The design and implementation of biological evolution as a video game mechanic.

Barrie D. Robison[1][0000-0003-4458-926X] and Terence Soule[1]

[1] University of Idaho, Institute for Interdisciplinary Data Sciences, Moscow, ID, USA, 83844
brobison@uidaho.edu

Abstract. Video games have the potential to help teach evolutionary biology, but most commercial games misrepresent evolutionary principles by allowing player choice to dictate evolutionary trajectories. Our game studio aims to incorporate scientifically accurate evolutionary models into gameplay mechanics. In our previous games Darwin's Demons and Project Hastur, we designed digital genomes and implemented evolutionary models to create enemy populations that adapt to player strategies. However, accurately simulating evolution can sometimes conflict with crafting an enjoyable game. Here we examine balancing scientific realism with fun in the game design process. Using experimental data from Project Hastur, we show enemies evolve increased size and sensory abilities to counter player defenses, demonstrating the game mechanic's adaptive capabilities. We discuss how mutation rates, population sizes, generation times and other parameters can be adjusted to balance accuracy and enjoyment, with the goal of creating engaging games that reinforce and demonstrate, rather than misrepresent, evolutionary principles.

Keywords: Evolution, Video Games, Adaptation, Gamification, STEM.

1 Introduction

Biological evolution is an important process that frames much of our understanding of the natural world. Unfortunately, public misconceptions about evolution are common. Acceptance of evolution as a real process is low in many demographic groups, and in the United States educators often face direct opposition to teaching evolution in their classrooms[1]. Evolution is also a concept that can be difficult to present in a formal learning setting but is very understandable when directly observed. Games and simulations could therefore be useful for engaging students, demonstrating evolutionary concepts, and correcting pervasive public misconceptions about the discipline.

Game designers have long recognized the potential of evolution to sell games. Unfortunately, the most successful commercial games that emphasize evolution (for example, Spore™, Evolve™, and Pokemon™ Evolution) aren't evolutionary at all. Most of these games substitute player choice for true evolutionary processes, allowing for example, the player's character to choose whether to "evolve" a beak, or a lightning attack. This approach inadvertently represents intelligent design, a pseudoscientific

reskinning of young earth creationism. These games, which use evolution as a marketing angle, potentially exacerbate the public's confusion about evolution and reinforce existing misconceptions. The question then emerges, can accurate models of biological evolution be incorporated as a game mechanic?

There have been a few games that incorporate reasonably accurate models of evolution. *Intelligent Design: An Evolutionary Sandbox* [2] is an evolutionary game in which players create species that are placed in an evolving ecosystem - once released, the creatures continue to evolve on their own. *Niche* [3] is a turn-based game in which the player breeds members of a species to avoid extinction. Although they have not been released commercially, the game *NeuroEvolving Robotic Operatives* (NERO) and a spin-off, *EvoCommander*, are also examples of games that incorporate evolution.

Incorporating accurate models of biological evolution has interesting implications for game design. Encoding evolution as a game mechanic could create a system that adapts to player strategy, produces unexpected results and emergent properties, and inherently scales difficulty in response to player skill. These features have the potential to increase replayability and decrease development effort on game balancing.

While the gamification of evolution can potentially help expose players to real evolutionary models in an engaging way, there are several tradeoffs between playability and realism that should be considered when developing an evolutionary video game. In this paper, we discuss the lessons learned from developing and releasing games that feature evolution as a core mechanic, and the design decisions inherent to balancing gameplay with scientific realism.

2 Polymorphic Games

Polymorphic Games is an interdisciplinary game studio at the University of Idaho that explicitly focuses on the gamification of evolution. The studio is motivated by the idea that evolution can make games more compelling and more fun because the game adapts to the player. By implementing evolution as a core game mechanic, we seek to make players inherently motivated to learn the underlying evolutionary concepts in order to succeed at the game objectives.

Polymorphic Games has commercially released two games that incorporate evolution as a central game mechanic. Darwin's Demons [4] is an arcade style space shooter in which the player battles an evolving population of aliens. The enemies' traits are encoded by a digital genome, and the fittest enemies reproduce to create the next generation. The enemy population then adapts to the player's strategy. To our knowledge, Darwin's Demons was the first commercially released game to fully feature evolving enemies [5]. Project Hastur [6] is an evolutionary tower defense game that features a 3D game environment with enemies procedurally generated from a digital genome to produce a wide variety of morphologies, behaviors, capabilities, and other game traits. Both games feature experiment mode, in which the evolutionary parameters can be altered, and game data written to a *.csv* file. Experiment mode allows exploration of the model on which the game is built and allows the player to autonomously test "cause

and effect" as they adjust parameters. Darwin's Demons and Project Hastur are both available for free on Steam [4, 6]. For the purposes of this paper, we focus primarily on empirical examples using our more recent game, Project Hastur.

Gameplay: In Project Hastur, the player must defend their base against waves of enemies called the Protean Swarm. The player places defensive towers with uniqe capabilities, strengths, and weaknesses in strategic locations. Enemies appear in "waves", a classic trope of tower defense games in which the developers script the number and type of enemies that appear over the course of the level. Project Hastur replaces scripted waves of enemies with a generational evolutionary model - each new enemy wave is comprised of the offspring of the most successful enemies in the previous wave. This leads to an enemy population adapts to the player defenses as the game progresses.

3 Evolution as a Game Mechanic

Here, we explain our approach to the gamification of evolution using four commonly taught components of the evolutionary process: Variation, Inheritance, Selection, and Time. When these components are present, the evolutionary process leads to Adaptation, which is the foundation of the evolutionary process and the core component of engaging games that feature evolution as a central mechanic.

3.1 Variation

Most video games feature some kind of variation among enemy types. Even the first arcade games, such as Space Invaders, Asteroids, Centipede, and Tempest, used variation among enemies or obstacles to add interest and challenge. In Space Invaders for example, there were three types of aliens in each wave, along with the occasional bonus flying saucer. Centipede and Tempest featured categorically different enemy types that varied in their appearance and capabilities. Variation is a central component of game development and is a big part of what makes video games exciting. The difficulty in video games is often increased over time by introducing new variations of enemies, changing the enemies' capabilities (like speed or fire rate) over time, or increasing enemy number.

When implementing evolution as a game mechanic, we encode variation of enemy traits using a digital representation of a genome. Often, the digital genome is a simple vector of numbers. Each individual has the same structure (e.g. everyone has a vector size of 10), and the variation among individuals is represented in the different numbers contained in the vector. The game engine then uses the values in each individual's genome to calculate game traits, such as speed, hit points, etc.

Genetic variation can be classified into two categories, Standing Genetic Variation [7] and Mutational Variation. Standing genetic variation is the genetic differences among individuals at the start of the game. This can be adjusted as design principles dictate, but a general rule of thumb is that more variation in the initial population corresponds to a faster rate of adaptation in the early phases of the game. Were the game

to proceed solely using the standing genetic variation present at the beginning, the combination of selection and random genetic drift would quickly result in a homogeneous population and a rather bored player.

Mutational processes can be used each generation to introduce new genetic variation into the enemy population. During gameplay, selection imposed by the player can act on this new variation. This is akin to adding more "fuel" to the process of adaptation, sustaining the evolutionary game play longer than one would observe with only standing genetic variation. In Darwin's Demons and Project Hastur, we used a mutational step between generations, mutating the digital gametes passed on by the parents to the next generation. It is here that we must acknowledge a tension between designing a game for playability and creating an accurate evolutionary simulation. Mutation rates in biological populations are well studied, and rates range between 1 in 10^6 and 1 in 10^9 per locus [8]. These rates are much too low to make for very compelling game play, as new variation would take much too long to accumulate in the population. In our games, we typically use mutation rates that are much higher than those found in nature, accelerating the infusion of mutational variance into the population.

Another link between the model parameters and playability is related to game difficulty. Increasing genetic variation in the population tends increase game difficulty. In our previous games, we have used the rate and effect size of new mutations as one of the key differentiators between modes of difficulty. If these parameters are too low, the game is boring and takes much too long for evolution to occur. If these parameters are too high, enemies with extreme trait values tend to destroy the player in the early game.

Genetic Variation in Project Hastur: Each member the enemy population is defined by a digital genome of 80 genes encoded as real numbers. Enemies are diploid (they have two copies of each gene, one from each parent), such that its final trait value is calculated as the sum of the genetic values for the given locus on each chromosome. All genes are used by the game engine to render a wide variety of visually distinct game enemies, and a subset of the genes also affect traits that are relevant to game play. When the game begins, trait values are instantiated using starting values modified by mutations from a Gaussian distribution. This creates the standing genetic variation in the initial population upon which selection (imposed by the player) will act. When offspring are created in subsequent generations, additional mutational variance is introduced. In campaign mode, the per locus mutation rate and effect size vary depending on the map's degree of difficulty - more mutational variance causes the game to become much more difficult. In experiment mode, mutation rate and effect size are determined by the user. Mutation Rate (u) is defined as the per locus probability that the genetic value of the locus will be changed by a number drawn from a Gaussian distribution with mean zero and standard deviation defined by the variable Mutation Effect Size (e).

3.2 Inheritance

In most video games there is no relationship between enemies within a wave or between waves. They are instantiated (spawned) with developer defined traits at a specified rate, location, and time. In an evolutionary game, enemy traits are specified by

digital genomes which are passed to offspring through a model of reproduction. Reproduction models can be asexual (in which one parent is sufficient to create one or more offspring) or sexual (two parents contribute to the genome of the offspring).

Inheritance in Project Hastur: Each "wave" of enemies in Project Hastur is a discrete generation created from the previous generation using a tournament selection algorithm. Individuals are sexually reproducing hermaphrodites and individuals that are selected as parents each contribute one of their chromosomes, determined randomly, for each locus. Project Hastur uses a free recombination model (each locus acts as its own chromosome), but this can be modified in experiment mode. The chromosomes passed to the offspring are each subjected to the mutation algorithm, and each locus has a chance (specified by u) to receive a mutation of effect size e. Once the new offspring's genome is instantiated, it is passed to the population for the next generation. On game maps that contain human civilians, enemy creatures can also reproduce using an asexual model. In this case, if a creature kills a civilian the creature clones itself. This cloning process produces an offspring that is genetically identical to its parent and the number of clones produced by this process depends on the parent creature's size (smaller creatures produce more clones).

3.3 Selection

Selection occurs when there is a correlation between a trait (such as Health or Speed) and Fitness, which in biological populations is defined as the number of offspring produced by a given individual. We often measure proxy traits in empirical biology that are correlated with fitness, such as seed-set in plants, survivorship, or number of eggs in a nest. In our games, we can make explicit linkages between performance and fitness using Fitness Functions.

In evolutionary games, the selection process should include a stochastic component that favors more fit individuals but leaves opportunities for less fit individuals to be selected to maintain diversity within the population. Two common approaches are roulette wheel selection [9] and tournament selection [10]. Roulette wheel selection is 'fitness proportional' the probability of being selected is proportional to an individual's relative fitness. Tournament selection is a 'rank based' process in which a small (typically 3-5) subset of the population is picked at random and the highest fitness individual in that subset wins the 'tournament' and thus the opportunity to reproduce.

Tournament Selection in Project Hastur*:* Project Hastur specifies two Fitness Functions that determine the probability that enemies will be selected to reproduce at the end of each generation. The first Fitness Function, called Base Fitness, specifies the closest distance to the player's base that was achieved by a given enemy. Should the individual reach the base (a distance of zero), then Base Fitness includes the Damage the Protean does to the base. Tower Damage is the other fitness function used to calculate whether an individual will reproduce. In this case, the game sums the total Damage done by the individual to any defensive structure. Total Fitness is calculated by combining Tower Damage and Base Fitness.

At the end of each generation, we use tournament selection to identify individuals that will serve as parents. A random sample of individuals from the previous generation

is drawn, and the individual with the best value for Total Fitness is selected as a parent. A new tournament is then conducted to select the mate. Reproduction proceeds as described above and a single offspring is passed to the population for the next generation. This process is repeated until the population size for the next generation has been reached.

