2002

TR-2002007: Ubiquitous Puzzle Pieces: 3D Tangible Interfaces for Collaborative Learning Environments

Lori L. Scarlatos

Shalva S. Landy

Saira Qureshi

Ubiquitous Puzzle Pieces: 3D Tangible Interfaces for Collaborative Learning Environments

Lori L. Scarlatos, Shalva S. Landy, Saira Qureshi
Department of Computer and Information Science
Brooklyn College, CUNY
2900 Bedford Ave.
Brooklyn, NY 11210,USA
Tel: 1-718-677-6170
E-mail: lori@sci.brooklyn.cuny.edu

ABSTRACT

3D puzzles provide valuable opportunities for both individuals and groups of people to explore, learn, and create in a natural way. In educational settings, learners using these tools can benefit from occasional intervention by a knowledgeable "guide on the side". In our project, Tangible Interfaces for Collaborative Learning Environments (TICLE), we are exploring innovative ways of enabling a computer to take on that role as children manipulate physical puzzle pieces.

In this paper, we describe recent work on a system for tracking and responding to manipulations of 3D puzzle pieces. This work is unique in that we have developed 1) strategies for tracking multiple wireless objects in a true 3D space, and 2) translation- and rotation-invariant representations of the 3D puzzle state that are used to trigger appropriate responses. We describe how these strategies and representations are used in a system that "watches" as children play with a Soma cube puzzle.

KEYWORDS: Tangible interface, ubiquitous computing, collaborative learning, educational applications, K-12 math and science education, puzzles, Soma cube, stereo vision, sensors, guide on the side.

INTRODUCTION

Blocks and 3D puzzles provide people of all ages and abilities with invaluable opportunities to explore, learn, and create in a natural way [5, 14]. Young children learn about their world by manipulating objects within it. Older children can develop a better understanding of spatial relationships and mathematical concepts by playing with puzzles. Occupational therapists give blocks and puzzles to people whose functions have been impaired by physical illness or injury, to help them develop their cognitive, perceptual, and problem solving skills. For animators, architects and designers, physical models aid the creative process by helping them to better visualize and analyze their creations. In all cases, the opportunity to work collaboratively is an added benefit of working with physical objects.

In learning situations, however, students working with physical objects sometimes need the intervention of a teacher or knowledgeable guide. This "guide on the side" can reinforce key concepts, provide encouragement, get learners to think in innovative ways, and help keep them focused on the task at hand. Yet in these days of shrinking budgets, instructors are rarely able to provide that level of attention; they simply have too many students. What these instructors need are teams of teachers' aides to keep all of their students on track while they (the real teachers) are working with other students.

In answer to this need, we are developing Tangible Interfaces for Collaborative Learning Environments (TICLE). This project strives to create computer-based "guides on the side" that "watch" as children play with physical puzzles, and offer help or suggestions as needed. Our approach is to regard the physical puzzle pieces as elements of a tangible interface. With this system, children are free to explore and collaborate without having to share a computer or learn to manipulate 3D objects using a mouse. Yet they can still benefit from the computer's instruction as needed (and only when they want it). Previously, we had successfully implemented and tested a 2D Tangram puzzle using this approach [12, 13].

In this paper we present our most recent work, which extends these ideas to the third dimension and applies them to a Soma Cube puzzle. This work is unique in that we are tracking multiple wireless 3D objects simultaneously in a small space. Here, we present two approaches to this problem. We have also developed novel representations for the state of the 3D puzzle, which are translation- and rotation-invariant. These representations enable our system to select appropriate hints, give encouragement as progress is made, and offer congratulations when the solution is found. We describe how these techniques are applied to one particular 3D puzzle, the Soma cube.

Although our focus has been on learning math concepts with puzzles, the ideas presented here will readily extend to many other applications. Potential applications include puzzles used by occupational therapists, models used to teach science concepts, and design tools for animators, architects, and designers. Applying our techniques to these other applications will be addressed in our future work.

BACKGROUND

As we looked into the range of possible techniques for tracking 3D puzzle pieces, we found two approaches to be promising. One is to use stereo vision; the other is to use sensors with wireless communication.

Stereo Vision

Human beings use a variety of senses to learn about the three-dimensional world we live in. Yet most people rely primarily on our sense of sight, or vision. Because our eyes are separated by a few centimeters, each eye sees a slightly different image which is sent to the brain for processing. The mind combines the two images by matching up the similarities and adding in the small differences. This three-dimensional perception is useful in making judgments about distances, angles, shapes, volumes, and spatial relationships. Like human binocular vision, stereo vision uses two cameras to capture images of the world; stereo algorithms reconstruct the structure of the scene. The greatest difficulty in stereo vision is something that people do quite naturally: identifying correspondence in the two images.

Stereo vision has many application areas, including measurement and controls for industry, biological sciences, and surveillance and security. For example, José is a visually guided mobile robot that can safely map, explore and navigate unknown indoor environments [8]. It uses three identical wide angle cameras and a frame grabber to capture images that are sent via a radio modem to a host computer for processing. Here, stereo vision enables the robot to operate in unknown, unstructured environments. As another example, the Mechatronic Systems and Robotics Research Group at the University of Surrey is developing robotic stereo head systems for autonomous robot control and augmented reality [7]. Here, too, stereo vision is used to intelligently derive information about a scene, guide a robot through an unknown space, and enable the robot to accomplish a variety of tasks (requiring object recognition) once the robot gets there. Point Grey Research, Inc.'s proprietary Digiclops Stereo Vision System is a full software-hardware solution for real-time applications such as 3D object modeling, face recognition, gesture interfaces, and people tracking in surveillance systems [11].

Stereo vision has also been used extensively to do motion tracking for animation. Here, multiple cameras track the paths of tags placed at key positions on human actors. This technology is continually being enhanced. For example, Utsumi et al. [17] have developed a system for integrating data from non-synchronous cameras. They use an understanding of the human model to help decide which camera view to use. Their system has the advantage of scalability, in terms of both observable region and number of observation nodes, because the cameras don't need to be synchronized.

Sensor Solutions

Although some tangible interface projects have used computer vision to track objects on a 2D surface [12, 16], many others explore alternative ways of sensing where the elements of the tangible interface are. Anderson et al. use modified Lego bricks that have circuit boards inside to

build objects that the computer can recognize [1]. Electricity is sent through the bricks in such a way that determines the shape of the structure. All of the information is then passed to a special brick, called a *drain*, which has a serial connection to the computer. Although this is an excellent idea, it is not wireless; nor does it work in real-time. ActiveCube [6] is a set of cubes that connect to each other in any arrangement, since the faces are the same. Although this is done in real-time, this implementation does not eliminate the need for one cube to be directly connected to the computer.

In a related effort, Gorbet et al. built a system that uses tangible triangles to allow users to arrange presentations by manipulating the triangles [4]. When two triangles are connected or disconnected, they trigger events accordingly. The applications described in their paper are concerned solely with discerning which pieces are touching, and do not require the distinction between which edges are touching, although the paper states that this distinction can be made. Similarly, Camarata et al. use objects—cubes—to affect a presentation based on their orientation and connection to other pieces [2]. Depending on which faces of the cubes are upturned, different information is displayed. Pieces can also be connected to each other if they are related. This is controlled with electromagnets. In their system, infrared is used to relay the state of the cubes to the host computer.

APPROACH

For our application, we were faced with the task of keeping track of where multiple 3D puzzle pieces are in relation to one another, in real time. Our software also had to respond to interrupts from learners, selecting appropriate hints based on the current state of the puzzle. This meant that at least once every second, our software must 1) sense where the puzzle pieces are, 2) generate a representation of the current state, and 3) respond appropriately to the current condition.

Working in educational settings imposed some additional requirements. One is that our system had to work in an environment where several students might be playing with a puzzle simultaneously. Naturally, this environment is filled with noise, both auditory and visual. A second constraint is that the puzzle pieces had to be wireless. We did not want children to get tangled up in wires, or accidentally yank wires out of the back of a computer. A third constraint is that the technologies we employ had to be relatively inexpensive. Ultimately we would like to make these puzzles available to educators, therapists, schools and museums, all of whom have to work with relatively low budgets.

Yet although we wanted to develop an approach that was general enough to support a wide range of 3D puzzles, we found that we could make several simplifying assumptions. One assumption is that the precise positions and orientations of the puzzle pieces are irrelevant; in fact, we want to minimize such constraints on the children. What's really important are the relationships between the pieces. A second assumption is that precise measurement of distances between the puzzle pieces is unnecessary. Instead, we are only concerned with whether two pieces are touching one another and, if so, how they are touching one another.

We developed two parallel approaches for tracking the puzzle pieces. Our first approach is to use stereo vision to determine the positions and orientations of individual pieces. We then convert these values to relative orientations and positions in our representation of the current state. Our second approach is to use sensors to detect touching pieces and wireless communication to transmit that information to the computer. We then use these data to derive a representation of the whole puzzle. Both approaches produce a translation- and rotation-invariant representation of the state of the puzzle, which aids the selection of relevant hints. Our strategies for representing the puzzle support a range of possible puzzle types, and are independent of the tracking method used.

Seeing Puzzle Positions

Our first approach to the tracking problem is to use stereo vision, with images taken simultaneously by a pair of digital cameras. For any computer vision system, the proper input is the backbone of the system. For our system, we paint the puzzle pieces to make them more recognizable. We use distinctive fluorescent colors to identify the pieces, selecting those colors such that they have the widest possible distribution in the YUV color space. We also mark the edges of the puzzle pieces with reflective tape so that they stand out in the images. Controlling the lighting in the learning environment, with the light sources located near the cameras, enables us to survey these edges properly.

Given a stereo pair, we first remove the noise from an images using median filtering. Then we convert the RGB images to YC_bC_r (luminance/chrominance color space) and generate two temporary images for each original: a Y

image, containing just the luminance values, and a C_bC_r image, containing just the chrominance values. Next, we perform the morphological operation of erosion on the Y images, which causes the bright areas to shrink. This reduces the apparent width of the tape on the puzzle edges, making those edges more defined.

The next step is feature extraction. We first detect the eroded edges in the Y images using the Sobel gradient method, applying two 3x3 kernels that detect horizontal and vertical gradients respectively. We then use Harris corner detection to find corner points in the image. These corners and edges define the boundaries of the polygonal faces of the puzzle pieces. Then, the color in the center of each polygon (found in the C_bC_r image) determines which puzzle piece that polygon belongs to. Because the colors captured are bound to vary, as shown in figure 1, we use nearest neighbors to find the best match in the C_bC_r color space. We then use these results to tag the corner points with puzzle piece identifiers.

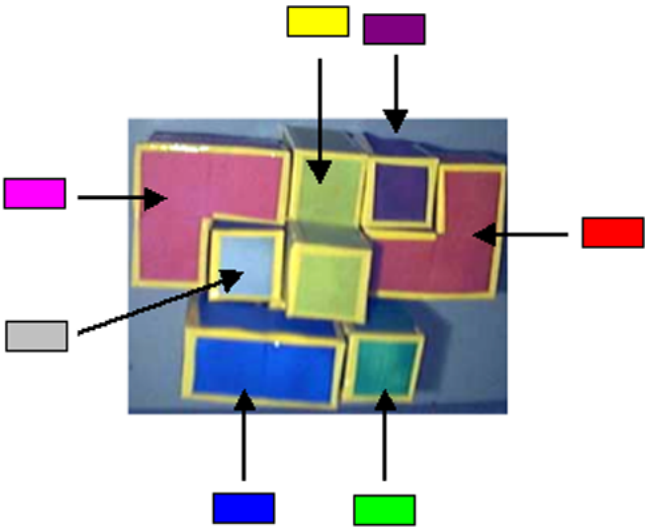


Figure 1: due to a variety of factors, colors in the images will diverge from the colors being sought.

With the corners detected, the next problem is correlation of the stereo pair. We do cross-correlation based on intensity by comparing local neighborhoods of corners. As a neighborhood, a small window of pixels centered around the corner is considered. Epipolar geometry aids the search for corresponding points, reducing the search space from the entire second image to a single epipolar line. So, each pixel in the reference image is compared with pixels along the epipolar lines in other image. The comparison measure is obtained as follows:

$$C = \sum_{i=-N}^{N} \sum_{j=-N}^{N} (I(x-i , y-j) - \hat{I}) (I'(x'-i , y'-j) - \hat{I}')$$

where I and I' are the intensity values at a certain point, and $\hat{I}$ and $\hat{I}'$ are the mean intensity values of the considered neighborhood. For a corner located at **(x,y)**, this means that the corner in the other image will have coordinates located in the interval $\mathbf{[x-w_i, x + w_i] \notin [y-h_i, y+h_i]}$, a small portion of the overall image. For each point found in both stereo images, we calculate the three dimensional world coordinates using:

$$X = \frac{D\,(xL + xR)}{2\,(xL - xR)}$$

$$Y = \frac{D\,y}{xL - xR}$$

$$Z = \frac{D\,f}{xL - xR}$$

where **xL** and **xR** are the **x** coordinates for an image point in the left and right images respectively, **D** is the separation between the cameras' centers and the constant **f** is the focal length of the camera.

The final step is to determine the position and orientation of each piece based on the positions of the corners. We apply affine transformations to the corner points, rotating and shifting the values until we achieve a reasonable match with our internal representation of the puzzle piece. These position and orientation values help us to determine which pieces are touching and how.

Sensing Puzzle Positions

Our second approach to the tracking problem is to use sensors within the puzzle pieces to detect their condition. Although we experimented with strategies for sensing absolute positions and orientations of the puzzle pieces, we were not able to detect these values with the required level of accuracy. However, we are able to detect which puzzle pieces are touching one another, and how.

We extended Anderson et al.'s approach for computational building blocks [1], which uses touching pieces to generate a current that can be detected by the sensors. But instead of

transmitting this information along wires, we use wireless communications via radio frequency (RF) transceivers to send information about the state of each puzzle piece to the computer.

We use a configuration similar to the one shown in figure 2. Two touching puzzle pieces complete a circuit, which allows the passage of an electrical current through a current sensor. By placing distinct resistors at each possible circuit, we can tell which sides of which pieces are adjacent to each other. With additional resistors, we can detect relative orientation as well. The current sensor inputs data to a Basic Stamp, which then formats a message and sends it to the computer through the RF transceiver. The setup of the leads on the outside of the pieces ensures that every adjacency will be sensed twice, once by each of the two pieces. These face-to-face relations then lead to a representation of the current puzzle state.

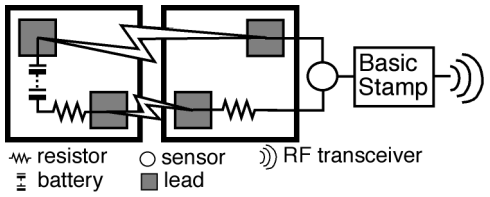


Figure 2: two touching faces complete a circuit. The resulting current is detected by the sensor, interpreted by the Basic Stamp, and transmitted to the computer.

Representing Puzzle State

In order to provide appropriate hints and responses to what the children are doing, the tracking data must produce a rotation- and translation-invariant representation of the state of the puzzle. For most 3D puzzles, we are only concerned with pieces that touch one another, face to face; disjoint puzzle pieces are not yet part of the puzzle, and so we don't really care where they are. In addition, we are only concerned with how they touch in general: precise measurements are not required. Therefore, it is sufficient to represent these relationships only, and then combine them in a unique way. For this, we use a variation of our representation of 2D puzzle states [12].

For puzzles that have only one solution, this representation is sufficient. However, some puzzles such as the Soma cube provide numerous ways of producing a solution. In these puzzles, it is the resulting shape that is most important. Therefore, we have developed a supplemental strategy that allows us to quickly check the overall shape of the combined puzzle pieces. We describe both representations here.

Face-to-Face Relationships. We represent two touching faces on two different pieces with a string of the form:

$p_1.f_1.p_2.f_2.angle.relation$

We presume that in a puzzle with **n** pieces, each puzzle piece has been assigned an identifier, and that these identifiers may be ordered such that $\mathbf{P_1} \le \mathbf{P_2} \le \ldots \le \mathbf{P_n}$. If $\mathbf{P_i} = \mathbf{P_{i+1}}$, then the pieces are identical and therefore interchangeable in the solution. Then $\mathbf{p_1}$ and $\mathbf{p_2} \in [\mathbf{P_1}, \mathbf{P_2}, \ldots \mathbf{P_n}]$ identify the touching puzzle pieces. We also presume that each puzzle piece $\mathbf{p_i}$ is made up of $\mathbf{m_i}$ polygonal faces, each of which is labeled with an identifier such that $\mathbf{p_i.F_1} \le \mathbf{p_i.F_2} \le \ldots \le \mathbf{p_i.F_{mi}}$. In this case, faces that have the same identifier represent symmetry in the piece, allowing for alternative orientations. For example, a cube may be turned any of six ways and still yield the same result. Then $\mathbf{f_1} \in [\mathbf{p_1.F_1}, \mathbf{p_1.F_2}, \ldots \mathbf{p_1.F_{m1}}]$ and $\mathbf{f_2} \in [\mathbf{p_2.F_1}, \mathbf{p_2.F_2}, \ldots \mathbf{p_2.F_{m2}}]$ represent the faces that are touching on pieces $\mathbf{p_1}$ and $\mathbf{p_2}$ respectively. In our string representation, we require that $\mathbf{p_1} \le \mathbf{p_2}$; and if $\mathbf{p_1} = \mathbf{p_2}$, then $\mathbf{f_1} \le \mathbf{f_2}$. This ensures that there is exactly one representation of each possible face-to-face relationship.

Every face of every puzzle piece has an orientation vector which is used to determine how that face is turned relative to another face. If face $\mathbf{f_1}$ has an orientation vector $\mathbf{v_1}$, and face $\mathbf{f_2}$ has an orientation vector $\mathbf{v_2}$, then **angle** in the string representation indicates the angle from $\mathbf{v_1}$ to $\mathbf{v_2}$. This value may be quantized to eliminate small measurement errors and represent relevant changes in orientation only.

Finally, two faces can touch any one of several different ways. The ways that these faces can touch is most generally represented by the 2D topological relationships used in mapping systems [3]. We represent this in the string with **relation**, which can take on any one of the eight possible topological relationships, as shown in figure 3.

Our string representation yields a single substring for every possible 3D face-to-face relationship. Given a set of these substrings, we can produce a unique string representation of the puzzle's current state by sorting and then concatenating the substrings. We may then examine this state by looking

for the presence or absence of particular relationships required in the final solution.

