Evolvable Modeling: structural adaptation through hierarchical evolution for 3–D model-based vision

Thang C. Nguyen, David E. Goldberg†, Thomas S. Huang
Beckman Institute and Coordinated Science Laboratory
†Department of General Engineering
University of Illinois, Urbana, IL 61801, USA
Email: cthang@uirvld.csl.uiuc.edu, goldberg@vmd.cso.uiuc.edu, huang@uicsl.csl.uiuc.edu

Abstract: We present a system that lets 3–D models evolve over time, eventually produce novel models that are more desirable than initial models. The algorithm starts with some crude models given by the user, or randomly-generated models from a given model-grammar with generic design rules and loose constraints. The underlying philosophy here is to gradually *evolve* the initial models into better models over many generations. There is a close analog in the evolution of species where better-fit species gradually emerge and form specialized niches, a highly efficient process of complex structural and functional optimization. Our simulation results for model jet plane designs illustrate that our approach to model design and refinement is both feasible and effective.

Keywords: model-based vision, model acquisition, hypothesis generation, 3-D modeling, CAD, stochastic optimization, genetic algorithms/programming, machine learning.

1 Introduction

The motivations for our work are:

a. Currently, some existing systems (e.g, [Ikeuchi88] and others) allow us to generate recognition strategies/programs, once we have an appropriate 3–D model of the target. A bottleneck in the development of these systems however, is the high costs of good model base design, indexing and coding.
b. Real-world objects are very diverse. For cars, planes, chairs, etc. there is much greater variability/parametrization than could be captured by hand-crafting of models. The number of designs is expected to grow exponentially over time. We observe that even man-made objects such as real airplanes have their evolution trend of progress towards higher complexity and specialization. Thus if a system can automatically evolve its model base then it can achieve a higher degree of autonomy in dealing with new-object cases, with lower requirements in human "teaching" or tedious hand-crafting.

Though still limited, GA, or genetic algorithms, has been applied successfully to a few vision and graphic applications, such as: segmentation of intensity images by Bhanu et al [Bhanu90], and range images by Meygret et al [Meygret92], pattern recognition [Calloway87], and automatic generation of abstract color art [Sims91]. Especially, Hill et al [Hill92] applied GA to generate flexible 2–D templates for recognition of left ventricles in echocardiograms. However, no known work for 3–D model-based recognition has been previously available. We attribute this to the difficulty of defining a framework for 3–D model evolution.

2 Evolvable 3–D modeling with multi-level Genetic Algorithms

Our approach is to define models in such a way that one can crossover (or “mate”), in the spirit of GA, between 2 models and get 2 new models (offsprings), sometimes with new characteristics not possessed by either parents. Goldberg’s book [Goldberg89] and Koza’s book [Koza92] are good references on GA and GP principles that we further extend:

