An Experiment in Simulation Coercion

Joseph C. Carnahan, Paul F. Reynolds, Jr., David C. Brogan
Department of Computer Science, University of Virginia
151 Engineer's Way, PO Box 400740
Charlotesville, VA
{carnahan, reynolds, brogan}@cs.virginia.edu

ABSTRACT

If simulations could be coerced -- literally reshaped -- to conform to requirements different from those for which they were originally designed, many of the challenges facing the simulation community should become less daunting. Success could foster reuse, enable linkages between multi-resolution models, and increase the chances of success for interoperability and composability. We consider coercing as it applies to multi-resolution modeling. Given two simulations of the same phenomenon at different levels of detail, we explore coercing the lower resolution simplified simulation to produce results that correspond satisfactorily with the detailed high resolution simulation. The product is a simulation possessing the speed of the low resolution simulation along with the desired accuracy of the high resolution simulation.

Coercing involves a subject matter expert and optimization. The subject matter expert selects simulation parameters and establishes constraints within which those parameters are allowed to vary without compromising the validity of the simulation. Then, an optimization technique is employed to search through the possible parameter values and to select that set for which the simulation results most closely reflect the ideal; namely, conformance with new requirements stemming from the high resolution model.

To explore the concept of coercing simulations, we selected a high resolution three-dimensional bicyclist simulation and a simple simulation of a particle moving in two-dimensional space as the low resolution simulation. For these two simulations, we were able to modify, using optimization, the parameters of the low resolution simulation to follow a route that more accurately reflected the route traced by the three-dimensional bicyclist on a given course. We report on our coercing experience, providing detailed insight into the process we have designed for coercing, and we describe results. Also, we discuss automating additional phases of the coercion process and their integration into our evolving coercion tool, SimEx.

ABOUT THE AUTHORS

Joseph Carnahan is a Computer Science graduate student at the University of Virginia, working with Professors Paul Reynolds and Dave Brogan on coercible simulations research. Joseph earned his B.S in Computer Science at the College of William and Mary, and has held the position of Scientist at the Naval Surface Warfare Center, Dahlgren Division, before joining the UVa graduate program.

Paul Reynolds is a Professor of Computer Science at the University of Virginia. He has conducted research in Modeling and Simulation for over 25 years, and has published on a variety of M&S topics, including parallel and distributed simulation, multi-resolution modeling and coercible simulations. He has advised industrial and government agencies on matters relating to modeling and simulation. He is a plank holder in the DoD High Level Architecture.

Dave Brogan earned his PhD from Georgia Tech and is currently an Assistant Professor of Computer Science at the University of Virginia. For more than a decade, he has studied simulation, control, and computer graphics for the purpose of creating immersive environments, training simulators, and engineering tools. His research interests extend to artificial intelligence, optimization, and physical simulation.

An Experiment in Simulation Coercion

Joseph C. Carnahan, Paul F. Reynolds, Jr., David C. Brogan
Department of Computer Science, University of Virginia
151 Engineer's Way, PO Box 400740
Charlotesville, VA
{carnahan, reynolds, brogan}@cs.virginia.edu

INTRODUCTION

Computer simulations, like all software, can be reused and combined with other simulations in order to address newer and larger problems. An existing simulation may be reused with different parameters to model a similar phenomenon in a new setting, or a set of existing simulations may be joined to form a simulation of a more complicated phenomenon. Simulation reuse and composability both require a certain degree of flexibility. A simulation may need to be adjusted to serve as a valid model of a new scenario or to meet the requirements of the other components in a federation of simulations.

We address a semi-automated approach to coercing simulations that supports increased levels of flexibility. The primary goal of simulation coercion is to tune an existing simulation as closely as desired to a specified target. In this context, tuning means changing parameter values and applying small code modifications to cause the simulation to produce different output. This paper begins with a few conceptual examples of situations when simulation coercion would be useful, as well as a few examples of where simulation coercion has been applied in the past. Next, we outline our approach to simulation coercion, which uses a combination of subject matter expert insight and optimization techniques to drive a simulation toward a desired goal. Then, we describe our experiment, in which we coerce a simple two-dimensional model of a particle's movement to reflect the movement of a physically simulated three-dimensional model of a bicyclist. Lastly, we analyze some of the possible limitations to this approach to simulation coercion, and we identify areas that we would like to explore further.

