherent" (a property of trap structure defined in Section 3.1) than unintended traps generated uniformly at random, and used this coherence as a indicator of designed traps.

There are two limitations with Hom et al.'s work: first, their design of an intention perception algorithm implicitly assumed a correlation between trap structure and its functionality, a relationship that requires further evidence to affirm; second, their experimental framework only investigated the survival of gophers for two types of traps (human-designed traps and traps generated uniformly at random). Whether their results hold more broadly, for different trap-generating processes, remains a relevant question.

In the present study, we examine the correlation between trap structure and trap functionality in a more general context. The relationship between structure and function is central, as machines can generally be defined according to their structure and function. While Hom et al. proposed that trap coherence implies intentional design and intentional design implied functional lethality (so that coherence implied lethality), we test this chain of inference by looking at trap coherence and trap lethality for machines produced by *genetic algorithms*. Genetic algorithms are a metaheuristic search method modeled on natural selection, consisting of biologically-inspired operators like selection, crossover (recombination), and mutation [7,16,18]. Genetic algorithms are known for their ability to generate solutions to optimization and search problems defined on complex, high-dimensional discrete spaces, and have become a popular tool for solving *structural optimization* problems, which is the automated synthesis of mechanical components based on structural considerations [3,21]. Generating traps with desirable characteristics (e.g., coherence and functionality) can be viewed as a simplified structural optimization problem on a high-dimensional discrete space. By simply changing fitness functions, one can obtain traps optimized for a variety of traits. Thus, we employ genetic algorithms as our primary trap-generation mechanism.

A second goal of this paper is to continue exploring the influence of intention perception on survival of artificial gophers, possibly drawing conclusions on intention perception more generally. We investigate whether intention perception can still provide gophers with survival advantages when gophers are faced with a variety of traps generated by genetic algorithms. In particular, we generate traps with structure (coherence) and with no function (no lethality), traps with function but no structure, and traps with both structure and function. We test whether the intention-perception algorithm of Hom et al. is able to distinguish such traps from human-designed traps, and how this affects the survival rates of gophers.

We find that while lethality requires some baseline level of coherence, the relationship is weak and the correlation between coherence and lethality is almost non-existent for traps generated by genetic algorithms. As such, the survival advantages of intention perception observed by Hom et al. no longer hold in this more general case. We also find that producing function (lethality) is much easier than producing coherence for trap-generating genetic algorithms. Lastly, we observe that local structure and clustering within fitness functions (*order*) is not sufficient to guarantee genetic algorithm success; biasing *alignment* with the target set is also necessary.

2 RELATED WORK

One of the focuses of the present work is the correlation between trap coherence and trap structure, namely, the relationship between structure and function of simple, combinatoric machines. Given the general importance of the relationship between structure and function, there has been a wide range of research on this topic, with objects ranging from organisms to engineering products.

Weibel, for example, posited that all functions depend on structural design in biological organisms, specifically for the morphometric characteristics of the organs. He proposed a *theory of symmorphosis* and attempted to quantify the relationship between structure and function in different organ systems [22]. Bock and Wahlert similarly approached the relationship from an adaptationist perspective, suggesting that structure and function "constitute the two inseparable dimensions of biological features when considering morphology and evolutionary biology" and "must always be considered together" [2].

In contrast, Gero and Kannengiesser argued that no direct connection exists between structure and function for human-designed objects [6]. Instead, they proposed the *Function-Behaviour-Structure* ontology which asserts that for any object, function is "ascribed to" behavior, and behavior is "derived from" structure. Though structure and function may affect each other, they are not directly connected.

Building on a presumed correlation between structure and function, the notion of *intention perception* was introduced by Hom et al. [10] for a kind of risk assessment by artificial agents. More broadly, there has been rich array of work on risk assessment that may also provide insights to intention perception specifically. Vorhees and Williams studied rodents' spatial learning and memory as they tried to maneuver safely through environments [20]. They argued that rodent survival depends on the ability to learn and remember locations, and this ability relies on two systems: *allocentric* navigation that uses cues outside the organism and *egocentric* navigation that uses internal cues. In our work, the intention perception of simulated gophers relies only on allocentric navigation, as decisions are made based on observations of the environment.

As mentioned in Section 1, our choice of genetic algorithms is motivated by their ability to produce high-quality solutions to a variety of search and optimization problems [7,16,18]. However, traditional genetic algorithms face several difficulties. One such difficulty is uncertainty—fitness functions are often noisy or approximated, environmental conditions change dynamically, and optimal solutions may change over time [13,19,1]. A variety of techniques have been developed to combat such problems [11].

2.1 Relation to Kamath et al.

This paper is an expanded supplement to Kamath et al. [12], adding additional experiments to the work done there. In the present manuscript we investigate how changing the trap generation process affects the intention perception algorithm and the survival of gopher agents, along with revisiting the work done in the original paper.

3 METHODS

Our goals are to investigate the link between structure and function in simple combinatorial machines, and to explore how an agent's survival is impacted by intention perception under more complicated scenarios. We adopt the "trap-gopher" framework of Hom et al., in which simulated gopher agents analyze a series of combinatorial traps containing food, and decide whether or not to enter the traps using their coherence-based intention perception algorithm [10]. Extending this framework, we introduce genetic algorithms that allow us to generate traps with different desirable traits, produced by a variety of fitness functions.

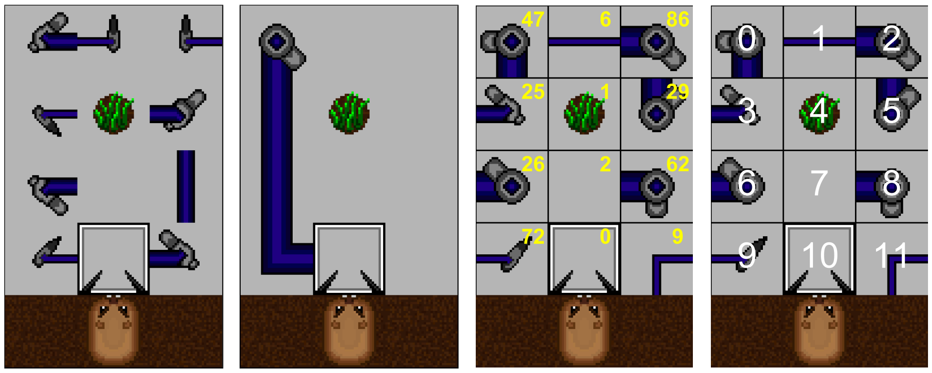
3.1 TRAPS

As mentioned, our trap framework is taken from Hom et al. [10]. Under this framework, each trap is a combinatorial machine embodying both structure and function.

TRAP STRUCTURE A trap's structure is simple; each trap is designed as a 4×3 grid that contains 12 tiles. There are three fixed tiles for all traps: one in the middle of bottom row that acts as the "door," allowing gophers to enter and triggering the trap; a fixed blank "floor" tile directly above the "door" that is traversible; and a "food" tile, enticing the gopher to enter the trap. Besides the three fixed tiles, each remaining tile can be either: a laser gun that we call an **arrow** tile; a blank **floor** tile; or a **wire** tile meant to propagate pulses from the door to arrow tiles. The arrows and wires have various rotations and thicknesses, and accounting for all possible variations, there are a total of 91 possibilities each of these 9 tiles can take, with details of tile variations given in the Appendix of Hom et al. [10]. Hence, there are a total of $91^9 \approx 4.28 \times 10^{17}$ possible traps in this framework. We let $\mathcal{X}$ denote the set of all valid traps, with $|\mathcal{X}| \approx 4.28 \times 10^{17}$. Examples traps are shown in Figure 1.

To quantify the structure of a given trap, we define the *coherence* of the trap, in agreement with Hom et al. [10], to represent how "connected" a given trap is. First, we say that a *coherent connection* exists between two non-empty (wire or arrow) tiles if: (1) the thicknesses of the two elements match, and (2) the two elements share an endpoint (i.e., the rotation of the elements align). The *coherence* of a trap is then defined as the number of coherent connections per non-empty (wire or arrow) tile.

TRAP FUNCTION *Functional* traps have at least one arrow properly connected to the sensing door. That is, functional traps have an arrow directly connected to the door or connected to the door through a series of wires with matching orientations and thicknesses, for which the door will send a "pulse" to the arrow after sensing an entering gopher. Once an arrow receives the pulse it will fire a laser, and if the laser hits the gopher, it may kill the gopher with certain probability (decided by the thickness of the arrow). We associate a larger probability of killing gophers with thicker arrows: in particular, the probabilities of killing a gopher on a successful hit with a wide, normal, skinny arrow are $P_{k,w} = 0.45$, $P_{k,n} = 0.3$, and $P_{k,s} = 0.15$ respectively.