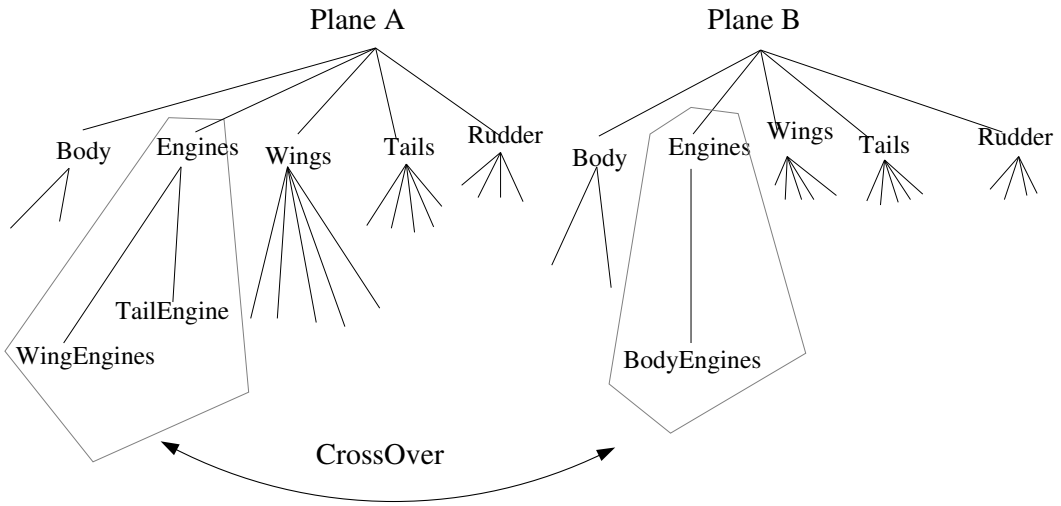
a. Grammar-based, structured object modeling: a plane is defined structurally as composed of a body, wings, engines, stabilizer tails and vertical rudder. Each of these structure is composed of geometric primitives like cylinders, cones, ellipsoids and polygons. Our implementation allows us to start with a symbol “Plane” and let the grammar expands fully, and then all terminals evaluate to function calls that eventually give a set of all 3–D geometric primitives ready for graphical display or printout. Important configuration variations like engines and tails placement (tip-of-rudder or tail-body) are also incorporated into the grammar. We are thus able to “crossover” structural elements among the symbolic designs.
b. Parametrization of numerical model features and feature relations and constraints. For example, we define many numerical parameters in terms of other parameters. The numerical parameters are coded into binary strings like DNAs so a pair of them can be crossed and give children binary strings.

c. Performance feedback, either from human user or automatic recognition process, contributes to shape the path of model evolution: current models that are more similiar to desirable models have more chance of surviving and evolving further, while bad models are weeded out soon. Eventually a good model within given constraints will emerge. Evolution processes have been shown to be more efficient than random search [Goldberg89].

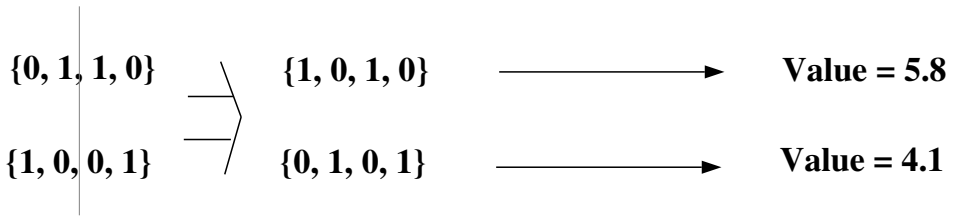
With respect to genetic operations (crossover and mutation), each of our models has 2 levels:

- Structural level: with a production grammar and symbolic parameters. A model design expressed at this level has the form of a tree, with no numerical values associated. A crossover at this level is performed by exchanging 2 subtrees of the same functionality: e.g., 2 Wings subtrees.
- DNA–string level: for the numerical values of the parameters, all coded in binary strings. At this level, a crossover between 2 models can be preformed by crossing 2 binary strings, both of which refer to some common parameter, e.g. WingHalfSpan.

In practice, some substructure is picked randomly, from among all substructures of a symbolic design, for crossover or mutation. This has the same effect of hierarchical sampling of both structures, like GP, and parameters (with GA string operations at parameter level).



(a) Structural crossover (Engines)



(b) Binary strings crossover.

Figure 1. Crossover between 2 planes at the structural level (Engines), and binary string level for numerical parameter values.

