Epigenetics & Genetic Algorithms for Inverse Kinematics

Multigeneration Effects On Children & Grandchildren Populations

Benjamin Kenwright, Edinburgh Napier University

This paper exploits a recent biological discovery of a popular evolutionary concept. The well-known genetic algorithm methodology mimics organic life through gene reproduction and mutation. However, recent research has pointed out that additional information embedded alongside individual chromosomes transmits data onto future offspring. This additional transmission of information onto child generations outside DNA is known as epigenetics. We incorporate this cutting-edge concept into a genetic algorithm to steer and improve the evolutionary development of the solution (i.e., achieving an optimal result sooner). We investigate the epigenetic principle of data that persists over multiple-generation (i.e., multiple generation inheritance or family tree analogy). Since epigenetics supports an important role in the evolutionary process and provides an additional mechanism to help model and solve complex problems more efficiently. We apply the enhanced genetic algorithm to solving inverse kinematic (IK) problems (e.g., linked kinematic chains). Solving inverse kinematic problems is important and challenging in multiple disciplines, such as, robotics and animation (e.g., virtual animated character control) and is difficult to obtain an optimal solution using transitional methods (e.g., geometric, algebraic, or iterative). We demonstrate the viability of our approach compared to a classical genetic algorithm. We also incorporate engineering enhancements (i.e., a non-linear mutation probability) to achieve a higher precision solution in fewer generation while avoiding prematurely converging on local minimums.

Categories and Subject Descriptors: C.2.2 [Mathematics of computing]: Bio-inspired optimization

General Terms: computational, iterative, evolving, epoch, population, genetic algorithm, evolutionary, design, algorithms, performance

Details:

1. INTRODUCTION

Preface. Genetic algorithms (GAs) are a theoretically and empirically proven technique [Goldberg 1989; Nearchou 1998] for providing robust and efficient solutions to complex non-linear problems. The powerful cross-discipline genetic algorithm at its heart is a stochastic-based method inspired by natural evolution. While the genetic algorithm continues to grow in popularity due to its flexible, robustness, and effectiveness for finding global optimization solutions [Grefenstette 2014; Tarokh and Zhang 2014; Grefenstette et al. 2011], it is still an active area of research with multiple avenues of exploration (e.g., immunization, self-adapting, and multi sub-population).

Motivation. Science and technology are increasingly hitting new limits and problems based on traditional disciplines. One avenue of inspiration for developing new and innovative ways of thinking and solving difficulties is biology. For example, biologically inspired systems have demonstrated across multiple disciplines the ability at all lev-
Epinetetics Concept - A visual depiction of the inspirational principle of epigenetics from biology and transgenerational inheritance point of view by Skinner [Skinner 2014].

To create optimal solutions for complex and chaotic systems. The popular process of evolutionary adaptation represents millions of years of design and testing. These adaptations may be more efficient than modern engineering solutions while utilizing numerous advantages, such as scalability and flexibility. As our understanding grows, nature is still showing us new ways to improve and expand, which we demonstrate in this paper by incorporating a novel enhancement from biology to improve the convergence of the classical genetic algorithm on an optimal inverse kinematic solution.

**DNA & Reproduction.** A genetic algorithm is a search heuristic that mimics the process of natural selection. This heuristic is effective for generating solutions for complex non-linear optimization problems. The principle is based on the original concept that a child’s heritable information is the combination of its parent’s DNA. However, in biology, we discover that additional inherent information is sprinkled on top and can remain over multiple generations to influence the evolutionary process (i.e., known as epigenetic inheritance). This inspiring insight, leads us to modify the classical evolutionary process (i.e., of selection and crossover) to include epigenetic data so that we may remedy inherent shortcomings within the uncomplicated GA model.