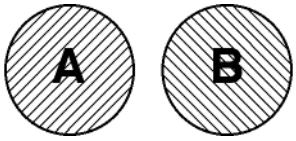
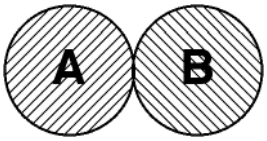
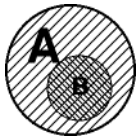
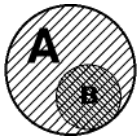
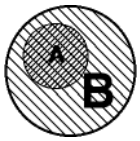
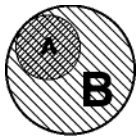
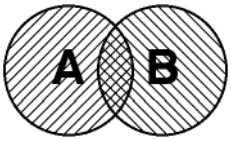
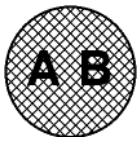
Relation Code	Example	Relationship
0		disjoint
1		A meets B
2		A contains B
3		A contains B and A meets B
4		B contains A
5		B contains A and A meets B
6		A intersects B
7		A coincides with B

Figure 3: eight possible face-to-face relationships and their codes.

Voxel Space Representation. For puzzles in which the overall shape of the solution is more important than face-to-face relationships, we need a volumetric representation of the space. Voxels are the best choice for puzzles whose pieces are composed of cubes or box-like pieces. Given the face-to-face relationships, it is trivial to build a volumetric representation of the touching pieces. To ensure a unique representation of the puzzle state, we translate the voxel model to align its origin with that of piece $\mathbf{P_1}$, and rotate the voxel model to align its axes with those of piece $\mathbf{P_1}$.

IMPLEMENTATION

For our first 3D puzzle, we decided to implement the Soma cube, invented by the Danish poet and inventor Piet Hein [9]. The seven pieces of the Soma Cube are the set of all irregular shapes formed by no more than four face-joined cubes (polycubes) as shown in figure 4. Hein first defined this set of pieces, then discovered that they could be put together to form a 3x3x3 cube. In fact, there are 240 possible ways to do this. As with the Tangram, there are many other figures that can be created with the Soma puzzle pieces as well.

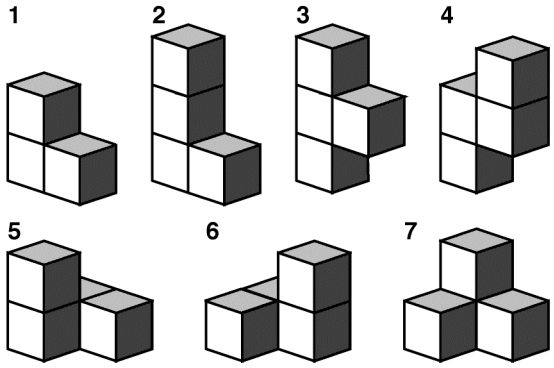


Figure 4: seven pieces of the Soma cube represent all irregular combinations of no more than 4 polycubes.

The Soma cube puzzle has actually generated a great deal of interest in the mathematics community, particularly in the area of combinatorics [10, 18]. Apparently, significant solution hints may be found by looking at the parity of the cube, and by treating it as a partitioning problem.

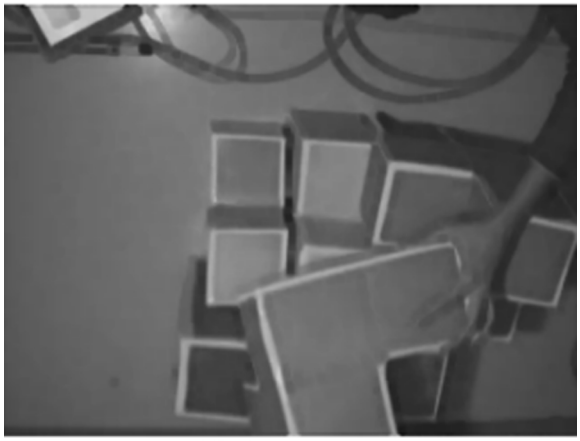
For this puzzle, we decided to implement two versions: one that uses stereo vision to track the puzzle pieces, and another that uses sensors to detect face-to-face relationships. We have not yet decided which approach is better for this puzzle. Yet our strategy for representing the current puzzle state, and the set of hints, are the same for both versions. In both cases, the goal is to construct a 3x3x3 cube using the seven Soma cube puzzle pieces.

Stereo Vision Version

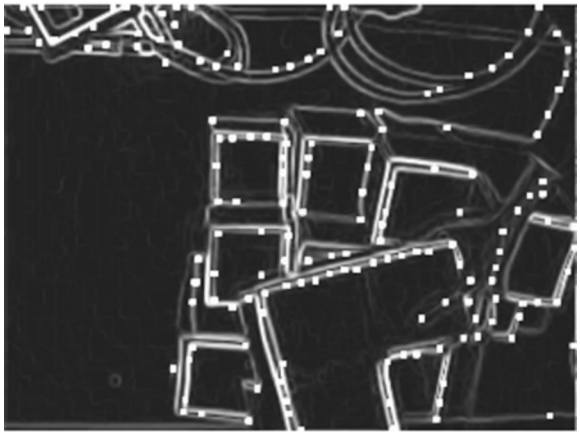
To help distinguish the Soma puzzle pieces from one another, we painted them with seven different colors: red, green, blue, yellow, magenta, gray and orange. The color selection was based on their distribution in YUV space. We

then marked the edges with reflecting tape, enabling us to survey edges properly.

(a) Original image



(b) Eroded image



(c) Detected corners superimposed on results of edge detection

Figure 5: sequence of image processing operations used to find corners for stereo correlation.

Figure 5 shows some of the image processing steps taken to derive the puzzle pieces' positions and orientations in space. Although we have been discussing operations on one pair of stereo images, we actually need to have multiple pairs of cameras capturing images simultaneously. This is because computer vision suffers from the problem of obscuration. Although six orthogonally positioned pairs of cameras would be optimal, we have found that two pairs of cameras, mounted overhead, are generally sufficient.

The information returned by our algorithms is converted to our Soma puzzle representation as described in the section below.

Sensor Version

Using cardboard boxes to construct the Soma cube, we place a Basic Stamp and radio frequency transceiver in each of the seven pieces. Attached to the Basic Stamp is a current sensor for each face of the piece. In order to achieve greater accuracy, we consider each outward facing side of a polycube to be a separate face on the puzzle piece; and so, each Soma puzzle piece has either 14 or 18 faces.

Each current sensor detects an initially incomplete circuit which is completed when two pieces touch. We have arranged the leads such that each face has the potential to complete one of two possible circuits, as shown in figure 6. In this arrangement, opposite passive leads are connected by a wire, a battery, and resistors. Each side also contains one set of active leads, which are attached to the sensor and Basic Stamp.

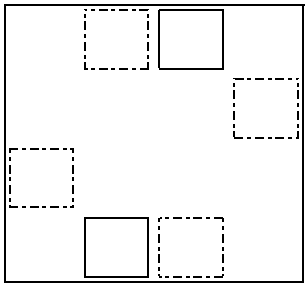


Figure 6: arrangement of leads on each face of the Soma Cube. Solid lines = active leads, dashed lines = passive leads.

When the active leads of one piece touch the passive leads of another, a circuit is completed. The current sensor detects the current on the circuit to determine which side of which piece is being touched. The rotation is also determined this way. Since all possible circuits will have different amperage, 90/270 degree rotations are easily differentiated from the 0/180 degree rotation. To

differentiate the 0 degree rotation from the 180 degree rotation, and 90 from 270, an extra resistor is put into the circuit on the active leads half. This resistor will affect the amperage of the 180/270 degree rotations, but not the 0/90.

The Basic Stamp interprets the current data as adjacency information, and then transmits it to the computer via the radio frequency transceiver. The setup of the leads on the outside of the pieces ensures that every adjacency will be sensed by the two pieces. However, we eliminate the duplication by having only the lower-ID piece transmit that adjacency. This information readily translates to our face-to-face representation.

Soma Puzzle Representations

With 240 possible solutions, we need a volumetric representation of both the current state and solution state. Because the Soma cube pieces are built from polycubes, a voxel model is ideal for this puzzle. However, the transition from tracking information to this representation requires several intermediate representations. These include the face-to-face representations described earlier, internal models of the 3D puzzle pieces, and translation- and rotation-invariant voxel models.

Representing 3D Puzzle Pieces. We identify the seven pieces of the Soma cube using the numbers typically used in the literature and shown in figure 4. Three or four polycubes, arranged on a grid, define the local geometry of each Soma puzzle piece. For simplicity, we define the grid such that each polycube is unit size. Then, the position of a polycube in the model may be indicated by a coordinate triple (x, y, z) representing the location of its lower-left corner. Our representation of each Soma puzzle piece's local geometry therefore consists solely of three or four coordinate triples.

In order to represent puzzle pieces that are touching, we must map these local coordinates to a common coordinate frame. Although we do store an absolute position (x, y, z) and orientation (angles θ and ϕ about the x and y axes respectively) for each puzzle piece in the stereo vision version, we need a representation of the world that is translation- and rotation-invariant. For this representation, we use the position and orientation of the piece with the lowest-valued identifier (typically $\mathbf{P_1}$) to define this common coordinate frame. The origin of the frame corresponds to the origin of this piece, and the axes of the frame are aligned with the axes of this piece. Then, we store a rotation matrix and translation matrix for each puzzle piece to achieve this mapping. These matrices are used later on to create the voxel model described below.

Face-to-Face Relationships. For the Soma cube puzzle, we use the face-to-face representation to derive 3D puzzle information for the sensor-based puzzle only.

