# Responsive FEM for Aiding Interactive Geometric Modeling

Nobuyuki Umetani[*]
The University of Tokyo

Kenshi Takayama
The University of Tokyo

Jun Mitani
University of Tsukuba

Takeo Igarashi[†]
The University of Tokyo, JST/ERATO

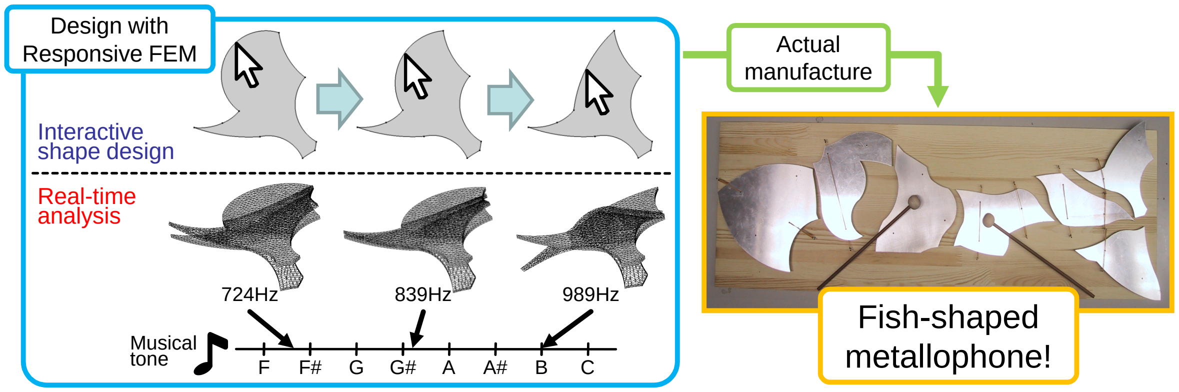


**Figure 1:** *Overview of our responsive FEM concept applied to the example of metallophone design. We always run real-time FEM analysis during interactive geometric design sessions, allowing the user to create real-world objects with physically desirable properties.*

## Abstract

Numerical simulation methods are used mainly as off-line verification tools in current computer-aided engineering systems to reject designs that fail to satisfy the required constraints, rather than to guide the user toward a better design. This paper presents the integration of a real-time finite-element method (FEM) analysis into interactive geometric modeling. Real-time feedback from numerical simulation during interactive editing can guide users toward improving their design without tedious trial-and-error iterations. We achieve fast FEM analysis during interactive editing by carefully reusing previous computation results such as meshes and matrices based on speed and accuracy trade-offs. We implemented several example applications to demonstrate the versatility of the system. We also present two informal user studies, metallophone and bridge design, to show the effectiveness of the approach. We envision that our tools can assist nonexpert users to design objects that satisfy physical constraints and help them understand the underlying physical principles.

**Keywords:** Modeling - Modeling Interfaces, Modeling - Geometric Modeling, , Modeling - CAD, Methods and Applications - Education, Real-time FEM

## 1 Introduction

Interactive numerical simulations can be a powerful tool for assisting the design of various items that satisfy specific physical requirements as described in Sutherland's SketchPad [1964]. However, most numerical simulation methods today, generally referred to as computer-aided engineering (CAE) tools, are used for the off-line verification of a given design. They are used to reject designs that fail to satisfy the requirements, but are not usually used to explore a better design. Real-time simulation is emerging, but typical applications are the simulation of deformation in animation [Mezger et al. 2008] and virtual training [Chentanez et al. 2009]. Real-time numerical simulation is not widely used as a tool for designing physical items.

This paper introduces our efforts at integrating a numerical simulation method into geometric modeling as described in Sutherland's [1964] vision. Our system runs a finite-element method (FEM) simulation in real time that responds to dynamic user input during geometric editing (Fig. 1). Unlike standard real-time FEMs for deformation that are applied to a single given initial geometry, our method continuously updates the simulation results responding to the initial geometry being modified. Real-time feedback during editing can provide guiding principles for better design and help the user approach a satisfactory design while avoiding the many trial-and-error experiments necessary with an off-line simulation. Responsive feedback is also useful for educational purposes.

The technical contribution of our work is the way in which we modify the traditional FEM framework to make it responsive, that is, to make it efficiently update the computation result responding to the continuously changing initial boundary geometry. The key observation is that the geometry only gradually changes when the user modifies the boundary by direct manipulation (i.e., by dragging the mouse). In this case, the mesh only slightly changes and then most matrix computations can be reused to accelerate the computation. The important questions are which computations to reuse and when. To answer these questions, we decompose the computation into multiple reusable components and perform the appropriate amount of recomputation by monitoring the dynamic user input. When the modification is small, the system only slightly updates the mesh, and most matrix computations are reused. For the large modification, the system gradually makes larger changes to the mesh and

[*]E-mail: n.umetani@gmail.com
[†]E-mail: takeo@acm.org

updates more matrix computations to maintain accuracy. We show that this method significantly improves the performance compared simply to running a monolithic FEM each time.

We present several example applications to explain our concept including structure vibration, structural analysis, fluid, and thermal fluid. We also performed two informal user studies to determine the effectiveness of our approach. One was to ask a professional artist to design a customized metallophone using responsive FEM analysis. The other was to ask nonprofessional test users to design a bridge with and without responsive FEM. Although our current implementation is limited to 2D problems with simplex first-order elements, the basic concept of responsive FEM is independent of dimensionality and element types.

Our contribution is summarized as follows:

- We propose a responsive FEM framework in which the simulation result is continuously presented to the user during geometric editing.
- We introduce a solid implementation to support the vision. It incrementally updates the FEM data structure to avoid redundant computation.
- We present several example applications, each of which is innovative and useful in itself.
- We conducted two informal user studies to demonstrate the effectiveness of our approach.

## 2 Related Work

Automatic optimization methods can be used to design objects that satisfy physical constraints. For example, Smith et al. [2002] applied an optimization approach to design truss structures. However, automatic methods have many difficulties in practice such as difficulty in explicitly specifying constraints and parameter spaces that are too large. The interactive approach is advantageous in that the users can use their own preferences and judgment during the design while considering other less tangible factors such as aesthetics.

Various methods have been proposed for making physical simulations interactive. One approach is to use precomputations. James and Pai [1999] presented an interactive physical simulation of deformable objects by precomputing the matrices of reference boundary value problems. Another approach is to use approximation for the nonlinear elasticity to achieve large deformations quickly and stably [Müller et al. 2002]. These methods take user input as an external force in a continuously running simulation. In our system, the user directly modifies the initial geometry and the system reruns the simulation with the new geometry.

Our goal is to aid the design process using physical simulation. A number of CAD systems provide an embedded simulation tool for design evaluation. These systems have been developed to switch environment seamlessly from design to off-line simulation. Masry and Lipson [2005] presented a sketch-based 3D modeling interface capable of FEM evaluation in the early stages of the design. However, these approaches are not very different from conventional CAD and CAE systems in that the analysis is performed *after* some modeling process has occurred, while the simulation is performed *during* the modeling in our system. Several systems have been proposed for end users to design physical objects such as stuffed animals with the aid of interactive simulation [Mori and Igarashi 2007]. However, the focus in those systems was mainly on the user interface and they used simplified simulation methods.

## 3 Algorithm

This section describes how to make the FEM framework responsive, that is, to provide immediate feedback during geometric editing. We achieve this by maximizing the reuse of intermediate computation results and carefully scheduling the computation pipeline to provide the best user experience. We first briefly describe the basic FEM framework as the basis of our algorithm. We then describe our proposed method to make FEM responsive, followed by detailed description of our current implementation.

### 3.1 FEM background

FEM finds an approximate solution of partial differential equations by spatially discretizing the field. The system first constructs a mesh inside of the boundary geometry and then solves a linear system $A\boldsymbol{x} = \boldsymbol{b}$ that is defined by the relationships among nodes (note that for nonlinear problems we need to solve such linear systems iteratively). Since the matrix $A$ is sparse, it is compactly represented by the combination of the nonzero pattern $A_p$ that represents the location of nonzero elements, and the value list $A_v$ that represents the values at the nonzero elements. Iterative methods are commonly used to solve sparse linear systems and their performance is often improved using a preconditioner. A preconditioner $B$ is used to approximate the inverse of $A$ which is not necessarily sparse. $B$ is usually represented as a sparse matrix with its nonzero pattern $B_p$ and the value list of the nonzero elements $B_v$.

These data ($A_v$, $A_p$, $B_v$, and $B_p$) must be constructed before actually solving the system. The construction of the mesh and the matrices can be considered as a precomputation. When solving a linear problem, the system runs the entire precomputation only once. The system finds a solution without changing the matrices and reuses them multiple times to solve a time-varying problem. In contrast, the system needs to solve the problem iteratively updating $A_v$ and $B_v$ each time to solve a nonlinear system.

Traditional FEM frameworks run the reconstruction of mesh and matrix computations all at once for every change of the geometry. When the user applies the same analysis to even a slightly modified geometry, the system discards the result of all precomputations and starts construction of the mesh and matrices from scratch. This is a waste of time because most of the computations are redundant and can be reused. The next section describes how we modify the FEM computation process to achieve this goal.

### 3.2 Our approach: multilevel reuse

In our system, the user modifies the boundary geometry by dragging vertices, edges, or regions, and the system continuously runs FEM analysis on the domain. Note that in the case of structural analysis, the user modifies the rest shape, not the deformed shape emerging as a result of simulation. The challenge is to provide immediate feedback to the user while maintaining a certain level of accuracy. Making a system *responsive* is not the same thing as simply making the system *fast*. One needs to be careful in distributing the computation corresponding to the degree of change in data caused by the user to maximize the speed–accuracy trade-off. We achieve this goal by reusing intermediate computation results instead of recomputing everything every time the boundary geometry is modified.

The basic concept is as follows. When the geometric modification is small, we can reuse most of the previous intermediate computation results to obtain an accurate result. If the accumulated geometric modification becomes too large, then we stop reusing previous results and run the costly computation to maintain accuracy. To implement this concept in a FEM framework, we divide the computation into multiple stages and choose the appropriate amount of recomputation depending on the current situation.

As we described in the background section, the FEM main computation is divided into mesh construction and matrix computation. When the boundary geometry is modified, then the mesh and matrices need to be recomputed. We define three levels of recomputation and choose the appropriate one to balance speed and accuracy (Table 1). Note that the reusing techinique doesn't change the final solution from regular FEM analysis if the mesh is same, because the same coefficient matrix is solved iteratively with a same convergence criterion. Continuous update of a mesh during simulation is already used to solve problems that involve geometry changes such as fluid–structure interaction based on Arbitrary Lagrangian Eulerian methods. However, such off-line methods do not selectively apply different update procedures responding to the user input as in our method.

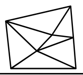
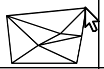
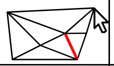
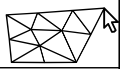
| | User operation | Idle | Dragging | | |
|---|---|---|---|---|---|
| ✓ = reusable | |  | Level 1 (Relocation)  | Level 2 (Reconnecting)  | Level 3 (Reconstruction)  |
| Coefficient matrix $A$ | Value list $A_v$ | ✓ | | | |
| | Non-zero pattern $A_p$ | ✓ | ✓ | | |
| Pre-conditioner matrix $B$ | Value list $B_v$ | ✓ | ✓ | ✓ | |
| | Non-zero pattern $B_p$ | ✓ | ✓ | ✓ | |

**Table 1:** *Multilevel reuse. A check mark indicates that the data can be reused. A blank means that the data needs to be recomputed.*

*Level 1.* When the modification of the boundary geometry is small, we only change the position of the mesh nodes (relocation). This does not change the topology of the mesh. Therefore, we only need to update the value list of the linear system ($A_v$), while reusing all the other data ($A_p$, $B_v$, and $B_p$). We can also reuse the FEM solution in the previous configuration. Since the nodes are moved only slightly, the solution (field values at the nodes) does not change very much. We therefore reuse it as an initial guess in running an iterative solver; this is faster than starting from an arbitrary guess.

*Level 2.* When the modification of the geometry becomes larger, node relocation is not sufficient to eliminate distortions in the mesh and we change the topology of the mesh locally to improve the mesh quality (reconnecting). In this case, we need to update the nonzero pattern $A_p$ as well as the value list $A_v$. However, we can still reuse $B_v$ and $B_p$ because nodes are not added or deleted, and they are only slightly moved. Reuse of the preconditioner is a known technique, but it is used mainly for solving nonlinear problems. The solution can also be reused as the initial guess in the iterative solver as in the Level 1 case.

*Level 3.* Even reconnecting is not sufficient when the modification of the geometry is significantly large. In this case, we stop the incremental update of the mesh and reconstruct the entire mesh from scratch (reconstruction). This might sound too radical, but a global reconstruction is often faster and yields a better mesh than local optimization with node insertion and deletion when the distortion has accumulated or the boundary geometry is too different from the current boundary. In this case, we recompute all data: $A_v$, $A_p$, $B_v$, and $B_p$. In addition, we cannot reuse the previous solution because the old nodes are completely replaced by new ones. Therefore, we need to start with a new initial guess.

We reuse FEM data to maximize the responsiveness of the analysis by considering the cost required for each level of recomputation; we try to rely mostly on the lightweight Level 1 recomputation while performing the expensive Level 3 recomputation only when necessary. The basic concept described above applies to both linear and nonlinear problems. However, the details are slightly different in nonlinear cases. Specifically, the value lists $A_v$ and $B_v$ need to be updated each time when solving a nonlinear system, so we cannot reuse them. However, we can still reuse the nonzero patterns $A_p$ and $B_p$, which significantly contributes to improving the performance.

## 3.3 Implementation details

The reuse of FEM data is divided roughly into the reuse of the mesh and the matrix computations. The multilevel reuse first determines what part of the mesh structure to reuse and then uses this to decide what part of the matrix computation to reuse. The system changes the mesh to a limited extent of element destortion. When the user edits the boundary geometry, the system first relocates the nodes to minimize mesh distortion. If the system detects an inverted element after the relocation, the system pushes the nodes back to the previous positions and applies reconnecting. If reconnecting does not occur, it means that reconnecting does not improve the mesh quality and the system reconstructs the entire mesh. If no inverted element is detected after relocation, the system checks for the existence of distorted elements. If distorted elements exist, the system applies reconnecting.

We use a simple mass-spring system for the node relocation. The rest length of spring is zero and we solve equilibrium iteratively. Since the nodes move only slightly each time, the computation converges quickly. An even number of iterations is desirable to avoid oscillation; we currently perform two. The distortion metric is based on the ratio of an inscribed circle and the maximal edge length. We apply edge swapping for reconnecting. The criterion of an edge to be swapped is whether the edge violates the Delaunay condition. Mesh reconstruction is based on Delaunay triangulation and local optimization.

The conjugate gradient method is used for solving symmetric matrices, while the Bi-CGSTAB method is used for solving asymmetric matrices. We improve the convergence of these iterative methods by using the preconditioner based on the incomplete LU factorization with level of fill-in (ILU(k)). The ILU factorization method computes a sparse matrix $B$ that approximates the inverse of a sparse matrix $A$ (note that the exact inverse of $A$ is not sparse in general). The method takes an integer called 'level of fill-in' as a parameter, which specifies the level of the approximation. It affects both the improvement of the convergence and the cost of the factorization; the higher the level, the more closely $B$ approximates the inverse of $A$ leading to a faster convergence, while requiring more computations for the factorization. A preconditioner with a high level fill-in benefits more from our multilevel reuse scheme, because the number of the preconditioner recomputation is much reduced in our method. However, the best level of fill-in is heavily dependent on the target problems, and we experimentally chose appropriate ones for each application (e.g., we used the three level of fill-in for the vibration analysis and the cantilever deformation examples, while we used the zero level of fill-in for the fluid and the thermal fluid examples in Section 4).

## 3.4 Performance

We demonstrate the effectiveness of our multilevel reuse described above through an example modeling sequence shown in Figure 2. Table 2 shows how many times each level of recomputation occurred during the mouse dragging. It shows that the Level 1 recomputation accounts for a large share of the total computation while the Level 3 recomputation occurs only occasionally. Figure 3 shows the cost required for each level of recomputation. It is measured on the same FEM problem as the vibration analysis example in Section 4, tested with a 2.5-GHz CPU and 2.0 GB of RAM.