**Beyond The Classical Genetic Algorithm.** One possible feature of epigenetic modifications and transgenerational effects via DNA mixing is the ability to control the number of variant individuals in the population. Natural selection then picks the fittest individual from the population to thrive and carry on the genome and epigenome. For example, the epigenetic information can be used for:

— controlling factors (e.g. in biology the endocrine disruptors) which reprogram the germ-line (i.e. sperm) to influence heritable traits of all subsequent descendants (i.e., influences a major determinant in the selection process)
— epigenetic transgenerational phenotype suggests secondary interventions influence biology through epigenetic alterations in genome activity
— supports an important role for epigenetics in evolutionary biology and provides an additional molecular mechanism to help understand evolutionary phenomenon
Contribution. The key contributions of this paper are: (1) the investigation of epigenetic as an additional mechanism in the classical genetic algorithm to improve convergence on an optimal solution for a non-linear search problem; (2) we modify the linear mutation probability to speed up convergence of an optimal solution in an inverse kinematic system; and (3) we focus on resolving the IK problem of virtual links (e.g., chains and limits) with applications in a wide variety of fields, such as robotics and computer animation (e.g., animated characters have a large number of degrees of freedom due to their flexibility). We propose an innovative and original approach for enhancing the GA method for solving IK problems which improves the search speed and solves the GA falling into ‘premature’ lows (i.e., local minimums).

2. RELATED WORK

Inspiration. Early investigatory research in biology [Bartolomei and Tilghman 1997; Gibbs 2003] questioned the evolutionary genome process, such as, what additional factors influence evolution outside DNA recombination. The inspiring work by Giggs [Gibbs 2003] opens our eyes to the fact that DNA was once considered the sole repository of heritable information. However, a malleable layer of information encoded within the chromosomes provides fresh insight into genetic mutation that we can exploit and build upon to create solutions to problems (e.g., modifying the traditional genetic algorithm methodology to include additional concepts and overcome shortcomings and problems without sacrificing features). The work by Jirtle and Skinner [Jirtle and Skinner 2007] showed preliminary investigations into environment epigenetic modification causing alterations in gene expression leading to disease phenotypes. We explore this approach as a method of investigation and the relevance of this phenomenon in genetic algorithms. Skinner [Skinner 2011] later explored the epigenetic effects via DNA methylation on information transference. Demonstrating for the first time that heritable conditions induce epigenetic modifications and transgenerational alterations in phenotypes.

Inverse Kinematics. Inverse kinematic systems are a multi-discipline field (e.g., animation, robotics, and biomechanics). However, tuning inverse kinematic variables to match multiple fitness targets is challenging because the search space can be highly non-linear and discontinuous with parameters constantly changing to accomplish specific goals, such as end-effector positions, centre-of-mass location, and matching motion capture data [Parker et al. 1989; Yang et al. 2007; Espinoza 2012]. Thus GAs because of their random search approach, prove to be an efficient tuning method. GA are able to find optimal solutions to inverse kinematic problems even though the search space is highly ambiguous, non-continuous (e.g., joint-angle limits), and contain singularities (e.g., no solution and desire a best guess). All the same, GA search times can be impractical and time consuming (i.e., require a large numbers of generation to converge on an optimal solution). Keeping this fact in mind, this paper presents an improved version of the classical genetic algorithm in order to improve the convergence speed on an optimal solution. The concept of epigenetics with elitism [Gibbs 2003], variable probability functions [Meng and Song 2007] and non-linear mutation distribution are integrated to make the GA work faster and better.

Inverse Kinematics and Animation. Control of joint angles is the key technique in virtual human motion control. The popular approaches include geometric control, behavior control, physical control, motion capture control, and synthesis control [Wang et al. 2009], which have different characteristics [Liu et al. 2005; Gengdai et al. 2010]. The kinematics control, one of the geometric control methods, has been the most widely used in human motion control while the end-effectors position and orientation are determined. Kinematics control algorithm contains forward kinematics (FK) algorithm.
and inverse kinematics (IK) algorithm. Calculating the position and orientation of the end-effector from the joint Cartesian space is known as FK problem and calculating the joint angles from position and orientation of the end-effector is called IK problem. Many traditional methods have been used to resolve the IK problem such as, algebraic [Hai et al. 2011], geometric [Chen et al. 2012], and iterative methods [Tolani et al. 2000]. However, these methods have their own demerits in solving the IK problem of the mechanical structure or the body structure. In recent years, Fuzzy [Qu et al. 2013], Artificial Neural Networks (ANN) [Goléa et al. 2012; Zhang et al. 2012] and Genetic Algorithm (GA)-based evolutionary approaches have been applied to solve the IK problem for many kinds of manipulators, and great progress has been achieved. Satish [Kumar and Irshad 2012] uses a 3-layer perceptron neural network to resolve the IK problem in a two degree-of-freedom (DOF) serial chain manipulator, as the training data subdivide degree is only 1.125 degrees, the precision of solution is not good enough. Bassam [Daya et al. 2010] uses a 4-layer perceptron neural network to resolve the IK problem in 2 DOF manipulator, which can avoid the emergence of redundant solutions and improve the precision by dividing the solution space into many sub-spaces, but the complexity of the algorithm is increased on one hand. Reference [Nanda et al. 2012] researches the multi-layer perceptron and functional link artificial neural network, respectively, to compute 2 and 3 DOF serial manipulators inverse solution. When the robotic arm is 3 DOF, the solutions precision is less than satisfactory. As the training samples subdivision degree determines the precision of the solution in neural network, when the body has more than 3 DOF, due to the limitation in the number of training samples, it is difficult to obtain a high precise solution. The GA is widely applied to solve the IK problem due to its excellent characteristics of evolutionary optimization; Saleh [Tabandeh et al. 2006] obtains the IK solutions of the three joints which decide the end-effector position of PUMA560 robot by using the genetic algorithm based on the niche and clustering technique. The method can search four optimal solutions at a time but it is time-consuming every generation computation spent 4 s; Banga [Banga et al. 2007] combines GA and Analytical Hierarchy Process (AHP) to resolve three nodes manipulators IK problem, AHP is used to select a different fitness function to prefer a one robotic arm indicator (such as attitude, friction, adjust time), and GA optimizes the inverse solution according to the fitness function selected by AHP. Due to the fitness function selection mechanism, the method can meet multiple-objectives optimization, but it also increases the probability of obtaining suboptimal solutions. Due to the random selection of initial population and the large searching domain, the computational time of GA optimization is usually long and the algorithm is prone to premature and later search slowly. So, scholars study another approach, using GA to optimize the parameters of neural network, and then use this neural network to compute the IK solutions. Liu [Liu et al. 2008] uses GA to optimize weights and thresholds of BP neural network by applying partition coding method; then the BP neural network is used to compute the 2 DOF robots IK solution. Carlos [Kavka and Crespo 1998] combines Fuzzy neural network (FNN) algorithm and GA for solving the inverse kinematics problem of a 2 DOF manipulator. This method uses GA to optimize the network structure, the set of rules and membership functions of the FNN, and then this FNN algorithm is used to compute the two joint angles values of manipulator. These methods will accelerate the optimization process of the inverse solution when the algorithm completes the parameter optimization. The disadvantage is that the precision of the solution depends on the neural network, and the accuracy of the neural networks output depends on its structure, parameters, and training samples. When the dimension of the input and output increases, the number of parameters also will be exponentially increased. And it requires a lot of training samples, which would make the training inefficient or even unattainable. As a simple neural network algo-
algorithm, the method mentioned above is only suitable for solving the IK problem in less than 3 DOF; when DOF is equal to or larger than 3, the precision of solutions will be far from satisfactory.

Our work: One of the primary goals of our approach is to build upon the simplicity and straightforwardness of the classical genetic algorithm without incurring high additional computational overheads (e.g., memory or performance) or system complexity (e.g., inflexible frameworks). The epigenetics’ principle does not modify the DNA sequences, but factors around the genome that regulates the genomic activity (e.g., synonymous to the chemical modification of the DNA - known as DNA methylation). This can alter the gene expression and determine if genes are turned on or off (e.g., mutation). A subset of genes called imprinted genes can transfer their epigenetic pattern, (i.e., through methylation of DNA), to the next generation and effect the activity of DNA. Endocrine disruptors have been shown to modify a set of new imprinted-like genes. The frequency of an epigenetic effect is high compared to that of genetic sequence mutations. We incorporate these epigenetic principles from biology by modifying the classical genetic algorithm to solve an optimization problem more efficiently. We represent the epigenetic concept through a feedback bias influence of each individual’s parent within the population that fades over multiple generations (e.g., echo of the individuals family tree successfulness). Hence, the evolutionary data contains more than just chromosomes DNA but also a fitness factor. The principle is analogous to each individual having a recollection of their family-trees’ prosperity.

3. GENETIC ALGORITHMS (GA)

3.1. Classical Genetic Algorithm Definition

The genetic algorithm is used to solve a problem (i.e., find an optimal solution) given some set of parameters that define the search space. The parameters are commonly represented as a binary sequence. Initially, a random population is created. Thereafter, three genetic operations (or evolutionary operators, selection, crossover, and mutation, are performed to create subsequent generations. The process is repeated until a termination criterion is reached (e.g., maximum iterations or acceptable optimal solution is reached). The pseudo code for the classical genetic algorithm is given below in Algorithm 1 and is illustrated in Figure 2.

The uncomplicated genetic algorithm has a population made up of individuals who represent single or multiple solutions (and potential solutions). Each individual is rated based on a fitness function (or objective function). The breeding operation (i.e., creation of the new population) takes the fitness into account when selecting individuals to ensure that each subsequent generation has a higher chance of improving (i.e., the best characteristics are passed onto the offspring).

3.2. Genetic Algorithm Problems

While the genetic algorithm is an intuitive, flexible, and robust method for solving complex non-linear problems, the research literature agrees that the algorithm has a number of serious flaws. The key limitations of the classical genetic algorithm are:

— inability to adapt on-the-fly to changing characteristics of the search space as the population evolves
— difficulty converging if the parameters are not optimal (i.e., hill climbing algorithms suffer the same problem)
— because genetic algorithms are heuristic in nature, the solution is not always optimal

[Richter and Paxton 2005; Michalewicz 1996]
Algorithm 1 Straightforward Genetic Algorithm

1: Initialize parameters
2: Generate Population P
3: generation ← 1
4: while generation ≤ maxNumberGenerations do
5:     Evaluate each individuals' fitness function \( f(.) \) in P
6:     Sort by fitness
7:     Select N best from population
8:     Create new empty population list \( P' \)
9:     while generation ≤ maxNumberGenerations do
10:        Select two parents \( \triangleright \) (e.g., roulette-wheel)
11:        Perform crossover (c)
12:        Perform mutation (m)
13:        Insert children into \( P' \)
14:     end while
15:     \( P ← P' \)
16:     generation ← generation + 1
17: end while

The genetic algorithm is a popular heuristic technique (i.e., random probabilistic search methodology) evolving around crossover, mutation, and selection. While the random search methodology was advantageous for vast search spaces, it is not so good...
for convergence. Hence, enhancements to the model to control the genetic process is valuable to aid convergence on an optimal solution (i.e., which we go into in the next section).

4. MODIFIED GENETIC ALGORITHMS (EPIGENETICS)

What does the ‘epigenetic’ bring to the genetic algorithm? What information does the additional epigenetic information modify? i.e., how does it influence, selection, crossover, and mutation? The epigenetic principle acts like a simplified family tree for each individual; providing an echo of past parent experiences (e.g., long history of poor fitness). The epigenetic process influences the probability parameters (i.e., mutation and crossover) and selection process so the system is more likely to convergence on a global optimal solution sooner rather than later.

4.1. Epigenetic Genetic Algorithm

The epigenetic principle offers a powerful tool for steering the population evolution based on non-genetic influences (e.g., diversity, trends, and patterns). Through inclusion of an epigenome, we can include properties about the individuals’ lineage (i.e., a reflection of parents and grandparents) to bias the epigenetic selection and mutation and crossover probabilities.

