Appeared: AAAI Fall Symposium, 1992

Modeling and Simulation for Automated Yacht Design

Andrew Gelsey
Computer Science Department
Rutgers University
New Brunswick, NJ 08903
gelsey@cs.rutgers.edu

Abstract

Computational simulation is an important tool for predicting the behavior of physical systems. Many powerful simulation programs exist today. However, using these programs to reliably analyze a physical situation requires considerable human effort and expertise to set up a simulation, determine whether the output makes sense, and repeatedly run the simulation with different inputs until a satisfactory result is achieved. Automating this process is not only of considerable practical importance but also raises significant AI research issues in the areas of spatial reasoning and modeling of physics and numerical methods. The application domain described in this paper is the design of racing yachts.

Introduction

Computational simulation is an important tool for predicting the behavior of physical systems. Many powerful simulation programs exist today. However, as illustrated in Figure 1, using these programs to reliably analyze a physical situation requires considerable human effort and expertise to

- set up the simulation by transforming a description of the physical situation into a representation the simulation program can successfully process,
- analyze the output of the simulation program to extract desired information and in particular to

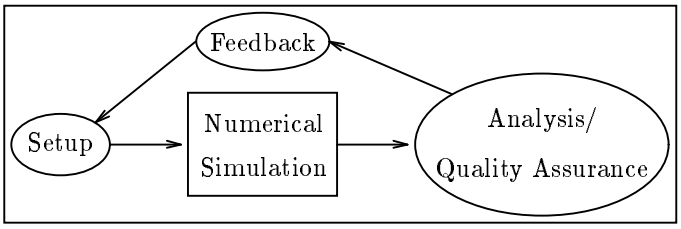

Figure 1: Analyzing a physical situation

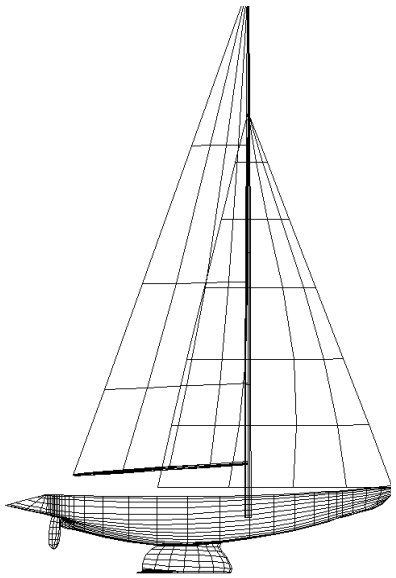

Figure 2: *Stars & Stripes*, winner of the 1987 America's Cup competition

- determine whether the output makes sense and how accurate it is likely to be, and if the output is not acceptable, to
- determine how to change the simulation program's input so that it will more reliably predict the behavior of the physical system being analyzed

As a result, these simulation programs typically can't be run successfully by inexperienced users. Perhaps more importantly, these simulation programs can't be reliably invoked by other programs. For example, human designers of complex objects like ships and airplanes typically run sophisticated simulation programs to analyze the object's physical behavior, but an automated system for designing complex objects could not easily include such a computational simulation as part the process it uses to evaluate new designs.

Artificial intelligence techniques seem essential in order to automate the processes of setup, analysis, qual-

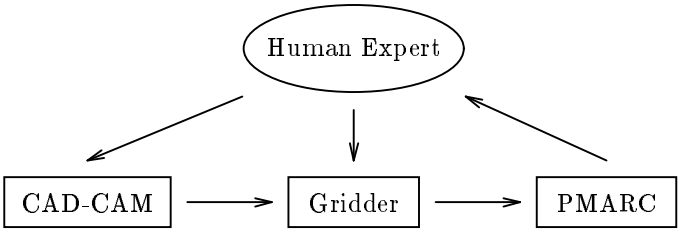

Figure 3: PMARC target environment

ity assurance, and feedback for computational simulation. However, simple application of known AI technology appears inadequate for the task of automating these processes. Basic AI research is needed, particularly in the areas of spatial reasoning and modeling physics and numerical methods.