3 Experiments on modeling of jetliners through simulated evolution

We present results of a typical run of interactive design of jetliners. However, with appropriate augmentation, the same system here can be embedded as a hypothesis generator and model acquisition module within a fully automatic object recognition system.

a. The first population of models consists entirely of randomly generated models from the model grammar, model primitives, parametric relations and constraints. For simplicity, and due to space limitation here, each generation consists of only 4 jetliner models.
b. Each jetliner model is then given a score, judged by the user (or by some internal performance measure, if within an object recognition system).
c. The two models with highest scores are then selected for crossover, or "mating". One randomly selected model (among the current 4) is then "mutated". And one new model will be randomly generated (in the manner of the first-generation models.) This process creates the new generation of 4 models.
d. The above loop in (b) and (c) is continued until at least one model is given a score higher than a threshold of acceptability (say, any model to be accepted has score > 100.) Alternatively, a maximum number of generations can be set beforehand, and many runs can be tried.

We keep a simple format for quick comprehension of the process. See figure 2: there are 4 populations, from the 0th generation to the 3rd generation of the evolution process. In each population (except the 0th generation), models at positions 1 and 2 are children of a crossover operation (sexual production) from the last generation; model at position 3 is a mutated offspring; and model at position 4 is a new randomly generated model. One of our contribution to GA/GP here is that we view a mutation as a crossover with a random unknown model, which is appropriate in our framework. Figure 3 shows 4 views of the best model found.

4 Comments and Conclusions

We have demonstrated that it is feasible to evolve 3–D jet models to search for better models, starting from random models. There are many ways to apply our evolvable modeling system. This is one method to let a system learns to acquire a new model with only implicit and partial, non-specific information (human user or recognition modules need only give some "fuzzy" score to the quality of any model proposed.) In many realistic situations, the evaluation criteria of a model cannot be put down very concretely, especially with an object unknown to even the system designer.

We believe our system to be among the first to perform 3–D model evolution that have a high degree of realism for model-based recognition, despite the fact that, for clarity of the figures, we had simplified some graphical rendering parameters to avoid clutters by too many details at small scales. We could render the model at greater details, or incorporate shading, but we chose not to. (See figures 2 and 3.)

We have run many simulations of this nature and found the results to be very interesting, with some novel and pleasing designs. On the other hand, conventional jetliner models are just as conveniently generated. The best feature of our system is its high autonomy, thanks to the underlying philosophy of evolutionary processes.

References

[Bhanu90] Bhanu, B., Lee, S., and Ming, J., Self-optimizing control system for adaptive image segmentation, *Proc. Image Understanding Workshop*, 1990, pp. 583–596.

[Calloway87] Calloway, D., Using a genetic algorithm to design binary phase-only filters for pattern recognition, *2nd Intl. Conf. on Genetic Algorithms*, 1987, pp. 422–429.

[Goldberg89] Goldberg, D. E., Genetic Algorithms in search, optimization and machine learning, *Addison-Wesley*, 1989.

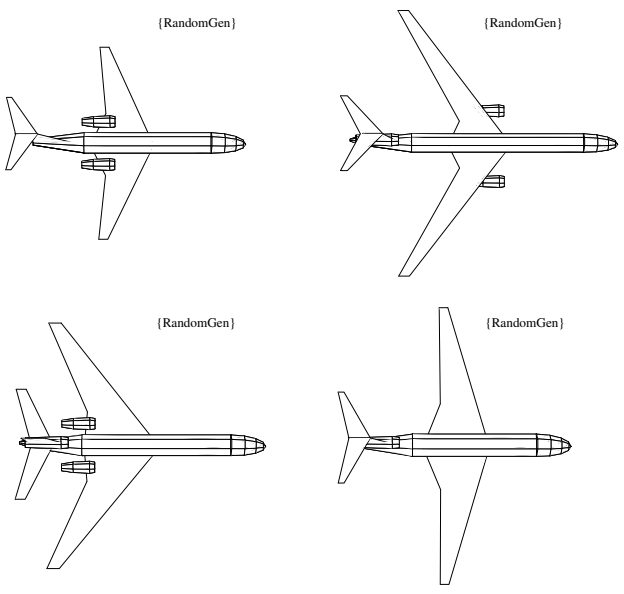
[Hill92] Hill, A., and Taylor, C. J., Model-based image interpretation using genetic algorithms, *Image and Vision Computing*, vol. 10, no. 5, June 1992, pp. 295–300.