4.2. Epigenetic Selection

A common approach is a roulette wheel selection based upon the parent fitness with no governing influence of the past. However, with the epigenome parameter, we can test if there has been a serious degeneration of the individuals’ fitness over-generations. We employ this in the selection process, since we can mix in chromosomes from an individual, which have had a high gain in fitness over-generations. We want to identify which individuals are changing over multiple population updates (e.g., analogy of a vector field with each individual have a vector indicating the direction of fitness gain/loss over-population epochs).

\[
\begin{align*}
  f_i &= f_i - f^\text{worst}_{i-1} \\
  \text{epiFitness}_i &= \text{epiFitness}_{i-1} + (\Delta f_i - \text{epiFitness}_i) \alpha
\end{align*}
\]

(1)
Algorithm 2 Modified Genetic Algorithm

1: Initialize parameters \( \Diamond \) (i.e., size and probabilities)
2: Randomly Generate Population \( P \)
3: \( \text{generation} \leftarrow 1 \)
4: while \( \text{generation} \leq \max\text{NumberGenerations} \) do
5: Evaluate each individual's fitness function \( f(.) \) in \( P \)
6: Sort by fitness
7: Select \( N \) best from population
8: Calculate each individual's epifitness in \( P \)
9: Create new empty population list \( P' \)
10: while \( \text{generation} \leq \max\text{NumberGenerations} \) do
11: Select two parents based on epifitness \( \Diamond \) (e.g., modified roulette-wheel)
12: Perform crossover (c) \( \Diamond \) (i.e., store worst parent fitness)
13: Perform mutation (m) \( \Diamond \) (i.e., with saw-tooth bias)
14: Insert children into \( P' \)
15: end while
16: \( P \leftarrow P' \)
17: \( \text{generation} \leftarrow \text{generation} + 1 \)
18: end while

Fig. 4. Non-linear Mutation Distribution - Saw-tooth bias to improve convergence speeds on local optimal (i.e., small angular changes in the inverse kinematic search problem).

where subscript \( i \) indicates the individual with the population, \( f_{\text{worst}}^i \) is the worst fitness for of the individuals' parent (i.e., after crossover), and \( \alpha \) is the epi-factor (0 to 1) - for example, when the epi-factor is 1.0 the previous parent influences are ignored (typically a value between 0.8 and 0.9 provide reasonable gains). \( \Delta f_i \) dictates the amount of gain of the individual over its worst parent gain. This generation performance echo fades away over update generations. As shown in the simulations, if this influence is too strong, it can impede the converge by acting damping out crucial high fidelity changes.

**Mutation Bias.** We add a fine grain optimization to the mutation procedure so it converges on an optimal solution sooner. Typically, the mutation probability is evenly distributed across each individuals' chromosome (i.e., the full set of data for the individual). For the inverse kinematic problem, the chromosome is composed of genomes each 10 bits long. Each genome within the individuals chromosome represents a joint angle. As we converge on a solution, it is more desirable to search small angular changes, hence, we bias the mutation probability across the chromosome using a linear saw-tooth bias so that we can search smaller angular spaces, as shown below in Figure 4 and Algorithm 3. The mutation bias is mixed with the underlying mutation rate which we can vary over generations to improve the genetic algorithm converging.

Technical Article - Experimental Algorithms, Vol. 9, No. 4, Article 39, Publication Date: August 2014.
Algorithm 3 Modified Mutation Procedure

1: while chromosomeBits ≤ numBits do
   chromosomeBits[..] holds all binary of the individual
2:   bias = (chromosomeBits[curBit] % (genomeLength)) / (genomeLength-1)
   bias 0 → 1
3:   if RandFloat() ≤ mutationRate + bias*0.01f then
5:      do we flip this bit?
6:         chromosomeBits[curBit] = !chromosomeBits[curBit]
7:         flip the bit
8:   end if
9: end while

Variable Mutation & Crossover Probabilities. Crossover rate $p_c$ and mutation rate $p_m$ decide the performance of the GA. In the conventional mechanism, the crossover rate and mutation rate are held constants. Higher crossover rate can increase the generation of new individuals in the searching process, it also raises the probability of destroying superior individuals in the crossover process, so it hinder the system converging to a local optimal solution. In order to prevent converging to local optimal solution, we do not want to employ a mutation rate that is too small, but a higher mutation rate will make the GA act in a completely random search. Hence, we use adaptive probabilities for the crossover and mutation mechanism, so they adaptively change according to the current evolution status. This approach presents a method in which crossover probability will diminish as the generation count increase. The crossover probability scaling is given below in Equation 2:

$$p'_c = p_c(1 - \frac{m}{M})$$

where, $m$ is evolution generation, $M$ is the total generation, generally $p_c$ is 0.8. Mutation probability can keep the algorithms from local convergence, while fixed probability may destroy perfect individuals easily. The mutation probability scaling is given below in Equation 3:

$$p'_m = p_m(1 - \frac{m}{M})$$

where, $m$ is evolution generation, $M$ is the total generation, generally $p_m$ is 0.01. The advantage of this method is that initially as we are starting from a random point, we need to cover a large search space but as number of generation is increasing and we are moving towards a better solution we can decrease probabilities and hence number of crossovers and mutations to make genetic search faster and better

5. IMPLEMENTATION DETAILS

We employ variable probabilities for the GA to move the population towards a better solution (i.e., the number of crossovers and number of mutations are decremented to reduce the simulation time). A non-linear mutation distribution helps search fine grain solutions to improve final convergence. Simulation analysis is carried out to show the viability of our idea for an enhanced GA to overcomes some of the difficulties associated with classical GA and conventional tuning. We also shown that the enhanced GA produces optimum or near optimum results for inverse kinematic solutions that are normally considered difficult to tune.
Fig. 5. **Non-Linear Mutation Graph** - Plot the population fitness against generation epoch for the linear mutation (i.e., 0.01) and a linear mutation with a saw-tooth bias (i.e., 0.01) on top. As we can see from the graph the mutation bias aids in speeding up the optimal solution (i.e., small angular changes).

**Parameter Genome Encoding and Decoding.** We map the problem space parameters (e.g., joint angles) using a binary system. We code each joint angle to an eight digit binary code (e.g., 40 joints we have 40x8 bit length individual sequence). Mapping each binary sequence to a real-number (e.g., floating point precision) is done using Equation 4 below:

\[ f = \text{lo} + \frac{hi - \text{lo}}{2^n - 1} b \]  

(4)

where \( b \) is the binary number, \( \text{lo} \) and \( hi \) are the upper and lower limits, and \( n \) is the binary length (e.g., for 8-bit number 0x00 to 0xFF, we can scale to 0.0 to 1.0 enabling us to map it easily to 0 to \( 2\pi \)).

6. EVALUATION
We evaluated our modified system for using an inverse kinematic problem (e.g., multiply connected serial linked chains). The enhanced genetic algorithm did not hinder the original classical GA model’s simplicity or incur high computational overheads (e.g., system resource, such as memory). The engineering enhancements (e.g., epigenetic parameter and non-linear mutation) enabled the algorithm to converge on an optimal inverse kinematic solution with fewer generations (as shown in Figure 5 and Figure 6).

7. CONCLUSIONS
In this paper, we developed an innovative genetic algorithm enhancement based on a recent biological discovery for controlling the population convergence on an optimal solution. The modified technique builds upon the popular breeding mechanics inherent in genetic algorithm while incorporating a epigenetic feedback parameter that keeps
Fig. 6. **Epigenetic Influence** - Scale the epi-genetic influence factor over multiple simulations to see how it influences the classical genetic algorithm (epi constant of 0.1, 0.5, 0.8, and 0.9). As we can see, if the epi-constant is too low it can degrade the convergence speed, while for higher factors we gain a noticeable speed-up (simulation of a singly linked chain of 40 links, 10 bit precision each angle, 0.01 mutation and 0.8 crossover for 1000 generations, population size 100 and 10 elite).

track of individuals progress over multiple populations. We compared the traditional genetic algorithm with our enhanced version and showed interesting improvements. While the focus of the method has been on an inverse kinematic problem, it also paves the way for multiple future avenues of research in other optimization problems.

**Acknowledgements**

A special thanks to reviewers for taking the time to provide insightful, invaluable comments, and suggestions to help to improve the quality of this paper.

**REFERENCES**


Fig. 7. **Serial Inverse Kinematic Chain Search Problem** - Simulation of a singly linked chain of 40 joints with a single end-effector (fitness objective the distance the end-effector is from the target).


Zbigniew Michalewicz. 1996. *Genetic algorithms+ data structures= evolution programs*. Springer. 5