(a) Example of a functional trap created by the genetic algorithm. (b) Example of a designed trap created by Hom et al. (c) Encoding for the twelve trap tiles. (d) Indexing the twelve trap tiles.

Fig. 1: Example traps, trap encoding, and trap location indexing. Figures 1-7 are reproduced from [12].

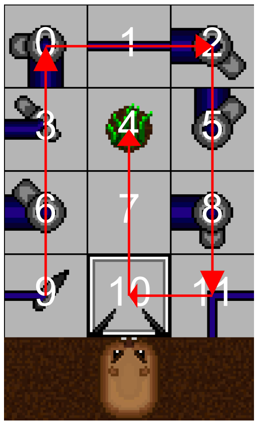
To quantify the function of a given trap, we define *lethality* of a trap as the probability that it kills a gopher entering it. We are able to compute the lethality, also referred to as functionality in this context, of a trap analytically and verify the analytical results through empirical simulation.

TRAP ENCODING To generate traps using genetic algorithms, we introduce a genotypic trap representation called an *encoding*. We code each trap as a finite length vector of components, analogous to a chromosome, with the variable components corresponding to genes. We consider the 93 possible individual tiles as variable components and map each of them to a unique integer $x \in [0, 92]$. For instance, the door tile is given the code 0, the food tile the code 1, the floor tile the code 2, and the skinny arrow with right-acute angle rotated at $0°$ is represented by 33. The encoding of an example trap is given in Figure 1c.

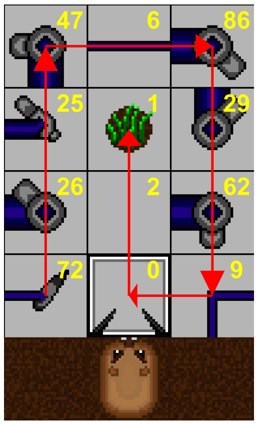
For convenience, we enumerate the twelve tiles in a trap as shown in Figure 1d. Our next step is to consider how to order the twelve tiles in the genotypic representation of a trap. Though simply encoding a trap by listing the codes for its tiles in the order $(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11)$ is acceptable, we instead encode a trap by listing the codes for its tiles in the order $(9, 6, 3, 0, 1, 2, 5, 8, 11, 10, 7, 4)$, using the wrap-around pattern shown in Figure 2. This genotypic representation better reflects the actual spatial layout of the trap.

3.2 GOPHERS

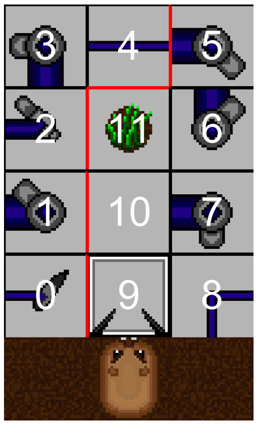
GOPHER BASICS As with traps, we adopt the simulated gophers from Hom et al. as the artificial agents in our experiments [10]. These gophers have intrinsic goals: survive traps and eat more food. Thus, each gopher will encounter traps, decide whether or not



(a) Encoding with permutation (9, 6, 3, 0, 1, 2, 5, 8, 11, 10, 7, 4).



(b) Encoded trap: [72, 26, 25, 47, 6, 86, 29, 62, 9, 0, 2, 1].



(c) An example of recombining at the 5th cell with our encoding methods.

Fig. 2: (a) Second encoding permutation and (b) corresponding encoded trap, for Method 2. (c) A recombination example split, under Method 2.

to enter them, and attempt to eat the food once inside. A gopher will repeat this process until it gets killed by a laser strike, dies from starvation, or successfully finishes going through a predetermined number of traps.

Following Hom et al., we include *intention gophers* and *baseline gophers* in our experiments. Intention gophers possess intention perception—the ability to assess the coherence of the trap and then determine whether the trap is randomly generated or deliberately harmful, based on its coherence. If the intention gopher senses the trap as sufficiently coherent, it will conclude the trap is intentionally designed (rejecting the null hypothesis of random generation), and skip over the trap unless it is forced to enter due to starvation. Baseline gophers, in contrast, simply enter traps according to some predetermined probability.

To simulate behavior, we first assign each gopher (whether intention or baseline) with a hunger level $H \in [0,1)$ to indicate how hungry a gopher is. We set Maximum Fasting Interval (MFI) as the maximal number of traps a gopher can endure without eating, and define the hunger level by $H(n) = \frac{n+1}{\text{MFI}}$, where n is the number of traps the gopher has already endured without food. The probability that a baseline gopher will enter an arbitrary trap is defined as $P_e'(H) = P_e \cdot (1 - H^{10}) + H^{10}$, where P_e is the default probability of entering and $P_e'(H)$ is the adjusted probability of entering. While the baseline gopher will decide whether to enter based on the adjusted probability of entering, the intention gopher will decide based on the intention perception algorithm described in the next section.

For convenience, we use discrete time steps, called *frames*. After entering a trap, a gopher will head directly toward the food in the center of the trap at the rate of 1 tile/frame and eat for a certain amount of time. As pulses take time to travel, eating longer puts gophers at higher risk of being hit. Thus, we design the eating time of a gopher to be based on its confidence of entering a trap, i.e., the default probability of entering a trap. Specifically, the eating time t_{eat} of the gopher is selected according to the probability vector $\vec{p} = [p_1, p_2, p_3, p_4, p_5]$, calculated based on P_e through methods in [10], where p_j represents the probability that a gopher eats for j frames.

If no arrow in the trap fires, after eating for t_{eat} frames, the gopher will exit the trap from the way it came at the same speed. However, if any arrow in the trap fires a laser, the gopher will leave immediately regardless of its current hunger level, what it was doing, and whether it was hit. This allows us to model the "skittishness" of gopher, and it does not count as having eaten if a gopher leaves before it finishes eating.

INTENTION PERCEPTION We simulate gophers' ability to determine whether the external agents intend to harm through assessment of trap structures using the intention perception framework adopted from Hom et al. [10]. The intention perception model is built upon the *functional information* model introduced by Hazen et al. [9] within Montañez [**?**], which "evaluates the surprise level (S) of a random configuration variable meeting or exceeding a given level of function" [10]. The intention perception model evaluates the surprise level S of any trap meeting a certain level of coherence. Following Hom et al., the surprise level S is computed as

$$S(x) = -\log_2 \left[|\mathcal{X}|(1 + \ln|\mathcal{X}|) \frac{p(x)}{F_g(x)^{-1}} \right] \tag{1}$$

where x is any trap configuration, $\mathcal{X}$ is the space of all possible traps, $g(x)$ is the coherence of a given trap x, $p(x)$ is a probability measure on space $\mathcal{X}$ (we set as $p(x) = 1/|\mathcal{X}|$ by default in experiments, namely, a uniform distribution), and $F_g(x)$ is the proportion of traps with at least the level of coherence of trap x. In other words, $F_g(x)$ is calculated by $F_g(x) = M_g(x)/|\mathcal{X}|$, where $M_g(x)$ is the number of traps with coherence greater or equal to $g(x)$. In order to implement the model, we use precalculated $M_g(x)$ terms for each possible coherence value $g(x)$, as given in the Appendix of Hom et al. [10].

After computing the surprise level S of a certain trap x, we reject the null hypothesis that a trap is randomly generated at a significance level of $\alpha = 0.0001$, which corresponds to a surprise level of $S = 13.29$ bits. That means that the probability of a trap with surprise level of at least 13.29 bits being generated by the null hypothesis process does not exceed 0.0001 [10,17]. Rejecting the null hypothesis implies that the intention gopher regards the trap as being intentionally designed rather than generated uniformly at random, and therefore the gopher will try to avoid the trap.

3.3 GENETIC ALGORITHMS

While Hom et al. considered only two types of trap-generating processes (namely, human design and uniformly random generation), we investigate a wider variety of traps generated by genetic algorithms. Similar to natural selection, individuals in a genetic algorithm population reproduce at a rate proportional to their fitnesses, producing offspring that largely inherit the characteristics of their parents. Populations shaped by the algorithm will tend toward increased fitness. Thus, by defining different fitness metrics one can produce individuals that are optimized for different desired characteristics.

For our genetic algorithm, we begin with a search space $\mathcal{X}$, which is the set of all valid traps defined in Section 3.1, a randomly generated initial population $X \subseteq \mathcal{X}$, and a fitness function $f : \mathcal{X} \rightarrow \mathbb{R}$ that evaluates the fitness of each individual trap (details given in Section 3.4). As discussed in Section 3.1, we represent each individual in the search space by a string of variables known as *genes*, with the joint string being called a *chromosome*, using the genotypic representation system defined there. Concretely, each gene is an integer encoding a tile and a chromosome is a string of twelve such genes. At each step, the genetic algorithm will go through the process of *(roulette-wheel) selection*, *recombination*, and *mutation* to generate a new offspring population $X' \subseteq \mathcal{X}$. This process is repeated for ten thousand generations.

For recombination, we select (in a fitness-proportional, roulette-wheel manner) two elements from the population, splitting the chromosomes of both elements into two parts at the same position, and combine the first slice of the first element with the second slice of the second element. This generates a new trap that inherits genetic information from both parents. Figure 2c illustrates an example of how we might split a trap at a certain cell. Finally, each recombined new element will undergo a mutation step: for a single trap, the algorithm uniformly picks a gene (that is not the door or food) from its chromosome and switches that gene to some random gene $y \in [0, 90]$, mimicking a genetic point mutation. Each trap generated through the steps of selection, recombination, and mutation becomes an element of the new population $X' \subseteq \mathcal{X}$. We repeat the process until the size of the new generation matches that of the old.

We iteratively create new generations until a specific number of generations is generated. Finally, we output the single trap with highest fitness value from across all generations. The complete process of our genetic algorithm is described in Algorithm 1.

Algorithm 1 Sample Genetic Algorithm

```
 1: procedure GENETICALGORITHM
 2:     globalBest ← none
 3:     population ← generateRandomPopulation()
 4:     while not terminationConditionsMet do
 5:         newPopulation ← empty
 6:         for element in population do
 7:             selectedPair ← roulette(population)
 8:             combined ← recombine(selectedPair)
 9:             mutated ← pointMutate(combined)
10:             newPopulation.add(mutated)
11:         popBest ← bestTrap(newPopulation)
12:         if f(popBest) > f(globalBest) then
13:             globalBest ← popBest
14:         population ← newPopulation
15:     return globalBest
```

3.4 FITNESS FUNCTIONS

Fitness functions are critical in generating desirable traps. They impose an ordering on all elements in the search space, based on certain criteria. When those criteria align with one's goals, it endows the genetic algorithm with the capacity to generate traps with particular traits as expected. However, not all fitness functions are valid and effective. We define two properties of fitness functions useful for discussing their effectiveness. First, we say a fitness function is **spatially ordered** if it assigns fitness values to individual elements based on some pattern or local clustering. Second, we say a fitness functions is **correctly aligned** if it is spatially ordered and embodies a preference for elements in a specific target set, which, in our setting, is the set of all traps that are either coherent, functional, or both.

We next introduce the set of fitness functions used in our experiments, and explore how the properties of each fitness function contribute to its effectiveness.

RANDOM FITNESS Given any trap $x \in \mathcal{X}$, the random fitness function $r : \mathcal{X} \to \mathbb{R}$ is a function defined by $r(x) = n$ where n is a real number chosen uniformly at random from the interval $[0,1]$. In other words, we assign each trap in the search space a random fitness value that is not based on any property of the trap itself. Thus, the random fitness function is neither spatially ordered nor correctly aligned, as the spatial distribution of its values reveal no regularities but only randomness.

BINARY DISTANCE (HAMMING) FITNESS The hamming fitness function begins by picking a "target" trap $t \in \mathcal{X}$ uniformly at random from the search space. It then measures the distance between a given trap and the fixed target trap. Given any trap $x \in \mathcal{X}$, the hamming fitness function $h : \mathcal{X} \to \mathbb{R}$ is a function defined by $h(x) = \frac{\#_{\text{diff}}}{9}$, where $\#_{\text{diff}}$ is the number of differing genes in the chromosomes of x and t, while $9 = 12 - 3$ is the largest possible number of differing genes, as there are three fixed cells for all traps. Since the hamming fitness function imposes ordering on traps based on their distance to the target trap, it is spatially ordered. However, this criteria doesn't align with our target set, so this fitness function is not correctly aligned.

COHERENT FITNESS Given any trap $x \in \mathcal{X}$, the coherent fitness function $c : \mathcal{X} \to \mathbb{R}$ returns the coherence of trap x defined in Section 3.1. This fitness function assign values based on trap coherence and gives higher values to more coherent traps, which are included in our target set. Thus, the coherent fitness function is both spatially ordered and correctly aligned.

FUNCTIONAL FITNESS (LETHALITY) Given any trap $x \in \mathcal{X}$, the functional fitness function $f : \mathcal{X} \to \mathbb{R}$ is defined by $f(x) = \frac{P_{\text{kill}}(x)}{P_{\text{max}}}$, where $P_{\text{kill}}(x)$ is the probability of x killing an entering gopher, also known as *lethality* of x as defined in Section 3.1, and P_{max} is the largest probability of killing an enter gopher which is realized only in the case that two arrows both hit the gopher. Since this fitness function rewards lethal traps, and since lethal traps are often similar to other lethal traps, this indicates that it is both spatially ordered and correctly aligned.

MULTIOBJECTIVE FITNESS We now define a fitness function that evaluates both the coherence and lethality of any given trap. Moreover, it prioritizes traps that are both coherent and lethal instead of traps that are only coherent or only lethal by penalizing the gap between the trap coherence and lethality values.

We define two types of multiobjective fitness functions. The first is a *local multiobjective fitness function* $\varphi : \mathcal{X}^n \to \mathbb{R}^n$ ($n \in \mathbb{N}$ is the population size) that takes in a population of traps and outputs an array containing fitness values of all traps in that population. The local multiobjective fitness function imposes a proper ordering on the input population based on the relationships between traps, and this effectively boosts the performance of the genetic algorithm when optimizing for more than one objective. However, the local multiobjective fitness value of each trap only depends on its relative rank within the population, which implies that the same trap may have different fitness values in different generations. Therefore, we define an additional *global multiobjective fitness function* $g : \mathcal{X} \to \mathbb{R}$ as a universal measure of trap coherence and lethality. We employ the local fitness function during the selection process within the genetic algorithm and the global fitness function to record the quality of generated traps.

Local Multiobjective Fitness Function. This fitness function is a variation of a standard method for multiobjective evolutionary optimization which relies on the notion "dominance" among traps [4]. For any traps $x, y \in \mathcal{X}$, we say that x *dominates* y if x has both a greater coherence fitness and a greater functional fitness than y. Given a trap $x \in \mathcal{X}$, let $\#_{dominant}(x)$ denote the number of traps x dominates within the population. The base score for x is then $\#_{dominant}(x) + 1$.

Moreover, we wish to add more diversity into the selected population by disincentivizing sampling traps that are too similar to each other. Let $N(x) = \{y \in \mathcal{X} \mid \#_{dominant}(x) = \#_{dominant}(y), y \neq x\}$ be the set of neighbors of x containing traps that have same base value as x. Then, we compute the normalized distance between x and its closest neighbor $d_{normalized}(x) = \min_{y \in N(x)} \|x - y\| / \sqrt{2}$, where $\|x - y\|$ denotes the point-wise Euclidean distance between trap x and y, and $\sqrt{2}$ is the maximum possible distance between two traps. If $N(x) = \emptyset$, the normalized distance is set to be 1. Then, for each $x \in \mathcal{X}$, we add this normalized distance to its base score, obtaining the boosted score. Each boosted score is then divided by the maximum boosted score across the population, leaving the most fit trap with a fitness value of 1. Finally, we return the array of normalized boosted scores contained in range $(0, 1]$ as the fitness values of traps in the population.

In this way, the local multiobjective fitness function can not only serve as a good measure of relative fitness of traps within the population, accelerating the genetic algorithm, but also promotes diversity within the population.

Global Multiobjective Fitness Function. Given any trap $x \in \mathcal{X}$, the global multiobjective fitness function is defined as

$$g(x) = \begin{cases} \frac{f(x)+c(x)}{e^{2|f(x)-c(x)|}} & |f(x) - c(x)| \leq k_{\text{diff}}, \\ \frac{1}{10k_{\text{diff}}} \frac{f(x)+c(x)}{e^{2|f(x)-c(x)|}} & |f(x) - c(x)| > k_{\text{diff}}, \end{cases}$$

where $f(x)$ and $c(x)$ are the functional and coherence fitness values of x respectively, and k_{diff} is some pre-defined constant. The intuition behind this definition is to reward both coherence and lethality of a trap while penalizing the difference between coherence and lethality. The design of a threshold k_{diff} intends to disincentivize solely optimizing coherence or lethality alone (which would lead to large gaps between the functional and coherent fitness values). Furthermore, since the global multiobjective fitness function assigns higher values to traps that are both lethal and coherent, it is both spatially ordered and correctly aligned.

4 EXPERIMENTAL SETUP

4.1 Generating Traps Through Genetic Algorithms

Our goal is to generate traps with specific traits using genetic algorithms equipped with different fitness functions. In the process, we observe the performance of each fitness function, in terms of their convergence speed and the quality of the traps produced.

