Development of an Interactive Augmented Reality Book

Daniel Bentall

Supervisor: Allan McInnes

Abstract

An Augmented Reality framework was developed to allow creation of educational Augmented Reality books. The framework features support for certain types of user interactions with the content; model and texture animations; and an enhanced marker design. The framework is written in C++ and utilises the Open Scene Graph library for 3D graphics, and the OPIRA library for marker tracking. Book pages are defined in a structured configuration file specifying the markers and scenes of each page, as well as operating parameters. Page specifications can also include standard interactions, which are built in to the framework. Pages with new behaviours are defined in the same way, but require a unique name and a class defining their behaviours.

A prototype book has been developed using the framework, featuring two examples of interaction. The educational topic chosen for this book was magnetism. Content development consisted of 3D modelling, texturing and animation, customised coding for each page with interactive content, and preliminary testing. Further books, continuing onto the theme of electromagnetism, have also been planned. These books will be used in a study comparing the learning of two groups of students, one using the augmented books and one using the same books without augmentation.

1. INTRODUCTION

Some topics in education contain inherently 3-dimensional and dynamic structures and processes, which are poorly dealt with by traditional teaching techniques, for example, chemistry [1]. Text and static 2D pictures and schematics do not intuitively demonstrate these difficult concepts, as the student must mentally translate, interpolate and extrapolate from these highly abstracted representations to a more realistic mental model. Concepts with time-varying aspects are especially difficult to represent with static images. Recently, technology has started to become more prevalent in education [2, 3], with computers able to display interactive projections of animated 3D scenes.

Augmented Reality (AR) is an emerging technology which allows users to view the real world augmented with digital content that in some way relates to the world. The most common form of AR involves viewing printed images with a webcam, which are known to the computer and are tracked using computer vision techniques. Digital content is then added to the video stream, typically in a manner which makes it appear that the content is physically attached to the printed image. In an education setting, the augmented content would relate to the concepts being taught, with the aim of improving the learning of the student.

AR offers some advantages for education that other technologies do not, especially for spatially complex topics [4]. The virtual environment is safe, and can be much more tightly controlled and monitored by educators, as well as providing content digitally, which can be inexpensive compared to physical teaching aids. Students who struggle with standard teaching techniques have also been shown to perform at a similar level to other students when using AR [5]. The importance of interaction has also been shown: Engineering students who took a special hands-on course over the course of their degree performed significantly better at the standardised spatial visualisation Mental Cutting Test than they had at the beginning of their degree, while other engineering students showed no improvement over the course of their degree [6]. Others [7, 8, 9] also showed benefits of using virtual reality, a closely related technology which also uses virtual environments, but does not contain the real-world aspects.

In this project I have developed an AR framework to provide a method of creating educational AR books. A book is under development with this framework, and will be used in a study to determine the effectiveness of AR for teaching the topic of Electromagnetism at the mid high-school level. This is a topic with complex 3D concepts, and one which many students struggle to grasp [10]. The book contains text and diagrams similar to standard high school text books, and is structured similarly, in terms of teaching content. The students will use a handheld AR display to view augmented educational content, as shown in Fig. 1. This display incorporates a webcam and two SVGA LCD screens, allowing the user to view the scene in a more natural manner, as the screens and camera are positioned roughly co-linearly to the user's line of sight.

The purpose of the framework is as a platform to create educational AR books; however it could be used for AR applications in general. The technologies used to create this framework are described in Section 2, along with an outline of the framework and its usage. In Section 3 the design of the framework is examined and discussed, including consideration of the weaknesses of the implementation and the potential for further work.

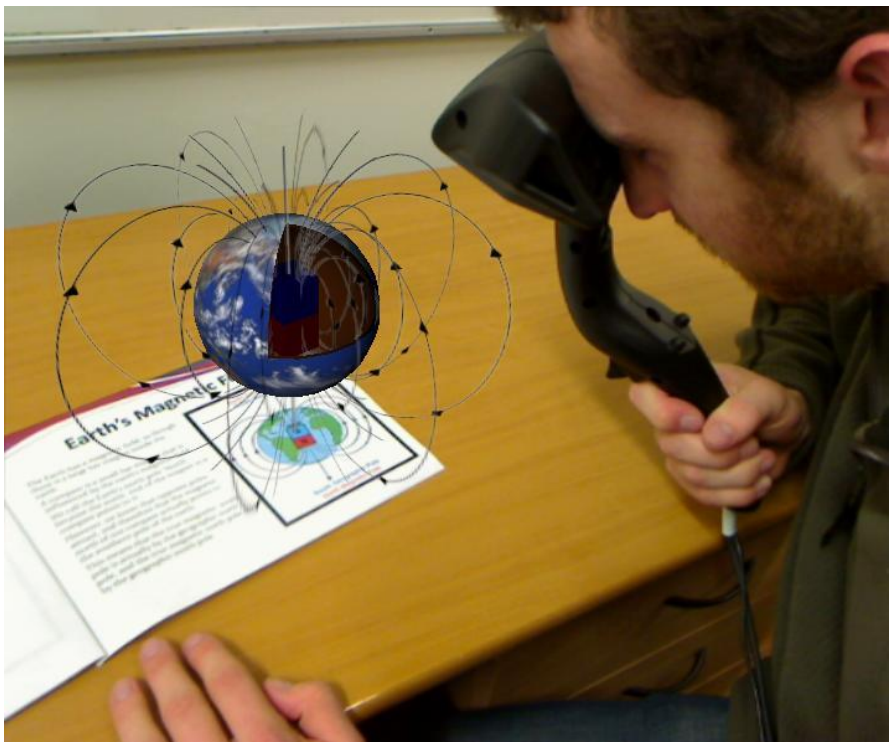

Figure 1 The handheld AR display and prototype content developed for the study.

2. THEORY OF OPERATION

At the highest level, the system operation is simple, consisting of an initialisation phase and a continuous loop which terminates when the user presses the escape key (Fig. 2). The important stages of the main loop are as follows:

1. An image is retrieved from the camera object and passed to the tracking object, which performs registration on it.
2. Registration searches the image for markers and returns the transforms of any that are found.
3. The transforms are used by each page to determine what the scenes should look like. This is where interaction behaviour is specified.
4. The frame is then processed and rendered by the OpenSceneGraph viewer object.
5. The main thread then waits for a set time, which allows processing time for the output image to display correctly.
6. If a key is pressed during the wait time, this time is interrupted and the key is returned. If that key is the escape key, the loop is exited and the program terminated.

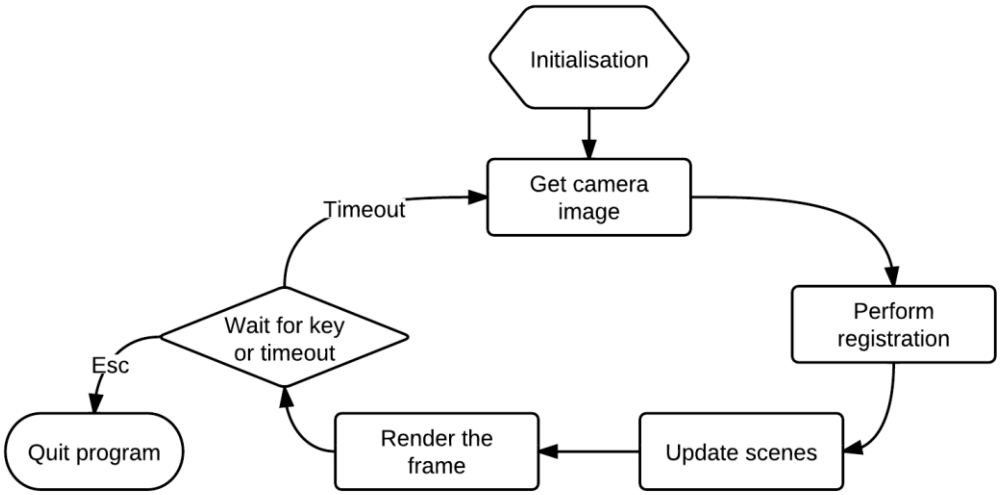

Figure 2 Flowchart outlining the system.

The class structure of the program is based on the physical structure of the book (see Fig. 3). There is one book object, corresponding to the physical book, which performs most of the program initialisation code and manages the main loop. The book contains the registration object which tracks the markers, and a list of page objects. Each page corresponds to one of the book's augmented pages, and contains a list of scene objects, each of which corresponds to a 3D scene and a marker. The scene objects contain information on any model or texture animations they contain, as well as providing methods to manipulate the state of the scene.

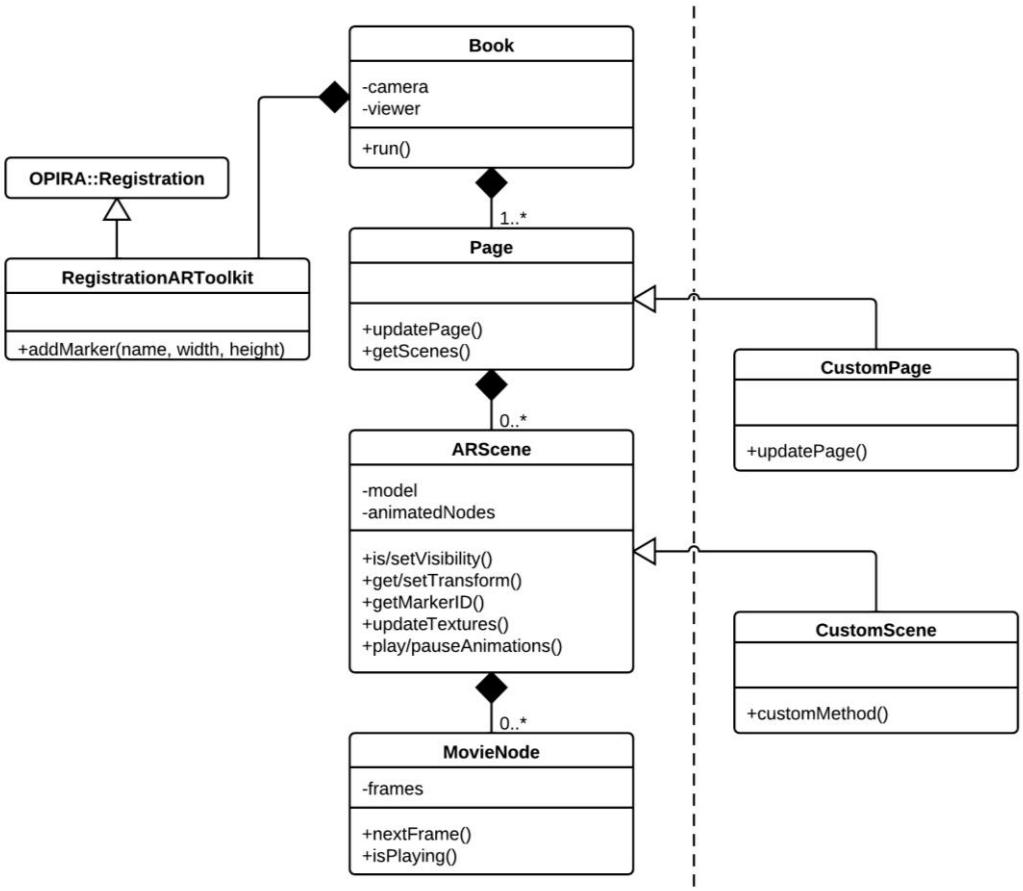

Figure 3 UML Class Diagram of framework.

The primary process to define custom behaviour is to define extensions of the Page class. The updatePage method is overridden, and the behaviour defined through calls to the scene objects. Most of the data of a scene can be accessed in this way, including its visibility, its transform from the origin and animation controls, both texture and model. If these methods do not provide the flexibility required, the ARScene class can be extended for more complex behaviours. CustomScenes would then be instantiated in the constructor of the CustomPage in place of the standard ARScenes.

2.1 Configuration

The system is configured by loading a configuration file at run-time, using the libconfig library. This library supports structured configuration, so that a hierarchy of settings may be defined. External configuration files can also be "included" from the main file, resulting in a more compact and readable file format. The main purpose of configuration file is to define the pages that the book will contain. Each page has an optional type referring to its behaviour, and a list of scenes, and each scene has a single marker and 3D model defined. The preferred method is to write separate configuration files defining the markers and models, so that they can be reused on different pages. Each marker configuration defines the pattern file to use and the dimensions of the marker. Each model configuration defines the model file to use and optionally a translation, rotation and scaling to apply. To allow texture animations the texture files and the nodes of the model which the textures are applied to must be specified in this file, and to allow model animations the animated nodes must be specified.

An example of the format expected by this system is shown in Appendix 1A, where two pages are loaded. The first page has two scenes, and is of type "SelectablePage", which is one of the built-in interaction types. With this type of page, only one of the scenes present on the page will display at once. When the user touches one of the markers and pulls their finger away, that marker's scene becomes the active one. This provides a basic selection interface. The second page has one scene and no type field, which means that it does not have any interactions defined for it. Extra parameters can be added to custom typed pages, so when users define custom interactions they can add other fields in the page definition, which they can extract and use within the custom page's constructor.

Marker and model files are defined in separate files which are included inline with the @include statement. An example of a model file with texture animations is shown in Appendix 1B. The 3D model file is specified, along with any model animation nodes and texture animation nodes. For the latter, the texture files used for the texture animations must also be specified. An initial scale, translation and rotation relative to the scene origin can also be specified.

2.2 Marker Tracking

Markers are specially formatted images on each book page which the application tracks using computer vision to obtain the position of the page relative to the camera. They consist of a border, which is used to identify the marker and calculate position information, and an image, which is used to differentiate between different markers. Typically the image chosen is only a symbol representing the content, and is not used to display information. In this project the roles of markers have been expanded to double as diagrams. Through minor modification of the libraries used, rectangular markers are possible, and the use of colour and thin borders has been experimented with and guidelines for the use of these elements have been established.

The framework employs the OPIRA framework [11] with the ARToolkit Registration Algorithm [12] for finding the pose of markers. For each frame, ARToolkit thresholds the image to a binary

image and searches the camera image for black-edged rectangles with uninterrupted borders. The pixels within the central area of the rectangle are sampled, where the central area is a rectangle starting from one quarter and going to three quarters of the marker's width and height. The sampled image is typically a 16 by 16 pixels 8-bit greyscale square (See Fig. 4), but for this project, to increase marker image differentiation, a 32 by 32 sampled square was used. This square is compared to the program's database of markers, and if a match is found, the pose of the marker is returned. The pose is calculated based on fitting a plane through the four corner points of the marker border, with the origin corner determined by the orientation of the marker image. The pose is represented by a 4 by 4 matrix containing the rotation and translation of the marker's origin, relative to the camera's origin.

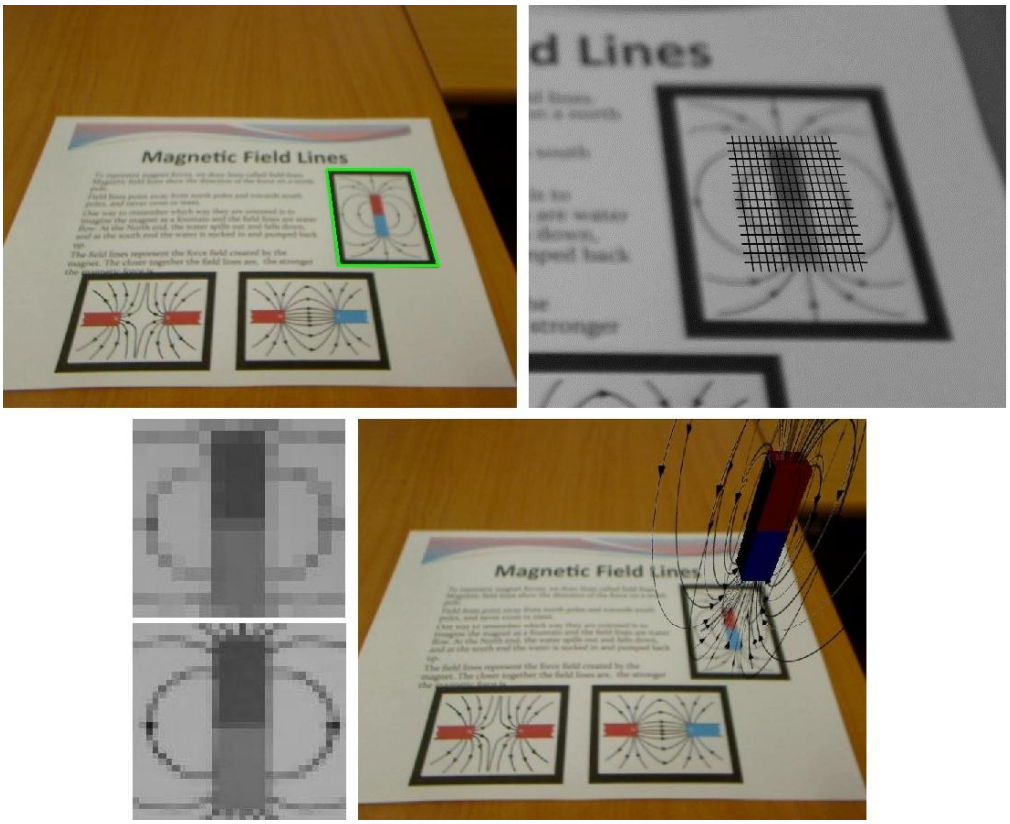

Figure 3 Marker tracking of a page from the prototype book. From top left: rectangles are found; the central area is broken into 16 by 16 squares and reduced to 8 bit greyscale; a comparison of 16 by 16 and 32 by 32 sampling; the content drawn with correct perspective.

2.3 3D Graphics

OpenSceneGraph (OSG) is an open-source 3D graphics library, which is used in this framework to render and manipulate the 3D content. This library provides high-level functionality such as scene graph manipulations; saving developers from having to directly call low-level OpenGL functions, on which the library is based. This was crucial in the framework development, and also gives users developing pages with custom behaviour the same functionality. OSG's scene graph implementation is a tree-based data structure containing the spatial and visual information of a scene, with each node acquiring the combined properties of its ancestors.

An example of the motivation for using a scene graph is specifying the positions of naturally hierarchical objects. The positions of the wheels of a car are defined relative to the position of the car, and the position of the car is defined relative to the world origin. To find the absolute position

of a wheel the transformations down the branches of the scene graph leading to the wheel leaf are accumulated. Special nodes defining visibility and rendering modes can also be defined, with their affects applying to all of their descendents. Hierarchies allow intuitive manipulation of complex 3D assemblies, simplifying the task of writing scene interaction code.

The process for creating a scene to be used with the book consists of modelling the scene in a modelling program and saving or exporting the scene to one of the formats compatible with OSG. A key-frame animation can be used to add animations to the scene from within the modelling software. By default, this animation will continuously loop; to control its playback, the animated node must be specified in the configuration, which provides access to that node for the behaviour code to control it. To create texture animations, the node and all of the textures which will be cycled through must be specified.

3. DISCUSSION

The system uses the ARToolkit Registration Algorithm for marker tracking and recognition; however other algorithms are supported by OPIRA, and can be substituted for ARToolkit. One of the alternatives is the Open Computer Vision Speeded Up Robust Feature (OCVSURF) algorithm, which performs Natural Feature Tracking (NFT). Instead of searching for black rectangles it finds unique visual features of the tracked images to identify them and calculate their poses. Therefore no borders are required, allowing images to be more seamlessly integrated into the design of the tracked object.

Initially, this algorithm was going to be used in the application; however it was found that it performed poorly with the diagrams of the book, since they are diagrams with few visual features. In addition, the algorithm is more computationally expensive, and resulted in low frame-rates. Instead, ARToolkit was adapted to be flexible enough to allow each marker image to double as a diagram, where the only extra requirement is to have 5 mm thick black borders around the diagram.

Three basic interactions were created and added to the framework. The SelectablePage interaction allows multiple separate scenes on a page to be selected and viewed individually. The PlayablePage allows an animated scene to be played and paused by touching a marker designated as the play button. The SwitchablePage allows a switch node in a scene to be toggled by the user, which could be used to change between two animations or toggle the visibility of part of the scene, for example. The latter two of these interactions allow the user to create custom behaviours through creating animations or switch nodes in the modelling program they are using, so that users with modelling experience but no programming experience can create custom behaviours. Creating complex animations through modelling is also much more straightforward than programmatically creating the animations, which requires difficult 3D mathematics. Another interaction where a digital nail appears to be magnetised by a cardboard mock-up magnet is not considered part of the framework as it is a very specific interaction.

3.1 Prototype

A prototype book has been developed which demonstrates the key features of the framework. This book aims to teach high-school students the basics of magnetism. The provided SelectablePage interaction is used, along with an interaction where the user picks up a real-world mock-up magnet and can use it to magnetise and move a virtual nail. Animated models are used, where magnets attract and repel according to their polarities. Animated textures are used to display arrows moving along magnetic field lines, reinforcing the convention of field lines running from the north pole to the south pole. The main focus of the first book is to introduce students to magnets, starting with

attraction and repulsion, covering magnetic field lines and induced magnetism and finishing with application to the earth's magnetic field and the use of compasses.

The designs for two more books have been laid out, which extend the content of the first book. In the second book, electromagnetism is introduced. The magnetic field caused by a current in a straight wire is explained, including the Right Hand Grip convention. The affect on the magnetic field of forming a loop from the wire is shown, followed by the explanation of a solenoid. In the third book, the force on a current-carrying wire resulting from intersecting a magnetic field is introduced, including the Right Hand Slap convention. The idea of using electrical energy to produce mechanical motion is presented, and the purpose of split ring commutators is explained. Finally, the concepts are combined to explain how a simple DC motor works.

3.2 Limitations

The tracking process imposes limitations on the marker. To achieve reasonable tracking the border must be robustly detected. The border is not required to have an inner edge, but it must have a channel of plain black fill at least 4 mm thick around the edge. When the border thickness was tested at less than 4 mm the tracking became erratic. Printing with colour with a laser printer was also found to have a negative impact on tracking, as laser printer ink is relatively reflective, and this makes the borders difficult to identify. Printing with an inkjet printer was found to lower the reflectivity, providing improved tracking. Another limitation is the ratio of the width to the height of the marker, which was found to affect the tracking at low camera angles when it exceeded 2:1.

The marker image also faces certain restrictions. If the image, once stretched to a square shape, down-sampled and converted to 8-bit greyscale, has rotational symmetry, then the algorithm will not be able to determine which corner of the border is the origin of the marker. This is because the image could be oriented in one of two or four directions (depending on the order of rotational symmetry). Thus the sampled image must not have rotational symmetry if the orientation of the marker is important. In addition, the sampled image must not be too similar to other marker images used by the application, or the application will not be able to differentiate between them.

If a user partially occludes the border of a marker, the tracking typically fails, as the entire border must be visible for proper tracking. This limits the interactions to those where the user is not led to occlude the markers, or to those where the occlusion is a part of the interaction, as with the SelectablePage interaction. ARToolkit supports multi-marker tracking, which is when the translations and rotations between multiple markers is fixed, and known to the program. This allows the program to track the position of the set of markers, so that as long as one marker is fully visible, the occlusion will not affect the tracking. This feature has not been integrated into the framework, as the modifications to ARToolkit to provide rectangular markers would be more difficult to extend to multi-markers.

A computer vision based hand-tracking system was previously developed [13], which produced accurate co-ordinates of finger tip locations in the marker plane. Originally this system was going to be integrated into the framework to provide a more natural interaction style, where the scene could be made to directly respond to the user's gestures. However the hand tracking uses the saturation of the image to extract the user's hand, meaning that the pages would have to be greyscale, or the algorithm modified. Since colours are an important element in teaching using diagrams, greyscale images were not an option. The simplest alternative to a thresholded saturation image is calculating the difference per pixel between the observed image and the image of the page from memory, but this was found to give inferior tracking. In addition, if hand interactions were added to the framework, multi-markers would need to be used so that when the user's hands obscured one of the

markers during interaction, the tracking was not disrupted. As a result of these complicating factors, hand tracking was not implemented as was originally planned.

3.3 Further Work

The next step is to perform a study using the prototype book and the two following books. An informal expert evaluation and a pilot study will take place to provide two new perspectives on the content to uncover any issues with the system's use and content. