VOL. 9, NO. 10, OCTOBER 2014 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

PUZZLAR, A PROTOTYPE OF AN INTEGRATED PUZZLE GAME USING MULTIPLE MARKER AUGMENTED REALITY

Marcella Christiana and Raymond Bahana
Computer Science Program, Binus International-Binus University, Jakarta Pusat, Indonesia
E-Mail: [2]rbahana@binus.edu

ABSTRACT

Jigsaw puzzles are an old concept but many people still enjoy playing them. Living in an ever-evolving world, people and technology have become inseparable. One technology that is on the rise is Augmented Reality (AR), which combines the real world with virtual or computer-generated data. This research is based on developing a combination of jigsaw puzzles with AR (multiple marker-based) called Puzzlar, using FLARManager as an AR tool. The application also uses Adobe Flash (game design) and PHP. The uniqueness of the prototype program is in its use of 2D physical markers to represent 3D jigsaw puzzle pieces that are animated in the playing mode. In user acceptance testing (UAT), 70% of the testers were satisfied with the game play of Puzzlar. However, 30% of the testers thought that it could be improved.

Keywords: 3D puzzle, augmented reality, game, jigsaw puzzle, puzzlar.

INTRODUCTION

Technology plays a big role in people's lives and has led recent generations toward intellectual development and advancement. One of those technologies, Augmented Reality or AR, combines the virtual and real worlds. AR has provided a new way to interact with the world. AR has been developing for many years and has been integrated with many applications, such as mobile applications, sports, utilities, social networks and games. Unlike other virtual games, using AR people can interact with the virtual object by manipulating objects in the physical or real world.

There are many games that are integrated with AR, particularly puzzle games. Jigsaws are a classic puzzle game. Integration with AR will provide a new experience in completing jigsaws. According to Carol Finch [1], jigsaw puzzles have many benefits; puzzles can help the development of children's problem solving, hand-eye coordination, understanding of shapes and color, and as educational support. The aim of this research is to create jigsaws by integrating them with multiple markers using FLARManager as an AR tool with Adobe Flash (ActionScript 3) and PHP.

BACKGROUND

Augmented Reality

AR is an artificial environment created through the combination of real world and computer generated data. AR is a variation of virtual environments. AR allows the user to see the real world, with virtual objects superimposed upon it. Therefore, AR supplements reality, rather than completely replacing it [2]. AR integrates the environment from the real world with any data assigned from the computer.

According to Milgram and Takemura [3], AR in the context of the reality-virtuality continuum is included in 'mixed reality'. In Figure 1, 'real environment' refers to any environment in the real world and virtual environment refers to the environment in a virtual or computer generated world. In between the real and virtual is called mixed reality (MR), such as AR and 'augmented virtuality' (AV). The difference between AR and AV is that AR is mostly in the real environment whilst AV is mostly into the virtual environment. AR not only combines with the real world, it also runs interactively and updates in real time. AR gives users the ability to interact with the virtual environment in their own surroundings. According to Moniz [4], AR has three types: marker-based, markerless, and hardware.

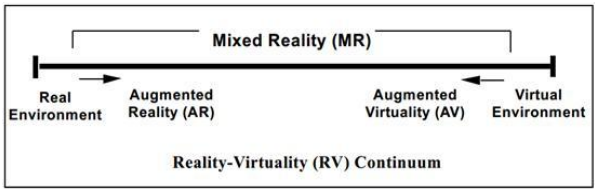

Figure-1. Simple representation of an RV continuum [3].

Tools Used in AR

There are many tools that have been developed to work on creating AR. Some of them are ported from another version. The following is by no means an exhaustive list, but serves as a guide for the commonly used tools in the AR sphere.

ARToolKit is a library developed for building AR. It uses the C and C++ languages [5]. Some features of ARToolKit are: single camera position; tracking code using simple black squares; easy camera calibration; fast enough for real time AR and free and open source.
NyARToolKit is a Java-ported library for developing AR [6]. Some features of NyARToolKit are: class based API; support for many image formats; faster fitting process than ARToolKit.

FLARToolKit is the AS3 ported version of ARToolKit and based on NyARToolKit [7]. It includes major flash 3D engines such as Papervision3D, Away3D, and Alternativa3D.

FLARManager is a framework that eases the build of AR for Flash [8]. There are many tracking libraries compatible with FLARManager and it is also supported by many 3D frameworks. It provides a more robust system in detection and management of markers, and also supports multiple markers. FLARManager is used in this research.

SOLUTION DESIGN AND IMPLEMENTATION

The solution chosen to solve the stated problem is to create a game that can entertain people and give a better experience when solving jigsaws. The proposed game is called 'Puzzlar' (Puzzle with AR).

System Architecture

The Puzzlar system uses MVC (Model View Controller) to handle the components inside the system. MVC provides a clear classification by having the ability of encapsulating project data [9]. Figure 2 shows how Puzzlar handles the components and which components interact with other components.

The user provides input to the system. User inputs vary according to what function the user wants. Puzzlar uses a webcam, with markers as the input to the AR. For the webcam, the FLARManager has created a default function for video capturing.

The controller is the link between the user and the system. It is the one that handles all of the logic that happens inside the system. It receives the user requests and calls appropriate resources to carry them out [10]. Two languages are used in the development - Actionscript 3.0 (AS3) Flash and PHP. The system has a controller for page navigation, and result calculation, while the AS3 files contains the logic for creating the AR page.

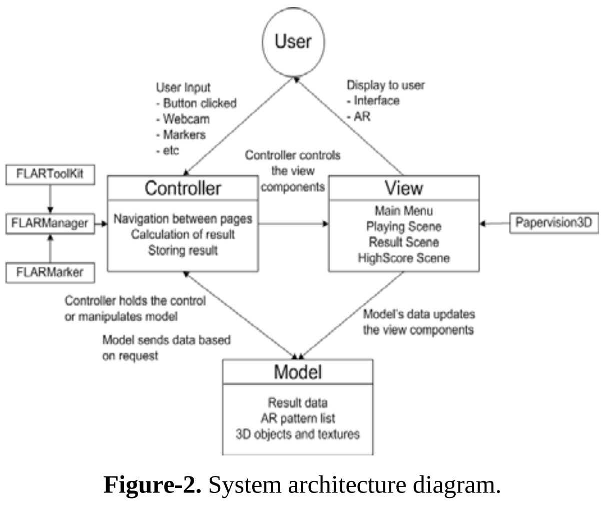

Figure-2. System architecture diagram.

The model component holds the data that is required to make the system run. The model provides data as it is requested by the user and it is presented to the user by the view component [10]. Data used by the PHP files such as scores are stored into a text file. For data used by the AS3 in order to create the AR application, such as the AR pattern list to match to the presented marker, the 3D object models are rendered out as the output of AR. Model components will send the required data if any request from the system's controller is accepted. The model also updates the data shown in the view components.

View is the visual representation of the model. The view component handles the user interface shown to the user as a bridge to interact with the system. In Puzzlar, the view component is based on Flash. FLARManager supports many 3D engines and Papervision3D was used in the Puzzlar. Papervision 3D is a 3D engine that is compatible with Flash. The view handles the output or result of the system process which is the 3D object rendered on top of the marker which can be seen by the user in the computer's/laptop's monitor.

Result Generating / Scoring Activity

Figure 3 depicts the process of generating a result in the game. The input in this process is when the user clicks the button that functions for to submitting the user's AR result. When input is received, the system would access the information ofrelating to the marker(s) and calculates them. There are three aspects considered in deciding the result which are: the distance between the markers ion the X-axis coordinate,; the distance between the markers ion the Y-axis coordinate and the gap between the markers' rotation of Z axis.

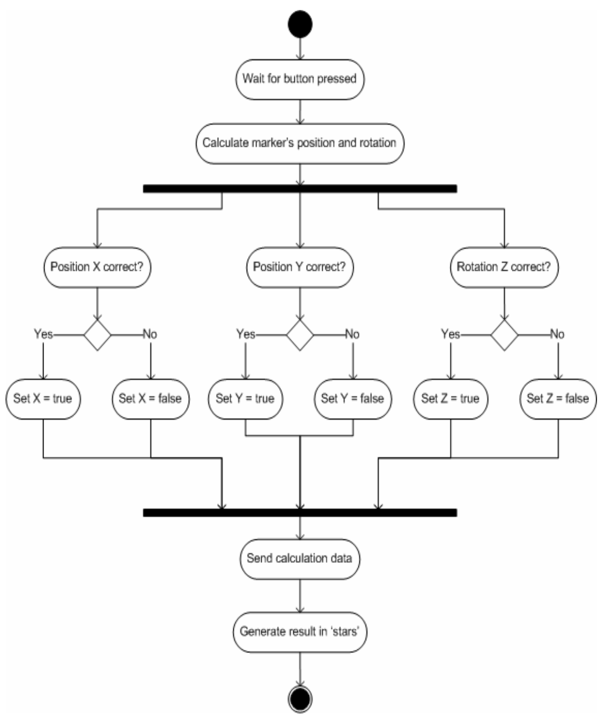

Figure-3. Result generating / scoring activity.

The calculation of the distances (1) and gaps (2) are usinge the following:

Markers distance = abs (Axis marker1-Axis marker2) (1)

VOL. 9, NO. 10, OCTOBER 2014 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences
©2006-2014 Asian Research Publishing Network (ARPN). All rights reserved.
www.arpnjournals.com

Axis markerN = round (markerN.axis) (2)

Markers distance represents the distance of markers whether it is in the X-axis, Y-axis or Z-axis. Later, it will be compared with the range of relevant axies. If it falls under within the acceptable range, it will be considered as true,. hHowever, if it falls outside the range, it will be set as false. The markers distance uses absolute values to omit anyfalse sign of the results (definitely positive).

Axis markerN, represents the value, whether it is X-axis, or Y-axis or Z-axis of marker N. N represents the number of markers involvesd in the calculation. If there are 2 markers, therese will be Axis marker1 and Axis marker2. This value is obtained from rounding the marker axis. This rounding is useful for minimizing the calculation complexity done in determinesing the Markers distance.

markerN.axis, represents the value of X-axis or Y-axis or Z-axis of marker N which is obtained from the FLARMarker class.

After the calculation is done and the system knows whether the positions of markers in the X, Y and Z axes are correct or not, the system will sends the result of calculation to be processed into a score shown to the user. In Puzzlar, the score is shown as in stars, with three stars as the top score. The calculation result (3) is processed as follows:

$$Amount\ of\ Stars = ceil\left(\frac{Amount\ of\ TRUE}{Amount\ of\ axis\ checked/3}\right) \quad (3)$$

‘Amount of Stars’, represents the stars awarded to the user as the score in Puzzlar. The stars are calculated based on the amount of correctness obtained from the previous calculation. It uses ‘ceil’ (as in ceiling) to round the value up to maximize scoring potential. It will be explained in the later example. The amount of ‘TRUE’ represents the amount of correctness. From previous calculations, the correctness of the X, Y and Z positions of the markers are deemed correct or not. ‘Amount of axis checked’ represents the number of axes that need to be checked to make sure the markers are correctly placed. The number of markers can vary between two and four in this prototype.

Determining Range

In determining the result, the Puzzlar system would compare the obtained axis with the acceptable range for the relevant axis. These ranges differ based on the height of the camera from the surface, the angle of the camera, and the distance between the camera and the marker. In order to determine a very specific range, it would require more experimenting and research to be done. In order for Puzzlar to generate results, the authors experimented in order to at least have a close range.

Determining the Z-axis is less complicated compared with the X-axis and Y-axis. The logic for determining the range of Z-axis is by considering the orientation of the markers. The rotation in Z-axis between markers would have to be the same since the 3D object model is made as one whole model rather than using the same model but different markers. To clarify how to determine the Z-axis, below is an example. For example, there are 2 markers that have the default or early state of rotation Z=0. When the Z is in the default state, the marker reaches the correct condition.

Figure-4(a) shows the default state of the markers. In Figure-4(a), marker 1 is assigned with the left side of a model (the star) and marker 2 is assigned with the right side of a model. At this point, the rotation of Z-axis will be 0 since there is no rotation made in the marker. Also at this point, the correctness status of Z-axis is TRUE or CORRECT. Figure 4(b) shows when the markers have been rotated in the amount of 180 degrees. However, this state would result on the Z-axis status to be TRUE again. It is because with the consideration that the user places the marker 2 first, then marker 1 which would result in Figure 4(d). For the case in Figure 4(c), the Z-axis will not be correct since the model will not be placed correctly. From those cases, the logic of Z-axis range determination is to subtract the rotation of the Z-axis between each marker. The correct rotation of Z-axis is 0, however, it can only be achieved if the rotation is exactly the same.

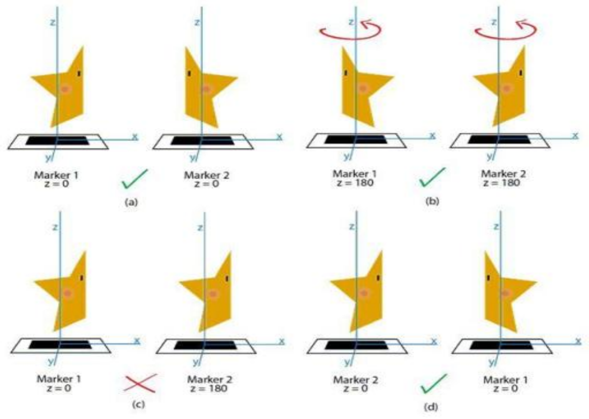

Figure-4. Example of determining correctness range in the Z-axis.

For determining the range of Y-axis correctness, it starts becoming a little bit complicated. FLARMarker has the information of the Y-axis which could help in determining the range. However, with the constraints of height, angle, and distance between camera and marker, more calculations are needed. The Y-axis will have 3 cases as listed in Figure 5. By following the cases, the Y-axis will be from each corner of the monitor as shown in Figure 6.

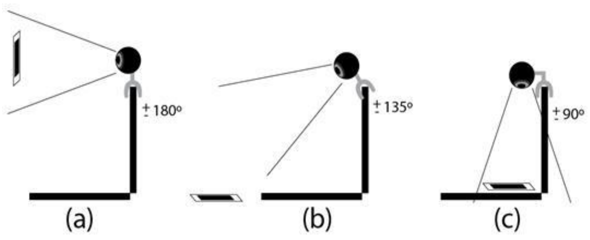

Figure-5. Example of determining correctness range in the Y-axis.

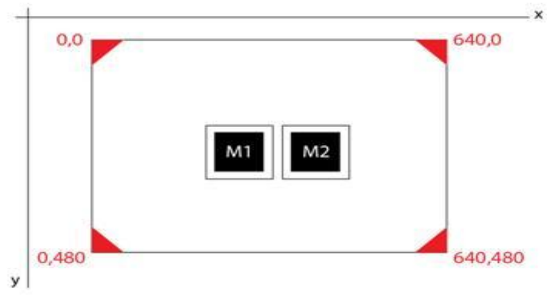

Figure-6. Monitor flash screen experiments.

Table-1 shows the results from the experiment. The range of the Y-axis could be from 2 to 15 (in pixels). Although it is not perfectly correct, it could help to decide the result of the puzzle. Different positions of camera, different angles of the camera, and different position of markers will affect the content of the table. So this experiment is only to show from where the authors could determine the range of the relevant axis.

Table-1. Table of Marker's distance in Y-axis to determine the range.

	◤	◥	◣	◢	▣
90°	5	7	7	7	9
135°	9	7	9	5	5
180°	11	14	2	8	15

The X-axis is also determined using the same method. Table 2 shows the results from the experiment. Based on the result, the range could be varied between 103 and 237. From the results for the X-axis, a camera with 1800 angle has a bigger range. It is because the system calculates that closer objects are larger and do not match the acceptable ranges.

Table-2. Table of Marker's distance in X-axis to determine the range.

	◤	◥	◣	◢	▣
90°	140	130	103	118	122
135°	127	126	177	179	138
180°	236	231	237	232	230

Component Diagram

The component diagram (Figure 7) shows the relationship between components in the system. The first component the user opens will be the Puzzlar.html which uses the printer library built-in Flash to print the marker. Choosing puzzle for Puzzlar is also done in Puzzlar.html. Data of the chosen puzzle is sent to the PHP file called controller.php. The controller.php directs the user to go to another page. This controller.php will call the HTML files of the AR puzzle according to the chosen puzzle. The HTML files are generated from the correspondent Actionscript file which utilizes FLARManager to produce the AR. FLARManager also uses the FLARToolKit library to track the markers and uses Papervision3D to render the 3D objects and set the environment. The Actionscript file also utilizes the packages from the Flash itself. When a user submits the puzzle result, the calculation data is sent to the puzzle.php which does a computation on the correctness and shows them as stars to the user by using Result.swf. The user is able to input his/her name and controller.php will store the score inside a text file along with his/her name.

Testing

Before playing, the user will be asked what puzzle they want to play (Figure 8). This prototype only provides three choices of puzzle - star shape, candy cane shape, and car shape. Each of the puzzles requires certain markers to play. The required markers are stated below the puzzle preview. Since there are specific required markers to play the game, the application allows the user to easily print the markers without the need to open a specific program (Figure 9).

Figure-10 shows the playing scene of the game. It will contain the video capture from the webcam, the timer of the game and a red button at the bottom of the video capture. The red button functions as a submit button for the puzzle. After submitting the result, the user sees the result scene where the score is represented in stars and the user could input his/her name to submit the score to the high score rankings. The highest scores are displayed on a 'high score' page.

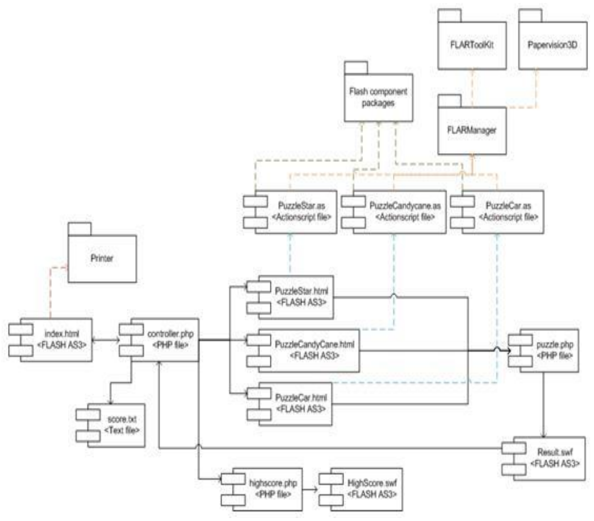
FLARToolKit
Papervision3D
Flash component packages
FLARManager
PuzzleStar.as <Actionscript file>
PuzzleCandycane.as <Actionscript file>
PuzzleCar.as <Actionscript file>
Printer
PuzzleStar.html <FLASH AS3>
index.html <FLASH AS3>
controller.php <PHP file>
PuzzleCandyCane.html <FLASH AS3>
puzzle.php <PHP file>
PuzzleCar.html <FLASH AS3>
score.txt <Text file>
Result.swf <FLASH AS3>
highscore.php <PHP file>
HighScore.swf <FLASH AS3>

Figure-7. Component diagram.

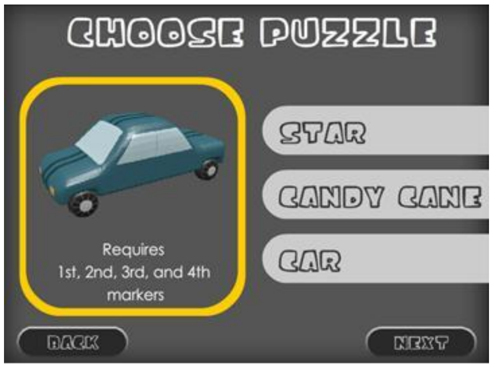

Figure-8. Choose puzzle.

