Incompatible BLOCK: Wonders Accompanied Interface

Jun Fujiki
Graduate School of DESIGN, Kyushu-University
4-9-1 Shiobaru, Minami-ku, Fukuoka, 815-8540 JAPAN
fujiki@gsd.design.kyushu-u.ac.jp

Taketoshi Ushiama
Faculty School of DESIGN, Kyushu-University
4-9-1 Shiobaru, Minami-ku, Fukuoka, 815-8540 JAPAN
ushiama@design.kyushu-u.ac.jp

Kiyoshi Tomimatsu
Faculty School of DESIGN, Kyushu-University
4-9-1 Shiobaru, Minami-ku, Fukuoka, 815-8540 JAPAN
tomimatsu@design.kyushu-u.ac.jp

CHI 2006, April 22–27, 2006, Montréal, Québec, Canada.
ACM 1-59593-298-4/06/0004.

Abstract
For interface design, improving user curiosity is important, as is intuitiveness and intelligibility. We think the wonder of 2D property input becoming 3D, as expected by the user, in the range of matching will be effective for enhancing not only curiosity but also availability. In this study, we developed the 3D modeling software "Incompatible BLOCK" with an interface of such wonders. This paper describes the four wonders of the interface of Incompatible BLOCK, introduces the mechanism, and discusses the usefulness of a wonders-accompanied interface from user evaluations.

Keywords
3D Modeling, interface, wonder, curiosity

ACM Classification Keywords
[H.5.2]: User Interfaces

Introduction
For interface design, the elements of enhancing user curiosity as well as intuitiveness and intelligibility are important. Recent studies of 3D modeling interfaces propose operational interfaces that provide intuitiveness and intelligibility by combining user experiences and impressions [1-3]. Under these

circumstances, we incorporate wonders born from the discrepancy between three dimensions and two dimensions into the interface to enhance user curiosity. The wonders of the conversion into three dimensions, as expected by the user, are projected to make the interface predictable to some extent. In this study, we experimentally created the 3D modeling software "Incompatible BLOCK" with an interface of such wonders. This paper describes the features of the Incompatible BLOCK and discusses its effectiveness from verification-based evaluation results.

Incompatible BLOCK

The Incompatible BLOCK is 3D modeling software that can generate a form by combining cubes and can draw lines for the form, floor, and background by a 2D operation. When modeling with a combination of cubes, it is difficult to create a precise shape. By applying techniques such as subdivision [4] and meta-ball [5] to a created form, however, a highly precise form can be obtained. This function is not implemented at this stage but can be added.

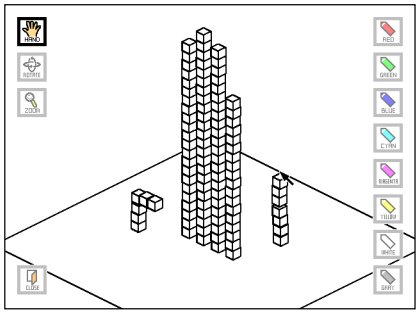


figure 1. The screen of Incompatible BLOCK.

Figure 1 shows a screen of the Incompatible BLOCK. The pens lined on the right side of the screen are tools to draw lines on the background. On the left side of the screen are the hand tool for managing a cube, the rotation tool for rotating the space, and the zoom tool for zooming the space. The door tool, on the lower left side, is used to close a session.

Features of the Incompatible BLOCK

The Incompatible BLOCK has four wonders and also an operational interface that allows results to be predicted to some extent. The following sections describe the features and mechanisms.

Feature1: Moving a Cube

If a cube is moved to a screen position on the screen by dragging, the Incompatible BLOCK judges the destination and places the cube at the expected position in 3D space, as described below.

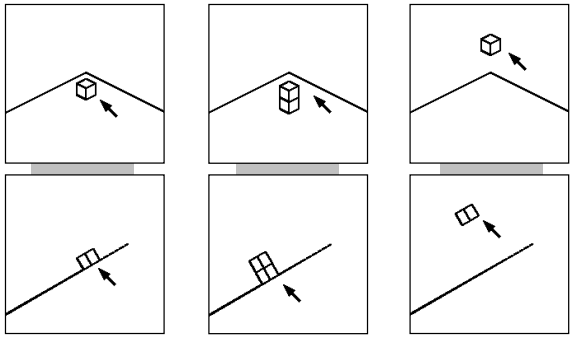
1. If a cube moved onto a 2D floor, the software places the cube on a 3D floor (Fig. 2a).

2. If a cube is moved to a look-as stacking position on the screen, it can be placed to be touched other cubes (Fig. 2b).

3. A cube in empty space is moved vertically or horizontally from the start position (Fig. 2c).

The destination is judged from the overlapping of the 2D-projected cube to be moved and the 2D-projected floor or cube already placed. If the 2D cube to be moved overlaps the 2D floor, Rule 1 applies. The cube is placed where the vector from the center of the cube in the direction of view intersects with the floor. If the

2D cube to be moved overlaps the 2D cube already placed, Rule 2 applies. The position coordinates are converted into 3D so that the cube will adjoin the destination cube. If the 2D cube to be moved overlaps nothing, Rule 3 applies. This resulted from the idea of floating in air because the floor means the ground while the empty space means the air. By dragging, the floor can be moved along the XZ plane.



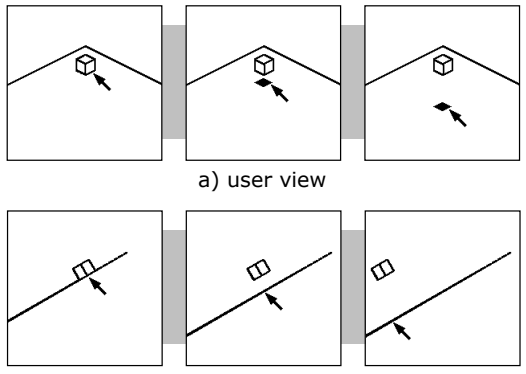
a) on the floor b) stacking a cube c) in the air

figure 2. Three cases of moving a cube (top: user view, bottom: side view).

Feature2: Changing a Cube Height

In addition to Rule 3 above, there are two methods of changing a cube height. One method uses a "shadow." A shadow can be pulled out from a cube by dragging the bottom of the cube down (Fig. 3a). The height of the cube is changed in the 3D space so that it will look afloat. As Fig. 3b shows, the cube moves in the direction of view each time the shadow is moved down.

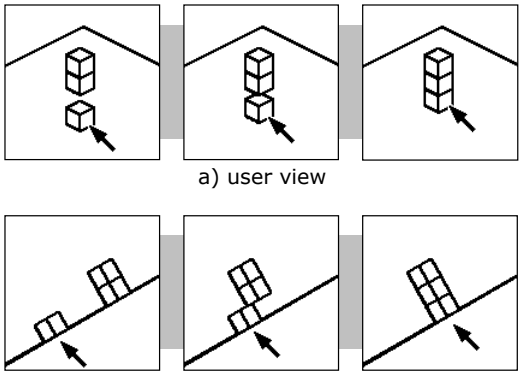
If a cube is added upward to the bottom of a form adjacent to the floor, the added cube looks as if it is under or over the floor. In the real world, however, a form buried in a floor cannot be seen (Fig. 4a). By considering this, the Incompatible BLOCK shifts the whole form up so that the form will be adjacent to the top of the floor (Fig. 4b).



a) user view

b) side view

figure 3. Changing a cube height by using shadow.



a) user view

b) side view

figure 4. The upward placement of a cube.

Feature3: Changing the Number of Cubes by Using Shadow

You can increase the number of cubes by pulling out several shadows from a single cube. In the real world, this can be compared to several building blocks lined in the direction of view, and the same number of cubes as shadows can be added to 3D space (Fig. 5). In contrast, the number of cubes can be reduced by pulling out a small number of shadows when several cubes are overlapping and look like one. In other words, the number of shadows corresponds to the number of cubes. These are the same as the copy and deletion functions of existing 3D modeling software. The existing 3D modeling software uses a menu or icon selection for switching, while the Incompatible BLOCK uses the hand tool to handle a cube without switching.