Examples

Almost unavoidably, simulations are written with a set of implicit assumptions. Consider a simulation of a car moving down a highway. Ideally, everything about the simulated environment would be user-controlled, so that this simulation could be used to represent any possible driving situation. More realistically, assumptions would probably be made about the weather or the road surface based on where the simulation designers actually expect the simulation to be used. These assumptions are usually made to simplify software development or to improve the simulation's performance. However, when the simulation developers are unaware of a potential use scenario (such as a different vehicle or road), the simulation may need to be modified or coerced to serve as a valid model.

For another example where simulation coercion could be necessary, consider a traffic simulation composed of multiple interacting vehicle simulations. To efficiently and correctly model flows of traffic that come together at a highway interchange, groups of cars could be simulated at multiple levels of resolution. On the highway, cars could be modeled as groups with a specified size and average speed. However, when a group approaches an intersection, the group would be disaggregated and modeled as individual cars, making it possible to model significant events such as traffic accidents between cars from different groups. In this multi-resolution traffic simulation, the federation designers may specify that the rate at which groups of cars reach an intersection should not differ by more than a specified amount from the rate at which cars would arrive at the intersection if they were simulated individually. If the existing model for a group of cars does not meet this requirement, it must be coerced to better reflect the behavior produced by simulating the cars individually.

Past Work

In the computer graphics community, the problem of coercing an animated model to imitate the motion of another character is called *motion retargeting*. This problem is analogous to the problem of trying to coerce a simulation to follow the behavior of a specified function or another simulation [Reynolds, 2002]. Gleicher showed that motion retargeting problems can be solved using an optimization technique for graphics problems called *spacetime constraints* [Gleicher, 1998]. With spacetime constraints, the animation designer specifies limitations on the movement of the animated figure and selects a characteristic of the movement that should be optimized, such as the amount of energy used in the movement. Then, at discrete points along the object's path, this optimization problem is solved to determine the correct parameters for the character's motion at that point [Witkin and Kass, 1988].

Drewry, Reynolds, and Emanuel first applied this approach to the area of general-purpose simulations by studying two different models of carbon dioxide consumption in forests [Drewry et al., 2002]. The two simulations differed significantly in resolution: CANOAK models individual leaf layers in the canopy and uses a time step of one hour, while DOLY uses time steps of one month and does not model individual canopy layers separately. With the help of several subject matter experts, three parameters to DOLY were selected and allowed to vary within specified ranges. Then, optimization was used to find values of these parameters for which DOLY produced monthly carbon dioxide consumption levels that were very close to the monthly average values produced by CANOAK. In this manner, DOLY was coerced to behave in a manner similar to CANOAK.

The study of simulation coercion can be compared to Davis and Bigelow's work on motivated metamodels [Davis and Bigelow, 2003]. Motivated metamodels are an extension of statistical metamodels, where a low-resolution model of an existing simulation is created by using statistical regression to fit a linear or quadratic model to the input and output data of the original simulation. Motivated metamodels improve on this by using subject matter expert insight to determine the form of the metamodel's equations, taking into account non-linear interactions of the simulation's parameters. This combination of insight and numerical techniques resembles the coercion process that is described in this paper. However, we are applying this technique to coerce a simulation, not to create an approximation of a simulation.

SIMULATION COERCION

This paper describes an experiment in simulation coercion. As stated above, the objective of simulation coercion is to tune an existing simulation to perform as closely as desired to a specified target. Ideally, we would accomplish this task without rewriting so much of the original simulation as to be actually writing a new simulation. Simulation coercion may be necessary in order to reuse a simulation in a new setting, or it may be required to bring a low-resolution model into agreement with a high-resolution model of the same phenomenon. In either case, the process of simulation coercion involves two significant participants, namely the subject matter expert and the simulationist. Figure 1 offers a graphical representation of the simulation coercion process and the roles played by the subject matter expert and the simulationist.