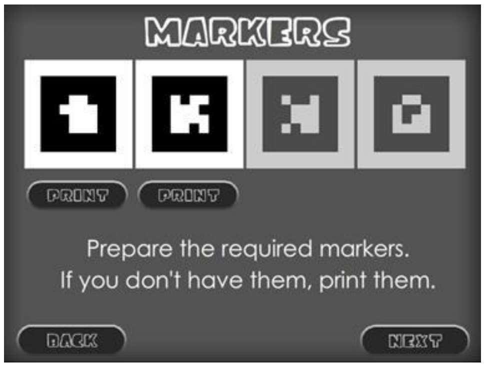
MARKERS
PRINT
PRINT
Prepare the required markers.
If you don't have them, print them.
BACK
NEXT

Figure-9. Print markers page.

User Acceptance Test

After the prototype was developed, testing was done on it. The testing was done to know how users accept the prototype as a solution. To test the prototype, a survey was done with 20 testers. Some testers feel the instructions could be improved by adding more details or descriptions of the game rules such as arranging the markers. Seventy percent of the testers were satisfied with the game play of Puzzlar. However, 30% of the testers thought that it could be improved by adding more animation when the puzzle has been completely arranged, adding more choices on the puzzle type and adding more sensitive marker detection. Some of the testers opined that it would be hard for those who do not possess a webcam or printers. Having to print the marker first was also deemed to be inconvenient but Puzzlar eased this by having the capability of printing markers from inside the application without the need to open another application.

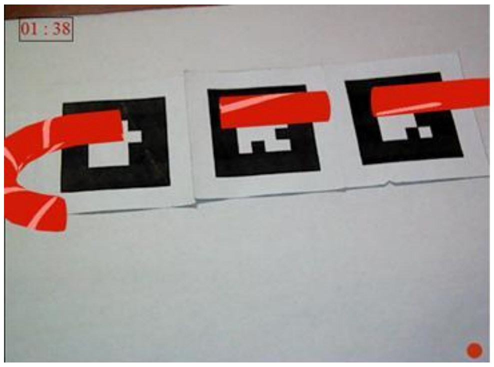

Figure-10. Playing scene of the game.

CONCLUSIONS

Based on the UAT, many of the testers found Puzzlar's game play interesting and challenging. Compared with the 2D jigsaw puzzle, testers also found that Puzzlar was more difficult than completing a jigsaw puzzle. However, the system still lacks in that it does not provide smooth game play since some of the testers encountered difficulties in the marker detection. This problem could be due to several causes. It could due to the bad quality of the printed markers such as those with low contrast. It could also be due to the lighting in the user's game playing location. Lighting plays a major role in determining the contrast of the image captured by the camera. It could also due to the threshold values. Although Puzzlar uses FLARManager that provides the capability of adaptive thresholds, it could not adapt to too bright or too dark lighting.

REFERENCES

[1] C. Finch. The Benefits of Jigsaw Puzzles: Jigsaws Can Educate, Stretch the Mind & Make a Great Gift. [Online]. Available: http://suite101.com/article/the-benefits-of-jigsaw-puzzles-a147474.

[2] R. Azuma. 1997. A Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments. 6(4): 355-385.

[3] P. Migram, H. Takemura, A. Utsumi and F. Kishino. Augmented Reality: A class of displays on the reality-virtuality continuum. Telemanipulator and Telepresence Technologies. 2351: 282-292

[4] J. A. Moniz. Augmented Reality Comparison of Different Types. 11 October 2011. [Online]. Available: http://josephamoniz.com/ augmented-reality/augmented-reality-comparison-of-different-types.

[5] ARToolKit. [Online]. Available: http://www.hitl.washington.edu/ artoolkit/.

[6] Welcome to NyARToolkit.EN. [Online]. Available: http://nyatla.jp/ nyartoolkit/wp/?page_id =198.

[7] saqoosha/FLARToolKit/en. [Online]. Available: http://www. libspark.org/ wiki/saqoosha/ FLARToolKit/en.

[8] FLARManager: Augmented Reality in Flash. [Online]. Available: http://words.transmote.com/ wp/flarmanager/.

[9] T. Reenskaug. Models-Views-Controllers. [Online]. Available: http://heim.ifi.uio.no/~trygver/ 1979/mvc-2/1979-12-MVC.pdf.

[10] E. Socolofsky. Inside FLARManager: Loading Collada Models. [Online]. Available: http://words.transmote.com/wp/flarmanager/inside-flarmanager/loading-collada-models/.