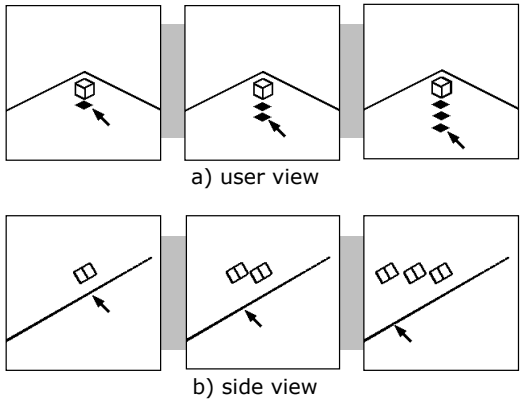


figure 5. Changing the number of cubes by using shadow.

Feature4: Changing a Pen Size

The Incompatible BLOCK enables the user to draw lines directly on the screen for the cube, floor, or background by 2D operation. Since lines can be drawn only on the visible sides, the back of the form shown in Fig. 6 is not painted. The apparent boldness of a drawn line is fixed, irrespective of space zooming (Fig. 7a-1, 7b-1). In other words, zoom-out makes a pen thick (Fig. 7a-2) and zoom-in makes it thin (Fig. 7b-2).

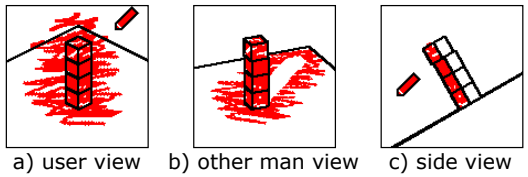


figure 6. Drawing cubes, the floor and the ground.

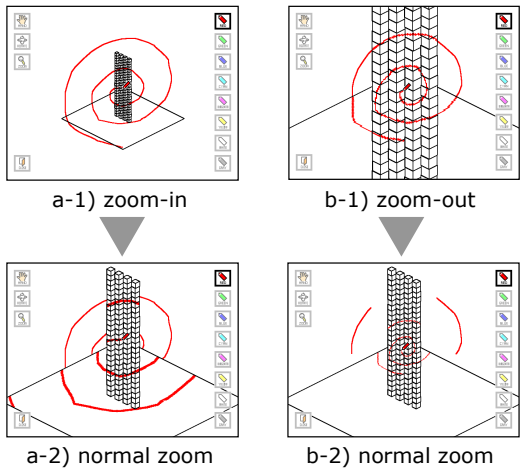


figure 7. Changing a pen size.

Verification and Evaluation

After hearing a brief explanation shorter than one minute, ten undergraduate and postgraduate students used the Incompatible BLOCK for verification. By observing the behaviors of the subjects and having interviews with them, we evaluated the Incompatible BLOCK. According to the evaluation results, the interface having 2D properties was used uniformly except for changing a cube height by using a shadow, and object forms could be created in a short time. In verification by the subjects without hearing the operational explanation, several but not all subjects used the function of upward placement. No subjects used the function of changing a cube height by using a shadow or changing the number of cubes by using a shadow. In both verifications, many subjects were surprised at and pleased with the wonder and intelligibility of viewing 2D results in 3D during the operation. During the interviews, they stated that they had enjoyed themselves. Some of them enjoyed operating the Incompatible BLOCK for 30 minutes or more. Figure 8 shows six models produced by using the Incompatible BLOCK.
Here are some opinions obtained from the subjects

[Affirmative opinions]
- Enjoyable and interesting
- Easy to understand
- Easy to operate
- Not boring
- Feeling wonderful to create a form as intended
- Persuasive even when a cube could not be created at an intended position
- Completely different concept from that of conventional 3D graphic software
- Faster input than that in existing 3D graphic software
- Would like to move several building blocks at a time
- Would like not to pull out a shadow but to write own shadow

[Negative opinions]
- Difficult to adjust a height by using a shadow
- Cube height unknown until turning
- Preferable to always have shadows