The Subject Matter Expert's Role

The subject matter expert (SME) understands the fundamental phenomena being modeled and the simulation's representation of the phenomena. However, the SME does not need to be able to program, nor does the SME need to know or understand any implementation-specific details of the simulation.

Given a simulation and a new requirement that the simulation must be coerced to meet, the SME and the simulationist observe the differences between the simulation's results and the desired results. Using his or her knowledge of the underlying model, the SME characterizes the differences between the simulation's current behavior and behavior that would satisfy the new requirement. Next, the SME selects one or more parameters to the model that could be allowed to vary in order to reduce these differences. Often, these parameters are represented as constants in the program due to the assumptions of the simulation developers. Finally, the SME establishes constraints on how widely these parameters may be varied, depending on the importance of these parameters to the validity of the model.

For example, consider coercing a traffic simulation to model traffic conditions in poor weather as opposed to the good weather that was assumed in the original simulation. The average speed of a car under the new set of weather conditions may not be known, but the SME can automatically rule out speeds less than zero and speeds greater than the top speed of the vehicle. Once those bounds are established, the simulationist can use optimization to search for the best value for average vehicle speed under the new conditions.

	Analysis Phase	*Decision Phase*	*Optimization Phase*	*Evaluation Phase*
SME Activities	Analyze differences between current simulation and desired results	Select parameters to be decision variables	(no activities)	Analyze differences between optimal simulation and desired results
Simulationist Activities	Provide visualization tools (if necessary)	Specify objective function Select optimization technique	Run the optimization	Analyze effectiveness of the optimization technique
Objects Used	Original simulation Desired results	SME's subjective description of differences	Decision variables Objective function Optimization technique	Candidate optimal simulation Desired results

Figure 1. The simulation coercion process

The Simulationist's Role

While the SME is analyzing the gaps between the simulation output and the desired results, the simulationist contributes by providing tools and graphs to help the SME visualize the data. The simulationist also converts the SME's description of the differences between the simulation's current behavior and target behavior into a computable objective function. Finding an optimal value for the objective function means minimizing the difference between the simulation and its desired output.

Once the SME has identified parameters and constraints, the simulationist modifies the simulation to allow these parameters to vary and to enforce the specified constraints. In the vocabulary of optimization, the parameters are called the *decision variables* of this optimization problem. After the decision variables are selected, the simulationist selects an optimization technique to explore the possible values for these variables. Available optimization techniques include gradient-based search, grid search, simulated annealing, and genetic algorithms. In spite of their differences, each of these optimization techniques all involve the following steps:

1. Select a new set of values for the decision variables
2. Run the simulation
3. Collect the simulation's results
4. Compute the value of the objective function
5. Compare this value to previously-computed values for the objective function, and
6. Return to step 1 and repeat the process. The termination condition of this loop depends on the specific optimization algorithm being used.

After the optimization has completed, the simulationist presents the optimal parameter values and the corresponding simulation output to the SME. Together, they may identify different parameters that should have been varied or constraints that should be changed. The simulationist may also decide that changing the optimization technique or objective function would yield better results. In either case, the simulationist repeats the optimization process for the latest set of parameters and constraints and the latest choice of objective function and optimization technique.

