

Multi-resolution Amplification Widgets

Kiril Vidimce
vkire@erc.msstate.edu
NSF Engineering Research Center
Mississippi State University

David C. Banks
banks@csit.fsu.edu
Computational Science & Information Technology
Florida State University

Abstract

We describe a 3D graphical interaction tool called an amplification widget that allows a user to control the position or orientation of an object at multiple scales. Fine and coarse adjustments are available within a single tool which gives visual feedback to indicate the level of resolution being applied. Amplification widgets have been included in instructional modules of The Optics Project, designed to supplement undergraduate physics courses. The user evaluation is being developed by the Institute of the Mid-South Educational Research Association under the sponsorship of a 2-year grant from the National Science Foundation.

Nous décrivons un outil graphique de l'interaction 3D q'on appelle un "widget d'amplification" qui permet à un utilisateur de contrôler la position ou l'orientation d'un objet aux échelles divers. Les réglages fins et approchés sont disponibles dans un outil simple qui donne le feedback visuel pour indiquer le niveau de la résolution étant appliquée. Des widgets d'amplification ont été inclus dans des modules d'instruction de The Optics Project, conçus pour compléter des cours de physique. L'évaluation d'utilisateur est développée par l'Institute of the Mid-South Educational Research Association sous le patronage du National Science Foundation.

Key words: 3D widgets, user interface, interactive control.

1 Introduction

This paper describes amplification widgets that deliver to the user an amplified version of changes that occur in a 3D object. Instead of offering just one level of amplification, these 3D widgets [Conner] allow multiple scales of operation. Our reason for creating such widgets was specific and practical, but the widgets themselves can be used in more general and even fanciful settings.

During the past four years we have developed several interactive 3D graphical tools as part of The Optics Project (TOP) to teach principles of optics to undergraduate physics students. The student sometimes needs to apply very fine control to an optical element in a simulated system. For example, moving a mirror or changing the wavelength in a Michelson interferometer produces large changes in the interference patterns on an observation screen. The resolution of a mouse is insufficient for providing satisfactory control.

Fine control is important in other application areas of computer graphics. A surgeon makes precise cuts with a scalpel, and a surgical simulation should offer such precision. Tele-operation of robotic controls may require precision beyond what a typical input device can afford. For individuals with motor disabilities, it is desirable to construct a user interface that improves the resolution of a non-dextrous hand.

If you want to translate an object with precision, one standard strategy is to zoom, move, and unzoom. If you don't zoom, the resolution of the mouse maps a 1-pixel motion into a relatively large translation. But when you zoom, the 1-pixel motion of the mouse converts to a small translation. It would be nice to have fine control without demanding the zoom/unzoom sandwich around the translation.

One strategy is – by fiat – to decree that the 1-pixel motions of the cursor are converted to tiny translations of the object. But that sacrifices the virtuous goal of direct manipulation: the object should move together with the cursor. How can you simultaneously provide direct manipulation and fine control?

Rotation creates similar problem. To exercise fine control over the orientation of an object you might use a knob. When you rotate the knob, the object rotates too. But how do you make the object turn by a millionth of a degree? You might make the knob be *really* big; then a 1-pixel motion on the knob would only rotate by a millionth of a degree. But there is not enough screen to

make the knob that big. Alternatively, you could decouple the knob's rotation from the object's, so one degree on the knob becomes a millionth of a degree for the object. But that seems to sacrifice direct manipulation of the object. A solution to one problem promises a solution to the other.

The single most influential work that inspired our efforts was the 1992 paper by the group at Brown University on 3-dimensional widgets [Conner], which demonstrated how effectively a 3D object can be manipulated through geometric elements that visually connect a representation of the input device (a cursor for a mouse, a hand for a 3D tracker) to the object being manipulated. When the object undergoes a simple 1-to-1 motion that tracks the device, such widgets may be gratuitous. But indirect control is mediated very effectively by an intervening widget.

Other researchers have addressed the general problem of how to design a "natural" 3D interaction technique that allows a user to manipulate an object without demanding a 1-to-1 correspondence to the input device. Poupyrev's "go-go interaction" applies a non-linear mapping of the user's hand position to the world, so that a reaching gesture will grab objects that are much farther than an arm's length away [Poupyrev]. Mark Mine took the idea a step further by automatically stretching the manifestation of the user's arm as long as is required to grab the object beyond the hand [Mine].

Bier's group developed "magic lenses" to permit multiple styles of display and interaction through a see-through interface [Bier]. They designed several overlay tools for 2D displays. They describe the notion of composing a widget with a lens:

... consider a click-through button on top of a magnifying lens. Mouse events pass through the button, are annotated with a command, and then pass through the lens, which applies the inverse of its transformation to the mouse coordinates.

Although they did not illustrate this idea for multiple scales of translational or rotational control, the notion was certainly part of their overall view of a modular layered interface.

Mackinlay's perspective wall [Mackinlay] and Furnas's fish-eye view [Furnas] share the strategy of warping a display in order to selectively scale (zoom) a region of interest. We depart from this scheme by restricting the deformations to the widgets rather than to the entire display, and by concentrating on translation and rotation rather than scale.

Ahlberg and Masui both addressed the problem of positioning a slider bar during a search of a large dataset [Ahlberg] [Masui]. Both coarse-scale seeking (finding

the right chapter of a book) and fine-scale seeking (finding the right sentence or word) can occur within moments of each other. Masui's rubber-band slider offers two levels of resolution. Ahlberg's offers two or three levels.

Mackinlay's work on rapid controlled movement demonstrated that an exponential relation between fine control and coarse control is effective in navigating a 3D environment [Mackinlay2]. We pursue a similar approach, although for manipulation rather than navigation.

Stoakley's worlds-in-miniature addresses a complementary problem to ours [Stoakley]. Rather than apply micro-scale changes to sensitive objects, his users manipulate physical props to speed up interaction, providing a means to produce large changes in translation. We have not yet applied amplification widgets to produce gross translations (or rotations for that matter, but rotations are essentially bounded by 360 degrees which makes them uninteresting for coarse control), but we plan to do so in the future.

2 Amplification Widgets

A 3D "amplification widget" is a cascade of 3D components, one controlling the next, with one component directly controlling a given object in a scene. The hierarchy of these components propagates increasingly fine control outward from the object being manipulated so that direct manipulation occurs at the object. For example, a box can be positioned at a scale of 1:1 by a widget component, but can also be translated at a scale of 2:1 by the neighboring component, which in turn is translated at a scale of 2:1 by its neighbor, and so on. In a similar fashion, an object can be rotated at a 2:1 ratio; the rotating element can be similarly rotated by another, and so forth, until the outermost component amplifies a minuscule rotation of the object into a large sweep of the mouse or tracker.

The benefit of amplification widgets is that they (1) eliminate the requirement that a user zoom, micromanipulate, then unzoom (a situation that arises when small changes are made within large environments); (2) preserve the spirit of direct manipulation by visually connecting the point of control with the point of attachment; and (3) provide multiple scales of resolution for manipulation.

2.1 Translation Amplifiers

Our strategy for giving interactive control works like this. A widget has a sequence of components to govern the movement of the object they are attached to. The position of component number u is $p(u)$, with $p(0)$ indicating the point of attachment that provides direct control of the object. The user grabs component number

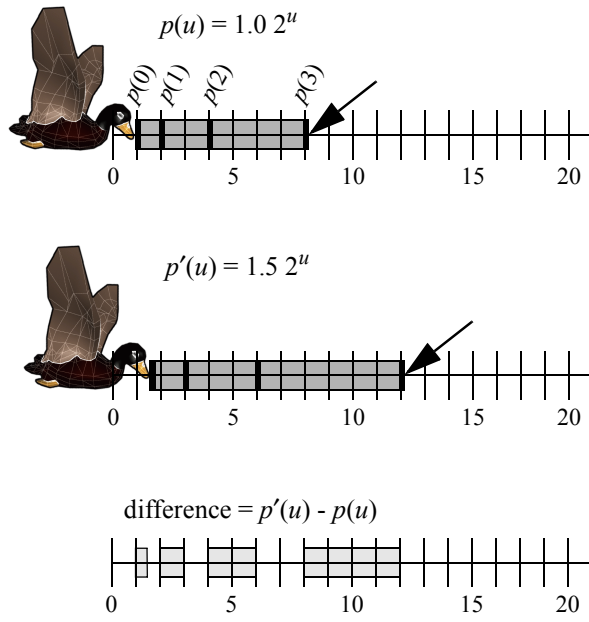


Figure 1. (Top) The mouse (indicated by the arrow) drags knuckle number 3, beginning at position 8. Other knuckles in the widget move according to the equation shown. (Middle) When the mouse reaches position 12, an appropriate scale factor is computed to maintain the form of Equation 1. As a consequence, the other knuckles move too. (Bottom) The gray regions show how much each knuckle moved. The duck moves with knuckle 0; the other knuckles amplify this small translation.

u and drags it. Direct manipulation applies to the component of the widget, not to the object itself. The sequence of widget elements ties the cursor to the object in a natural, visual way.

There are many mathematical functions that exaggerate motion away from the zero point, but the exponential function

$$p(u) = sb^u - q \quad (1)$$

is perhaps the most convenient way to capture this desired behavior. We chose this equation since it can be used to encapsulate the notions of widget scale s , exponential base b , and the widget's point of attachment q to the object. The base b of the exponent produces the multi-resolution effect: the k^{th} knuckle amplifies the object's motion by b^k . The scale term is applied to the ensemble of knuckles as a whole so that the user can dictate how large the entire collection is on the screen.

The translation amplifier (transAmp) has n segments that stretch or squish and $n+1$ knuckles that can be dragged by the cursor. Since (by Equation 1) the widget attaches at the point $p(0) = s - q$, changes made in

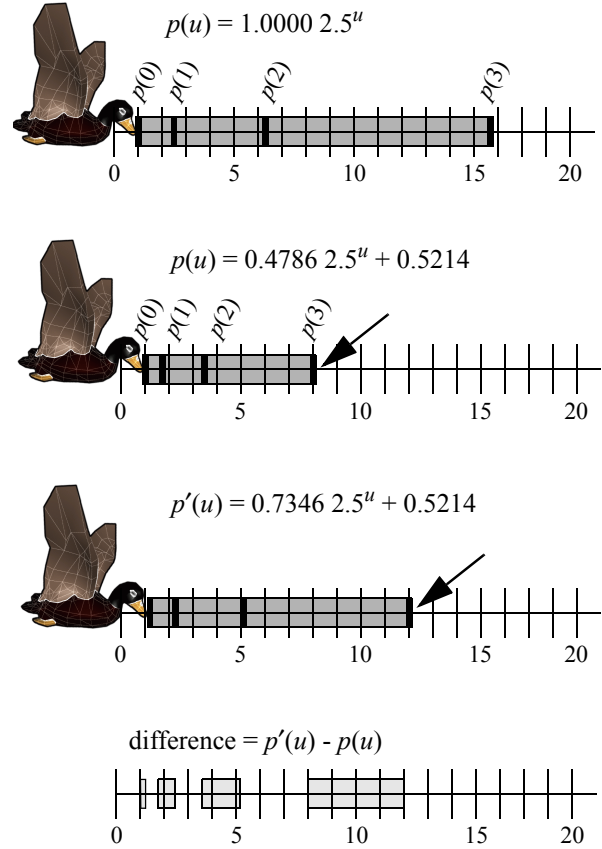


Figure 2. Changing the sensitivity of the transAmp without changing its overall appearance.

(Top) The base of the exponent in figure 1 is increased from 2.0 to 2.5, making the widget longer and more sensitive at the rightmost end.

(Upper middle) The widget is scaled and offset to match the length of the transAmp in figure 1. This requires calculating a different scale and offset.

(Lower middle) The user again moves the transAmp's right end from position 8 to 12, but the duck moves less than in figure 1 because the exponent is now larger.

(Bottom) The gray regions show how much each knuckle moved.

the scale s produce translation of this point of attachment. Changes to the exponential's base b have a different effect: the point of attachment $p(0)$ is independent of b , but the relative lengths of the segments change. When b is large, the u^{th} knuckle moves considerably compared to the motion of the attachment point.

Figures 1 and 2 illustrate a transAmp with three segments. The putative goal is for the user to move the duck by a small amount. The left end of the transAmp attaches to the duck at position $p(0) = 1$. With the scale $s = 1$ and the base $b = 2$, we must have $q = 0$. Thus the other knuckles lie at $p(1) = 2$, $p(2) = 4$, and $p(3) = 8$.

We leave the base b and the offset q fixed, allowing s to vary. When the rightmost knuckle $p(3)$ is dragged to a new position $p'(3) = 12$, the scale s is no longer valid and must be recalculated. According to Equation 1, this knuckle lies at $p'(3) = s' 2^3 - q = 12$, which is satisfied by a new scale factor $s' = 1.5$. The positions of the other knuckles are updated accordingly using the new s' . The point of attachment therefore changes from $p(0) = 1$ to $p'(0) = 1.5$, moving 0.5 units, while the rightmost knuckle moves 4 units. The third knuckle amplifies the object's motion by a factor of 8. This is how the motion of the transAmp is amplified with respect to the motion of the object. The second knuckle at point $p(2)$ only moves 2 units, so if it had been selected and dragged the cursor's motion would amplify the object's motion by a factor of 4. Similarly, point $p(1)$ amplifies by a factor of 2.

We can calculate this amplification factor by writing the position of the u^{th} knuckle in terms of the 0^{th} knuckle. Substituting in Equation 1 for the scale s when $u = 0$ we have

$$p_u = (p_0 + q) b^u - q,$$

where p_u denotes $p(u)$. When p_0 changes, so does p_u . The amplification is the rate of change of one with respect to the other and is given by the derivative

$$\frac{dp_u}{dp_0} = b^u$$

So the knuckles exhibit exponentially increasing amplification. We can limit the maximum amount of amplification for a given configuration of the widget. A transAmp with n knuckles and total length l , having a maximum amplification of B , satisfies $b^n = B$, so the desired base of the exponent is $b = B^{1/n}$. Furthermore, we note that $s = l/(B-1)$ and $q = p_0 - s$. Thus the scale, the base, and the offset can be easily calculated to satisfy a desired configuration for the widget.

When a knuckle is dragged, the new value of s is computed as $s = p_u/b^u$. We also allow the user to drag a knuckle for the purpose of changing the base b , rather than for producing a translation via a change of the scale s . In that case, the new value of b is given by the equation $b = (p_u/s)^{1/u}$.

2.2 Look and Feel

As the components move, the transAmp's shape distorts because they move by unequal amounts. The *object* moves rigidly, the *cursor* moves rigidly, each *knuckle* moves rigidly, but not the widget as a whole. When the user releases the widget, how should it return

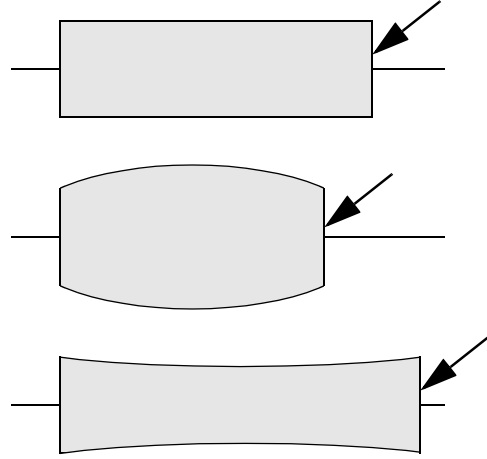


Figure 3. (Top) A cylindrical component of the translation amplifier in its rest state is grabbed (arrow).

(Middle) The cylinder bulges when it is compressed.

(Bottom) The cylinder squeezes in when it is extended.

to its undistorted rest state? One choice is to simply switch from the stretched state to the rest state discontinuously. But a smoother transition is produced by briefly animating the widget's return (as a visual inverse operation of the dynamic stretching the widget undergoes when it is in use). This strategy gives the widget a more physical look and feel than does a discontinuous pop. The only issue is how to provide a reasonable animation. To restore the widget to its original shape, we animate the transition from distorted to undistorted when the user completes the drag (releasing the mouse). We experimented with a damped spring and various other oscillating behaviors to animate the dragger's return to its rest state, but we were eventually the most satisfied with a simple cubic "return function"

$$c(t) = 2t^3 - 3t^2 + 1$$

which has velocity zero at the endpoints of the interval $[0,1]$. This cubic is the lowest-degree polynomial that has zero-derivatives at both ends of the interval, which makes it the simplest choice for animating the widget's return to its rest state. The point of attachment remains fixed at p'_0 during the animation, while the other knuckles return to their original rest positions, translated by p'_0 . The behavior is given by the equation

$$p_u = c(t) p'_u + (1-c(t)) (p'_0 + p_u).$$

To give additional visual feedback during dragging, we make the widgets bulge outward when compressed and squeeze inward when stretched (figure 3). This is the appropriate behavior for a cylinder of constant volume

as its length changes. We give the cylinder a parabolic profile that preserve its original volume. Why parabolic? Because a second-degree polynomial is the quickest non-linear shape to compute on the fly. In normalized units, the axis of a cylinder runs from -1 to 1 in the x -direction, its radius is r , and its volume v is $2\pi r^2$. When the length of the axis is changed to some other value l , the radius is found by the quadratic expression $r = au^2 + b$, where a and b are chosen to preserve the volume of the cylinder as it bulges or contracts. This quadratic is chosen so that the radius at the endpoints of the cylinder remains fixed.

We apply texture mapping to the cylinders forming the transAmp segments so that each local neighborhood looks like it is stretching or contracting. Ideally the texture would stretch non-linearly to match the exponential stretching, but for segments of modest length the difference is not objectionable. We created periodic texture maps so that there would be no visible seam where they wrap, and constructed the textures from multi-frequency noise so their appearance would remain acceptable throughout varying degrees of expansion or compression.

We experimented with segments whose lengths formed a geometric progression so that segment k has length $l_k = b l_{k-1}$. The visual effect is not appreciably different from the exponential function already described, but the math is considerably more tedious.

After we used the transAmp for awhile, three improvements to the widget became immediately evident. First, there is no need to scale the knuckles beyond the one being dragged. This is especially obvious when the knuckle at the point of attachment is moved. All the other knuckles spread out exponentially, which really isn't their purpose. We therefore restrict the scaling to the knuckles between the one being dragged and the one at the point of attachment. The others simply translate by the same amount as the dragged knuckle.

The second improvement is an implementation detail. Rather than calculate the absolute position of each knuckle, it is more convenient to calculate the relative position of one knuckle with respect to its neighbor. Since the scale s and the base b are actually variables, the change of position p is given by the total derivative

$$\begin{aligned} dp &= \frac{\partial}{\partial u} s b^u du + \frac{\partial}{\partial s} s b^u ds + \frac{\partial}{\partial b} s b^u db \\ &= s \ln b b^u du + b^u ds + s b^{u-1} db \end{aligned}$$

Therefore

$$dp(u-1) = \frac{1}{b}(dp(u) - s b^{u-1} db)$$

Thus we have an expression for the change in position of component $u-1$ (which is closer to the object) as a function of the change of its neighboring component u . In this way, the motion of the mouse can be propagated from knuckle to knuckle, all the way back to the point of attachment to the object.

When the user is not adjusting the value of the base b , then $db = 0$ and the right hand side is simply $dp(u)/b$. This means that the knuckles can be recalculated progressively toward the object by moving each one by a fraction $1/d$ of the distance its neighbor moves. The knuckles can be recalculated progressively outward from the dragged knuckle by moving each one the same distance $dp(u)$ that the point p_u moved.

The third improvement exploits the relative calculation of distance from knuckle to knuckle. The calculations were designed to agree with the exponential function for the positions p_u . But we can continue to blithely use these relative calculations even if the initial positions of the knuckles result from a completely different distribution. For example, the knuckles can be evenly spaced rather than crowding together near the point of attachment (as in figure 2), but still retain the exponential scaling of their motion amplification. In other words, they may be displayed as if $b = 1$, but have an effect as if $b = 10$. At first blush this may seem jarring: why should we be justified in applying phoney derivatives that don't match the actual set of positions? But the idea is actually familiar in a different guise. Bump mapping assigns "fake" tangents to a polygon in order to produce more complicated reflectance effects than a flat surface can afford. So there is a precedent in cheating on a derivative in computer graphics.

As a final note, two 1-dimensional transAmps can be attached to an object to permit fine control in two different dimensions. Likewise, three transAmps can be fitted to it to permit three degrees freedom for precise translation.

2.3 Rotation Amplifiers

If you can amplify translation, you can amplify rotation. This is useful in cases when small rotations produce large results, because unlike translation (where zooming provides a simple scheme for converting mouse motion into small absolute motion in world space), rotation angles do not change when a scene is zoomed. Let $\theta(u)$ represent the angle as a function of the distance u from the point of attachment connecting a rotation amplifier (rotAmp) to an object. Initially we set $\theta(u) = \theta(0)$ for every knuckle u (which means that the scale s is zero for all the knuckles). When one knuckle is rotated by some amount $d\theta(u)$, the neighboring knuckles (working

inward toward the object) are updated in the same manner as for the transAmp:

$$d\theta(u-1) = \frac{1}{d}(d\theta(u) - sb^{u-1}db)$$

Just as for transAmps, knuckles on a rotAmp may be placed according to an arbitrary initial distribution $\theta(u)$; the values of u need not even be uniformly spaced. For knuckles outside the one being dragged, we apply the same incremental rotation as that of the dragged point. The inner ones are adjusted according to the iterative scheme described in the section above. Figure 4 shows

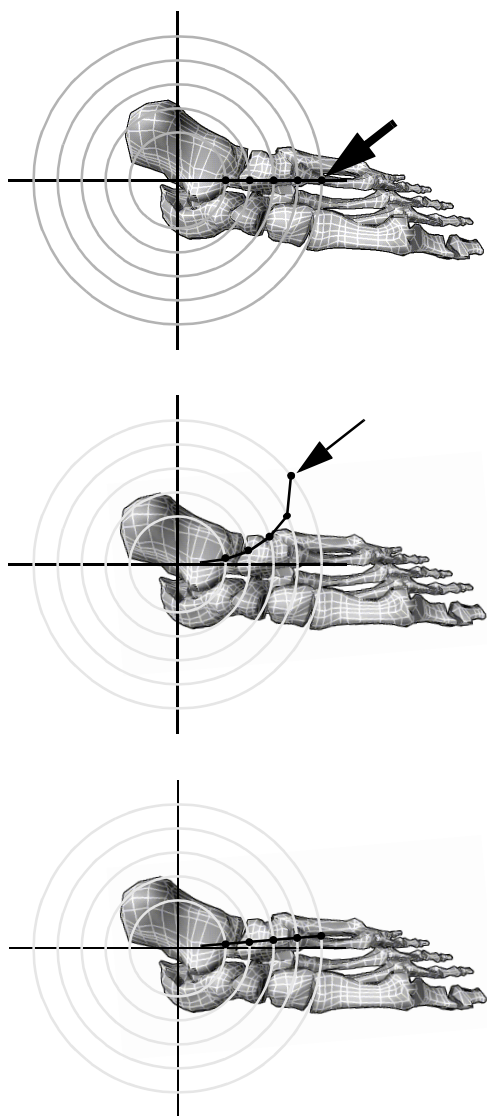


Figure 4. (Top) The outermost knuckle of the rotation amplifier is selected and dragged (arrow). The outer knuckle is rotated, changing the scale of the governing equation and producing a slight rotation of the object.

(Bottom) The knuckle is released and the widget returns to its straight-line rest state, but rotated now by a slight amount..

an example of a rotAmp before, during, and after the outermost knuckle is moved.

3 Implementation and Evaluation

The original motivation for the amplification widgets was very practical – we wanted students to exert extremely fine control over sensitive elements in a 3D simulation of optical experiments. Sometimes the user wants to quickly move from a blue wavelength (400 nm) to a red wavelength (700 nm); however, the user also needs to be able to commit minute changes (on the order of one or less than one nanometer) in the wavelength in order to observe the rapid changes in the interference patterns. For this reason we inserted amplification widgets into components of The Optics Project where they are most needed.