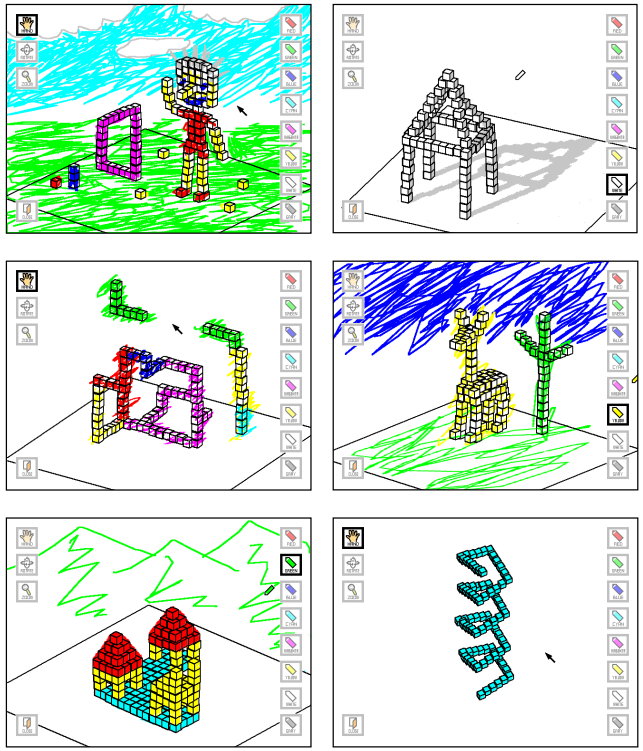


figure 8. Six models produced by using Incompatible BLOCK.

Discussion

The function of changing a cube height by using a shadow was not used much among the subjects who received the operational explanation. This is probably because a height increase moves the cube in the direction of view and the height is difficult to adjust to an expected position. The upward placement of a cube also moves a form in the direction of view. The subjects used this function without a feeling of wrongness, probably because they wished to create an object form quickly and did not mind where the form would be positioned.

The subjects who had not received the operational explanation did not notice that a shadow could be pulled out from a cube. The discrepancy from the real world where a shadow is not touched but produced from light might have prevented them from making this association. In addition, several subjects used the upward placement function but did not notice that the whole form moved. This is probably because priority is given to the creation of an object form rather than to the form position, as mentioned above. The other subjects might have avoided this function by considering it unnatural that a cube added by upward placement was buried in the floor. Under both conditions, those who had ever used existing 3D graphic software were surprised by the fact that the operation, as viewed, produced an expected form. Many of the received opinions were affirmative, but negative ones pointed out the issue of height. This will be a future subject to solve.

Conclusion

In this study, we developed the 3D modeling software "Incompatible BLOCK" with a wonders-accompanied interface using 2D properties and evaluated the software by verification with student users. In the range of matching in 3D, this software motivates the user to work easily with the user-expected 3D design. By verification, we could prove the interface of the wonders is useful with its intelligibility, input speed, and entertainment. However, negative opinions were also received and there are still issues to solve. We will next seek expressions of more wonders, verify their effectiveness, and extend their application to other interface environments.

References

[1] T. Igarashi, S. Matsuoka, H. Tanaka. Teddy: A Sketching Interface for 3D Freeform Design. Proceedings of SIGGRAPH'99. ACM Press, (1999), 409-416.

[2] S. Kasai, S. Hagihara, M. Sano, K. Kusida, K. Ohshiba, N. Kiyohiro and Y. Sugita. Development of The Data-Glove for Design-CAD which Embodies a Real Feeling. 2004.

[3] D. Anderson, J. L. Frankel, J. Marks, A. Agarwala, P. Beardsley, J. Hodgins, D Leigh, K. Ryall, E. Sullivian and J. S. Yedidia. Tangible Interaction + Graphical Interpretation: A New Approach to 3D Modeling. Proceedings of SIGGRAPH'2000. ACM Press, (2000), 399-402.

[4] L. Kobbelt, $\sqrt{3}$ subdivision. Proceedings of SIGGRAPH 2000. ACM Press (2000), 103-112.

[5] H. Nishimura, M. Hirai, T. Kawai, T. Kawata, I. Shirakawa,K. Omura. Object modeling by distribution function and a method of image generation. Transactions of the Institute of Electronics and Communication Engineers of Japan, J68-D(4) , Information Processing Society of Japan (1985), 718-725.