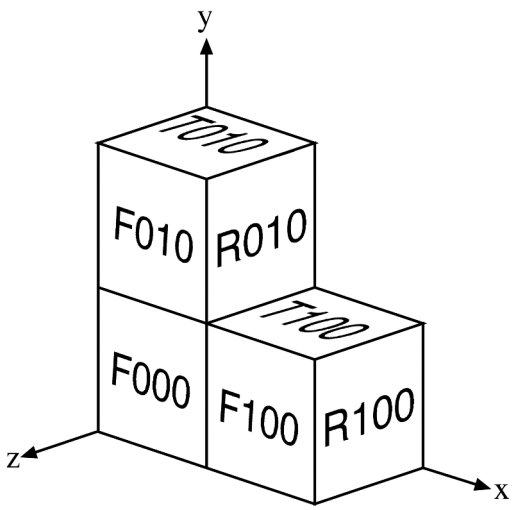


Figure 7: face labels relate to the geometry of the Soma cube puzzle pieces.

We have tried to pack as much relevant information as possible about the 3D puzzle state into this representation. Each face on a puzzle piece is labeled with a coordinate triple representing the location of its corresponding polycube, concatenated with a representation of where the face appears on the polycube: R(ight), L(eft), T(op), B(ottom), N(ear), or F(ar). Figure 7 shows some of the face labels for piece $\mathbf{P_1}$.

For the Soma cube, the only valid angles of orientation between touching faces are increments of 90 degrees. We use orientation vectors corresponding to the **j** vector (y axis) on the 3D model for all R(ight) and L(eft) faces, and to the **i** vector (x axis) for all other faces. Furthermore, the only relevant relationship between faces is 7 (i.e. the two faces are coincident). Figure 8 gives the pseudo-code for deriving the rotation (R) and translation (T) matrices that align puzzle piece $\mathbf{p_2}$ with $\mathbf{p_1}$., and position $\mathbf{p_2}$ within the voxel model.

```
Align ( p1, f1, p2, f2, angle, relation, R1, T1      // input values
        R2, T2)                                      // output matrices for p2
{
        // use angle to determine first rotation
        R2 = rotation by –angle about axis normal to the face
        // use table to determine second rotation;
        // concatenate to first rotation
        R2  = R2 * rotation to align normals to p1 and p2
        // apply R2 to local coordinates of face f2
        f2' = R2 * f2
        // calculate offset needed to align p1 and p2
        T2 = f2' – f1
        // concatenate p1's transformation matrices
        R2 = R2 * R1
        T2 = T2 * T1
}
```

Figure 8: pseudo-code for aligning puzzle piece p2 with p1, given their face-to-face relationship.

Voxel Space Representation. We represent both the current state and the solution state of the Soma cube with voxel models. Once again, each unit in the voxel grid corresponds to the size of a single polycube. Because we are only interested in puzzle pieces that have been put together (i.e. are touching), the current state only represents clusters of touching Soma puzzle pieces. This means that the current state may actually be represented by up to three voxel models. Yet this actually simplifies our representation. For any voxel model, the piece with the lowest ID value defines the coordinate frame. Subsequent pieces have relative orientations in 90 degree increments and offsets expressed in polycube units. We do not have to worry about free rotations or polycubes that are not aligned with grid boundaries.

Voxel models are created dynamically, with dimensions corresponding to the bounding box about the set of touching pieces. Then, each voxel is either filled (i.e. contains a polycube) or empty. A filled voxel specifies which Soma cube puzzle piece fills that space. This helps to determine which hint is appropriate when one is requested.

Guide on the Side

We generate our voxel model(s) of the current state approximately once every second. If the current state is represented by a single 3x3x3 voxel model in which each voxel is occupied, then the solution has been found and congratulations are offered. Otherwise, the current state can be used to trigger a hint. Our basic hints are the following:

- The solution must be 3x3x3; remind users of this if the voxel model extends beyond these dimensions in any direction.
- Disjoint voxels cannot be filled by any of the Soma cube pieces; remind users of this if these are the only unfilled spaces.
- Sometimes there will be a space where a piece could fit, but the user just doesn't see it; suggest turning the piece around and trying to fit it in the available space.
- $\mathbf{P_5}$, $\mathbf{P_6}$ and $\mathbf{P_7}$ are the most complex shapes, and have the smallest number of possible positions in the 3x3x3 cube; suggest putting these puzzle pieces together first.

We are also considering adding the following advanced hints:

- Using parity, one can prove that the central polycube of $\mathbf{P_1}$ cannot lie in any of the corners of the final cube [18]. Parity may also be used to determine whether the current locations of $\mathbf{P_3}$ and $\mathbf{P_7}$ will prevent a solution from being found.
- Viewing this as a set partitioning problem, one can evaluate a starting configuration and determine whether it will be possible to find a solution [10].