For each fitness function (i.e., random, hamming, coherent, functional, or multi-objective) we generate 1,000 optimized trap examples. Since each run of the genetic algorithm outputs a single best trap as the final step, we run the genetic algorithm for 1,000 independent trials. For each run of the genetic algorithm, we set the population size to be constant 20, terminate the algorithm after 10,000 generations, and output the trap with best fitness value among all 200,000 traps generated in this run. After repeating this process for all five fitness functions, we also generate 1,000 traps uniformly at random and make use of the designed traps from Hom et al. [10] for comparison.

5 RESULTS

5.1 Generating Traps Through Genetic Algorithms

First, we aim to understand how the genetic algorithm traverses through its solution space when optimizing for different attributes. Moreover, we use three methods to understand how the algorithm searches the space: we calculate the average number of generations until the best-fitness trap was found, the proportion of traps found with a given lethality/coherence, and an estimated probability distribution for traps over the range of lethality and coherence values.

Time to Optimal Trap First, we calculate the average number of generations until the best-fitness trap is found. Figure 3 is a boxplot which shows the number of generations until the best trap was found in each of the 1,000 trials. As stated above, the experiment was split based on the fitness function used to generate the traps. The plot shows that the functional fitness function has a median at around 100 generations, along with a small interquartile range and some outliers up to 500 generations. On the other hand, the multiobjective and coherence fitness functions seem to have medians around 3,000 generations and ranges that span all 10,000 generations. The only notable difference between the multiobjective and coherence fitness functions is that the interquartile range of the coherence function is smaller than that of the multiobjective function. Specifically, it seems that the coherence boxplot has an interquartile range of around 1,200 generations to 5,200 generations, while the multiobjective fitness function seems to have an interquartile range of 200 generations to 6,200 generations. Hence, the coherence fitness function seems to more reliably find its optimal trap in the 2,500-5,000 generation range, while the multiobjective fitness function seems to be less reliable, finding its optimal trap in the 200-6,200 generation range. Since the range of the functional boxplot is so much smaller than that of the coherence and multiobjective boxplots, it is evident that finding a maximally lethal trap is much easier than finding a maximally coherent trap. Furthermore, it is evident that coherence is the bottleneck for the multiobjective fitness function as the multiobjective boxplot strongly resembles the coherence boxplot.

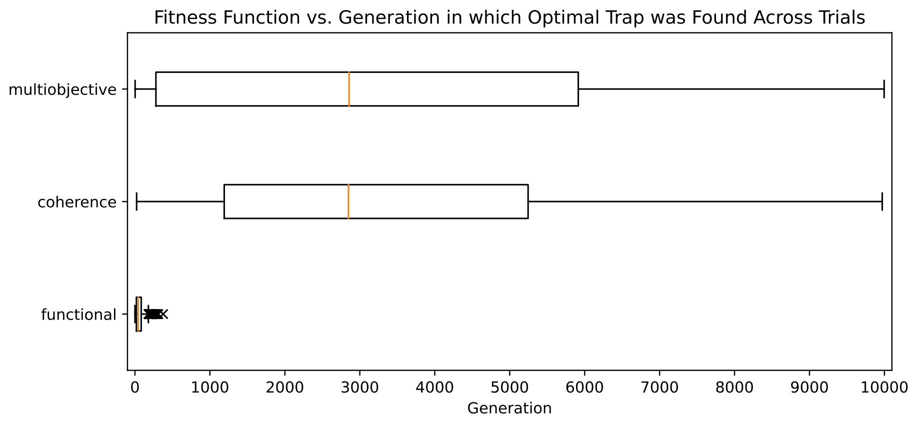
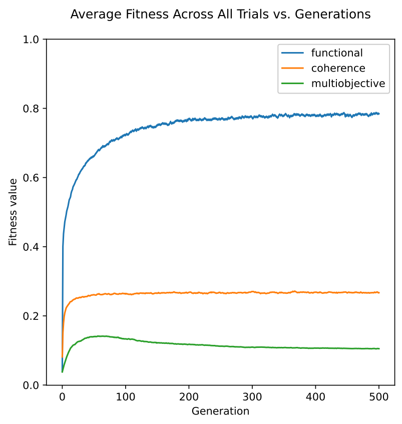


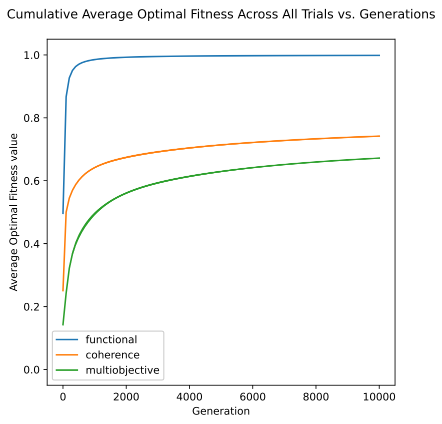
Fig. 3: Boxplot showing the distribution of when the optimal trap was found across all generations. Reproduced from [12].

Figure 4 shows two line plots depicting the average time to optimal trap. Figure 4a depicts the average fitness across all 1,000 trials for a given generation, and Figure 4b depicts the cumulative average optimal fitness across all 1,000 trials for a given generation. Unlike the average fitness, the cumulative optimal fitness takes “the best trap seen until generation *i*” for each trial and averages those across all trials, whereas the average fitness just takes the average fitness of all traps in generation *i* across all trials. Additionally, the lack of visibility in the error bars can be attributed to the low variance in the data.

In Figure 4a, notice that each of the functional, coherence, and multiobjective fitnesses plateau after around 300 generations. After this plateau, there were no deviations from the observed trend, and hence we focus on the first 500 generations of the plot. Notice that the functional, coherence, and multiobjective curves all settle at values around 0.75, 0.25, and 0.10, respectively. It makes sense that the functional fitness function converges at a high average fitness value, since this indicates that the function is able to find traps that are maximally lethal quickly (notice that the plot should not converge to 1, or maximal lethality, since we are averaging all traps in a given generation across all



(a) Line plot showing the average fitness across all trials over generations. The (imperceptible) shaded region again represents the 95% confidence interval.

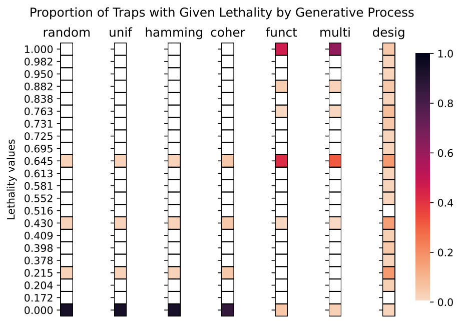


(b) Line plot showing the cumulative average optimal fitness across all trials over generations. The shaded region represents the 95% confidence interval.

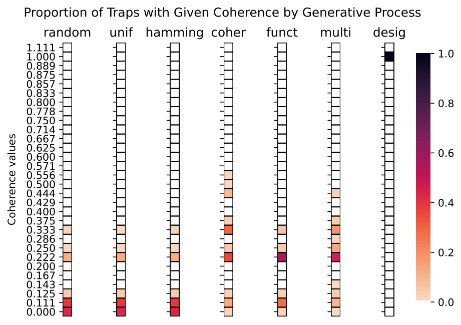
Fig. 4: Line plots showing the trends of average fitness over generations, from [12].

trials. By virtue of recombination and mutation, there are bound to be some defective traps within each generation, thereby lowering the total average fitness). However, it is interesting that the coherence and multiobjective functions converge at lower values of 0.25 and 0.10. Since the optimal solution likely wasn't found (an intuition we will soon confirm), we would expect the average fitness of these traps to generally increase since our genetic algorithm has the potential to generate coherent structures. Finally, note the peak in the multiobjective curve. This is likely due to the genetic algorithm choosing to sacrifice lethality in favor of coherence to generate traps with higher coherence and lethality. Such a choice would temporarily decrease overall fitness.

Next, consider Figure 4b, the optimal cumulative fitness graph. Notice that the functional, coherence, and multiobjective lines converge to around 1.0, 0.75, and 0.60, respectively. Furthermore, since this is a graph of cumulative optimal fitness, notice that all of our plots depict monotonically increasing averages. The functional graph converges to 1.0 within about 500 generations. This tells us that all 1,000 trials were able to find a maximally lethal trap within that same period. Likewise, the coherence and multiobjective plots converge to values below 1, which tells us that these functions were not reliably able to find traps of maximal fitness (either maximally coherent or maximally coherent and lethal). In fact, this plot tells us that the highest value for coherence and multiobjective traps that we can find on average is 0.75 and 0.60. Such a disparity between the functional and coherence/multiobjective line plots is evidence that coherence is a much harder problem to solve than lethality (at least for this problem instance). Finally, after around 4000 generations, notice that there is another dramatic decrease in slope among all of the lines. Such a decrease indicates that most trials have found their optimal traps, and corresponds to the end of the interquartile range in the respective box plot.



(a) Lethality.



(b) Coherence.

Fig. 5: Heatmap showing the proportion of traps with a given lethality / coherence, split by the parameter which the genetic algorithm was optimizing for (and designed traps). Reproduced from [12].

Proportion Vectors Figure 5 shows the proportion of traps with a given lethality and coherence. Each vector is separated by the method used to generate the trap. First, notice that the uniform random, (genetic algorithm) random, and hamming vectors are nearly identical in both figures. This supports the idea that all of these functions are equally effective at finding lethal and coherent traps. Specifically, this implies that the random and hamming fitness functions generate traps of similar quality to sampling our solution space uniformly at random (when using lethality and coherence as our proxy for success). Hence, sampling traps using a random fitness function is as effective as uniformly assigning fitness values to traps. Additionally, it is evident that being spatially ordered is not the only important aspect of our fitness function (contrary to [8])— there must also be correct alignment to encode some selection bias and guide the genetic algorithm to a target set of interest. In the case of the hamming function, notice that it is spatially ordered but not correctly aligned. Spatial-orderedness in the hamming case comes from optimization towards some random “goal” trap. Failing to align correctly can be seen in its inability to consistently generate traps with high lethality/coherence. Hence, spatial order is not sufficient for search success.

Additionally, notice that there are hot spots at non-zero values for traps generated uniformly at random. These hot spots are at lethalities 0.215, 0.430, and 0.645, and at coherences less than or equal to 0.333. The lethalities correspond to simple, lethal traps, such as traps where there is an arrow right by the door. Examples of such traps are shown in Figure 6. The coherences correspond to random, coherent connections formed within cells. Note that it is not hard to form these connections by themselves, but it is the act of chaining long coherent connections together that becomes difficult.

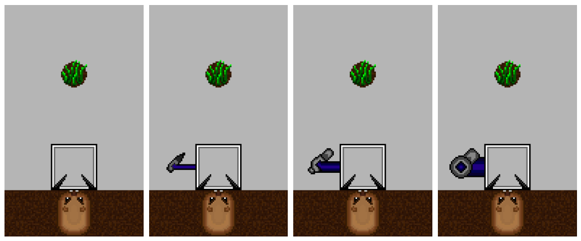


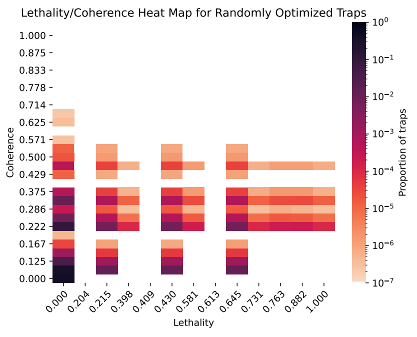
Fig. 6: Four simple traps of lethality 0, 0.215, 0.430, and 0.645, respectively, from [12].

Notice that the coherence vector resembles the uniform random vector in 5a and the lethality vector resembles the uniform random vector in 5b. As seen in the hamming case, this resemblance to the uniform random vector is indicative of a mismatch in alignment. In other words, optimizing for lethality does not beget coherence and vice versa, a trait that is attributable to our problem instance: the intersection of coherent and lethal target sets is too small.

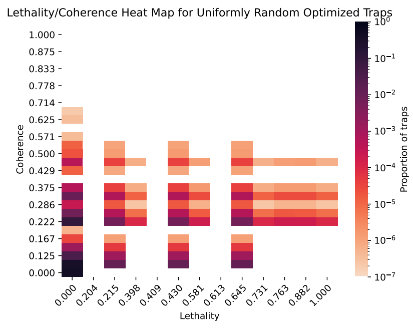
In Figure 5a, notice that there are a considerable proportion of traps that have achieved maximal lethality (namely, traps generated by the functional and multiobjective fitness functions). However, the same cannot be said for traps reaching maximal coherence in 5b. Specifically, about 40% of the traps generated by the functional fitness function had maximal lethality, but only about 10% of the traps generated by the coherence fitness function had a coherence value of 0.556 (which corresponds to the last color of sufficient magnitude to be non-white). It is important to note that our genetic algorithm was able to find traps with higher coherence than 0.556, but there were not a significant enough proportion of those traps to show up in Figure 5b. Hence, this is further evidence that finding lethal traps is significantly easier than finding coherent traps.

Finally, note that the proportion vectors for the designed traps are starkly different from the other vectors. Recall that these traps were created by the Hom et al. to zap gophers in a variety of ways—they were not intentionally designed to be maximally lethal or maximally coherent. Despite this lack of intentional effort, notice that every designed trap has a coherence of 1 and a near-randomly distributed lethality. Hom et al. attributed this unintended coherence to an innate sense of structure in human design. Furthermore, since coherence was an unintentional side-effect of construction, we see that form is a good indicator of intentional construction, as Hom et al. assumed. However, as shown by the inability for our genetic algorithm to reliably produce highly coherent traps directly, it is hard to reproduce this affinity for structure *in silico* [12].

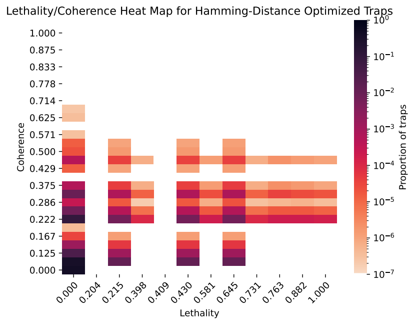
Frequency Density Heatmaps Finally, Figure 7 shows the log proportion of traps generated with a given coherence and lethality. Again, each of the heatmaps are split by the method used the generate the traps. These heatmaps are similar to the proportion vectors shown in Figure 7, but they plot both coherence and lethality as separate dimensions on the same graph (rather than being separate graphs). Note that all traps across all 1,000 trials and 10,000 generations are shown here, and hence we can see how the genetic



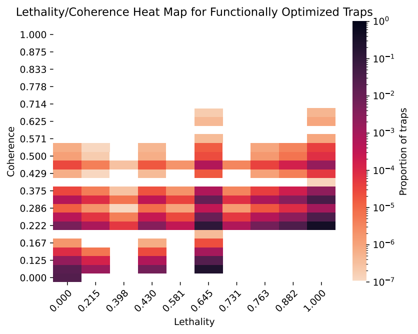
(a) Random.



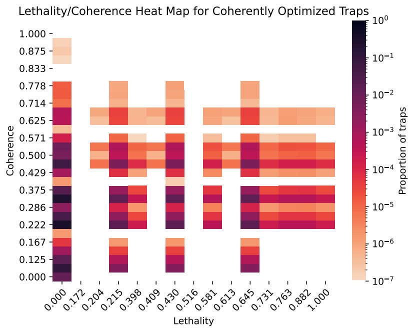
(b) Uniform-random.



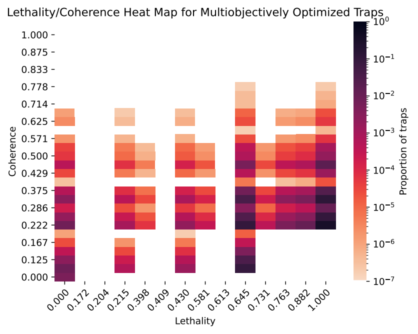
(c) Binary Distance.



(d) Functional.



(e) Coherence.



(f) Multiobjective.

Fig. 7: Heatmaps representing the log proportion of traps with a given lethality and coherence value separated by optimization parameter from [12].

algorithm traversed its search space among all iterations (again, using coherence and lethality as proxies for success).

First, notice that all distributions have hot spots at lethalities 0, 0.215, 0.430, and 0.645, which we have already noted correspond to simple, lethal traps (as shown in Figure 6). Furthermore, notice that there is some baseline level of coherence for such simple traps, since having a firing arrow implies that there must be at least one coherent connection. This relationship can be seen for any trap with non-zero lethality, since these traps also have non-zero coherence in the heatmap. Additionally, notice that the lethalities 0.215, 0.430, and 0.645 are present in traps with coherence less than 0.222 (meaning that there is at most one coherent connection in the trap). On the other hand, when we consider lethalities larger than 0.645, notice that all of traps have coherence greater than 0.222, since these traps must have two firing arrows. Hence, since traps can only be lethal if they are built upon coherent connections, all lethal traps must have some baseline level of coherence. In other words, an absolute lack of coherence implies an absolute lack of lethality [12].

Next, notice that in Figures 7a, 7b, and 7c have nearly identical distributions. Since these figures correspond to our unaligned fitness functions, the similarity of these distributions gives credence to the claim that these methods generate results of similar quality, and thus cannot reliably generate lethal or coherent traps.

