Abstract: Hyper-Paint is a software-toy that is a mix of a block-toy and a drawing-toy. A 3D figure is constructed by combination of single shaped block-toys. A linear tangent line can be drawn directly to a box applied to a shape or a base. Curiosity of children is aroused by its mysterious aspects: even if something is drawn considering two-dimensions (2D), it is rendered by the program as a three-dimensional (3D) object. It is expected that construction ability for the 3D figure and expression ability for drawings, among others, will be improved by expressions mixing a block-toy and a drawing-toy. This paper first respectively describes a block-toy, a drawing-toy and a drawing-toy look, which are features of Hyper-Paint. Next, it describes an implementation method to express these features effectively. Finally, effects of the play that is possible using Hyper-Paint are considered and discussed based on users' evaluations.

Key word: software-toy, block toy, drawing toy, curiosity, discovery, dimension

1. Introduction

Block-toys and drawing-toys are common traditional playthings; these have been expressed using software in recent years[1-6]. These software-toys include rich elements of simulation, in which a block-toy and a drawing-toy are recreated on a computer. A block-toy and a drawing-toy express action in the reality space: a 3D space. On the other hand, those playthings that have been recreated in a computer environment are executed through a 2D display window. Here, the difference from actual playthings becomes apparent. That difference from reality is not preferable for simulation, but we will examine that difference in terms of the use of this program for play.

The Hyper-Paint produced in this study is a software-toy that blends a block-toy and a drawing-toy while applying advantages of a computer. A 3D figure is constructed by combination of single shaped block-toys. A linear tangent line can be drawn directly to a box applied to a shape or a base. The curiosity of children is aroused by the program's mystery: even if something is drawn considering its 2D shape, it is rendered as a 3D object. It is expected that the expression mixes features of a block-toy and a drawing-toy. It is inferred to improve construction ability for 3D figures and expression ability of drawings, etc. This paper describes its features and one exemplary implementation method of Hyper-Paint. Its effects of play that arises from the use of Hyper-Paint are considered and discussed, based on users' evaluations.

2. Outline of Hyper-Paint

Hyper-Paint (Fig. 1) is a software-toy. In the software-toy, it is expected that curiosity of children is aroused by mystery that is aroused by its transformation of dimensions: a 2D drawing is reflected as a 3D object. We infer that construction ability for 3D figures and expression ability of drawings, among others, are improved as they would be by mixing block-toys and drawing-toys. Hyper-Paint works in a Microsoft Windows environment, in which a CPU having higher performance than a 1 GHz Celeron (Intel Corp.) and a DirectX9 runtime are installed. Hyper-Paint was developed using Visual C++ 6.0 @ (Microsoft Corp.) and DirectX9 SDK[7] (Microsoft Corp.).

Figure 1. The screen of Hyper-Paint

6.0 @ (Microsoft Corp.) and DirectX9 SDK[7] (Microsoft Corp.). The broad area under the right in Fig. 1 is the operation milieu (campus). A large box viewed from the inside in the campus is a georama box for a base to arrange block-toys. Lines can be drawn to the georama box. A shape can be drawn using a color pencil tool that has nine colors, depicted on the first line of the screen. The single line below shows a block-toy tool that has been selected from the nine colors. The tool group situated at the left of the campus displays, sequentially from the top, a hand tool to rotate a campus space, a delete tool to delete additional block-toys, a printer tool to print images of the campus, and a door tool to finish a session.

3. Hyper-Paint Features

3.1 Construction of 3D Figures using Block-Toys

In Hyper-Paint, a 3D figure is constructed using a combination of cubes to produce block-toys. In the state where the block-toy tool is selected, when the georama box in the campus or the side face of the existing block-toy is clicked, new block-toys can be arranged to be adhered to the side face (Fig. 2). Nothing is added when block-toys are not in the position that is clicked. In the
case of using this rule, a new cube cannot be arranged to a side face that is not displayed on the screen. By rotating the campus space, however, a new cube can be arranged to the side face that had previously not been displayed.

### 3.2 Function of the Drawing-Toy

In Hyper-Paint, a line can be drawn using a color pencil tool in the 3D space in a sense like a drawing-toy. A similar function called 3D paint exists also in 3D CG software. So as not to fail the users’ expectations, the 3D paint function includes, for example, a limitation function, by which colors are not drawn except for a currently selected object, and a function for quickly returning to the last action: a so-called undo function. For toys having educational elements, these functions decrease “attention” and “discreetness”. For that reason, curiosity that is fueled by mysterious results, where protruding lines are reflected in a 3D space, cannot be cultivated. In consideration of the above matter, no state can be returned to the last one and overrun of lines is allowed in Hyper-Paint. Color is sometimes not drawn to the reverse part of the block-toy when lines that are roughly drawn to the block-toy are rotated in a space, as shown in Fig. 3. The design was intended as follows “all actions taken as a drawing-toy are acceptably reflected to a 3D space: curiosity and applied skill are introduced from the user’s own actions”.

### 3.3 Drawing-Toy Look by Drawing of a Profile Line

In Hyper-Paint, shade and shadow processing is not carried out. All coloring is executed by the user in Hyper-Paint; along with all coloring, shadows are executed by a user in a drawing-toy of the actual world. However, a block-toy without shade and shadow processing sometimes cannot portray a 3D figure. Using a drawing-toy of the actual world, a 3D figure can be grasped and a base of an original drawing-toy is expressed by drawing of profile lines of the block-toy as well as drawing of objective profile lines. Consequently, color drawing on a profile line is impossible, and the profile line is always displayed.

### 4. Implementation

For implementation of features described in section 3, some methods exist for their respective implementations. For both cases of addition of a block-toy and a drawing-toy, some processing is required to calculate one point in a 3D space from the position clicked by a mouse. Regarding this method, in one technique, a nearest intersection point is calculated based on judgment of crossing between a vector heading toward the line of sight from the position clicked in 3D space and all block-toy faces. Furthermore, another technique previously embeds an address of the face in each pixel. The address of the face that is embedded in the correspondent pixel is referred from the clicked position. Then an intersection point is calculated from this face. Profile lines are drawn using various methods, e.g., ID edge detection, depth edge detection, and normal edge detection. The ID edge detection technique embeds unique ID values into each pixel at rendering and to process neighboring pixels having different IDs as a profile. Depth edge detection embeds the
depth (distance from a camera) into each pixel at rendering; then the depth of neighboring pixels is compared and profile processing is executed in a similar manner to that of ID edge detection. In normal edge detection, normal line data of the face are embedded into a pixel and are compared with neighboring pixels in a similar fashion.

In depth-edge detection, no profile is drawn between identical depths; no profile can be drawn between block-toys that are adjoined right and left. In normal edge detection, a profile can not be drawn either in right and left directions, or between block-toys adjoined in any direction. In consideration of this matter, ID edge detection was employed in Hyper-Paint. By specifying this ID as an address of the face, this ID is useful for calculation of points from mouse click.

In Hyper-Paint, drawings are renewed 60 times each second at most. The above processing is executed through refreshing of the renderings. This processing requires some time of the CPU, but the action of Hyper-Paint is not affected, even in a notebook personal computer with a 1 GHz Celeron processor.

Implementation procedures are explained in more detail in order. The entire processing flow is portrayed in Fig. 4.

1. Initialization of Pixel Data
All pixels in the off-screen are drawn with background colors; NULL values of invalid ID are installed. In Hyper-Paint, the address of a block-toy’s face is specified as an ID value. The state in which the ID value is NULL indicates the state that the face is not stored. In drawings of the face, when drawings are executed directly on the screen, the process for drawing of the face is seen and a flicker occurs. A pixel buffer of the same size with a drawing range, called an off-screen, is retained in the memory space to prevent this flicker. Processing of all drawings is executed in this off-screen, and the flicker is thereby prevented by final transfer of the image to the screen.

2. Application of the Z-Buffer Algorithm
Simultaneously with rendering by the Z-buffer algorithm to the off-screen, an address of the face is stored as an ID value in each pixel of the off-screen. The Z-buffer algorithm is a method whereby depth data are provided to each pixel constructing a screen, and in drawing of a face for which a coordinate was transformed, depth data of each pixel of the same coordinate are compared. Then depth data in front of the face and colors of the face are written. An address value of the face is also written into the pixel when data of each pixel are rewritten by the Z-buffer algorithm (Fig. 5).

3. Drawing of the Profile Line
An ID value of a pixel is compared with those of surrounding pixels. When pixels have a different ID value, the profile lines are replaced with profile colors. In Hyper-Paint, two pixels are compared: the right pixel and lower pixel (Fig. 6). A bold profile line can be drawn when the number of pixels for comparison increases, but processing time might thereby be increased.

4. Transfer of off-Screen to Screen
An off-screen image is transferred to the screen. It is indicated on the display.

5.6 Mouse Click Judgment and ID Valid Judgment
In the case that a mouse is clicked or dragged, ID values of pixels in the mouse coordinate are referred, and in the case of valid IDs, processing proceeds to the next step.

7. Calculation of the 3D Position at the Clicked Position
The 3D position in a virtual space is calculated from the 2D position that is clicked by a mouse. The ID value obtained in clause 5 indicates an address of the face; the determined 3D position is derived as an intersection point between this face and a vector heading toward the sight direction from the position...
clicked in the 3D space (Fig. 7). Here, position D(x,y,z) of a screen coordinate pntSt(x,y) in the 3D space is derived as the following equation.

\[
D_x = \frac{((2.0f*pntS.x)/(float)VW)}{matP._11} - 1.0f \\
D_y = \frac{-(2.0f*pntS.y)/(float)VH)}{matP._22} - 1.0f \\
D_z = 1.0f
\]

where VW and VH respectively denote the width and height of the drawing range, and matP is a projective transformation matrix.

An intersection point between a plane and a linear line vector is determined as follows. When the determined point and the sight direction vectors are assumed respectively as P and V, a linear equation is expressed as below.

\[ P = V \cdot t + D \] 

\[ (1) \]

Variable t is a parameter. On the other hand, when a normal line of the plane is assumed as N, a plane equation can be indicated below.

\[ N \cdot P + d = 0 \]

\[ (2) \]

Here, d is included in another expression in the plane equation.

\[ ax + by + cz + d = 0 \]

Calculation is immediately executed when three points on the plane are given. Next, eq. (1) is substituted into eq. (2) for determining t.

\[ t = \frac{-N \cdot D - d}{N \cdot V} \]

An intersection point P is determined by substituting t into eq. (1).

8. Processing According to Mode
Processing is executed according to a present input mode, based on the 3D position obtained in clause 7.

In the case of a mode that adds a block-toy, a new cube is added to this position. At this time, new texture images are generated respectively on six faces of the cube; texture coordinates are distributed. The texture coordinate is a relative position of the texture image for which the vertical and horizontal dimensions to distribute to each point of the face are assumed as 1.

For a mode for drawing lines, a texture coordinate of the face is further determined from this 3D position. Colors of the texture image corresponding to the texture coordinate are also renewed.

5. Evaluation and Validation
At one exhibition, Hyper-Paint was used by groups of various ages from 12 to 60 years old (estimated age), and validation for feasibility was executed using an observational method for action and interviews (Fig. 8). Consequently, for generation of a block-toy, it was understood that it was not easy to create shapes according to considerations in the first stage. Moreover, a certain level of practice was required to become accustomed to using it. Especially, many situations existed in which generation of a block-toy could not be executed according to some prior plan. As one example, generation of a block-toy floating in midair was intended by one user in the first stage, but the block-toy adhered to the side faces of a georama box. A block-toy was constructed along the side faces of the georama box, and planar moldings were sometimes seen.

As for the drawing-toy, many users tended to draw lines in the assumption of 2D, but there were many expressions of interest in the fact that the drawn lines were reflected to a 3D space; some users became excited about that fact.

Among users' experiences of Hyper-Paint, there were some uses that we had not anticipated. Some of those ways of use are introduced below.

1. Generation of Shadows
All produced shapes and georama boxes are drawn with black. Subsequently, the whole scene is drawn with white from the present point of sight. White color is not applied to the back of a block-toy, and a "shadow" is thereby generated (Fig. 9). White was used to represent "light," and the principle of projection was used.

Figure 8. Verification

Figure 9. Creating shadow
2. Stencil Effect
Planar patterns are drawn on the side faces of the georama box using a combination of block-toys. The whole scene is drawn. When the block-toys are removed, parts that were positioned under the block-toys are not drawn with colors. Thereby, a stencil effect is obtained (Fig. 10). The block-toy was not used as an element for constructing shapes, but another use, as a tool for removing color, was applied.

3. Effect of Illusionism
After generating a 3D figure, parts that contacted the ground are removed, and a "floating" object is produced. Next, from the present point of view, a picture of shadow showing a shape that "might contact the ground" is drawn. The shadow apparently contacts the ground from the point of view, but when it is rotated, the shadow is understood as "a picture of shadow drawn to merely appear as though it is contacting the ground" (Fig. 11).

4. Hiding of Pictures
A picture is hidden by laying a block-toy on the picture drawn to the bottom face of the georama box (Fig. 12). Used in this manner, a user hides a picture and makes other users seek it. This also uses the block-toy for another application.

6. Discussion
Block-toy operation of Hyper-Paint requires a certain level of practice and experience. Therefore, it is hard to say that the operation is intuitively carried out. From an observational method, many appearances of arranging block-toys that floated in midair were seen at the start point for making shapes. This is inferred to be caused by the georama box enclosing around the block-toys. Ground is applied to other block-toy software, and the block-toy is initially arranged from the ground up. On the other hand, the block-toy of Hyper-Paint can be arranged from an upper face, a side face, or a bottom face of the georama box. Construction only from the ground promotes users' "arrangement from the bottom". It is presumed that an enclosed space like Hyper-Paint does not allow users to be conscious of directionality of space. The users eventually recognize that "arrangement from all directions is possible".

Regarding the drawing-toy, many users showed interest in the mystery by which lines that had been drawn under an assumption of 2D were rendered as 3D, as we expected. Various kinds of usage existed: users enjoyed transformation from 2D to 3D; they drew colors considering resultant 3D images after they understood the principle; and they enjoyed new uses by applying this principle. All of these applications were carried out by users who were older than high school students, but they exceeded our expectations. It is basic that users discovered a new point of view in their play without being constrained by any rules. Hyper-Paint used its differences from actuality to show its value as a software-toy. A production sample of Hyper-Paint is shown in Fig. 13.
7. Summary and Future Prospects
As for Hyper-Paint produced in this study, users' expressions confirmed that this blend of a block-toy and a drawing-toy encouraged the curiosity of children by the mystery shown by transformation of dimensions. As for the prototype produced in this time, problems were found in operation of the block-toy module, but good results were obtained for operation of the drawing-toy. Future prospects for study are explained below.

1. Improvement of Block-Toy Operation
Regarding the present Hyper-Paint, some problems existed in a block-toy operation. Those problems might be improved using a "ground" surface, similar to that applied in other block software, instead of the georama box.

2. Transfer of Block-Toys
Currently, an arranged block-toy cannot be transferred. The block-toy must be deleted and then a new one must be added when a block-toy is replaced to another position. In the future, the block-toy can be transferred using a transfer interface that is easily understood.

3. Generation of Colors
Now, colors for a block-toy and drawing must be selected from nine colors prepared beforehand, but colors other than the above nine colors cannot be used. Expressive powers of children will be improved when users can produce their own desired colors.

4. Network Correspondence
Normal power can be improved when plural persons can use the software. Plural users can play when remote collaborative work becomes possible through correspondence over a network.

The drawing-toy interface of Hyper-Paint can draw users' attention and curiosity through a mysterious experience. It is inferred from this result that we can elicit similar effects in other interfaces also. In an interface, understandability and easy understanding are important factors, but in recent interface studies, attractiveness for users is also noted. Users' motivation toward "using" the program is promoted in various situations of operation. As shown in Hyper-Paint, we expect to improve that curiosity through amenity by "mystery" in other interfaces also, and to enhance users' motivation. However, it is a concern that the user is adversely disturbed when mystery is expressed in a range of difficulty that is beyond the user's understanding. A design that can function within the realm of a user's understanding and intuition is required.

This study proposed and implemented a new expression of a software-toy. Its effects were observed experimentally through users' participation. When this study is developed as an interface, its effects that arise from its mystery and its expression method will be verified, along with its feasibility for education and expression.

Reference

Figure 13. Sample work