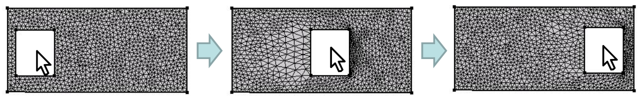


**Figure 2:** *An example modeling sequence used for the performance measurement. The user continuously drags the hole from left to right. The mesh consists of 1952 elements.*

| | Level 1 (Relocation) | Level 2 (Reconnecting) | Level 3 (Reconstruction) |
|---|---|---|---|
| # of occurrences | 117 | 39 | 3 |

**Table 2:** *The frequency of each recomputation during the example modeling sequence shown in Figure 2.*

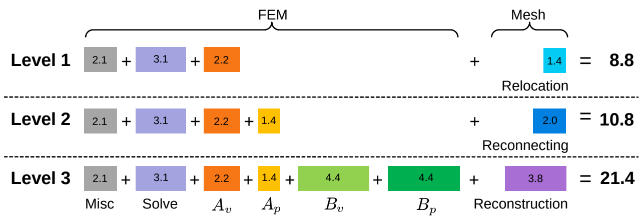


**Figure 3:** *The cost required for each level of recomputation (ms). The Level 3 recomputation is more than twice as expensive as the Level 1 recomputation, which is mainly due to the cost required for the construction of the preconditioner ($B_v$ and $B_p$).*

## 4 Application Examples

We show several preliminary examples of applying a responsive FEM framework to typical 2D design problems. In each of these examples, the user interactively manipulates the shape of an object within a certain physical constraint and the system returns the analysis result in real time. We envision that these responsive simulations for geometric modeling will be useful for both early exploration of new design problems and refinement of designs that are almost finished. We present the current implementations primarily as a proof of concept to clarify this vision. They may not necessarily be useful for practical real-world applications; building practical applications based on these examples remains a subject for future work. Still, we believe that the current implementation is already useful for some end-user design problems as shown in the next section, and to teach the general principles of physical phenomena.

**Vibration analysis of a structural object.** In this example, a structural object is fixed to the ground that is shaking constantly at a certain frequency, causing the entire structure to deform (Fig. 4). Resonance behavior appears when the user manipulates the object into a specific shape, one that would only be predictable through the use of simulation. We expect this application to be much more sophisticated in the future to help in the design of a building that would avoid collapsing due to resonance caused by an earthquake or wind.
*Equation.* The analysis is based on a nonstationary 2D linear solid without gravity:

$$\rho \ddot{\boldsymbol{u}} = \nabla \cdot \boldsymbol{\sigma} \tag{1}$$
$$\boldsymbol{\sigma} = \lambda (\mathrm{tr}\varepsilon) \boldsymbol{I} + 2\mu\boldsymbol{\varepsilon} \tag{2}$$

where $\boldsymbol{u}$ is the displacement, $\rho$ is the density, $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{\varepsilon}$ is the linearlized strain tensor, and $\lambda$ and $\mu$ are the elastic Lamé coefficients. The time integration is based on the Newmark-$\beta$ method.

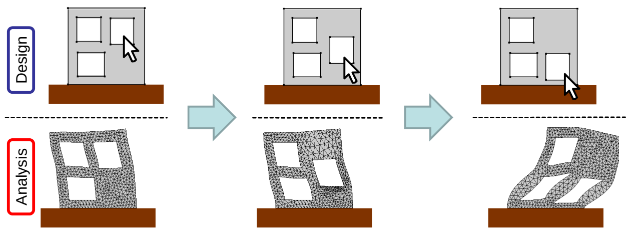


**Figure 4:** *Vibration analysis example. A structural object deforms due to the shaking movement of the ground. Notice that resonance occurs when the user moves the top right window toward the bottom, leading to a large destructive deformation. The displayed deformation is exaggerated for the purpose of visualization.*

**Cantilever deformation.** In this example, the leftmost part of a horizontal cantilever is fixed to a vertical wall while the remainder is free. The gravity causes the whole cantilever to deform (Fig. 5). This application can possibly be of benefit in the design of an airfoil, in which the designer is most interested in the hydrodynamic performance of the shape after the deformation caused by gravity and wind pressure, rather than the original undeformed shape. Automatic optimization is usually used for this kind of problem when the goal shape is clearly defined. However, the user may often have only vague ideas about the goal shape and wishes to try various designs before deciding on one; continuous feedback can be very useful in such cases. Also note that the design shape can be used as an initial guess for the automatic optimization problem.
*Equation.* We solve the St.Venant–Kirchhoff material equation:

$$\boldsymbol{S} = \lambda (\mathrm{tr}\boldsymbol{E}) \boldsymbol{I} + 2\mu\boldsymbol{E} \tag{3}$$

where $S$ is the second Piola–Kirchhoff stress tensor, $E$ is the Green–Lagrange strain tensor. Both $\lambda$ and $\mu$ are the same as in Eq. 2.