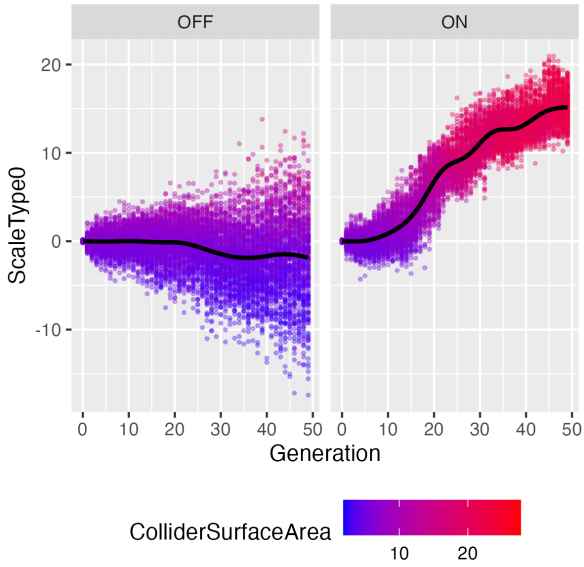

Figure 1. Values of the *ScaleType0* gene in two replicates of Project Hastur over 50 generations using a standardized defense of only Autocannon Towers. Each point represents an individual enemy, and is colored according to Collider Surface Area, which measures total body size within the game environment. *ScaleType0* controls the size of the main body of the individual. LEFT: Fitness functions are turned OFF, resulting in the accumulation of mutational variance over time, but no change in mean body size (black line) of the population. RIGHT: Fitness functions are turned ON, resulting in directional selection for increased body size.

In Figure 1, we present an example of how the combination of inherited variation and selection via fitness functions can produce directional selection, in this case for larger enemies (which tend to have more Health). Extending this example, we can show that the patterns of trait adaptation in Project Hastur are in response to game conditions and strategies instituted by the player. To demonstrate the effects of the evolutionary model, we used Project Hastur's experiment mode to run replicated instances of the same game play conditions. We therefore simulate the decision of a player in terms of the organization of a fixed number of defensive towers, but then let the game proceed without the ability of the enemies to destroy the player base. We refer to each individual game play session as a replicate.

The data presented in Figure 1 used a standardized arrangement of Autocannon turrets on the same game map ("Crater Mountain"). The Autocannon turret relies on a ray-

cast for targeting, firing numerous projectiles that do not miss. We ran a total of 9 replicates (separate instances of game play) under both conditions (with and without fitness functions) using the Autocannon. We ran additional replicates using the identical arrangement of towers using the Chip Shredder, a turret that fires a single, slower projectile that relies on collision detection. Figure 2 shows the pattern of trait evolution we observed under these conditions. This experiment is intended to show the potential outcomes if players specialize their defenses toward a single tower type.

We can test whether selection causes the observed changes in game traits by calculating the selection gradients within each generation. Selection gradients are standardized measures of the strength of selection, calculated by measuring the slope (Beta) of the relationship between a trait and fitness [11]. An estimate of Beta=0 indicates no selection. Positive values of Beta indicate directional selection for increased trait values, and negative values the opposite. Combining replicates within each treatment reveals a general pattern of positive directional selection on the ScaleType0 gene by the Autocannons, and negative directional selection on the ScaleType0 gene by the Chip Shredders (Figure 2).