CONCLUSIONS

We have presented two alternative strategies for tracking 3D puzzle pieces, and have described translation- and rotation-invariant representations for the current state of a 3D puzzle. We have also described how these are implemented in two versions of the Soma cube puzzle. In the future, we plan to apply these techniques to a puzzle that has a single solution, such as the Tower of Hanoi. We also plan to apply this to puzzles used by occupational therapists to evaluate cognitive abilities of clients. Finally, we would like to explore other potential educational applications in the sciences.

ACKNOWLEDGMENTS

We would like to thank the many people who have made invaluable contributions to this project. This research is based upon work supported by the National Science Foundation under Grant No. 9984385, and by a grant from PSC-CUNY.

REFERENCES

1. Anderson, D., Frankel, J.L., Marks, J., Agarwala, A., Beardsley, P., Hodgins, J., Leigh, D., Ryall, K., Sullivan, E. and Yedidia, J.S. Tangible Interaction + Graphical Interpretation: A New Approach To 3D Modeling. In *Proceedings of ACM SIGGRAPH 2000*, ACM Press / ACM SIGGRAPH, New York, 2000, 393-402.

2. Camarata, K., Do, E.Y. and Johnson, B.R. Navigational Blocks: Navigating Information Space with Tangible Media. In *Proceedings of ACM International Conference on Intelligent User Interfaces* 2002, ACM Press, New York, 2002, 31-38.

3. Egenhofer, M. *Spatial Query Languages*. Ph.D. dissertation, 1989.

4. Gorbet, M., Orth, M. and Ishii, H. Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography. In *Proceedings of SIGCHI 1998*, ACM / SIGCHI, New York, 1998, 49-56.

5. Hughes, F.P. *Children, Play and Development, 3rd Edition*. Allyn & Bacon, Newton, MA, 1998.

6. Kitamura, Y., Itoh, Y. and Kishino, F. Real-time 3D Interaction with ActiveCube. In *Extended Abstracts of SIGCHI 2001*, ACM/SIGCHI, New York, 2001, 355-356.

7. Lawson, S.W. and Pretlove, J.R. Augmented Reality And Stereo Vision For Remote Scene Characterization. In *Proceedings of SPIE Int. Symp. on Intelligent Systems and Advanced Manufacturing, Telemanipulator and Telepresence Technologies VI*, vol. 3840, Boston, Sept, 1999, 133-143.

8. Murray, D. and Little, J. Using real-time stereo vision for mobile robot navigation. Computer Science Dept., University of British Columbia, 1998. On the WWW at http://www.cs.ubc.ca/spider/donm/pubs/wpma/wpma.html.

9. *Introducing SOMA*, Parker Brothers, Inc., Salem, MA, 1969, available on the WWW at http://www.fam-bundgaard.dk/SOMA/NPARKER/NPARKER.HTM.

10. Peter-Orth, C., All solutions of the Soma cube puzzle, *[J] Discrete Math. 57*, 105-121 (1985).

11. Point Grey Research, Inc. Multiclops - Computer Vision Outside the Box. *News and Press Releases*, Aug. 13, 2001. Available on the WWW at http://www.ptgrey.com/corporate/news/index.htm.

12. Scarlatos, L.L. TICLE: Using Multimedia Multimodal Guidance to Enhance Learning. *Information Sciences 140* (2002), 85-103.

13. Scarlatos, L.L. An Application of Tangible Interfaces in Collaborative Learning Environments. To appear in *ACM SIGGRAPH 2002 Conference Abstracts and Sketches*, ACM Press / ACM SIGGRAPH, New York, 2002.

14. Sofilka, M. Work + Play = Learning. Pitt Campaign Chronicle, University of Pittsburgh, available online at http://www.discover.pitt.edu/media/pcc010312/profpuz.html.

15. Trucco, E. And Verri, A. *Introductory Techniques For 3-D Computer Vision*. Prentice Hall, 1998.

16. Underkoffler, J, Ishii, H. Illuminating Light: An Optical Design Tool with a Luminous-Tangible Interface. In *Proceedings of CHI '98*, ACM / SIGCHI, New York, 1998, 542-549.

17. Utsumi, A., Yang H. and Ohya, J. Adaptive Human Motion Tracking Using Non-Synchronous Multiple Viewpoint Observations. In *Proceedings of the International Conference on Pattern Recognition* (ICPR'00), Vol. IV, IEEE, 2000, 607-610.

18. Whiniham, M.J. and Trigg, C.W. Parity and Centerness Applied to the SOMA Cube. In *Mathematical Solitaires & Games*, B.L. Schwartz ed., Baywood Publishing Co., 2001, pp. 20-25, reproduced with permission on the WWW at http://www.fam-bundgaard.dk/SOMA/NEWS/N010110.HTM.