DESIGN OF THE EXPERIMENT

For this experiment in simulation coercion, we use a high-resolution, physically simulated model of a bicyclist and a low-resolution model of a particle moving across a flat surface. Using the method described in Figure 1, we aim to coerce the low-resolution simulation to follow the same path as the center of mass of the high-resolution bicyclist, given the same target path for both simulations to follow. This objective is reasonable because of the fact that the simulations already possess semantic similarities: Both models take a target path as input, and both models

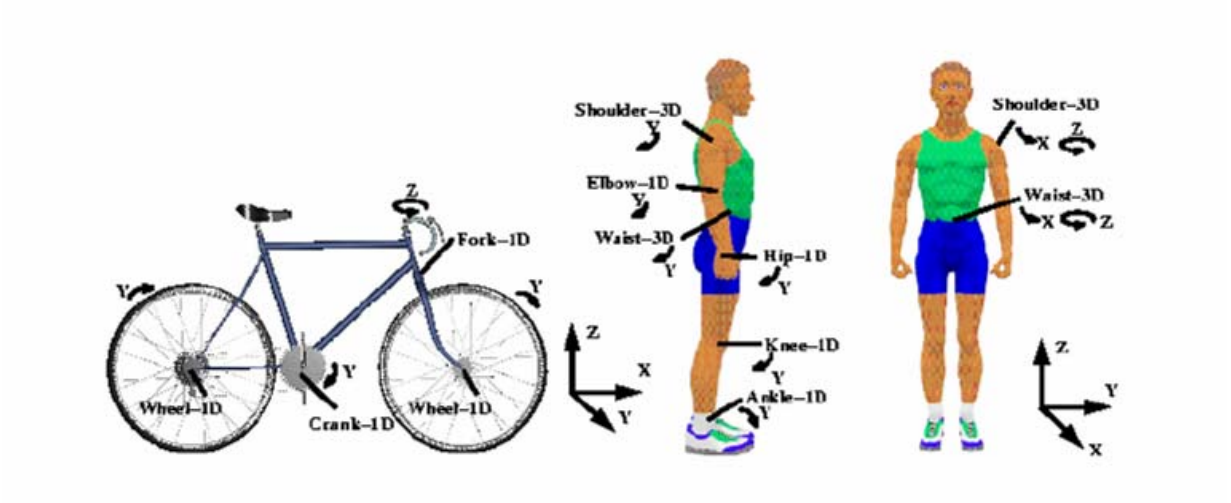

Figure 2. Diagram of the physically simulated bicyclist

represent the movement of an entity that is trying to follow the path but that must operate within physical constraints, such as only being able to change heading by a certain amount in a certain period of time.

The Simulations

The high-resolution simulated bicyclist was published previously in the graphics literature [Hodgins et al., 1995]. The bicyclist is modeled as twelve rigid segments connected by eleven joints, and the bicycle is modeled as five rigid segments connected by four joints
[Brogan and Hodgins, 2002]. Figure 2 depicts the models for the bicycle and the human bicyclist, as well as indicating the degrees of freedom that are available in the models. The simulation includes a navigation controller that specifies torques in each of the bicyclist's joints in order to make the center of mass of the bicyclist follow a given path as closely as possible at a nearly constant speed. The simulation is sophisticated enough to include leaning into turns, anticipating curves, and even falling down.

The low-resolution simulation is a simpler version of the bicyclist simulation, developed by Pascal Vicaire at the University of Virginia. Since the simulated object is modeled as a single point moving across a two-dimensional surface, we refer to it as the ``hockey puck simulation." Instead of simulating the hockey puck as a point mass that changes direction when forces are applied to it, the hockey puck model enforces limits on how sharply the hockey puck can turn within a given period of time. The hockey puck travels at a constant speed, and the hockey puck's navigation controller changes the hockey puck's heading in response to upcoming curves in the path. Like the bicyclist, the hockey puck does not follow a prescribed path exactly, but veers slightly off the path when it reacts to a curve too early or does not turn quickly enough.

The Target Paths

For this experiment, two different courses are used for the simulated bicyclists to follow. The first course is a 90-degree left turn, while the second course is a circuit involving a 45-degree left turn, a gentle 135-degree right turn, and then three successive 90-degree right turns, each sharper than the previous one. These two different courses are selected to demonstrate the effectiveness of coercing the simulation for a single specific task as well as for a sequence that involves multiple tasks of different types. The two target paths are shown in Figures 3 and 4.