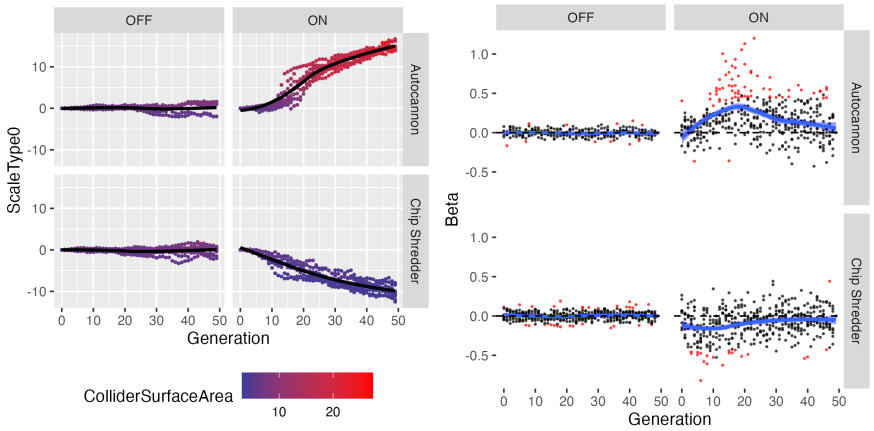

Figure 2. LEFT: Mean values of the ScaleType0 gene in 9 replicates of Project Hastur under four gameplay conditions (36 total replicates). Each point represents the mean value of a replicate at each generation, and is colored according to Collider Surface Area, which measures body size. When Fitness Functions are inactive (OFF column), mean trait values change randomly because of genetic drift. When Fitness Functions are active (ON column), the Autocannon towers select for larger individuals (which have more Health), while the Chip Shredder towers select for smaller individuals (which are faster and can evade projectiles). RIGHT: Estimates of the selection gradient, Beta, for the ScaleType0 gene in Project Hastur. Red points indicate estimates of Beta that are significantly different from 0 in individual tests, while the blue line indicates the Loess estimate of Beta across the entire experiment.

As Figure 2 shows, the type of tower chosen by the player has a dramatic effect on the evolutionary trajectory of the game enemies. This example is a simplified experiment intended to isolate a single trait's response to manipulation of a single variable. The enemies in Project Hastur have more than 20 traits, and the defensive combinations

available to the player are innumerable. This creates a great deal of potential variation in gameplay.

In project Hastur, we often observe variation in the evolutionary outcomes even when gameplay conditions are identical. In Figure 3, we show a heatmap of the standardized trait means across 9 replicates of the exact same game conditions. The first two rows of the heatmap (replicates 2 and 3) are a distinct cluster representing an outcome dominated by small and fast enemies with a large sight radius. The remainder of the replicates produced some variation of large, slow "tanks" that evolve to absorb most of the hits from the Autocannons. This variation in evolutionary outcomes is inherent to the stochastic nature of the evolutionary model. New mutations occur randomly, and the rather small population sizes cause random genetic drift. Even within a single game replicate, we can sometimes observe subdivision of the population as evolution drives individuals toward two separate optima in the fitness landscape.

For example, when we plot the individual values of the ScaleType0 gene and color by the Sight Range trait, we observe that replicates 2 and 3 are clearly evolving towards smaller individuals. Further, the individual patterns of evolution among the replicates differ in their temporal dynamics. The most extreme example is replicate 8, in which two sub-populations are clearly present between generation 25 and 40, after which point the larger individuals out-compete the smaller and the mean size of the enemies changes dramatically.

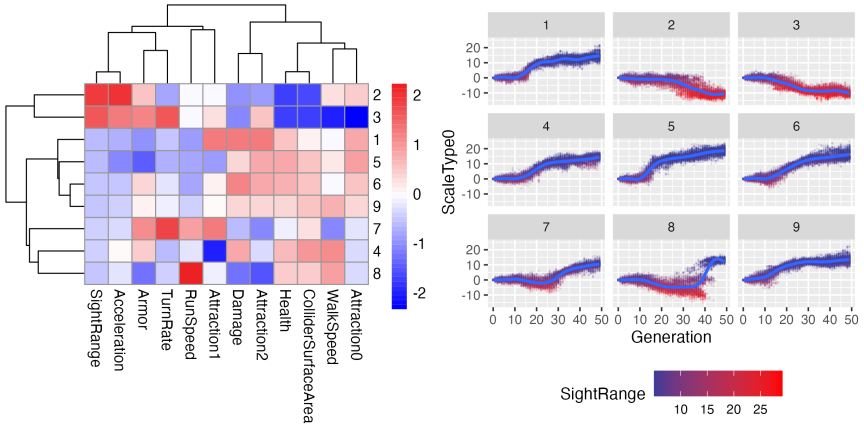

Figure 3. RIGHT: Heatmap of traits (columns) for 9 replicates (rows) of Project Hastur using the Autocannon conditions from Figures1 and 2. Each cell represents the Z-score of that trait, with each unit change representing one standard deviation from the overall mean of zero. LEFT: Temporal pattern of individual values of ScaleType0 over 50 generations, colored by the Sight Range trait.

