3D Visualization and Exploration of Relationships and Constraints at the Example of Sudoku Games

Gudrun Klinker* Florian Echtler†

Technische Universität München
Germany

Technical Report TUM-I-07-22

ABSTRACT

In recent years, many systems for visualizing and exploring massive amounts of data have emerged. A topic that has not yet been investigated much, concerns the analysis of constraints that different objects impose on one another, regarding co-existence.

In this paper, we present concepts toward visualizing mutual constraints between objects in a three-dimensional virtual and AR-based setting. We use 3x3 Sudoku games as suitable examples for investigating the underlying, more general concepts of helping users visualize and explore the constraints between different settings in rows, columns and blocks of matrix-like arrangements. Constraints are shown as repulsive and magnetic forces making objects keep their distance or seek proximity.

We expect our concepts to be valuable beyond gaming – both as didactic tools for teaching and research, and in computational steering scenarios where problems are too complex to be solved by the computer alone in a reasonable amount of time. We are in the process of developing several prototypical visualization and interaction environments for Sudoku puzzles, allowing users to explore constraints in virtual 3D settings on a desktop and in an Augmented Reality-based environment. This paper reports on the underlying visualization principles.

1 INTRODUCTION

In recent years, many systems for visualizing and exploring massive amounts of data have emerged [16]. Systems such as the Attribute Explorer [19, 16] and the EZ chooser [1] are very useful tools to help users understand the distribution of objects within a large parameter space and to select a subset of objects that satisfy user-defined parameter constraints. The systems provide users with sensitivity toward finding the most useful parameter settings by indicating objects that are barely out of bounds of selected parameter ranges, as shown in the Influence Explorer [18]. Other systems allow users to visualize and explore relationships between objects, e.g. in Cone Trees [12], Tree Maps [7], the Analyst's Notebook [5], InfoCrystal [17], and cluster maps [3].

A topic that has not yet been investigated much, concerns the analysis of causality. Relevant to this topic is the analysis of constraints that different objects impose on one another, regarding co-existence. Example applications include np-complete problems, such as route planning or scheduling [22, 21] for which users need to find and interactively explore solutions when the side conditions change in midst the daily business: parameters involved in a large number of parallel equations can be represented in large matrices, with the cells (the parameters) imposing constraints on one another. In order to find acceptable solutions, users have to determine acceptable settings for all parameters. In the process they have to iteratively explore different settings - experiencing the mutual influence between settings for different parameters, according to the given side conditions. Smith et al have presented an approach toward visualizing such dynamic inter-dependencies in the Model Maker system which allows them to interactively explore suitable parameter settings for high polynomials that are fitted to given data sets [15].

In this paper, we present concepts toward visualizing mutual constraints between objects in a three-dimensional virtual setting. We use 3x3 Sudoku games as suitable examples for investigating the underlying, more general concepts of helping users visualize and explore the constraints between different settings in rows, columns and blocks of a matrix-like arrangements. We consider this choice to be particularly attractive since Sudoku puzzles have recently gained significant public interest world-wide. In their unique way of expressing a high number of inter-related constraints between a very low number of symbols they capture people's interest in solving complex puzzles. 3D visualization provides excellent means to help people visualize such constraints, to make virtual notes, and to review, re-investigate and refine prior notes in the light of new insights. One critical aspect is here, not to spoil the game. The computer is an assistant to the human, supporting him or her in understanding the problem rather than automatically providing the final solution. We expect such scenarios to be valuable beyond gaming – both as didactic tools for teaching and research, and in computational steering scenarios where problems are too complex to be solved by the computer alone in a reasonable amount of time.

We are in the process of developing several prototypical visualization and interaction environments for Sudoku puzzles, allowing users to explore constraints in virtual 3D settings on a desktop and in an Augmented Reality-based environment. This paper reports on the underlying visualization principles, showing them most completely in the 3D desktop environment. We are currently solidifying the AR-based and the multi-touch versions, as well as various mixed combination of interactive facilities. Future papers will report on such versions, as well as on user studies that compare user performance across all platforms and across subsets of visualization facilities.

2 PHYSICAL METAPHORS FOR 3D VISUALIZATION OF GEOMETRIC CONSTRAINTS

At the very essence, Sudoku imposes geometric constraints on the positioning of digits on the board: every row and every column must contain every digit exactly once. The same is true for blocks. As a consequence of these constraints, digits that are already placed on the board have strong consequences regarding options for positioning further ones. We use a three-dimensional layout to present such constraints. In this paper, we suggest visualizing them as physical metaphors that most users are expected to be familiar with from daily life. To this end, the grid is shown on a horizontal plane in a

*e-mail: klinker@in.tum.de
†e-mail: echtler@in.tum.de

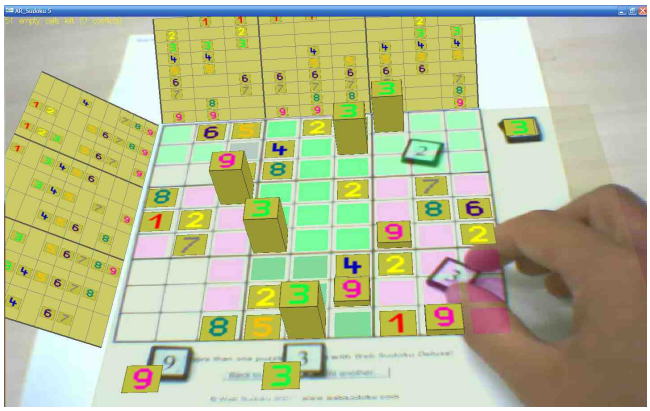

Figure 1: AR Sudoku.

three-dimensional scene. Constraints are shown as structures and physical effects arising from the plane and hovering above it.

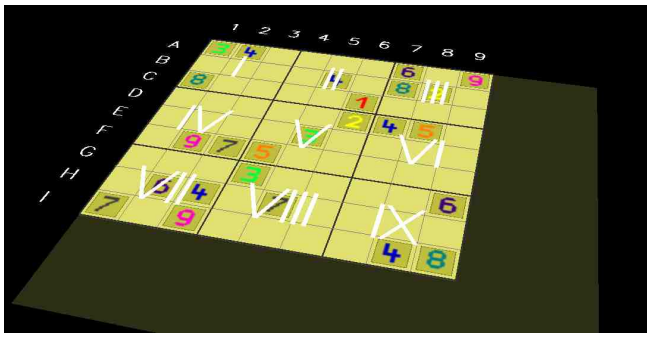

Figure 2: Layout of a 9x9 Sudoku board.

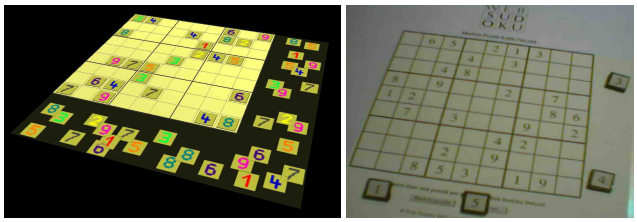

Figure 3: Three-dimensional arrangement of board with tiles in a virtual (VR) and a real (AR) setup. (Puzzles taken from WebSudoku[4]. Left: evil puzzle 4,006,671,090. Right: medium puzzle 6,266,736,022).

Figure 2 shows the arrangement for a 9x9 Sudoku board. The example is taken from the WebSudoku web site [4]. For later reference, rows are labelled A-I, columns are labelled 1-9, and blocks are labelled I-IX. The board is shown either virtually or as a real piece of paper on a table (see Figure 3). As the major means for interacting with the Sudoku game, users can place and move virtual [10] or real tiles with digits 1-9 on or next to the board.

2.1 Repulsive Forces

Digits that are already placed on the board inhibit other cells in the same row, column and block to accept the same digit. If the set of all still available digits is envisioned as a collection of loose tiles, waiting to be placed, the already placed digits perform a repulsive force on these tiles, pushing the ones that have the same digit away. These repulsive forces are imposed along rows, columns and within blocks. We visualize such forces by raising respective digits on the board to high towers that darken (cast shadows on) the row, column and block that they are occupying (see Figure 4).

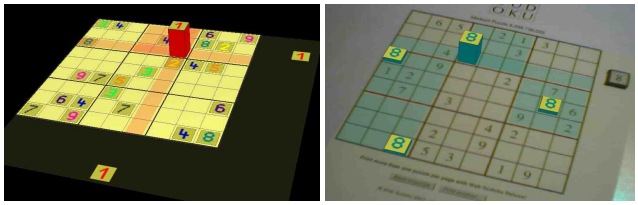

Figure 4: Left: Digit 1 exhibiting a repulsive force on cells in its row, column and block (responding to two tiles 1). Right: All digits 8 on the board are exhibiting repulsive forces on their blocks and on rows A-C (responding to tile 8).

By placing tiles with selected digits to the right or below the board, users can inquire which rows, columns and blocks have already been set and thus exert repulsive forces. The system groups such inquiries into threesomes of rows and columns in accordance with the blocking structure. The rationale for this visualization is based on one of the prime strategies for solving Sudoku puzzles: if two columns already contain a specific digit, the placement of this digit in the third column is constrained to lie in maximally three grid cells of one specific block. The same applies to threesomes of rows. The left picture in Figure 5 shows this situation for the digit 3. Since the 3 occurs already in columns 4 and 5, it is blocked off in blocks V and VIII. For column 6, it is thus restricted to block II. Within block II, placement of the digit 3 is restricted even further since row A is also blocked due to a 3 in cell A1. Furthermore, cell C6 is set to 1 and is thus not available. Judicious placement of two tiles 3 to the right of rows A-C and below columns 4-6 helps the user visualize this situation and identify cell B6 as a unique position for a 3 on the board. In a similar way, the placement of two tiles 8 in the right picture reveals a unique position for an 8 in position B2.

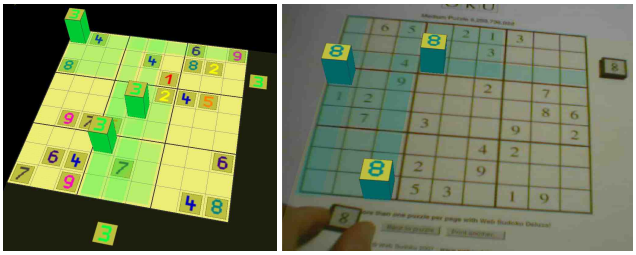

Figure 5: Left: Determining that cell B6 must be a 3 by judicious placement of two tiles next to rows A-C and columns 4-6. Right: Cell B2 has to be an 8.

2.2 Attractive Forces

In contrast to the constraint, that only one digit of each kind can be placed in a row, column or block, Sudoku players also have to remember that each row, column and block does have to contain every digit. As a consequence, rows, columns and blocks that are missing certain digits are direly attracting these. There are several ways of visualizing such attractive forces.

Funnels: Analogously to the exploration of repulsive forces, users can explore attractive forces by moving tiles along the sides of the board. In this case, a metaphorical funnel symbolizes the fact

that a cell in a respective row or column is open toward being set to the digit of the tile. Funnels are shown on all those cells of a row or column that are not yet set and that do not have a conflict with already existing digits in other rows or columns (repulsive forces).

In the left picture of Figure 6 columns 2 and 3 still require a 3. Funnels indicate cells in these columns that can be set to 3. Since there is only one such funnel in column 3 (above cell D3), this cell is uniquely determined to be set to 3. Similarly, in the right picture, there is only one funnel in column 2 at cell B2 for the digit 8 – supporting the finding that was already evident from Figure 5. The picture also determines a unique position for an 8 in row A, at cell A9.

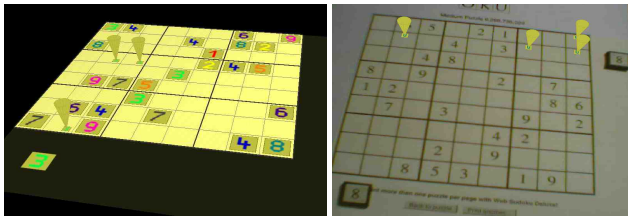

Figure 6: Representing attractive forces by funnels.

Overviews of Unsatisfied Requirements: In addition to indicating missing digits for individual cells upon interactive user request (by moving tiles), we also provide row-wise and column-wise overviews of all digits that have not yet been placed in a row or column. These overviews are shown as slanted two-dimensional panes hinged to the left and top sides to the board. The left pane is aligned with the board such that rows on the pane directly correspond to the rows on the board. Each overview row represents a vector of all possible digits 1-9; digits that have not yet found a place in this row on the board are shown, whereas those digits that have already been taken show an empty slot. The same applies to the overview pane at the top side of the board, indicating the availability of digits in columns of the board. As a physical metaphor, users can envision that the digits that are still on the slanted panes obey the attractive force of gravity and thus have to slide down onto the board and fill the respective rows and columns.

The overview panes help users to quickly obtain an understanding a) of how individual digits spread across rows and columns, b) their row-wise and column-wise distributions, and c) the overall availability of individual digits. Users can see at a glance which digits have only few left-over representatives. Starting with these is a good strategy because placement of the remaining few representatives is likely to be much constrained - and thus generally easy to resolve uniquely.

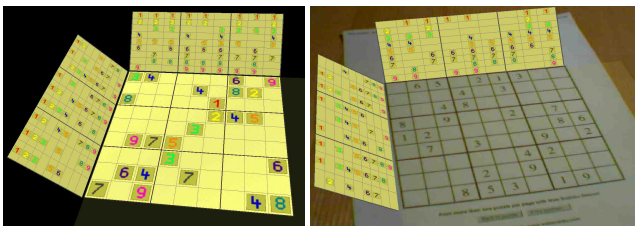

Figure 7: Presentation of all not-yet-set digits in overview panes for all rows and columns.

2.3 Individual Markups

Another technique that is much used in real Sudoku puzzles concerns the systematic cell-by-cell evaluation which digits can be placed there without conflict. Typically, results are penciled as tiny annotations into the grid. Books recommend that Sudoku players should have pencils and erasers close at hand when trying to solve a tough puzzle.

Virtual or AR-based Sudoku games are very suitable for supporting such annotations. For a selected digit (shown on a tile), the user can move the mouse across the board and explore, based on repulsive and attractive forces, whether a particular cell can take the digit or not. If yes, this can be annotated in the cell via a right mouse click. The cell then shows a miniaturized version of the tile in one of nine dedicated positions. Of course, the system can automatically determine for entire rows, columns, or even for the entire board, for which cells a given digit is still an option – as indicated by the funnels in Figure 6. The current version of the system allows users to either set markups individually for single cells, or for blocks of rows or columns. In the latter case, the user has to perform a right mouse click outside the board itself within the areas where tiles can be moved. All tiles showing a funnel are then marked up. With this facility, users can quickly determine all conflict-free positions of a digit – and even of all digits, as shown in Figure 8.

Currently, such interactivity is not yet provided during tangible use of real, physical tiles. Future work will investigate suitable gestures (or suitable use of dedicated special tiles) to request the markup-facility from the system (corresponding to a right mouse click). In the current version, virtual or real tiles can be placed on the board; the mouse has to be used for interactive markup.

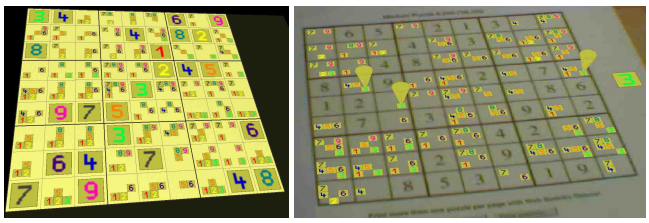

Figure 8: Marking up each cell with individual lists of yet eligible digits.

There are further logical lines of reasoning that skilled Sudoku players apply to difficult puzzles. In Figure 9, digit 3 in cell E5 partially blocks the setting of 3's in blocks IV and VI. The funnels indicate that block IV still has options in row D (cells D2 and D3), whereas block VI has options both in row D (cell D9) and in row F (cells F7, F8, and F9). These results can be marked up on the board, as shown in right picture in Figure 9. However, since block IV must place a 3 in row D (D2 or D3), cell D9 in block VI cannot have a 3. This means that block VI can ignore cell D9, since attractive forces in row D have to give priority to block IV. As a consequence of this analysis, users need to be able to indicate that cell D9 is off-limits for digit 3. The system allows users to make a negative mark in a cell by right-clicking on the cell a second time. (Toggle between three markup modes: *good*, *bad*, and *undetermined*). The lower picture in Figure 9 shows the visualization of a "bad" markup (darkened digit) in cell D9.

Figure 10 shows the limits of complete information visualization: although all information is presented consistently and succinctly, users may be overwhelmed by the amount of details in tough puzzles that they need to sort out by themselves. The picture on the right shows that funnels provide a means of focusing on selected aspects of the full view of all information. We will investigate and evaluate the relative contribution of different visualization schemes in user studies in the future.

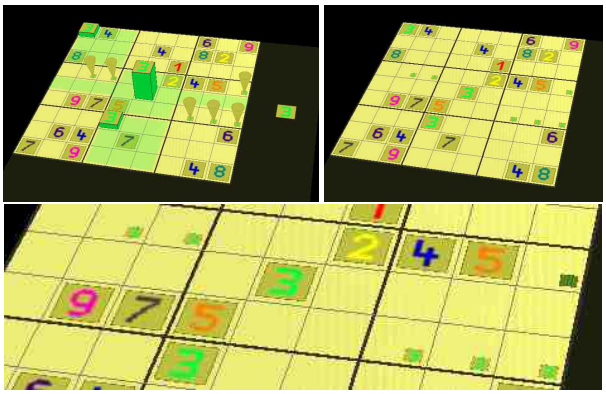

Figure 9: Presentation of negative markups: Cell D9 cannot be a 3.

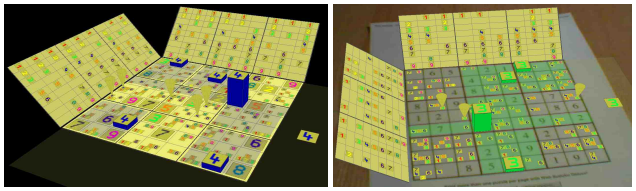

Figure 10: All presentation schemes combined (shadows, funnels, overview panes and markups).

2.4 Exploration by Trial and Error

In some situations, there is no evident, unique option for a user to proceed further – or, at least, the user may just not be able to realize that a unique option exists due to limited puzzle solving skills. At that point, a potential next step has to be guessed and explored. Should it prove to lead to no consistent interpretation for the entire board, backtracking has to be used in order to try out an alternate option.

The system provides trial-and-error facilities by pushing the current state of the board onto a stack and popping it off the stack during backtracking. The current stack level is indicated on the display.

3 Implementation

3.1 Long-term requirements for intended use

The system is intented to be used for investigations whether and in which way 3D visualizations, both in a desktop environment and in augmented, real settings can help users explore mutual constraints between objects. To this end, intended setups vary from 3D visualizations on a regular desktop computer with keyboard and mouse to an AR-based arrangement, in which users wear an HMD with a monocular camera and look at a planar piece of paper with a Sudoku puzzle that is taken out of an arbitrary periodical, booklet or is a printout from the web. In this aspired setup, users are not expected to prepare the paper in any way (e.g., by attaching a marker to it [8]). Interaction with the system is achieved via tangible objects (e.g., tiles). Both the Sudoku grid on the paper and the tiles are automatically recognized and tracked while users interact with the scene, partially obscuring parts of the paper and some of the tiles, or moving objects out of the field of view.

3.2 Current setup

While striving toward the long-term goals, the current setup makes some simplifying assumptions. We currently use a hybrid of a desktop-based setup showing purely virtual 3D presentations and a video-based AR setup.

The setup allows for traditional keyboard and mouse input whenever necessary. All visualizations are shown on the computer monitor rather than in an HMD. Optionally, a web camera can be used. If the camera is not plugged in, a description of a Sudoku grid is read from a file, and the user solves the puzzle in the purely virtual setting (left pictures in Figures 3-10).

If, on the other hand, the camera is plugged in, users see a video-based augmentation, merging raised tiles, funnels, overview boards and markups with the live video stream (right pictures in Figures 3-10). Typically, the camera is attached to the computer monitor and oriented at a piece of paper that lies on the table next to the computer – ideally halfway in between the user and the monitor. To avoid upside-down viewing of the Sudoku game when the camera is facing the user, the video image can be flipped and then texture-mapped onto a plane defined by the grid. In this case, the viewpoint is not aligned with the camera but can rather be set interactively by the user (presumably to coincide approximately with his/her viewpoint).

The system initially detects the Sudoku grid in the video stream and determines the pre-set numbers using optical character recognition. Subsequently, it tracks the grid. No large motions are expected, since the camera is attached to the monitor rather than to the user. Yet, the tracking system must be able to tolerate significant occlusions by the user's hands. Whenever tracking is lost, the system re-detects the grid (without re-reading the digits that are pre-set on the board).

For direct interaction, the system is able to track tiles that users can move about in order to explore repulsive and attractive forces. All further interaction is currently still provided via mouse and keyboard. One by one, we will investigate suitable 3D metaphors using direct tangible interaction to substitute the mouse clicks and keyboard entries.

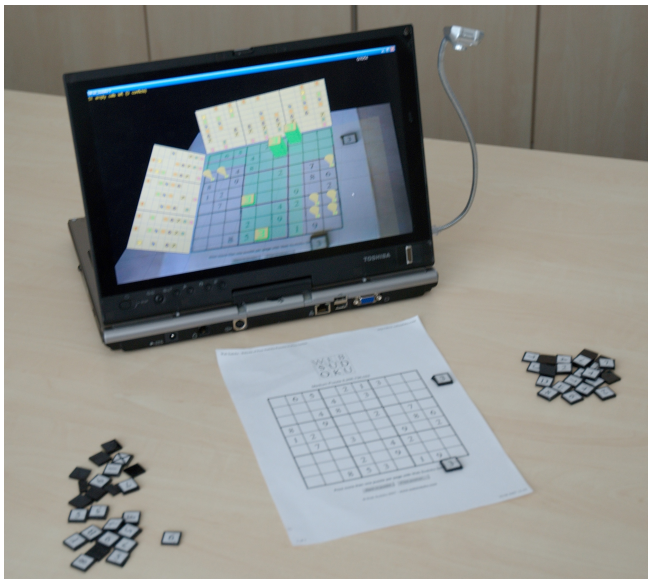

Figure 11: Setup for AR-based presentation.

The system is designed as a model-view-controller architecture, thereby allowing for easy interchange of display and interaction schemes in the future.

3.3 Board model

The system has an internal model of the current state of the board. Per cell it keeps track of the cell value, an array indicating which digits are still acceptable, the current cell state, and a list of user-provided markups. The cell state is *final*, if the cell value is part

of the initial puzzle definition and the cell value thus is known and fixed. All other cells toggle between states *unknown* and *guessed*. The values of these cells can be interactively set and modified, whereas the values of the final cells cannot be changed by the user. For the AR-based game, the system further differentiates between cell states *final* and *finalOCR* to account for potential errors when the system reads the digits from the piece of paper (see below).

Each cell keeps a list of user-provided markups. For each potential cell value 1-9, the markup can have one of three labels: *good*, *bad* or *undetermined*. Such markups are user-controlled (see below). To allow for user mistakes, they are not checked for consistency.

The board model further maintains three internal schemes to evaluate board-wide consistencies: one for row-wise consistency, one for column-wise consistency and one for block-wise consistency. In each case, arrays exist that indicate which values 1-9 per row, column or block have already been taken.

Finally, there is a list of all interactive tiles that are currently in use.

3.4 Display

The visualizations that were discussed and presented in section 2 are implemented using OpenGL and glut. No particularly complex operations are necessary. In the following, a few differences between the presentation in a purely virtual setting and in an AR-based setting are presented.

Virtual Sudoku: For the purely virtual setup, the system uses a black background on which it presents the Sudoku board. Optionally, users can include or exclude the visualization schemes of section 2.

AR-Sudoku: In the AR-based visualizations, the current video image serves as a back-drop to the visualization schemes. Upon request, the image is flipped and texture-mapped to the 3D plane defined by the Sudoku grid, as detected and tracked by the system. Alike the VR setup, various visualization schemes can be overlaid according to user requests.

In comparison to the virtual setup, the AR-setup has to be more careful with respect to the realistic rendering of added-in digits. Differences between initial and added cell values should be visible - yet, not so large that they refrain users from obtaining an integrative, overall understanding of the current state of the puzzle. Cells that are already set in the original puzzle are shown directly from the original video stream. For guessed cells, similar visualizations of the current cell value are created by mapping the texture of a representative digit, found during the OCR-phase, into the screen space of the cell in question (see Figure 12).

Figure 12: Texture-based visualizations of guessed cell values.

3.5 Interaction

There are several different groups of interactions: a) facilities for simple system control, b) schemes to manipulate the boards c) methods to trigger and manipulate interactive visualizations, and d) interactions to really play the game. For each group, it is necessary to consider, how control can be exerted in different system setting (virtual Sudoku, AR-Sudoku, and more in the future).

Simple control: Simple toggling operations allow users to change the viewing state of the system. They are also needed during system setup time in AR-Sudoku when the board is detected, and the inscribed digits are recognized (see section 3.6). Thus far, these operations are not strongly required during the game itself – since the different schemes can all co-exist. For both presentation setups, they are currently controlled via keyboard and mouse input.

Board manipulation: This aspect of the game depends heavily on the physical setup. Very different concepts exist for Virtual Sudoku and for AR-Sudoku. In *AR-Sudoku*, manipulations do not need a highly specialized interaction scheme – since it is the very essence of Augmented Reality that objects are part of the real world and can thus be manipulated and viewed directly and without specially provided system control. It suffices to merely move either the piece of paper or the camera. A hard consequence is, however, that complex and robust object detection and tracking methods are needed. Those will be discussed in section 3.6.

In *Virtual Sudoku*, 3D rotation of the board is provided via mouse interaction, using a needle map metaphor. Manipulation is enabled when the user clicks outside the interactive area of the board. A circle and an enclosing square appear (see Figure 13). Within the circle, left mouse clicks are interpreted to define a vector orientation on a Gaussian [6] or virtual [2] sphere, relative to the center of the circle which is positioned at the center of the Sudoku grid. Mouse dragging results in rotations relative to the normal of the plane that is spanned by consecutively measured vector orientations according to the changing mouse positions. Outside the circle, rotations are defined to be orthogonal to the current viewing direction (i.e., a roll around the z-axis). Right mouse clicks and drags are interpreted as object translations. Within the circle, translations are in x and y. Outside the circle (but within the square) right-clicked mouse motion is interpreted as a zooming operation (in z). Clicks outside the square terminate the transformation mode, thereby toggling back into the tile manipulation mode (presented next).

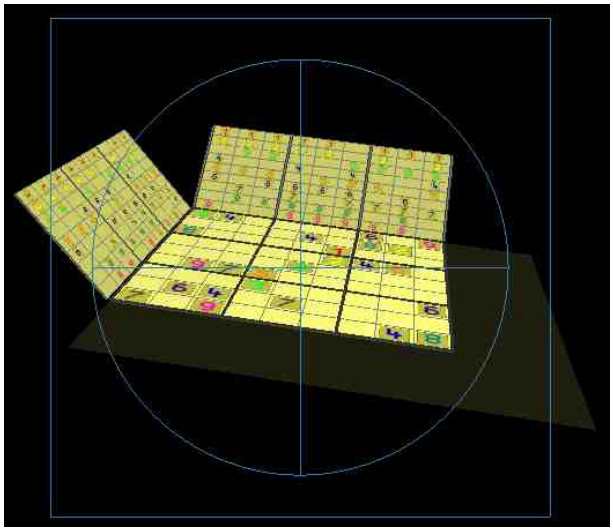

Figure 13: 3D rotations and translations in virtual setups.

Interactive visualizations: Most of the time during a Sudoku game is spent staring at the board and trying to deduce suitable next actions from the current setting and all currently applicable constraints. During this phase, interactive control of visualization aspects is essential to the overall performance. To this end, the virtual board is extended on the left side and in the front by an interactive area (width of 3 columns or rows, resp.). It is shown semi-transparently in dark gray (see Figures 2 and 13).

As presented in section 2, users can visualize repulsive and attractive forces by moving tiles below or on the right side of the Sudoku grid. In *Virtual Sudoku*, new tiles are entered into the visualization by pressing the respective key 1-9. A virtual tile is then placed at the current mouse position and added to the list of tiles. Multiple tiles of the same, as well as of other digits can be entered this way. A tile can be moved by pressing and dragging the left mouse button in the vicinity of the tile. The closest tile is selected to accept user manipulations. Respective repulsive and attractive forces are shown in row- and column-wise manner on the board. A tile is removed from the list of tiles by quickly dragging it off the interaction area or beyond the scope of the window. In *AR-Sudoku*, new tiles are entered into the game by picking physical tiles out of a container and placing them next to the board. They are removed from the list by moving them out of the interactive area or out of the field of view of the camera. Whenever they get back in sight, they are reentered into the list.

A further, very powerful means of exploration is provided by user-set markups. This concept allows users to make annotations to individual cells of the Sudoku grid, stating which values the cell can still accept and – in a more sophisticated annotation scheme – which values are off-limit due to complex lines of reasoning. In the current setting, the system can assist users by automatically performing simple evaluations of the currently applicable set of repulsive and attractive forces. More complex reasoning schemes (such as those outlined in section 2.3) are left to the creative and explorative skills of the users. In *Virtual Sudoku*, annotations are accepted via clicks of the right mouse button, interpreted with respect to the current position of the mouse and the most recently selected tile (the tile doesn't have to be dragged along, i.e. the left mouse button is not required to remain pushed down). If the right button is clicked within the board itself, the cell that is closest to the mouse reacts by adding the digit of the currently active tile to its list of user markups. If the markup was already listed, its state is toggled from *good* to *bad* to *undetermined* and back to *good*. If the right button is pressed outside the board, but still within the interactive area of the board, the right mouse click is interpreted as a request, asking that threesomes of rows (or columns) next to the tile are evaluated with respect to repulsive forces from cells that have already been set. Special care is taken not to overrule previous delicately provided markups of the user, i.e. cell values that a user has previously set to *bad* are not automatically re-set to *good*.

For the augmented version of our system, *AR-Sudoku*, we have not yet developed a tangible scheme that is equivalent to pressing the right mouse button. For this reason, users currently still have to use the mouse button – in combination with moving the physical tiles on the board. Alternatively, they can also switch to the completely virtual method, i.e. use the mouse to interact with virtual tiles on the video picture of the real Sudoku puzzle.

Game playing: Two actions can be performed for game playing: digits can be placed on the board, and they can be removed. In *Virtual Sudoku*, users can place digits on the board by clicking with the left mouse button in an empty cell. Digits that are not declared *final* can be removed from the grid by clicking on the cell again with the left mouse button. We have not yet devised a scheme for *AR-Sudoku* because we have not yet decided how to distinguish explorative tile motions to visualize repulsive and attractive forces from the action of placing or removing a digit. To this end, we need to find a concept to distinguish between hovering over a cell (i.e.: mouse motion without button presses) and actually selecting it (i.e.: a button press). As described above, users currently have to resort to virtual or mixed interaction that involves the mouse.

3.6 Board recognition and tracking

For *AR-Sudoku*, the printed grid and physical tiles need to be detected, recognized and tracked on the table such that augmentations can be properly aligned. As a basic design principle of the system, we want the setup to be as natural as possible. In particular, we want users to be able to use Sudoku puzzles right out of books and periodicals that they might own. Thus, algorithms cannot count on the existence of special markers or hardware.

For selecting suitable algorithms, the following requirements need to be considered: first, grid detection and tracking must be very robust against partial occlusions since the users' hands and some tiles will be in the scene most of the time. Second, the grid is a highly repetitive structure. On the one hand, algorithms can benefit from this. On the other hand, many current tracking algorithms count on irregular features.

Our system is based on a number of routines from OpenCV.

Grid detection: For initial grid detection, we expect users to arrange their setup initially such that the entire grid is within the field of view. The system detects lines in the image, using a Hough transform such that occluding hands are ignored as much as possible. It selects the maximum Hough entry to hypothesize a first line and then removes entries in the Hough space that can be attributed to this line. It repeats the process until a given number of lines is found. Figure 14 shows results for finding the lines of a grid that is partially covered by a hand. Line colors correspond to the colors of circles in the Hough space indicating the respective maximum. The picture shows that all grid lines are found reliably, yet, a number of further lines exist, e.g. along the edge of the paper, and aligned with the fingers.

To disambiguate better between lines, the system also detects feature points using the algorithm by Shi and Tomasi [14]. By checking whether feature points lie on edges, we are able focus on lines of the grid and their intersections. The latter ones are indicated in Figure 14 by thick white circles. Subsequently, the arrangement of lines and intersections is analyzed and interpreted to find the lower left corner of the grid and to fill in any lines that might have been missed (e.g., due to occlusion).

The system repeats this process until it was able to detect the complete grid. It then computes the camera pose relative to the grid [23] and projects all line intersections of the 3D grid model into the image in order to be able to refit the grid more precisely to the image. The camera pose is recalculated accordingly.

Recognition of digits: When the grid has been detected and the camera pose has been calculated, it is analyzed such that a grid model can be derived that describes the initial arrangement of digits in the puzzle. By projecting the grid model into the image, the system has a clear understanding where to look for cells that could contain a digit. It extracts and rectifies the inner pixels of a cell (excluding pixels close to the boundaries). If an area is homogeneously bright, is is assumed to represent an empty cell. Otherwise, the system compares the cell area to a small collection of image patterns that describe the appearance of the digits 1-9. The pattern with maximal correlation is assigned to the cell. Figure 15 shows a typical result from the digit recognition routine. To ensure that the board is correctly interpreted, users are shown the interpretation and asked to make interactive corrections and then to accept the interpretation. The real Sudoku game starts after the grid was accepted.

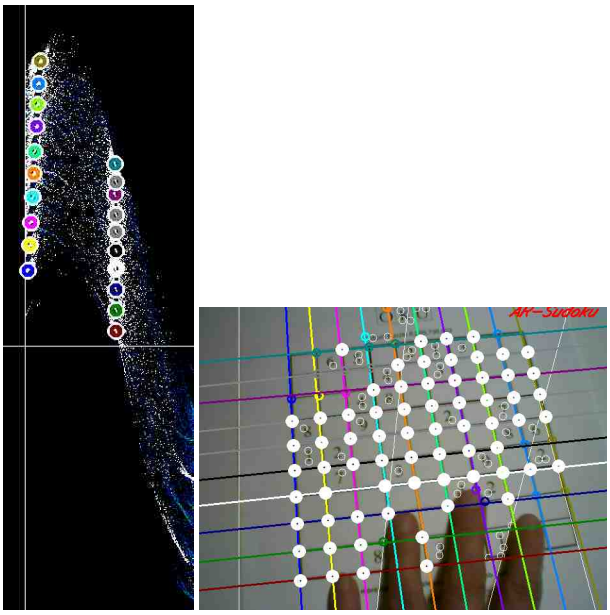

Figure 14: Grid detection (Hough space of line parameters, lines and cross points overlaid on the image).

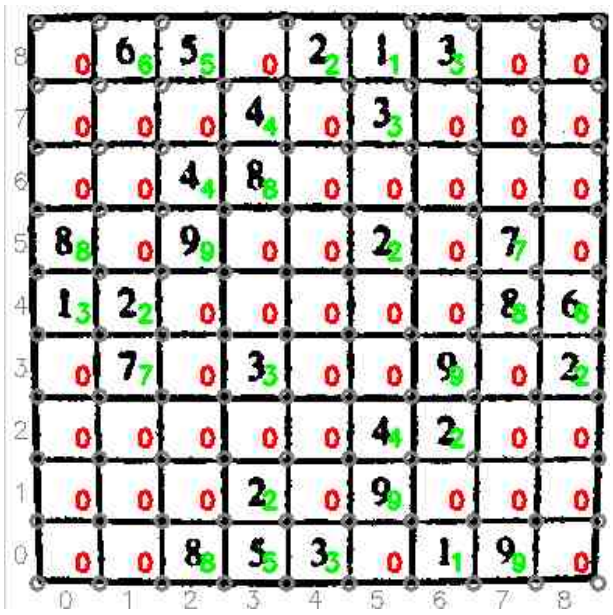

Figure 15: Recognized digits.

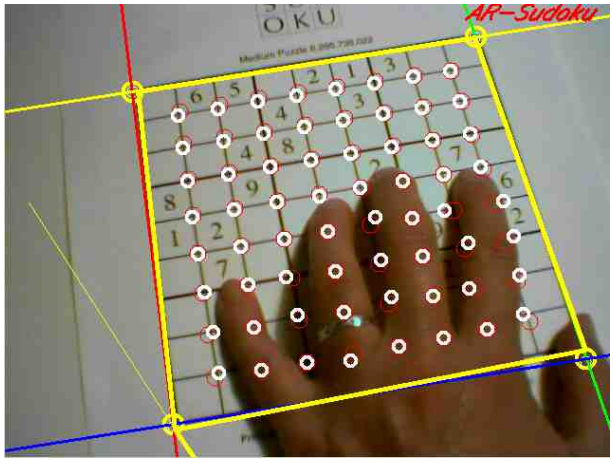

Figure 16: Grid tracking.

Grid tracking: After the grid has been detected and analyzed, it is tracked in subsequent images. The four outer edges of the grid are projected into the image, using the pose that was determined in the previous image. New positions for these edges are determined. To this end, for each edge, a Hough space is created for pixels in a strip around the edge (50 pixels wide). The line corresponding to the maximum in the Hough space is selected as the new candidate for the moved edge. After further local readjustments, the new camera pose is computed from the corners of the grid, i.e., the intersections of the edges. As in the grid detection phase, a more precise camera pose is obtained by projecting the grid model into the image using the new camera pose, repositioning these points with sub-pixel precision, and then re-estimating the extrinsic camera parameters. Figure 16 shows tracking results while the image was partially occluded by a hand.