The Objective Function

Optimization requires an objective function to quantify what is being minimized. Because the goal of this experiment is to coerce the hockey puck simulation to follow the bicyclist simulation as closely as possible, the objective function needs to capture the difference between two curves. As a result, the simulationist uses the root mean squared error as the objective function. For this experiment, the simulationist defines an objective function which samples 100 points along the paths traveled by the bicyclist and the hockey puck and averages the squared distance between each pair of points. The square root of this average is an estimate of the root mean squared error.

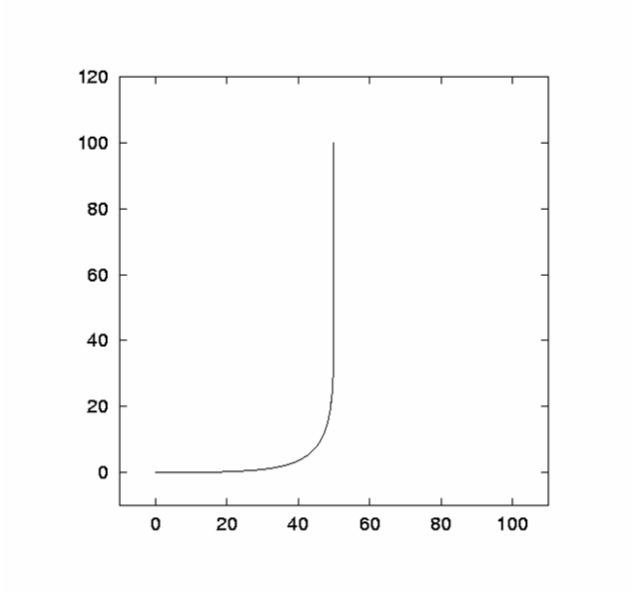

Figure 3. The left-turn target path

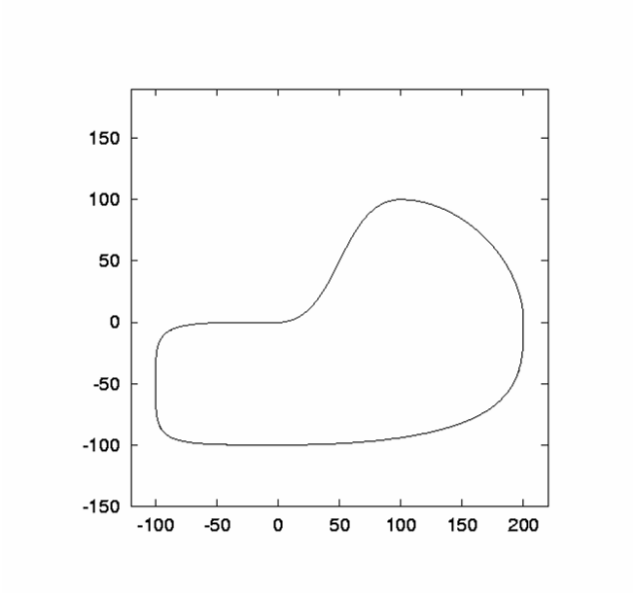

Figure 4. The looping target path

Simulation Parameters

The second step of applying optimization to a problem is to decide which parameters of the simulation will be used as decision variables. In the case of the hockey puck simulation and the bicyclist simulation, the SME selects the following three parameters as being relevant:

- maximum rotation speed (MRS), a limit on the number of degrees per second that the hockey puck is allowed to turn,
- maximum rotational acceleration (MRA), a limit on the number of degrees per second per second that the hockey puck is allowed to change turning speeds, and
- lookahead factor, which specifies the position on the curve where the hockey puck's navigation controller begins to change the direction of the hockey puck in order to anticipate a turn.

Optimization Technique

The optimization technique used for this experiment is a modified form of grid search. Stepping through the range of each of the two parameters by a discrete interval, the simulationist runs the hockey puck simulation and computes the objective function relative to the high-resolution bicyclist. The size of the interval steps is not constant: The SME hypothesizes that a lower lookahead value and a lower MRA would drive the hockey puck to behave more closely to the bicyclist. As a result, the discrete steps used are approximately equal to one percent of the parameter value at that point. For example, for values of the MRA between 18 and 180, steps of size 1 are used, while between 180 and 1,800, steps of size 10 are used. If the SME were certain that the optimal parameter values are in the lower ranges, the simulationist could constrain the search and not try using the higher values at all. However, the decision to use variable step sizes allows the simulationist to verify the SME's assumption that high parameter values are undesirable while spending most of the computational effort on the portion of the parameter space where the SME believes that the optimal result will be found.

Experimental Iteration

A certain amount of iteration is necessary in the simulation coercion process. By examining the optimization results, the SME may realize that a different set of decision variables should have been used. Alternatively, the simulationist may discover that the current optimal solution yields a low objective function value but still does not meet the SME's subjective criteria for being ``similar to'' the bicyclist. In that case, the simulationist would want to refine the objective function to better reflect what the SME wants the simulation to do.

An example of this occurred in our instance of this experiment. For the first iteration of the experiment, we selected the MRS and MRA as decision variables.

	MRA	Lookahead	Root Mean Squared Error
90-degree left turn, default values	100000 deg/sec^2	0.1	2.48 m
90-degree left turn, optimal values	280 deg/sec^2	0.01	2.24 m
Looping course, default values	100000 deg/sec^2	0.1	3.53 m
Looping course, optimal values	1580 deg/sec^2	0.008	2.32 m

Table 1. Results of simulation coercion applied to hockey puck simulation for two courses

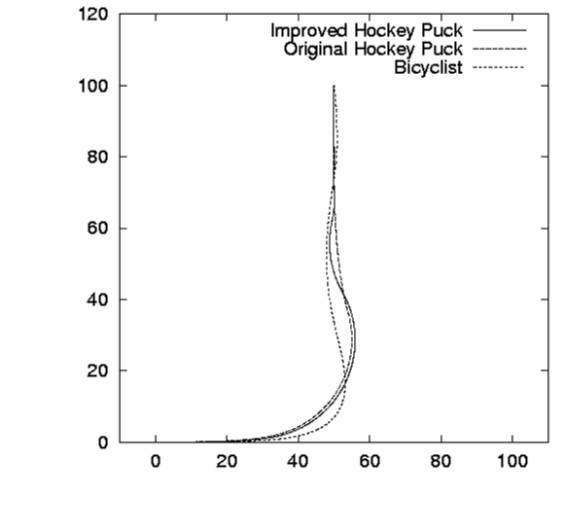

Figure 5. Results for the left-turn target path

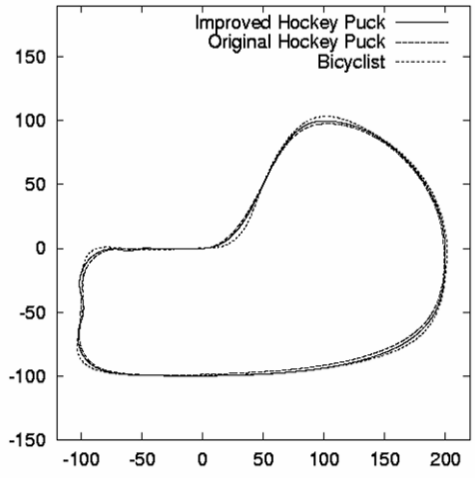

Figure 6. Results for the looping target path

However, we discovered that the limit on rotational speed had very little effect on the path of the hockey puck, since the hockey puck's turns were usually limited by the maximum rotational acceleration. At the same time, we also observed that the main difference between the two simulations' paths was that the hockey puck was reacting to curves too early and turning inside of the path of the bicyclist. In response to this, our SME identified the lookahead factor as another parameter that could be included in the optimization search. As a result, the MRA and the lookahead factor were used as the decision variables for the final iteration of the experiment.