3.4 Time

The concept of time in video games is often defined in terms of waves or levels that imply a progression toward the game's goal and a corresponding increase in difficulty. In our evolutionary games, time is specified in terms of generations. As the generations (waves) proceed, the enemies with traits that are best able to optimize the fitness functions have more offspring, and the population adapts to the game play conditions. Most of these conditions are created by player choices and playstyle, and thus the enemies adapt to the player. Difficulty increases organically and repeated gameplay often creates novel adaptive solutions to the same play style.

To make an interesting and challenging game, evolution must occur within the rather limited time frame of gameplay. In nature, biological evolution can take thousands to tens of thousands of generations to result in obvious changes. This is much too long for most games, necessitating sacrifices in realism of the model. Potential changes to model parameters include increasing the amount of genetic variance and selection pressure, both of which can speed up the evolutionary process.

4 Limitations

In this paper we have focused on the implementation of accurate evolutionary models as game mechanics, and how those models can be adjusted to accommodate engaging gameplay. A critical un-answered question is whether these types of evolutionary games result in measurable learning gains. We have recently begun formal research projects focusing on this topic, and invite interested educational researchers to contact the authors.

5 Conclusion

Video games present an engaging opportunity to teach core scientific concepts like evolution. However, most commercial games fail to accurately represent evolutionary processes, and some even mislead players. This paper discusses attempts to incorporate scientifically valid evolutionary models into game mechanics using digital genomes, variation, inheritance, selection, and generational time. Balancing realism and playability is challenging, but our data indicate the systems can demonstrate adaptive evolution in response to player actions. With further research into potential learning outcomes, evolution-driven games hold promise to improve public understanding of science while retaining intrinsic motivational appeal. Interactive games that reinforce evolutionary principles could ultimately help counter widespread misconceptions and have significant educational value.

References

1. Miller, J.D., Scott, E.C., Ackerman, M.S., Laspra, B., Branch, G., Polino, C., Huffaker, J.S.: Public acceptance of evolution in the United States, 1985–2020. Public Underst. Sci. 31, 223–238 (2022). https://doi.org/10.1177/09636625211035919.
2. Intelligent Design: An Evolutionary Sandbox, https://store.steampowered.com/app/627620/Intelligent_Design_An_Evolutionary_Sandbox/.
3. Niche, https://store.steampowered.com/app/440650/Niche__a_genetics_survival_game/.
4. Wood, N., Soule, T., Heck, S., Wright, L.R., Robison, B.D.: Darwin's Demons, https://store.steampowered.com/app/572020/Darwins_Demons/.
5. Soule, T., Heck, S., Haynes, T.E., Wood, N., Robison, B.D.: Darwin's demons: Does evolution improve the game? In: European Conference on the Applications of Evolutionary Computation. pp. 435–451. Springer, Cham (2017).
6. Heck, S., Soule, T., Wright, L.R., Robsion, B.D.: Project Hastur, https://store.steampowered.com/app/800700/Project_Hastur/.
7. Matuszewski, S., Hermisson, J., Kopp, M.: Catch me if you can: Adaptation from standing genetic variation to a moving phenotypic optimum. Genetics. 200, 1255–1274 (2015). https://doi.org/10.1534/genetics.115.178574.
8. Bergeron, L.A., Besenbacher, S., Zheng, J., Li, P., Bertelsen, M.F., Quintard, B., Hoffman, J.I., Li, Z., St. Leger, J., Shao, C., Stiller, J., Gilbert, M.T.P., Schierup, M.H., Zhang, G.: Evolution of the germline mutation rate across vertebrates. Nature. 615, 285–291 (2023). https://doi.org/10.1038/s41586-023-05752-y.
9. Blickle, T., Thiele, L.: A Comparison of Selection Schemes Used in Evolutionary Algorithms. Evol. Comput. 4, 361–394 (1996). https://doi.org/10.1162/evco.1996.4.4.361.
10. Fang, Y., Li, J.: A Review of Tournament Selection in Genetic Programming BT - Advances in Computation and Intelligence. Presented at the (2010).
11. Stinchcombe, J.R., Agrawal, A.F., Hohenlohe, P.A., Arnold, S.J., Blows, M.W.: Estimating nonlinear selection gradients using quadratic regression coefficients: Double or nothing? Evolution (N. Y). 62, 2435–2440 (2008). https://doi.org/10.1111/j.1558-5646.2008.00449.x.