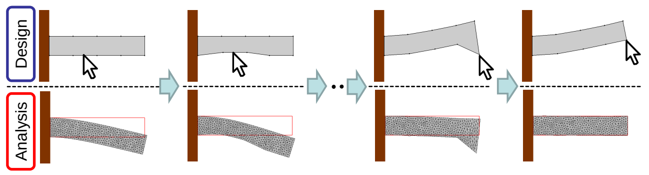


**Figure 5:** *Cantilever deformation. The user tries to fit the cantilever shape after deformation caused by gravity (bottom row) to a certain goal shape (shown in red lines) by continuously manipulating the undeformed shape (top row).*

**Fluid around an object.** In this example, an object is placed inside a space filled with air, and a certain velocity of wind blows constantly from left to right, creating complex flow around the object (Fig. 6). Depending on the object shape manipulated by the user, we can observe various kinds of phenomena such as boundary layer separation (Fig. 6a), which can cause a stall, or a Karman vortex street (Fig. 6b), which leads to an oscillation that may destroy the object. This application shows its potential utility for the design of various objects that are constantly exposed to a strong flow; this includes objects such as airfoils, car bodies, door mirrors, and air ducts.
*Equation.* We solve incompressible Newtonian flow stabilized with

the SUPG-PSPG algorithm [Tezduyar et al. 1992]:

$$\rho \frac{D\boldsymbol{v}}{Dt} = -\nabla p + \mu \nabla^2 \boldsymbol{v} \quad (4)$$

$$\nabla \cdot \boldsymbol{v} = 0 \quad (5)$$

where $\boldsymbol{v}$ is the velocity, $\rho$ is the density, $p$ is the pressure, and $\mu$ is the viscous modulus. We used the implicit method for time integration.

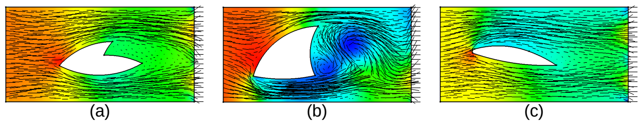


**Figure 6:** *Fluid around an object. The velocity field is displayed as line segments, while the pressure is visualized as color contours. As the user manipulates the object shape, various phenomena can be observed such as boundary layer separation (a) and a Karman vortex street (b).*

**Thermal fluid inside an object.** In this example, some kind of fluid (e.g., water) fills an object (e.g., a teapot) whose bottom is heated while other boundaries are constantly cooled. We observe how the complex nonstationary behavior of the thermal fluid changes according to the object shape manipulated by the user (Fig. 7). In addition to the design of a heat-efficient teapot, we expect this application could be useful for various problems concerned with thermal fluid phenomena such as the layout of room air conditioners or the design of a computer case.

*Equation.* We solve the Navier–Stokes equation with buoyancy proportional to the temperature, which is computed via a convection–diffusion equation:

$$\rho \frac{D\boldsymbol{v}}{Dt} = -\nabla p + \mu \nabla^2 \boldsymbol{v} + \rho \boldsymbol{g} \beta (T - T_0) \quad (6)$$

$$\nabla \cdot \boldsymbol{v} = 0 \quad (7)$$

$$\frac{DT}{Dt} = \alpha \nabla^2 T \quad (8)$$

where $T$ is the temperature, $T_0$ is the reference temperature, $\alpha$ is the thermal diffusivity, $\beta$ is the volumetric thermal expansion ratio, $\boldsymbol{g}$ is the acceleration of gravity, and $\boldsymbol{v}$, $\rho$, $p$, and $\mu$ are defined as in Eq. 4. We used the implicit method for time integration.

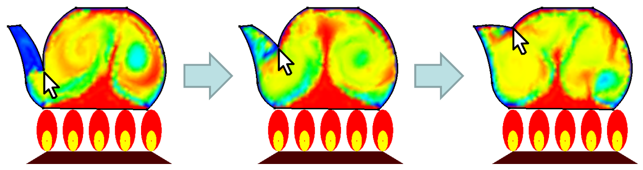


**Figure 7:** *Thermal fluid inside an object. The temperature is visualized as a color contour; blue and red correspond to low and high temperatures, respectively.*

**Performance.** Table 3 summarizes the performance of our application examples. Because the frames per second (FPS) depends on the user manipulation (i.e., the FPS decreases when the user makes a large shape modification very quickly), we averaged the FPS measured during the user manipulation which is similar to that in Section 3.4 at a moderate speed. These results show that our data reuse scheme allows our application examples to run at a quite high frame rate that is sufficient for the interactive modeling. We also measured the FPS without data reuse (the last column). It shows that our method significantly improves the performance and is particularly effective for linear or stationary problems.

| Title | linear | stationary | elem | fps (reuse) | fps (no reuse) |
|---|---|---|---|---|---|
| Vibration | yes | no | 1962 | 105.0 | 41.8 |
| Cantilever | no | yes | 990 | 37.6 | 7.1 |
| Fluid | no | no | 1971 | 30.0 | 18.2 |
| Thermal fluid | no | no | 1938 | 21.3 | 14.3 |

**Table 3:** *Performance of our application examples tested with a 2.5-GHz CPU and 2.0GB of RAM. The third and fourth columns show FPS with and without data reuse, respectively.*

## 5 Informal User Studies

We performed two informal user studies to verify the utility of responsive FEM, one with a professional artist and the other with nonprofessional students.

### 5.1 Metallophone design

We used responsive FEM for the design and actual creation of an artistically shaped metallophone to show that our current implementation is already useful as a practical tool for designing real-world objects. Metallophones are usually rectangular because that shape is suitable for predicting the instrument's tone analytically. The computer simulation has been applied for simulating sound from arbitrary shaped object. For example, Chadwick et al. [2009] proposed a computationally efficient framework to synthesize realistic sound from nonlinear thin shell vibration. However, we believe that designing a metallophone with a desired artistic shape and tone would be possible only with responsive FEM because this kind of highly constrained modeling naturally demands tight integration of design and analysis.

In our current implementation, the metallophone is modeled as a 3D thick plate extruded from the 2D shape designed by the user. Its tone, or frequency, is computed via eigenvalue analysis. The lowest nonzero eigenfrequency is assumed to be the output tone as other higher eigenfrequencies usually attenuate very quickly. The eigenmode, which tells how the metallophone oscillates, is computed along with the eigenfrequency because it is important for determining the positions on the metallophone to be fixed. Details on these computations can be found in the Appendix. Note that simulation parameters need to be calibrated for each specific material.

Figure 8 shows our software for metallophone design. The user can edit the shape in a 2D view in the left window while checking the eigenmode of the oscillation in a 3D view in the right window. The tone is updated in real time during the design, with aural feedback to the user using beep sounds and visual feedback in the status bar.

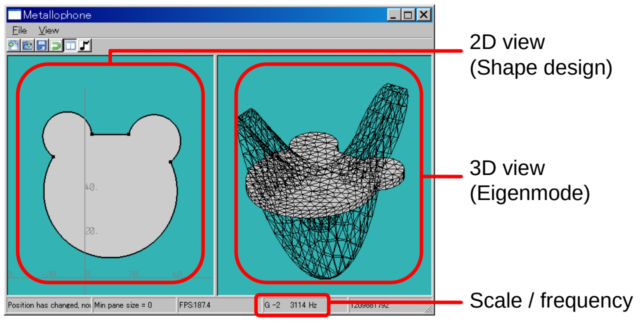


**Figure 8:** *Metallophone design software. The left window is used for the design in 2D, while the right window shows the analyzed eigenmode in 3D. The tone is updated in real time with both aural and visual feedback to the user.*

We asked a professional artist to design a metallophone using our software. We did not put any limit on the design time and provided instruction on the use of the software if required during the design period. Figure 9 shows the designed shapes corresponding to the sequence of musical scale notes from C (523Hz) to B (987Hz). We cut out these metallophone shapes from 4-mm-thick aluminum plate using a wire-electrical discharge machine, and fixed them onto a wooden board according to the analyzed eigenmodes (right side of Fig. 1). Top three rows in Table 4 show that the frequencies of the most pieces well conformed to each other for the target, the simulation, and the actual metallophone. To further improve the quality, we manually adjusted the tones of the actual metallophone pieces by trimming their edges (except for the piece of F). Note that our metallophone design software was also useful for this adjustment process, because it predicts the change of tone caused by the edge trimming. The last row in Table 4 shows the frequencies of the actual metallophone after the manual adjustment. We found that the sounds produced with the actual metallophone were of sufficient quality for a hobbyist.

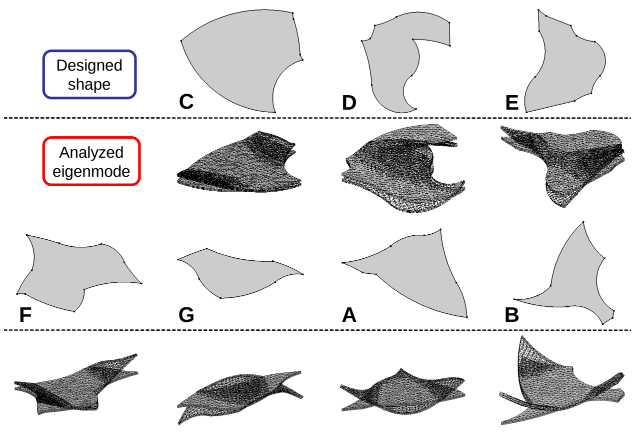


**Figure 9:** *The metallophone shapes designed by the artist. The upper row shows the designed 2D shapes while the lower row shows their analyzed eigenmodes in 3D. Note that the displayed eigenmode is much more exaggerated than the actual oscillation for the purpose of visualization.*

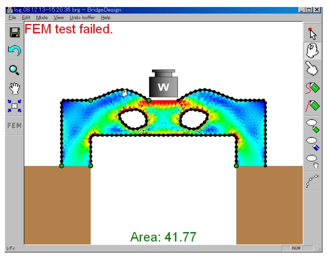
| Scale | C | D | E | F | G | A | B |
|---|---|---|---|---|---|---|---|
| Targeted | 523 | 587 | 659 | 698 | 783 | 880 | 987 |
| Simulated | 525 | 588 | 661 | 699 | 786 | 880 | 989 |
| Measured | 506 | 604 | 621 | 698 | 787 | 860 | 993 |
| Adjusted | 523 | 587 | 659 | 698 | 783 | 881 | 987 |

**Table 4:** *Target, simulated, measured, and adjusted frequencies of the metallophone for each musical scale, showing the accuracy of our analysis.*

We interviewed the artist after the design to obtain subjective comments. The artist reported that the design took roughly 5 h to complete, most of which was devoted to the C and D pieces. This was mainly because these lower tones tended to require larger areas than others, which greatly slowed the response of the analysis. One of the difficulties the artist found during the design was the requirement to consider overall shape balance among pieces while keeping their tones true to the intended ones; this was a huge design constraint. Another difficulty was that sometimes a small modification of the shape resulted in a large change in the tone; this demanded high responsiveness of the analysis. The artist noted that it would be almost impossible to design such an artistically shaped metallophone without using the responsive FEM.

## 5.2 Bridge design

The next study concerned the design of a bridge, with the aim of showing that responsive FEM can provide better support than traditional nonresponsive FEM for nonprofessionals in the design of objects with physically desirable properties.



**Task.** The task given to the users was to design the 2D shape of a bridge to span a certain gap and support a certain weight on its center, as shown in the inset. Its strength was tested through FEM analysis with the physical model based on equivalent stress. The system displays the amount of stress being applied to each region as color contours (blue and red correspond to low and high stress, respectively) and judges whether the bridge passes the test. The users were asked to design a bridge that passes this strength test with as small an area as possible. In other words, the goal was to design a strong bridge with the least amount of material.

The shape design software used in this study provides a set of tools such as pushpin-and-pull curve editing [Igarashi et al. 2005], curve smoothing, and holes creation. The area of the bridge is always displayed during the design. The software has FEM analysis functionality with two modes: responsive FEM mode and nonresponsive FEM mode. In responsive FEM mode, the analysis is always performed during the user interaction (i.e., mouse dragging) and the result is updated in real time. In nonresponsive FEM mode, however, the analysis is performed only when the user completes the design and presses a button on a toolbar. The analysis result immediately disappears when the user changes the design. This mode simulates the way most existing FEM systems are used, in which the design process and the analysis process are completely separate, and switching between these two involves a great deal of tedious work such as file export/import and FEM parameter setup.

**Experimental setup.** Six university students majoring in art and design participated in the study, all of whom were unfamiliar with FEM techniques and material mechanics. These participants were split into two groups, A and B. Participants in group A used the responsive FEM mode, while participants in group B used the nonresponsive FEM mode. Experiments for these two groups were performed separately. Each group was first given a 15-min lesson on software usage, followed by a 30-min main design session. After that, participants in each group were asked to try the other FEM mode in a follow-up session, and their subjective feedback on the two FEM modes was collected.

**Results.** Figure 10 shows the smallest area of the bridge that passes the strength test for each participant in the main design session. We observed that participants who used responsive FEM generally achieved better results than those who used nonresponsive FEM. The most common subjective feedback was that responsive FEM is very useful when the user wants to make a small adjustment to see how it affects the analysis. Another interesting feedback was that the analysis result displayed during the design in responsive FEM mode could be too conspicuous, making a large design change difficult. Some participants even pointed out that shape design without responsive FEM may be more appropriate for initial design exploration.

We should emphasize that the nonresponsive FEM mode used in this study is already much more efficient than current commercial FEM tools, which require many time-consuming procedures such

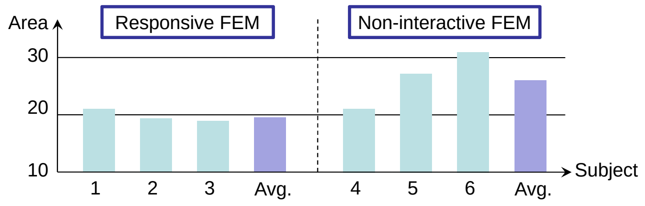


**Figure 10:** *Result of the bridge design study showing that subjects using responsive FEM generally achieved a better design.*

as switching between different tools and setting many parameters each time. This simple study obviously cannot *prove* anything but we believe that it at least *suggests* the potential of responsive FEM as a design aid for nonprofessionals, and it is our future work to perform more formal user studies.

## 6 Limitations and Future Work

**Limitations.** Most importantly, we need to extend our techniques to 3D so that they will be truly useful for many practical real-world problems. While solving linear systems in 3D itself is rather straightforward, the main challenge would be the continuous mesh update scheme in 3D. As noted by Labelle and Shewchuk [2007], existing methods for improving tetrahedral mesh quality by continuously moving nodes and changing connectivity have yet to guarantee sufficient quality for accurate simulation.

Another limitation is that currently we can use first-order elements only. Higher-order elements are much more desirable for some problems such as bending of thin-walled structure. However, using them may be problematic in our approach because the reconnecting of the mesh would probably change the relationships between the nodes and prevent the matrix reuse. In addition, we have yet to try non-simplex elements (e.g., quadrilateral in 2D and hexahedron in 3D) that can be more appropriate than simplex elements (e.g., triangle in 2D and tetrahedron in 3D) depending on the problems, although the reconnecting of such non-simplex meshes without adding and deleting points is generally known to be difficult.

Our approach cannot be applied to history-dependent problems such as plasticity processing because the solution in these cases needs to be computed sequentially from the initial state, and our data reuse scheme is inappropriate for that purpose.

**Future work.** We plan to test reusing various kinds of data other than the preconditioner matrix. This could include node reordering, which will also improve the responsiveness of analysis.

One future direction is to make the system more actively guide the design process using the result of simulation. For example, it would be useful if the system could assist the user design shapes that satisfy certain constraints (e.g., certain stress limits in certain areas) by guiding the user manipulation with instructions and suggestions whenever the user makes a design change that will not satisfy these constraints.

Another direction would be to let the user interactively control the simulation accuracy. We use fixed criteria for the speed–accuracy trade-off, but the user may want more explicit control over it during the design (i.e., the user may want more accuracy than speed in the design refinement stage, and vice versa). We also assume the homogeneous mesh density, but it would be useful if the user could control the simulation accuracy locally by manipulating the local mesh density. This would help the user examine the analysis results more closely in the specific region of interest, which would be difficult with existing automatic error estimation methods.

## References

CHADWICK, J. N., AN, S. S., AND JAMES, D. L. 2009. Harmonic shells: a practical nonlinear sound model for near-rigid thin shells. In *SIGGRAPH Asia '09: ACM SIGGRAPH Asia 2009 papers*, ACM, New York, NY, USA, 1–10.

CHENTANEZ, N., ALTEROVITZ, R., RITCHIE, D., CHO, L., HAUSER, K. K., GOLDBERG, K., SHEWCHUK, J. R., AND O'BRIEN, J. F. 2009. Interactive simulation of surgical needle insertion and steering.

IGARASHI, T., MOSCOVICH, T., AND HUGHES, J. F. 2005. As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3, 1134–1141.

JAMES, D. L., AND PAI, D. K. 1999. Artdefo: accurate real time deformable objects. In *Proceedings of SIGGRAPH '99*, 65–72.

LABELLE, F., AND SHEWCHUK, J. R. 2007. Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. *ACM Trans. Graph. 26*, 3, 57.

MASRY, M., AND LIPSON, H. 2005. A sketch-based interface for iterative design and analysis of 3d objects. In *Proceedings of Eurographics workshop on Sketch-Based Interfaces*, 109–118.

MEZGER, J., THOMASZEWSKI, B., PABST, S., AND STRASSER, W. 2008. Interactive physically-based shape editing. In *Proceedings of the 2008 ACM symposium on Solid and physical modeling*, 79–89.

MORI, Y., AND IGARASHI, T. 2007. Plushie: an interactive design system for plush toys. *ACM Trans. Graph. 26*, 3, 45.

MÜLLER, M., DORSEY, J., MCMILLAN, L., JAGNOW, R., AND CUTLER, B. 2002. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 49–54.

SMITH, J., HODGINS, J., OPPENHEIM, I., AND WITKIN, A. 2002. Creating models of truss structures with optimization. *ACM Trans. Graph. 21*, 3, 295–301.

SUTHERLAND, I. E. 1964. Sketch pad a man-machine graphical communication system. In *Proceedings of the SHARE design automation workshop*, 6.329–6.346.

TEZDUYAR, T. E., MITTAL, S., RAY, S. E., AND SHIH, R. 1992. Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements. *Comput. Methods Appl. Mech. Eng. 95*, 2, 221–242.

## A Computing tone and mode of metallophone

Assuming that the metallophone is floating in the nongravity space without any external forces and fixed boundaries, we obtain the following equation from the FEM discretization:

$$\bar{\mathbf{M}}\ddot{\boldsymbol{u}} + \mathbf{K}\boldsymbol{u} \quad = \quad \mathbf{0} \tag{9}$$

where $\boldsymbol{u}$ is the nodal displacement vector, $\bar{\mathbf{M}}$ is the lumped mass matrix, and $\mathbf{K}$ is the positive semidefinite stiffness matrix. We split the displacement $\boldsymbol{u}$ into the product of the spatially varying amplitude $\boldsymbol{\phi}$ and the harmonic oscillation at angle rate $\omega$ as $\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{\phi}(\boldsymbol{x})e^{i\omega t}$ , and substitute it into Eq. 9 yielding the following generalized eigenvalue problem:

$$\mathbf{K}\boldsymbol{\phi} = \lambda\bar{\mathbf{M}}\boldsymbol{\phi} \tag{10}$$

where the eigenfrequency $f = \omega/(2\pi) = \sqrt{\lambda}/(2\pi)$. We calculate the smallest nonzero eigenvalue $\lambda$ and its corresponding eigenvector $\boldsymbol{\phi}$ to obtain both the tone and the mode of the metallophone.

We perform Cholesky factorization on the lumped mass matrix as $\bar{\mathbf{M}} = \mathbf{L}\mathbf{L}^{\mathbf{T}}$ and multiply $\mathbf{L}^{-\mathbf{1}}$ to the both sides of Eq. 10 from left, obtaining the following eigenvalue problem:

$$\mathbf{A}\boldsymbol{\psi} = \lambda\boldsymbol{\psi} \tag{11}$$

where $\mathbf{A} = \mathbf{L}^{-\mathbf{1}}\mathbf{K}\mathbf{L}^{-\mathbf{T}}$ and $\boldsymbol{\psi} = \mathbf{L}^{\mathbf{T}}\boldsymbol{\phi}$. We solve this using the inverse iteration method. We modify the original iteration procedure adding a step that removes the component of all the zero eigenvectors of $\mathbf{A}$ from the current solution to obtain the smallest nonzero eigenvalue and its corresponding eigenvector. Thanks to our problem setting, we already know the zero eigenvectors of $\mathbf{K}$ as $\boldsymbol{\phi}_0^i$ $(i = 1 \ldots 6)$: the translations along the three coordinate axes and the rotations around them. We apply the modified Gram-Schmidt orthogonalization to $\mathbf{L}^{\mathbf{T}}\boldsymbol{\phi}_0^i$ $(i = 1 \ldots 6)$ to obtain the orthonormal basis vectors $\boldsymbol{\psi}_0^i$ $(i = 1 \ldots 6)$ that span the kernel of $\mathbf{A}$, and define a projection $\mathcal{P}$ that maps a vector $\boldsymbol{v}$ to the compliment space of the kernel of $\mathbf{A}$ as $\mathcal{P}(\boldsymbol{v}) = \boldsymbol{v} - \sum \boldsymbol{\psi}_0^i \left(\boldsymbol{\psi}_0^i \cdot \boldsymbol{v}\right)$. In each iteration step, we apply this projection to the solution vector and normalize it. W add a small positive number $\varepsilon$ to the diagonals of $\mathbf{A}$ in order to improve the condition number. Once the shifted eigenvalue $\lambda_1'$ and its corresponding eigenvector $\boldsymbol{\psi}_1$ of $\mathbf{A}$ are computed, we finally obtain the nonzero smallest eigenvalue $\lambda_1 = \lambda_1' - \varepsilon$ and the eigenvector $\boldsymbol{\phi}_1 = \mathbf{L}^{-\mathbf{T}}\boldsymbol{\psi}_1$ in Eq. 10. Our technique of reusing the solution of the previous time step significantly improves the convergence of the inverse iteration.

The simulation accuracy is very sensitive to the mesh density because we use linear tetrahedral elements to represent the bend of a 3D thin plate. This problem, called ‘shear-locking’, can be alleviated by preventing the excessive distortion of tetrahedral elements. We keep the longest edge shorter than the twice of the shortest edge.