EXPERIMENTAL RESULTS

Using the root mean squared error as the objective function, grid search as the optimization technique, and the MRA and lookahead factor as decision variables, we ran the optimization search and obtained optimal values for the two selected simulation parameters. These results are summarized here in Table 1. Figure 5 and Figure 6 contain graphs of the default and optimal paths for the hockey puck, overlaid with the path of the high-resolution bicyclist.

ANALYSIS

This experiment is an example of the process of simulation coercion. Using the combined approach of parameter optimization and subject matter expert insight, we expect to be able to coerce existing simulations to solve a variety of problems. The simulation coercion process can extend beyond the parameter manipulation used in this experiment to include other semi-automated techniques, such as modifying loop control structures or adding hooks into a program to monitor the values of variables at run-time and constrain them from exceeding specified values [Waziruddin et al., 2003].

However, the simulation coercion process does have some remaining challenges. First, optimization itself has limitations in the quality of results that it can produce in a limited amount of time. Continuous parameters have an infinite number of values, and for complex simulations, a small change in one of the parameter values may lead to an unpredictable change in the simulation's behavior. Exploring a large number of parameters and conducting a more detailed search of

the parameter space can increase the chances of finding a better set of parameter values, but they also increase the amount of computation required to carry out the search. Numerous techniques exist for performing numerical optimization [Ólafsson and Kim, 2002], but the simulationist may not be able to determine the best optimization technique.

More importantly, while the optimization provides parameter values that are guaranteed to give better results than the default values, there is no guarantee that the improved results are good enough for any specific application. For example, by changing the hockey puck simulation's parameters, we were able to reduce the root mean squared error for the looping course down to 2.32 meters. With a different optimization technique or a more finely grained search, better parameter values could probably be found. However, if we had a requirement to reduce the error below a given threshold, such as 2.0 meters, there is no guarantee that any value of the hockey puck simulation's parameters will reduce the error enough to meet that requirement.

Lastly, our current approach only optimizes the performance of the simulation for a specific set of inputs. In the case of the bicyclist simulation, the input to the system was the specified course, and the two different courses yielded different optimal values for the selected parameters. The performance of the hockey puck improved with lower lookahead values and lower limits on its rotational acceleration on both courses, but there is no way to know if a course exists for which either a higher lookahead value or a higher MRA value would have been beneficial. Optimizing the coerced simulation for a single set of inputs produces a set of parameter values that may or may not offer improvement for other sets of inputs.

CONCLUSIONS

Our semi-automated technique for simulation coercion attempts to force a simulation to produce results that are closer to a target goal. In the case of the bicyclist simulation, our hypothesis was demonstrated to be promising: Using simulation coercion, we discovered parameter values that drove the low-resolution hockey puck simulation to more closely follow the path of the high-resolution bicyclist simulation.

By definition, optimization selects the best known parameter values for the given objective function. This means that unless the default values are already optimal, this coercion technique will yield improved values for those parameters. Also, this technique is relatively simple, and nothing about this technique is based on characteristics that are unique to the bicyclist simulation. Therefore, optimization-based simulation coercion has the potential to improve the fit of any simulation to a specified objective.

Future Work

There are several areas where this work can be extended. First, it would be beneficial to establish a set of criteria for identifying the limits of coercion in a specific simulation. In a sense, any simulation can be coerced to behave like any other simulation, although it may be necessary to rewrite the entire simulation to achieve this. However, we are interested in coercing simulations with a minimal amount of internal modification. In addition, we would like to develop criteria to indicate when a simulation cannot be coerced enough to meet specific requirements, as well as providing guidance for how future simulations could be developed to be more easily coerced when needed.