Yachts

The Design Associate (DA) [Ellman *et al.* 1992] is an automated design system for racing yachts like the one in Figure 2. In the process of designing a yacht, the DA must repeatedly evaluate candidate yacht designs. A large number of these evaluations are required, so the capability to automatically evaluate the performance of a candidate yacht design without human intervention is crucial for the success of the DA.

Part of the process of evaluating a yacht's performance involves computing the yacht's *effective draft*, which measures the efficiency of its keel. Reliable computation of effective draft requires the use of computational fluid dynamics. For this purpose we use a program called PMARC, a product of NASA Ames Research Center. However, the PMARC target environment (Figure 3) is not compatible with automated use.

The input PMARC requires is a panelization — a discretization of a yacht's surface as a grid of planar panels. This panelization is normally created from a CAD/CAM model of the yacht by a human expert using an interactive gridding program. PMARC will produce poor results if applied to an inadequate grid, and using PMARC often requires several iterations of a loop in which the human expert looks at PMARC output, decides it is unacceptable, and uses the interactive gridding program to modify the grid to improve the PMARC output. In order for the Design Associate to evaluate candidate yacht designs without human intervention, PMARC must be invoked by an automated intelligent controller which can

- Use spatial reasoning to form appropriate grids, etc. for input to PMARC
- Use relevant physics and numerical analysis to
  - verify the quality of PMARC output
  - modify PMARC input to improve output quality

so PMARC can run reliably without the need for a human expert.

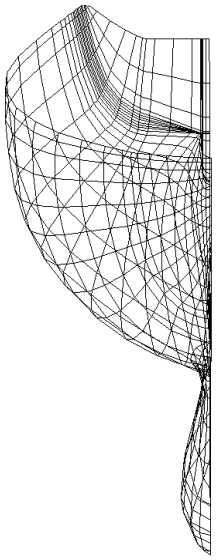

Figure 4: *Stars & Stripes* CAD/CAM surfaces for hull and keel

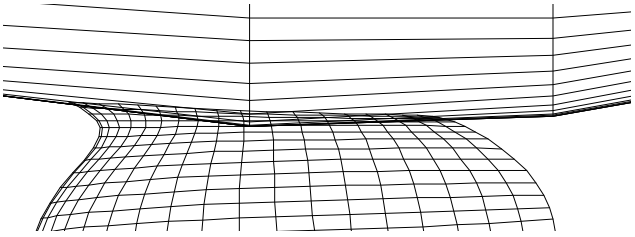

Figure 5: *Stars & Stripes* hull-keel intersection

Setting up the simulation

Considerable work has been done on the problem of automated gridding [Thompson *et al.* 1985], and many gridding programs have been developed. These programs are good at forming grids on surface patches, which is part of what is needed to run PMARC automatically. However, these programs are not capable of finding an appropriate set of patches to grid or of choosing appropriate input parameters for a gridding algorithm (e.g. how many panels to use to cover a patch). The programs can't make these decisions because they can't do spatial reasoning or reason about physics and properties of numerical methods.

Figure 4 illustrates one difficulty the intelligent controller for PMARC must deal with, which is that the input does not have a unique interpretation as a solid object. The human user of an interactive gridding program must use spatial reasoning in order to realize that the surfaces given cover half the boat and that the complete boat also must have surfaces on the other half of the boat and the top. Another difficulty is shown in Figure 5. A close examination of the place where the hull and keel meet reveals that the surfaces don't meet