Finally, in Figures 7d, 7e, and 7f, we have functional, coherence, and multiobjective distributions, respectively. We can see that the multiobjective heatmap shares traits with both the functional and coherence heatmaps. Namely, notice that the multiobjective heatmap resembles the functional heatmap with some of the probability mass shifted towards higher coherence values. This is as expected, since the multiobjective function tries to find traps with high coherence and lethality. However, since lethality is easier to obtain, it makes sense that the multiobjective plot more closely resembles the functional plot; the algorithm seems to be finding highly lethal traps first, and then it tries to increase the coherence of those highly lethal traps.

6 Intention Perception and Genetic Algorithm Traps

6.1 Examining Generated Traps Under Intention Perception

Having generated a series of traps with genetic algorithms, we evaluate them in terms of intention perception. We note that while the significance value of $\alpha = 0.0001$ and uniform distribution probability measure $p(x) = 1/|\mathcal{X}|$ were used by default in both Hom et al.'s and our own previous work [10,12], we add more variation to the intention perception model in the present paper. Considering that the traps in our study are generated by genetic algorithms rather than being selected uniformly at random from the sample space $\mathcal{X}$, a default, idealized uniform distribution model may affect the accuracy of intention perception. We test this claim.

Using the intention perception framework introduced in Section 3.2, we design four different intention perception models with varying probability measures $p(x)$ and significance levels α. Specifically, we use two different probability models: one representing an idealized uniform distribution and one estimating the real distribution of traps in the generating process, smoothed by the Simple Good-Turing (SGT) method [5]. We also test two significance levels, $\alpha = 0.0001$ and $\alpha = 0.05$. We then apply these intention perception models to the five classes of traps (random, functional, coherence, multiobjective, and designed) in an attempt to see whether gophers equipped with intention perception will regard the generated machines as designed traps, and if so, how this affects survival.

Survival of Gophers We investigate the status of gophers with and without intention perception surviving against a series of traps generated by the genetic algorithms. For each class of traps, we randomly sample fifty traps, and a gopher will decide whether to enter a trap or skip to the next trap until it gets killed by a laser, dies from starvation, or successfully survives the fifty traps. Note that the gopher will behave exactly as described in Section 3.2: after deciding to enter the trap, the gopher will directly head toward the food and eat for some amount of time. However, if any arrow fires, the skittishness of the gopher will force it to leave immediately. We repeat this for 1,000 independent trials, record the status of the gophers as they progress through fifty traps, and analyze the results across availability of intention perception and types of traps. The parameters in the experiment are given in Table 1.

Table 1: Default values of experiment parameters.

Param.	Description	Value
P_e	Default prob. of entering	0.8
MFI	Maximum Fasting Interval	4
$P_{k,w}$	Prob. of kill w/ wide arrow	0.45
$P_{k,n}$	Prob. of kill w/ normal arrow	0.30
$P_{k,s}$	Prob. of kill w/ skinny arrow	0.15

Survival of Brave Gophers Gophers are skittish by construction. However, we further investigate how the survival status of gophers is affected when the gophers are *brave*: instead of leaving a trap immediately whenever an arrow fires, a brave gopher will only leave the trap if it is hit or has finished eating. We re-run the same set of experiments for brave gophers as for default (skittish) gophers. That is, we run the experiment for each type of trap for 1,000 independent trials, average the measured outcomes, and analyze the results.

We investigate whether the generated traps are considered designed or randomly generated when gophers apply their intention perception algorithm. Moreover, by choosing different distributions and significance level parameters, we also study how varying of parameters affects the algorithm itself.

Table 2 shows the results of the intention perception algorithm for the uniform distribution model ($p(x) = 1/|\mathcal{X}|$) and the Simple Good-Turing (SGT) model. We test for

Table 2: Intention Perception Test Results.

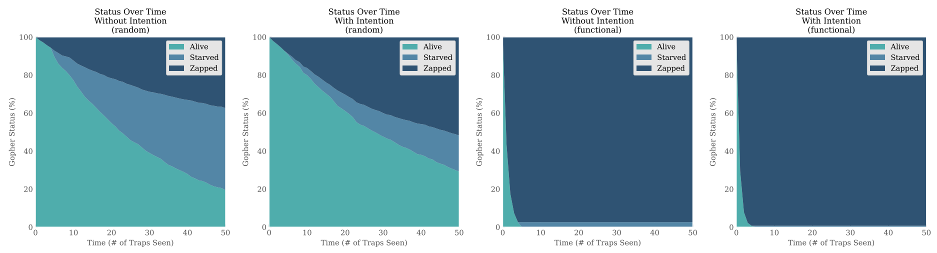
Fitness Function	α	**Unif. "Designed" %**	**SGT "Designed" %**
Uniform Random	0.05	0%	0%
Functional	0.05	0%	0%
Coherence	0.05	100%	8.9%
Multiobjective	0.05	97.5%	0.1%
Designed	0.05	100%	100%
Uniform Random	0.0001	0%	0%
Functional	0.0001	0%	0%
Coherence	0.0001	100%	0.1%
Multiobjective	0.0001	58.2%	0%
Designed	0.0001	100%	100%

α levels of 0.05 (corresponding to a surprise level of 4.33 bits) and 0.0001 (corresponding to a surprise level of 13.29 bits). As seen, for uniform random generated traps and functional traps, 0% of them are regarded as intentionally designed at both significance levels, under both distribution models. In other words, they are never mistaken for designed traps. In contrast, the designed traps themselves are judged as designed 100% of the time, at both significance levels under both models. For coherence and multiobjective traps, the results become more interesting. Under the uniform distribution model, coherence traps are deemed designed 100% of the time at both significance levels. Multiobjective traps are deemed designed 97.5% of the time at $\alpha = 0.05$, and 58.2% of the time at $\alpha = 0.0001$. However, once the more accurate SGT distribution model is used, the percentages drop precipitously for both types of traps.

We conclude that the intention perception algorithm finds coherence-optimized traps to be very similar to designed traps, but that there is a clear threshold between human-designed traps and others under the SGT model, at roughly the $\alpha = 0.0001$ level, where the intention perception algorithm becomes nearly perfect at distinguishing between human-designed traps and the others. Thus, changing the probability model from a misspecified uniform model to a more accurate SGT estimated model greatly improves the accuracy of the intention perception algorithm. Even though most of the coherence-optimized traps are considered to be intentionally designed under a assumed uniform distribution, very few of them are considered designed under the estimated SGT distribution model.

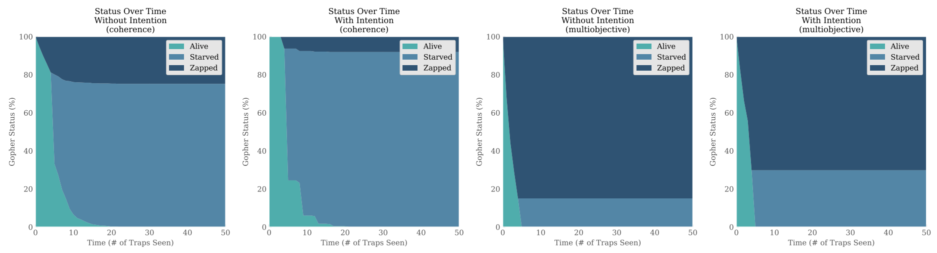
6.2 Survival of Gophers in Generated Traps

We investigate whether the intention perception algorithm of Hom et al. continues to confer significant survival advantages for artificial gophers when confronted with traps generated by a wider variety of processes. For these experiments we used the default parameter values of Hom et al. [10], including the misspecified uniform distribution probability model discussed in the previous section.



(a) Random Traps without Intention. (b) Random Traps with Intention. (c) Functional Traps without Intention. (d) Functional Traps with Intention.

Fig. 8: The effects of intention perception on the status of (skittish) gophers progressing through random and functional traps.



(a) Coherence Traps without Intention. (b) Coherence Traps with Intention. (c) Multiobj. Traps without Intention. (d) Multiobj. Traps with Intention.

Fig. 9: The effects of intention on the status of (skittish) gophers progressing through coherence and multiobjective traps.

Figures 8a and 8b show the status of gophers with and without intention perception as they progress through fifty random traps, and these fifty random traps are selected from among the random traps generated by the genetic algorithm equipped with the random fitness function. We observe that only about 21.8% of baseline gophers survive all fifty traps while around 30.8% intention perception gophers do. The greater survival rate for intention perception gophers can be attributed to the sharp decline in percentage of gophers starved (40.6% starved for baseline gophers compared to 18.7% starved for intention gophers). This is mainly because intention perception gophers are more likely to enter random traps and eat, confidently judging them as safe. However, intention gophers also have a higher probability of being killed by lasers (37.6% zapped for baseline gophers compared to 50.5% zapped for intention gophers). Thus, their confidence has a cost. Overall, intention perception gives gophers a survival advantage on random traps.