Second, the simulation coercion process can become more automated. In this experiment, only the optimization step was automated, using scripts written by the simulationist for this particular application. Additional automated features could include tools for visualizing the behavior of the coerced simulation, tools for automatically collecting the simulation's output, and a package of optimization tools to give the simulationist a variety of options for how to try to coerce this simulation. To this end, we are developing an application called SimEx (SIMulation EXplorer), which is designed to plug in to given simulations with relatively few modifications and which offers visualization, data collection, and optimization features. However, note that we are still envisioning a semi-automated process instead of a fully automated one, because the subject matter expert's assistance is a necessary part of identifying and constraining the simulation parameters that must be modified.

Finally, we are considering the problem of finding a generally applicable set of optimal simulation parameters for a given simulation. One possible solution is to generate a wide variety of inputs to the simulation and to define the new objective function as the average value of the original objective function for each set of parameter values over all of the different input values. Another solution is to find optimal parameter values for each member of a set of possible inputs. For the bicyclist example, this would mean finding the optimal parameters for the hockey puck to imitate the bicyclist on a straight path, a gentle turn, a sharper turn, and an even sharper turn. Then, when

presented with a new input, the simulation could use different values for the parameters for different phases of the new input, depending on which of the original input sets most closely matches the current section of the new input. Using this technique to simulate a bicyclist riding along an unfamiliar path, the hockey puck simulation would use the set of parameters from the training path that most closely resembled the upcoming section of the new path.

Given the possibilities for automating and expanding this process, we believe that simulation coercion is a powerful technique for solving problems in the areas of simulation reuse and multi-resolution modeling. Simulations can be semi-automatically tuned to validly represent new phenomena, even phenomena that violate the assumptions of the simulation's original developers. Meanwhile, low-resolution simulations can be made to closely conform to their higher-resolution counterparts. As the idea of simulation coercion is developed, future simulations can be designed with coercion in mind, which in turn will make existing simulation coercion techniques more effective and easier to apply.

ACKNOWLEDGEMENTS

We wish to acknowledge support from the Defense Modeling and Simulation Office. We are especially grateful to Sue Numrich, Phil Zimmerman, and Phil Berry.

REFERENCES

[Brogan and Hodgins, 2002] Brogan, D. C. and Hodgins, J. K. (2002). Simulation level of detail for multiagent control. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 199-206. ACM Press.

[Davis and Bigelow, 2003] Davis, P. K. and Bigelow, J. H. (2003). *Motivated Metamodels: Synthesis of Cause-Effect Reasoning and Statistical Metamodeling*. RAND Corporation.

[Drewry et al., 2002] Drewry, D. T., Reynolds, P. F., and Emanuel, W. R. (2002). An optimization-based multi-resolution simulation technology. In Yücesan, E. and Chen, C.-H., editors, *Proceedings of the 2002 Winter Simulation Conference*. WSC, ACM Press.

[Gleicher, 1998] Gleicher, M. (1998). Retargeting motion to new characters. In Cohen, M., editor, *SIGGRAPH 98 Conference Proceedings*, pages 33-42. ACM SIGGRAPH, Addison Wesley.

[Hodgins et al., 1995] Hodgins, J. K., Wooten, W. L., Brogan, D. C., and O'Brien, J. F. (1995). Animating human athletics. In Cook, R., editor, *SIGGRAPH 95 Conference Proceedings*, pages 71-78. ACM SIGGRAPH, Addison Wesley.

[Ólafsson and Kim, 2002] Ólafsson, S. and Kim, J. (2002). Simulation optimization. In Yücesan, E. and Chen, C.-H., editors, *Proceedings of the 2002 Winter Simulation Conference*. WSC, ACM Press.

[Reynolds, 2002] Reynolds, P. F. (2002). Using space-time constraints to guide model interoperability. In *Proceedings of the 2002 Spring Simulation Interoperability Workshop*.

[Waziruddin et al., 2003] Waziruddin, S., Reynolds, P. F., and Brogan, D. C. (2003). The process for coercing simulations. In *Proceedings of the 2003 Fall Simulation Interoperability Workshop*. SISO.

[Witkin and Kass, 1988] Witkin, A. and Kass, M. (1988). Spacetime constraints. *Computer Graphics*, 22(4):159-168.