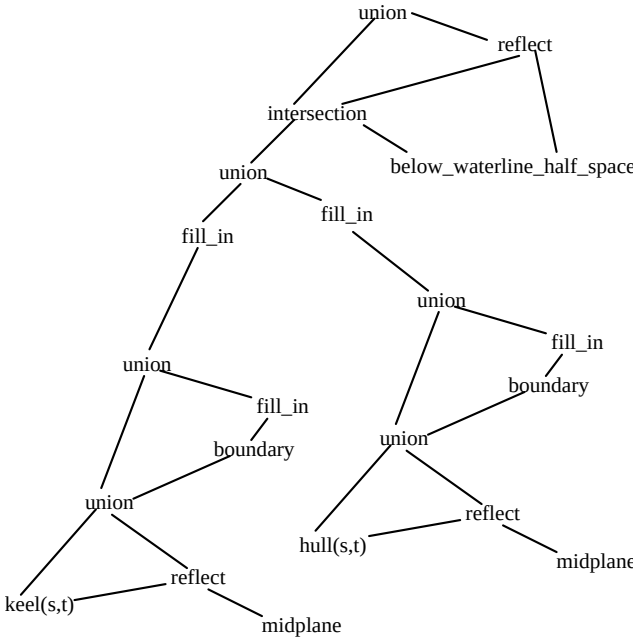

Figure 6: ECSG representation for yacht

neatly at their edges but in fact overlap each other. This overlap is useful because it allows the hull and keel to be modified independently while still remaining in contact. However, the overlap adds additional difficulties to the gridding process because the panelization for PMARC must include only the real parts of the CAD/CAM surfaces, not the fictional parts that are actually inside the yacht. An interactive gridding program has no spatial reasoning capability, and thus must rely on a human user to distinguish the real and false surfaces.

The first requirement for an intelligent PMARC controller is therefore a semantically clear input representation. Since standard CAD/CAM representations like B-spline surfaces appear to do an adequate job of representing surface details, our solution is to embed these CAD/CAM surface representations in a semantically clear solid modeling representation, which we call ECSG (extended Constructive Solid Geometry). ECSG is an extension of the standard Constructive Solid Geometry [Requicha 1980] solid modeling representation by adding additional operations and primitive shapes. The standard CSG set operations and rigid motions are supplemented by operations to take the boundary of solids and surfaces and operations to fill in the interior of closed surfaces and curves. The additional primitive shapes allowed are parametric surfaces like the B-spline surfaces representing the yacht shown in earlier figures.

Figure 6 shows an ECSG "tree" for a yacht. The internal nodes are operations which are applied to the children of the node. The leaves are primitive shapes. For example, hull(s,t) is a parametric surface representing half a hull. When it is reflected about the yacht's midplane, the other half of the hull's surface is generated. The union of these two surfaces then gives a surface for the hull which is open at the top. The boundary of that surface is a closed curve, which when filled in gives the top of the hull. We take the fill-in of a closed curve in space to be a minimum-energy surface like that of the bubble that would form if the curve were dipped in soapy water. The union of that with the rest of the hull gives a closed surface, which may be filled in to give a solid hull shape. The union of the solid hull and the solid keel gives a solid boat. PMARC can't handle water-air boundaries, so instead we chop the boat off at the water line and then reflect everything below the surface above it.

As mentioned above, existing automated gridding algorithms are good at forming grids on surface patches, but they are not capable of finding an appropriate set of patches to grid. For PMARC, the panelization of a surface patch is a matrix of adjacent panels. Thus every row in a patch must have the same number of panels, and every column in a patch must have the same number of panels. Whenever two adjacent parts of the surface require different numbers of panels, they must be input as separate patches. For example, the hull and keel would typically be separate patches.

Normally, the selection of patches is done by the human user of an interactive gridding program. In order to automate this process, we propose to use a known technique, streamline-based adaptive gridding, in a novel way. Streamline-based adaptive gridding involves running a computational fluid dynamics program several times. The grid for each run is constructed from the fluid streamlines found in the output of the previous run. Streamline-based adaptive gridding has been used successfully by other researchers to form grids for given patches [Chao and Liu 1991]. We propose to use the same technique as a way to recognize natural patch boundaries by looking for adjacent streamlines of different lengths. For example, the streamline on the keel nearest the hull will be quite near a streamline on the hull, but the streamline on the hull will be much longer, suggesting that the two should be in separate patches.

For the first iteration, we will create an initial set of "streamlines" either by adapting solutions of similar problems or by forming approximate streamlines based on flow direction and body contours. The implementation section of this paper describes an algorithm we have implemented which forms such approximate streamlines.