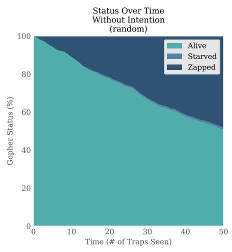
Figures 8c and 8d show how intention perception influences the survival of gophers as they progress through functional traps generated by the genetic algorithm optimizing for lethality. In this case, both baseline gophers and intention perception gophers have similar lifespans of around 5 traps. This is due to the high lethality of functional traps. Since we set the Maximum Fasting Interval (MFI) to be 4 (using the default value from Hom et al. and Kamath et al. [10,12]), every gopher will enter at least one trap before the 5th, leading to a high percentage of death from lethal lasers. The only difference between baseline gophers and intention perception gophers is that while 99.2% of intention gophers are killed by lasers, 3.3% of baseline gophers die from starvation as shown in Figure 8c. While the high lethality of the traps is responsible for the low survival rate for both gopher types, the relatively low coherence of the functional traps causes intention perception gophers to confidently enter them.

Figures 9a and 9b reveals another case in which intention perception fails to boost the survival rate of gophers even when traps are not lethal. We observe that both baseline gophers and intention perception gophers only survive through around 17 traps. Both baseline gophers and intention perception gophers die mainly from starvation, but intention gophers are even more likely to starve (73.5% starved for baseline gophers compared to 91.0% starved for intention gophers). As seen in Section 5, coherence traps usually have only low levels of lethality, resembling the lethality levels of traps generated uniformly at random. However, the high coherence implies a high probability of proper connectedness, which leads to a higher probability of firing. Even though the lasers may not hit the gophers, they can still frighten the skittish gophers and cause them to leave before finishing eating. This leads to the high percentage of gophers dying from starvation for both gopher types. However, the decrease for baseline gophers is relatively smooth, but the decrease for intention gophers is more periodic. We observe a sharp decline in the surviving gopher population at around 4 traps, corresponding to the Maximum Fasting Interval (MFI) of 4. This is because, as the name suggests, coherence-optimized traps have high coherence, leading intention perception gophers to treat them as designed traps and thus they refuse to enter them. For coherence traps, the intention perception algorithm becomes a liability, as the largely non-functional coherence confuses the decision apparatus, leading to increased, unnecessary starvation.

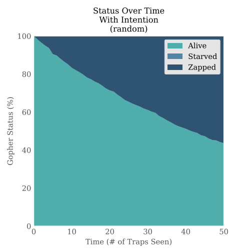
Finally, Figure 9c and 9d show the status of baseline and intention gophers when attempting to survive against series of multiobjective traps that are optimized for both coherence and functionality. In this case, intention perception again provides no survival advantage to gophers. Though intention perception gophers have a lower percentage of dying from lasers (88.1% zapped for baseline gophers compared to 71.3% zapped for intention gophers), they have an increased likelihood of starvation (11.9% starved for baseline gophers compared to 28.7% starved for intention gophers). The average lifespan for both gophers types is again approximately 4. Similar to case of functional traps, this means gophers are forced to enter at least one trap before the 5th. Once they entered the trap, they are either directly zapped by a laser or frightened to leave the trap without consuming food since the traps are both lethal and coherent. Thus, before entering the 5th trap they typically die from either laser strike or starvation.

6.3 Survival of Brave Gophers in Generated Traps

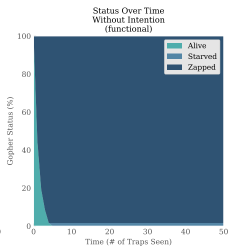
In contrast to Section 6.2, Figures 10 and 11 show the status of **brave** gophers with and without intention perception as they progress through fifty random traps generated by our genetic algorithms, beginning with the random fitness function results in Figures 10a and 10b. Notice that the number of gophers that starve has now sharply



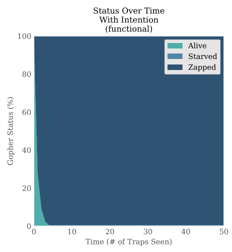
(a) Random Traps without Intention.



(b) Random Traps with Intention.

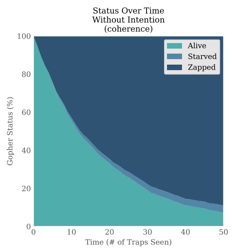


(c) Functional Traps without Intention.

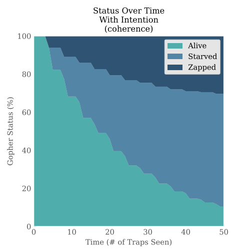


(d) Functional Traps with Intention.

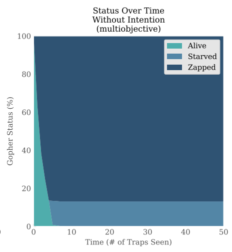
Fig. 10: The effects of intention perception on the status of brave gophers progressing through random traps and functional traps.



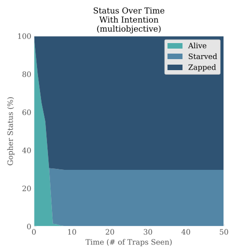
(a) Coherence Traps without Intention.



(b) Coherence Traps with Intention.



(c) Multiobj. Traps without Intention.



(d) Multiobj. Traps with Intention.

Fig. 11: The effects of intention on the status of brave gophers progressing through coherence and multiobjective traps.

decreased, regardless of intention perception. Furthermore, notice that gophers with intention (seemingly) do not starve at all. Both of these observations make sense since random traps are likely non-lethal. Even if these random traps are firing traps, it is unlikely that the orientation of such a trap would hit the gopher. Furthermore, if a gopher is hit by a rogue arrow, the probability that it will be hit in successive traps is equally small. Due to their bravery, these gophers will keep eating even when aberrant arrows fire. Starvation is still possible, though, such as when a gopher without intention perception decides not to enter MFI - 1 traps, and is hit by a rogue arrow in the final trap.

Notice that Figures 10c and 10d are nearly identical to Figures 8c and 8d, respectively. Effectively nothing changes in the setup since every trap is maximally lethal. Thus, having brave gophers does not make much of a difference, since any entering gopher will be shot immediately and then leave. Hence, the only factor in what proportion of gophers starve or are zapped is the probability of entering, which remains constant between the previous experiment and this one.

In Figures 11a and 11b the gophers survive much longer than in Figures 9a and 9b, respectively. This is expected, since these gophers are brave. We saw in the previous experiment that high connectedness implied a higher probability of firing, which would scare off skittish gophers. However, brave gophers are rewarded for their bravery—since the probability of a trap being lethal is low (as coherence does not necessarily beget lethality [12]), if a gopher enters a trap it will most likely be able to eat unharmed. However, the probability of entering a trap now causes a larger discrepancy between the intention perception and non-intention gophers. Since these traps are highly coherent, gophers with intention perception are more likely to consider these traps designed, and will not enter them until they are forced to (namely, every 4 traps). Furthermore, if these gophers get hit on the 4th trap, then they starve. Hence, we see sharp drops every 4 traps in Figure 11b. On the other hand, gophers with no intention perception are rewarded for their ignorance, and hence we see much smoother curves (since there is a fixed 80% probability that a non-intention gopher will enter a trap). This is another case where intention perception slightly helps gophers through genetic algorithm traps.

Finally, consider Figures 11c and 11d. Again, these plots are nearly identical to Figures 9c and 9d. This is due to the same reasoning as the functional case—all traps are somewhat lethal. Furthermore, note that these traps may not be maximally lethal, since we penalize large differences in coherence and lethality (thereby artificially lowering lethality in favor of coherence). Hence, we expect traps to fire, but not necessarily kill the gophers. Thus, when a gopher enters a multiobjective trap, it is likely to get either killed or starve before the 5th trap, and the gopher's boldness is unable to save it.

7 DISCUSSION

7.1 GENETIC ALGORITHM

First, we saw that it is easier to optimize for lethality than for coherence. This could be seen in Figure 4 since the average cumulative optimal fitness of the functional fitness function reaches a maximal lethality of 1.0, while the coherence fitness function was only able to reach a coherence of 0.75 on average. Furthermore, every trial for the functional fitness function converged to a maximal lethality within 500 generations, as per Figure 3. Additionally, we saw a considerable proportion of traps generated reach a maximal lethality under the functional fitness function in Figure 5, and we saw that the multiobjective heatmap shared a much stronger resemblance with the functional heatmap than the coherence heatmap in Figure 7. However, even though we saw highly lethal traps within 200 generations (when optimizing for lethality), we noted that these traps were only simple, lethal traps, where there were one or two arrows by the door. If we wanted to generate traps with only these 2 arrows by the door (and everything else as floor cells), then that would be a much harder problem. In other words, it is hard to generate highly coherent traps with just these two arrows, just as it is hard to generate long chains of coherent connections.

Next, we saw that most coherence-optimized traps did not reach their optimum fitness until around 2,500 generations, as per Figure 3. This can be attributed to a lack of "shortcuts" in optimizing for coherence, unlike lethality [12]. Specifically, when optimizing for lethality, the algorithm exploited the shortcut of putting two arrows near the door. There is no such shortcut for coherence—the only way to create highly coherent traps is to place the correct tiles with the correct thickness in the correct orientation at the correct position on the board. This discrepancy in problem difficulty is reflected in the target set sizes: among all possible traps, there are only 26,733 traps with a coherence of 1, while there are at least $91^7 \approx 5.1 \times 10^{13}$ traps with lethality 1. Furthermore, according to a uniform sampling of one million traps, 2.27% of those traps had a lethality greater than 0.5, whereas only 0.0037% of traps had a coherence greater than 0.5.

Finally, we note that the coherence of a trap is solely dependent on the connectedness of the wire, which is not a property that is preserved under recombination [12]. Since we encode our traps as an array, when we recombine and mutate, we effectively cut the trap in two. Thus, we can cut the coherence by up to a factor of one half if we cut along a coherent connection, and we can further lower the coherence if we then mutate a cell that is a coherent connection. On the other hand, if we only have arrows by the door, mutation and recombination are much less likely to affect these traps. Modifying the genetic algorithm to allow for coordinated mutations and recombination may overcome the current barriers to achieving high coherence levels.

7.2 INTENTION PERCEPTION

We tested gophers against traps generated by our genetic algorithms. Some of the gophers were equipped with intention perception algorithms, using a default, misspecified uniform distribution probability model. We saw that under the uniform model the gophers would always enter the trap, unless the trap was coherence-optimized. In the event that the trap was optimized for both lethality and coherence, we saw that a majority of multiobjective traps were thought to be designed. This suggests that intention perception may no longer confer strong survival advantages when presented with such traps under a misspecified model. Furthermore, as we saw in Sections 6.2 and 6.3, intention perception gophers no longer survive at a significantly higher rate than those without intention perception. When presented with traps optimized for only coherence, intention perception gophers actually starved more often, since they become too scared to enter traps they deem harmful. On the other hand, intention perception gophers enter every trap optimized for lethality, thereby dying instantly (and never starving). Hence, the gophers were confidently wrong in both cases, losing their survival advantage.

8 CONCLUSIONS

As discussed in the introduction, we set out to investigate two potential limitations of Hom et al.'s work. First, we conclude that the link between lethality and coherence is weak. We saw that while an absolute lack of coherence implies an absolute lack in lethality (cf. Figure 7), this is where the correlation ended. As noted by Kamath et al., even though highly lethal traps imply a baseline level of coherence, the genetic algorithm does not produce high coherence as a side product of high lethality [12]. Similarly, high coherence does not beget high lethality. However, high coherence may boost the probability of finding firing traps, but firing traps may not necessarily hit the gopher (and hence may not be lethal).

Second, we found that while intention perception helps intention gophers survive as they progress through random traps, the strong survival advantage disappears under other conditions. When traps are lethal, all gophers are killed quickly, within 5 traps.

When gophers progress through traps that are coherent and traps that are both coherent and lethal, intention perception leads to more gophers dying from starvation, but the average lifespan of intention gophers is the same as that of baseline gophers. Even if we attempt to reduce of number of gophers dying from starvation by abandoning the skittishness of gophers, intention perception does not significantly benefit gophers' chances of survival in any of the situations. Therefore, we can conclude that Hom et al.'s finding that intention perception provides significant survival advantage to artificial agents doesn't always hold. This is because the implicitly assumed correlation between trap coherence and trap lethality doesn't hold generally.

Finally, we explored the capacity of genetic algorithms to generate traps with specific traits. We saw that they were efficient in creating highly functional traps in a small number of generations, but the genetic algorithms struggled to optimize for coherence. The inability of genetic algorithms to directly generate highly coherent traps implies that trap coherence may still remain a plausible sign of intentional construction. Moreover, we conclude that generating traps with specific traits of interest requires our fitness functions to be both spatially ordered and correctly aligned. Being spatially ordered allows the genetic algorithm to perform a meaningful local search with neighborhood constraints on the elements of the search space, but simply being spatially ordered is not enough. Additionally, alignment allows us to target a specific set using selection bias.

ACKNOWLEDGEMENTS

This research was supported in part by the NSF under Grant No. 1950885. Any opinions, findings, or conclusions are those of the authors alone, and do not necessarily represent the views of the National Science Foundation.

References

1. Bhattacharya, M., Islam, R., Mahmood, A.N.: Uncertainty and Evolutionary Optimization: A Novel Approach. In: 2014 9th IEEE Conference on Industrial Electronics and Applications. pp. 988–993 (2014). https://doi.org/10.1109/ICIEA.2014.6931307
2. Bock, W.J., Wahlert, G.V.: Adaptation and the form–function complex. Evolution **19** (1965)
3. Chapman, C.D.: Structural Topology Optimization via the Genetic Algorithm. Ph.D. thesis, Massachusetts Institute of Technology (1994)
4. Fonseca, C.M., Fleming, P.J., et al.: Genetic Algorithms for Multiobjective Optimization: Formulation Discussion and Generalization. In: ICGA. vol. 93, pp. 416–423 (1993)
5. Gale, W.A.: Good-Turing Smoothing Without Tears. Journal of Quantitative Linguistics **2** (1995)
6. Gero, J.S., Kannengiesser, U.: A Function–Behavior–Structure Ontology of Processes. Artificial Intelligence for Engineering Design, Analysis and Manufacturing **21**(4), 379–391 (2007). https://doi.org/10.1017/S0890060407000340
7. Golberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley (1989)
8. Häggström, O.: Intelligent design and the NFL theorems. Biology & Philosophy **22**(2), 217–230 (2007)

9. Hazen, R.M., Griffin, P.L., Carothers, J.M., Szostak, J.W.: Functional information and the emergence of biocomplexity. Proceedings of the National Academy of Sciences **104**(suppl_1), 8574–8581 (2007). https://doi.org/10.1073/pnas.0701744104, https://www.pnas.org/doi/abs/10.1073/pnas.0701744104
10. Hom, C., Maina-Kilaas, A., Ginta, K., Lay, C., Montañez, G.D.: The Gopher's Gambit: Survival Advantages of Artifact-Based Intention Perception. In: Rocha, A.P., Steels, L., van den Herik, H.J. (eds.) Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 1: ICAART. pp. 205–215. INSTICC, SciTePress (2021). https://doi.org/10.5220/0010207502050215
11. Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments-a survey. IEEE Transactions on Evolutionary Computation **9**(3), 303–317 (2005). https://doi.org/10.1109/TEVC.2005.846356
12. Kamath, A., Zhao, J., Grisanti, N., Montañez, G.: The gopher grounds: Testing the link between structure and function in simple machines. In: Proceedings of the 14th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART. pp. 528–540. INSTICC, SciTePress (2022). https://doi.org/10.5220/0010900900003116
13. Krink, T., Filipic, B., Fogel, G.: Noisy optimization problems - a particular challenge for differential evolution? In: Congress on Evolutionary Computation, 2004. (CEC 2004). vol. 1, pp. 332 – 339 Vol.1 (june 2004). https://doi.org/10.1109/CEC.2004.1330876
14. Maina-Kilaas, A., Hom, C., Ginta, K., Montañez, G.D.: The Predator's Purpose: Intention Perception in Simulated Agent Environments. In: Evolutionary Computation (CEC), 2021 IEEE Congress on. IEEE (2021)
15. Maina-Kilaas, A., Montañez, G.D., Hom, C., Ginta, K.: The Hero's Dilemma: Survival Advantages of Intention Perception in Virtual Agent Games. In: 2021 IEEE Conference on Games (IEEE CoG). IEEE (2021)
16. Mitchell, M.: An Introduction to Genetic Algorithms. MIT press (1998)
17. Montañez, G.D.: A Unified Model of Complex Specified Information. BIO-Complexity **2018**(4) (2018)
18. Reeves, C., Rowe, J.: Genetic Algorithms: Principles and Perspectives: A Guide to GA Theory. Kluwer Academic Pub (2002)
19. Then, T., Chong, E.: Genetic algorithms in Noisy Environment. In: Proceedings of 1994 9th IEEE International Symposium on Intelligent Control. pp. 225–230 (1994). https://doi.org/10.1109/ISIC.1994.367813
20. Vorhees, C., Williams, M.: Assessing spatial learning and memory in rodents. ILAR journal / National Research Council, Institute of Laboratory Animal Resources **55**, 310–32 (09 2014). https://doi.org/10.1093/ilar/ilu013
21. Wang, S., Wang, M., Tai, K.: An enhanced genetic algorithm for structural topology optimization. International Journal for Numerical Methods in Engineering - INT J NUMER METHOD ENG **65** (01 2006). https://doi.org/10.1002/nme.1435
22. Weibel, E.R.: Symmorphosis: On Form and Function in Shaping Life. Harvard University Press (2000)