The current procedure is rather brittle – especially when large motions occur. In such cases, model edges are fitted to the wrong image edges, due to the repetitive grid pattern. Thus far, this has not been the major focus of this research. In the future, we will extend the system to include constraints defined by knowledge about the repetitive grid structure, as well as about texture patterns from the digits that are part of the puzzle. We will also compare the results to state-of-the-art algorithms for marker-less optical tracking such as [9, 13, 20, 11], checking how well they perform on repetitive structures.

Detection of tiles: During the Sudoku game, the user places and moves physical tiles on and next to the board. The system detects such tiles by looking for blobs against a bright background and matching their image against the texture patterns of the digits. The best match is assumed to represent the digit shown on the tile.

4 First Results

The system is still in the process of being developed. The *Virtual Sudoku* system is fairly robust by now and can be used to quickly solve real puzzles. The authors have been able to solve several so-called *Evil Puzzles* of the WebSudoku site [4] in about 30 minutes – which is much faster than the author's' success in solving paper-based puzzles.

No user tests have been conducted yet, since it is not fair to compare the computer-aided version with a paper-based version. Different digital versions (virtual Sudoku, augmented Sudoku, multi-touch Sudoku) will be compared in the future.

5 Discussion

We consider the Sudoku system to be a very promising start toward visualizing and exploring complex relationships between objects. In this sense, the major contribution of the work presented here lies in the overall vision, as well as in the developing concepts toward presenting and interacting with repulsive and attractive forces in 3D, both in virtual and in AR-based settings.

Up to now, we have focused on building and putting together the core components of the system. None one of them is highly sophisticated thus far. However, we are excited to be able to present a complete vision of our approach by now. Many enhancements will be brought in in the future – as discussed throughout the text.

We expect to see exciting user studies once the system is robust enough to provide solid test scenarios for different technical setups.

REFERENCES

[1] M. Apperley, R. Spence, and K. Wittenburg. Selecting one from many: The development of a scalable visualization tool. In *HCC '01: Proceedings of the IEEE 2001 Symposia on Human Centric Computing Languages and Environments (HCC'01)*, page 366, Washington, DC, USA, 2001. IEEE Computer Society.

[2] D. A. Bowman, E. Kruijff, J. Joseph J. Laviola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison-Wesley/Pearson Education, 2005.

[3] C. Fluit and J. Wester. Using visualization for information management tasks. In *IV*, pages 447–, 2002.

[4] G. Greenspan and R. Lee. Web sudoku, 2005. http://www.websudoku.com.

[5] i2D ltd. Analyst's notebook, 2006. Defence R&D Canada, Contract Report DRDC Atlantic CR 2006-122, Aug. 2006.

[6] K. Ikeuchi. Recognition of 3-d objects using the extended gaussian image. pages 595–608, 1981.

[7] B. Johnson and B. Shneiderman. Tree-maps: A space filling approach to the visualization of hierarchical information structures. In *Proc. 2nd IEEE Conference on Visualization*, pages 284–291. IEEE, 1991.

[8] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top ar environment. In *Proc. IEEE International Symposium on Augmented Reality (ISAR'00)*, pages 111–119, Munich, 2000. IEEE.

[9] A. Ladikos and S. Benhimane. A real-time tracking system combining template-based and feature-based approaches. In *International Conference on Computer Vision Theory and Applications*, 2007.

[10] R. Lee and G. Greenspan. Jigsawdoku, 2007. http://www.jigsawdoku.com.

[11] J. Platonov, H. Heibel, P. Meier, and B. Grollmann. A mobile markerless ar system for maintenance and repair. In *Proc. IEEE International Symposium on Mixed and Augmented Reality (ISMAR'06)*, Santa Barbara (CA), USA, October 2006.

[12] G. Robertson, J. Mackinlay, and S. Card. Cone trees: Animated 3d visualization of hierarchical information. In *Proc. ACM Conference on Human Factors in Computing Systems (CHI'91)*, pages 189–194. ACM Press, 1991.

[13] M. Salzmann, V. Lepetit, and P. Fua. Deformable surface tracking ambiguities. pages 1–8, 2007.

[14] J. Shi and C. Tomais. Good features to track. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, 1994.

[15] A. J. Smith, Z. Malik, J. Nelder, and R. Spence. A visual interface for model-fitting. *Quality and Reliability Engineering International*, 17(2):85–91, 2001.

[16] R. Spence. *Information Visualization: Designe for Interaction 2nd Ed.* Pearson Education, 2007.

[17] A. Spoerri. Infocrystal: a visual tool for information retrieval. In *VIS '93: Proceedings of the 4th conference on Visualization '93*, pages 150–157, 1993.

[18] L. Tweedie, R. Spence, H. Dawkes, and H. Su. The influence explorer. In *CHI'95: Proceedings of Sigchi*, pages 129–130, Denver, CO, USA, 1995. ACM.

[19] L. Tweedie, R. Spence, D. Williams, and R. Bhogal. The attribute explorer. In *CHI'94: Proceedings of Sigchi*, pages 435–436, Boston, MA, USA, 1994. ACM.

[20] H. Wuest, F. Vial, and D. Stricker. Adaptive line tracking with multiple hypotheses for augmented reality. In *International Symposium on Mixed and Augmented Reality (ISMAR'05)*, pages 62–69, Vienna, Austria, 2005. IEEE.

[21] P. Zhang. An image construction method for visualizing managerial data. *Decision Support Systems (DSS)*, 23:371–387, 1998.

[22] P. Zhang and D. Zhu. Information visualization in project management and scheduling. In *Prod. 4th Conference of the International Society for Decision Support Systems (ISDSS'97)*, Switzerland, 1997. Ecole des HEC, University of Lausanne.

[23] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.