Quality Assurance

Computational simulations of physical systems can easily produce poor behavior predictions, or even complete nonsense. The human experts who typically run these programs must use their knowledge to judge the

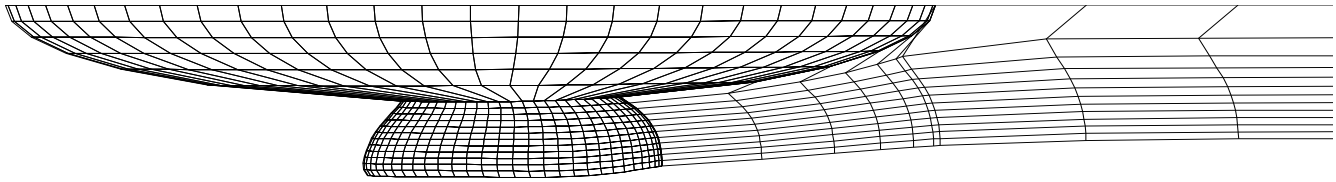

Figure 7: Panelization of hull, keel, and wake for *Stars & Stripes*

simulation output and modify the simulation input if the output is unsatisfactory. Robust automated simulation, which is essential for successful automated design, requires an intelligent controller which can apply expert knowledge to check simulation output quality.

Computational simulation of a physical system's behavior involves "executing" a model of the physical system. This model invariably involves approximations and simplifying assumptions. There appear to be two basic causes for "bad" simulations:

- The physical situation being analyzed is not compatible with the model being used
- The physical situation being analyzed is not compatible with the way the model is "executed"

If the simplifying assumptions of the model are violated by the physical situation, then the situation is not compatible with the model and the simulation will be unreliable. If the situations is compatible with the model, though, it still may not be compatible with the way the model is being used. For example, if the model is a partial differential equation which adequately captures the physics of the situation, a bad simulation can still result if the equation is solved using an inappropriate numerical method or too coarse a grid.

PMARC is based on a potential flow model. The key simplifying assumption behind this model is that the effects of viscosity are not important in the situation being considered, and may therefore be neglected. This assumption tends to hold for streamlined bodies like a yacht. However, an automated design system might change the yacht's shape enough to seriously impair its streamlining. Thus it is desirable to be able to test this simplifying assumption. One way to turn this assumption into an operational test is to look for predictions by PMARC of excessively high velocities. High velocities indicate unrealistic flows since viscous effects would tend to prevent excessive rises in velocity. Recognition of what velocities are "too high" is best done with a database of simulation results for similar situations.

Detection of problems in model execution is also more feasible with a database of past simulation results. These problems can be detected by applying knowledge of either relevant physics or relevant numerical analysis. For example, physics indicates that the flow around a body should exhibit "stagnation points": when the flow hits the front of a body it has to go around one side or the other, and where it divides the velocity must be zero. Such physics knowledge can be represented by a set of feature extractors, which examine a PMARC solution for features resembling stagnation points, and a set of feature evaluators, which judge whether an expected feature is sufficiently well resolved by the current grid.

c	hs	ks	T_{eff}	min c_p	max c_p
14	2	2	2.26864	-0.447619	0.36943
28	4	4	2.15689	-0.797459	0.747343
56	8	8	2.13218	-1.35133	0.958627
112	16	16	2.12021	-2.31862	0.999368

Figure 8: Simulation database entry for *Stars & Stripes*

Current Implementation

Our initial goal was to implement a complete working version of an intelligent PMARC controller which would be fully integrated with the Design Associate. We have achieved this goal for a certain subclass of possible inputs, the set of yachts consisting of a single hull and single keel with no other appendages.

Our current implementation does not include streamline-based adaptive gridding. The intelligent controller presently generates a grid that approximates the set of streamlines that we expect PMARC to compute. This grid allows PMARC to compute the initial solution which will be needed as input for streamline-based adaptive gridding.

Our current automatic gridder uses what we call "depth-based gridding". For a deep body with the same cross-section at every depth, streamlines will always remain at the same depth. A yacht can be considered a (major) perturbation of such a body. Our current implementation forms a grid with grid lines at constant depth, with some modification of the grid to follow body contours, thus better approximating actual streamlines. Figure 7 shows a panelization produced by our current implementation.

PMARC requires as input not only a panelization of a yacht, but also a panelization of the vortex wake shed by the yacht. Our intelligent controller constructs this wake by first running PMARC in a nonlifting orientation without a wake and then using the streamlines computed in that first PMARC run to form a wake to use in subsequent PMARC invocations.