[Ikeuchi88] Ikeuchi, K., and Kanade, T., Automatic Generation of Object Recognition Programs, *Proc. of the IEEE*, vol. 76, no. 8, August 1988, pp. 1016–1035.

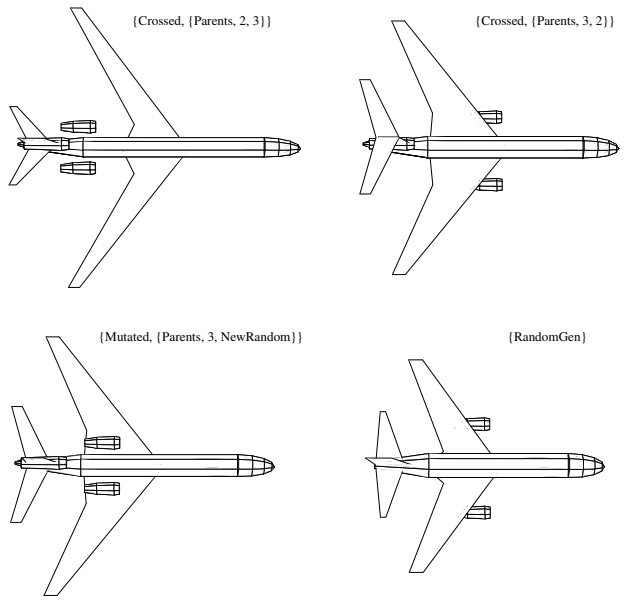
[Koza92] Koza, J., Genetic Programming, on programming of computer by natural selection, *MIT Press*, 1992.

[Meygret92] Meygret, A., Levine, D., and Roth, G., Robust Primitive Extraction in a Range Image, *11th Intl. Conf. on Pattern Recognition*, 1992, vol. III, pp. 193–196.

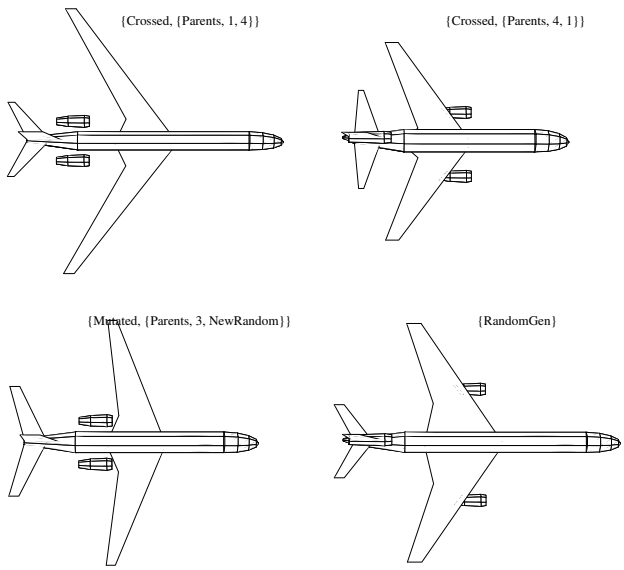
[Sims91] Sims, K., Artificial Evolution for Computer Graphics, Computer Graphics, vol. 25, no. 4, July 1991, pp. 319–328.



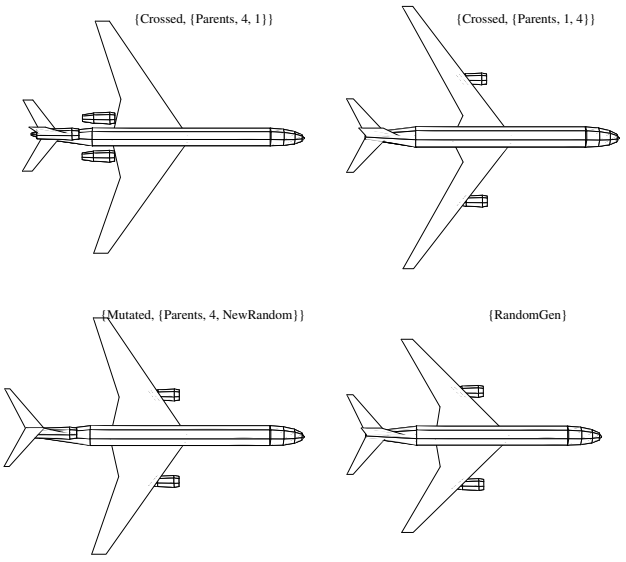
0th_generation: all random jets



1st_gen: planes 1,2 from crossover of 2, 3 in generation 0.



2nd_gen: planes 1, 2 from planes 1, 4 of 1st gen.



3rd_gen: planes 1, 2 from 1 and 4 of 2nd gen.

Figure 2. A typical run to evolve jet planes over 4 generations. In each generation, planes are numbered left to right, top to bottom, from 1 to 4.

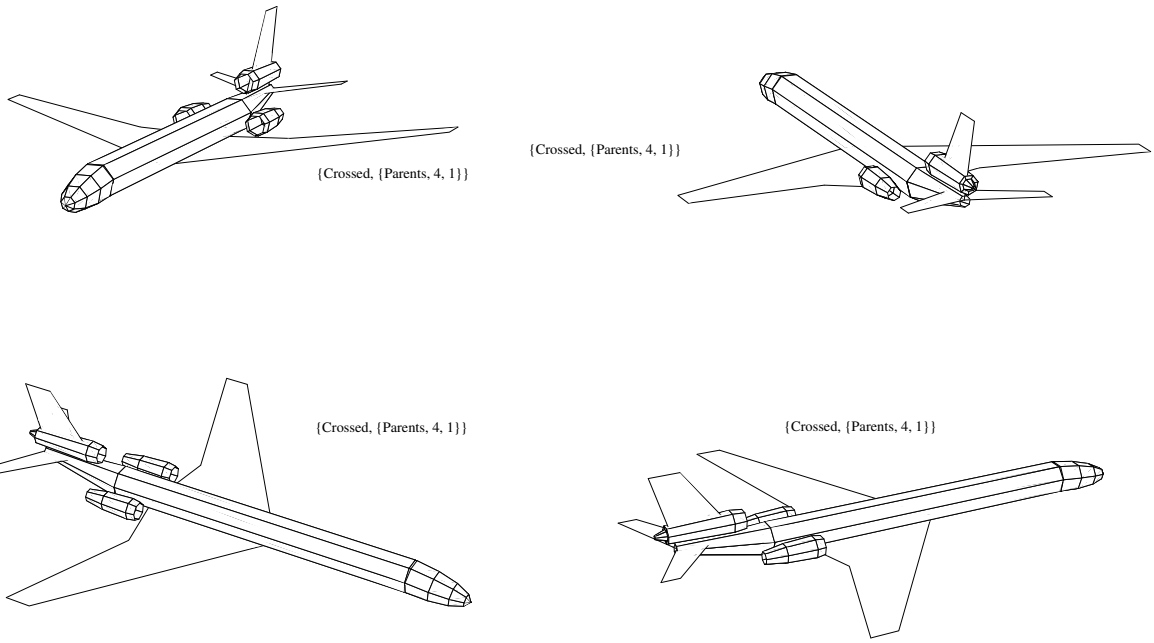


Figure 3. The best plane so far, plane #1 in the 3rd generation above. It has the following chracteristics: moderate body length, relatively small body diameter, moderate wing span, three engines, all mounted at the tail end of the body, a body-mounted stabilizer tails (as opposed to rudder-tip, high mounting position like plane #3 of generation 3).