The Optics Project now contains more than 10 modules and is being field-tested in physics courses at several universities. Our analysis of the students' performance using the amplification widgets will not be available until after the spring semester of 2000; we plan to include the results on the Optics Project home page at www.webtop.org. The user evaluation is being developed by the Institute of the Mid-South Educational Research Association under the sponsorship of a 2-year grant from the National Science Foundation.

We developed these amplification draggers on top of the Open Inventor framework from SGI. Open Inventor provides a convenient environment for designing new widgets. It already has translation widgets (called draggers) and rotation widgets that can control 3D objects in a scene. We have also implemented the widgets using VRML and Java as part of our Web-based deployment of The Optics Project.

We have applied the widgets in several test domains. These are summarized briefly below and illustrated in figures 5-7. (1) The Optics Project: Amplification widgets allow the user to control the wavelength of light at very fine resolution in the Michelson Interferometer module, but also permit gross changes in wavelength from blue to red. The widgets allow the user to make micro-adjustments to the optical elements along the workbench in the Geometrical Optics module, but also allow elements to be moved large distances out of the way. (2) Text scrollbar: The translation amplifier has been converted into a variable-resolution scrollbar to allow either fast sweeps through a document or slower line-by-line scrolling. (3) 3D control: Multiple translation amplifiers and multiple rotation amplifiers have been combined to provide multi-resolution control of 3D position and 3D orientation.

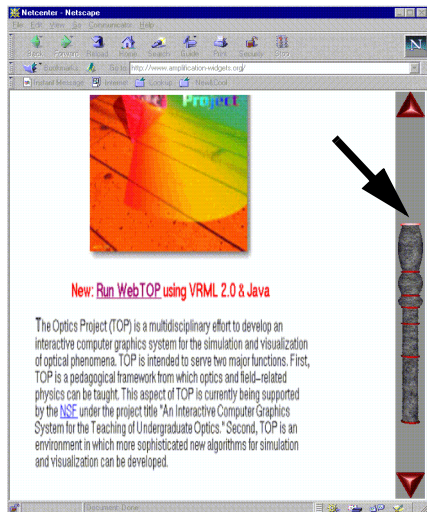


Figure 5. Trans.Amp scrollbar for browsing text. Top knuckle scrolls a pixel at a time, while bottom knuckle scrolls entire pages.

4 Conclusions

Amplification widgets balance (1) the need for direct manipulation with (2) the need to control the position or orientation at resolutions finer than a screen cursor ordinarily permits. An amplification widget deforms when it is in use, then recovers its initial shape for repeated usage. The widgets permit a user to translate or rotate an object at fine, coarse, and in-between levels of resolution. The widgets can be composed in 1D, 2D, and 3D. We implemented translation and rotation amplification widgets for instructional modules in The Optics Project, where both fine and coarse adjustments are needed.

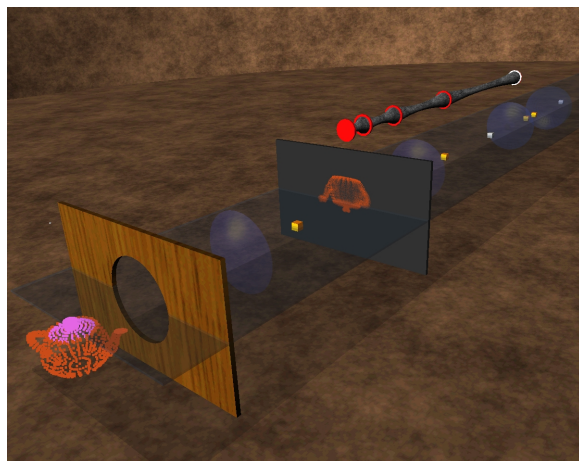


Figure 6. Simulation of geometrical optics. The trans.Amp (with red knuckles) allows the user to position the observation screen along the workbench, and then micro-position the screen precisely in order to bring the image into sharp focus.

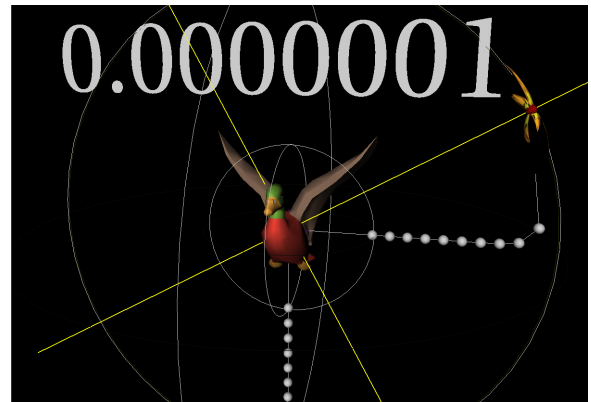


Figure 7. 3D rotation. The three gimbals provide 3D orientation control of the duck. The outermost knuckle of each rot.Amp provides control at the resolution of less than a millionth of a degree.

Acknowledgments

This work was supported in part by the National Science Foundation (grant numbers 9555053, 9624325, and 9950569). The authors gratefully acknowledge equipment provided by Intergraph Corporation. We thank Martin Reddy for his helpful suggestions on how to improve the readability of the paper.

References

[Ahlberg] C. Ahlberg and B. Shneiderman, "The Alphaslider: A Compact and Rapid Selector," *Proceedings of ACM CHI '94*, pp. 365-371.

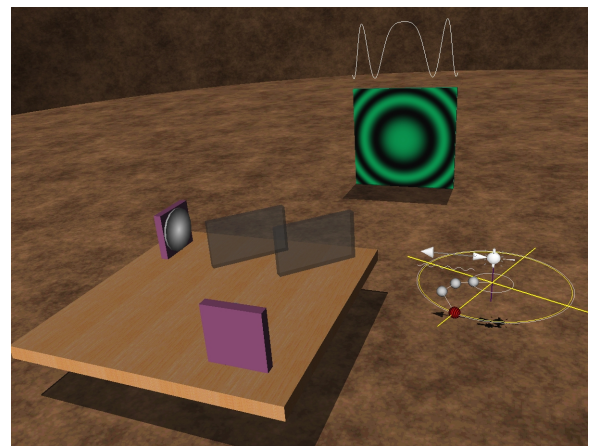


Figure 8. Simulation of the Michelson interferometer. A minuscule rotation of the elements of the apparatus produces a substantial change in the interference fringes on the observation screen. The rot.Amp on the right has 4 knuckles the user can control, providing 4 levels of rotational resolution.

[Bier] E. Bier, M. Stone, K. Pier, W. Buxton, and T. DeRose, "Toolglass and Magic Lenses: The See-through Interface," Proceedings of ACM SIGGRAPH '93, pp. 73-80.

[Conner] D. B. Conner, S. Snibbe, K. Herndon, D. Robbins, R. Zeleznik, A. van Dam, "Three-Dimensional Widgets," Proceedings of the 1992 Symposium on Interactive 3D Graphics, pp. 183-188.

[Furnas] G. Furnas, "Generalized fisheye views." Proceedings of ACM CHI '86, pp. 16-23, 1986.

[Mackinlay] J. Mackinlay, G. Robertson, S. Card, "The perspective wall: detail and context smoothly integrated," Proceedings of ACM CHI '91, pp. 173-176.

[Mackinlay2] J. Mackinlay, S. Card, G. Robertson, "Rapid Controlled Movement Through a Virtual 3D Workspace," Proceedings of ACM SIGGRAPH '90, pp. 171-176.

[Maui] T. Masui, K. Kashiwagi, G. Borden, "Elastic Graphical Interfaces for Precise Data Manipulation," CHI '95 Conference Companion, pp. 143-144. Addison-Wesley, May 1995.

[Mine] M. Mine, F. Brooks, C. Sequin, "Moving Objects in Space: Exploiting Proprioception in Virtual-Environment Interaction" Proceedings of ACM SIGGRAPH '97, pp. 19-26.

[Poupyrev] I. Poupyrev, M. Billinghurst, S. Weghorst, T. Ichikawa, "Go-Go Interaction Technique: Non-Linear Mapping for Direct Manipulation in VR." Proceedings of UIST '96, pp. 79-80.

[Stoakley] R. Stoakley, M. Conway, R. Pausch, "Virtual Reality on a WIM: Interactive Worlds in Miniature," Proceedings of ACM CHI '95 (Denver, CO, USA, 1995), ACM Press, pp. 265-272.