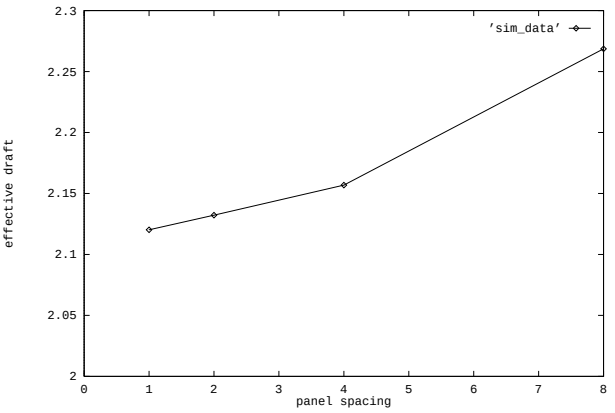

Figure 9: Plot of simulation data

Simulation quality assurance in our current implementation makes use of a database of past simulation results for similar shapes. Figure 8 shows an entry in this database. The first three columns specify the number of panels in the direction of flow, on the hull perpendicular to the flow, and on the keel perpendicular to the flow. The next column shows the effective draft (in meters) computed by PMARC to measure keel performance. The last two columns show the minimum and maximum pressure coefficients computed by PMARC. A c_p value of 1, the highest possible, indicates a stagnation point where flow velocity is zero. As discussed earlier, physics predicts that a stagnation point must exist, so the maximum computed c_p values measures how well the current grid resolves the stagnation point. The more negative a c_p value is, the higher the corresponding velocity (by Bernoulli's principle), so very negative c_p values may indicate a poorly streamlined body which violates PMARC's potential flow assumption.

The intelligent controller uses the database as follows:

1. Use database to choose coarsest grid likely to be converging
2. Run PMARC with that grid and next finer grid
3. Check c_p extrema for quality assurance violations
4. Extrapolate to estimate fully converged solution
5. Estimate error

The numerical method used by PMARC has a theoretical rate of convergence directly proportional to the panel spacing, and the intelligent controller makes use of this knowledge to choose an initial grid based on data in the simulation database. Figure 9 shows a plot of the simulation data in Figure 8 which indicates that three of the four grids are converging as predicted, while the coarsest of the four grids is not fine enough to be in the region of full convergence. The intelligent

c	hs	ks	j_c	j_h	j_k	min c_p	max c_p
44	11	11	.086	.06	.05	-1.76	.925
58	9	9	.065	.069	.056	-1.47	.955
60	10	8	.063	.065	.058	-1.43	.967
62	10	7	.061	.065	.061	-1.37	.98

Figure 10: Adaptive gridding

PMARC controller therefore runs PMARC in a new situation using the two middle grids in the simulation database. For each run, it checks that the maximum computed c_p value indicates that the stagnation point is being resolved as well as might be expected given the data from past simulations. The controller also checks that the most negative c_p is not much more negative than would be expected given the database.

The controller then applies knowledge of PMARC's theoretical rate of convergence to the two effective draft computations for the current situation and extrapolates to predict what the effective draft would be with an infinitely fine grid. (In effect, it draws a line through the appropriate two grid points in Figure 9 and extends the line to find out where it crosses the zero panel spacing axis.) The controller then uses the database to estimate the error in this extrapolation by comparing an extrapolation based on the two finest points in the database to one based on the two middle points. For example, the extrapolated value for effective draft using the two finest points in Figure 8 is 2.10824 m while the extrapolated value using the two middle points is 2.10747 m, so the error in extrapolation using the two middle points should be roughly 1 mm. Each halving of panel spacing increases the time to run PMARC by one to two orders of magnitude.

While our current implementation does not include streamline-based adaptive gridding, we have used it to experiment with other forms of adaptive gridding. We are particularly interested in applying knowledge of the numerical methods and approximations being used to the problem of choosing appropriate solution characteristics to use for feedback in adaptive gridding. PMARC uses a constant-coefficient approximation of the potential on each panel, which suggests adapting the grid to try to minimize the jumps in potential between panels. Figure 10 shows the results of such a feedback experiment. The column headings are the same as in Figure 8 except for j_c, j_h, and j_k which show the maximum jump in potential in the direction of flow, on the hull perpendicular to the flow, and on the keel perpendicular to the flow. The c_p values are not used for feedback, yet they are nevertheless improved by the adaptation process: the maximum c_p value rises, indicating that the stagnation point is being resolved better, and the minimum c_p value becomes less negative, indicating that artificially high velocities due to a poor grid are being reduced.

Related Work

Jambunathan *et al.*[1991] and Andrews[1988] discuss the use of expert systems technology to augment more traditional computational fluid dynamics programs. Most other artificial intelligence research concerning reasoning about physical systems has focused on qualitative rather than numerical simulation [Weld and de Kleer 1990]. Exceptions are the work of Gelsey[1990], Sacks[1991], Yip[1991], and Zhao[1991]; however, they have focused on numerical simulators for ordinary differential equations and have not addressed the issue of quality assurance. Forbus and Falkenhainer[1990] discuss the use of qualitative simulation to check the quality of numerical simulation results; however, the approach described appears limited to physical situations modeled by ordinary differential equations.

Conclusion

While the problems of setup, analysis, quality assurance, and feedback for computational simulation are clearly of considerable practical importance, their solution should also lead to significant insights into some basic AI problems. Successful computational simulation of complex physical systems appears to require a combination of spatial reasoning and reasoning about physics and numerical methods which has so far received insufficient attention from AI researchers.

Acknowledgments

The research on automated use of PMARC was done with fellow Rutgers Computer Science Dept. faculty member Gerard Richter and graduate student Ke-Thia Yao. We worked with hydrodynamicists Martin Fritts and Nils Salvesen of Science Applications International Corp., and John Letcher of Aero-Hydro Inc. Our research is part of the CAP (AI and Design) project, and benefited significantly from interaction with other members of the project. The CAP project is supported by the Defense Advanced Research Projects Agency and the National Aeronautics and Space Administration under NASA grant NAG2-645.

References

Alison E. Andrews. Progress and challenges in the application of artificial intelligence to computational fluid dynamics. *AIAA Journal*, 26(1):40–46, January 1988.

Y. C. Chao and S. S. Liu. Streamline adaptive grid method for complex flow computation. *Numerical Heat Transfer, Part B*, 20:145–168, 1991.

T. Ellman, J. Keane, and M. Schwabacher. The Rutgers CAP Project Design Associate. Technical Report CAP-TR-7, Department of Computer Science, Rutgers University, August 1992.

Kenneth D. Forbus and Brian Falkenhainer. Self-explanatory simulations: An integration of qualitative and quantitative knowledge. In *Proceedings, Eighth National Conference on Artificial Intelligence*, Boston, MA, 1990. AAAI-90.

Andrew Gelsey. *Automated Reasoning about Machines.* PhD thesis, Yale University, April 1990. YALEU/CSD/RR#785.

K. Jambunathan, E. Lai, S. L. Hartle, and B. L. Button. Development of an intelligent front-end for a computational fluid dynamics package. *Artificial Intelligence in Engineering*, 6(1):27–35, 1991.

Aristides A. G. Requicha. Representations for rigid solids: Theory, methods, and systems. *ACM Computing Surveys*, 12:437–464, 1980.

Elisha P. Sacks. Automatic analysis of one-parameter ordinary differential equations by intelligent numeric simulation. *Artificial Intelligence*, 48(1), February 1991.

Joe F. Thompson, Z. U. A. Warsi, and C. Wayne Mastin. *Numerical grid generation : foundations and applications.* North-Holland : Elsevier Science Pub. Co., 1985.

Daniel S. Weld and Johan de Kleer, editors. *Readings in Qualitative Reasoning about Physical Systems.* Morgan Kaufmann, San Mateo, California, 1990.

Kenneth Yip. Understanding complex dynamics by visual and symbolic reasoning. *Artificial Intelligence*, 51(1–3), October 1991.

Feng Zhao. Extracting and representing qualitative behaviors of complex systems in phase space. In *Proceedings, 12th International Joint Conference on Artificial Intelligence*, 1991.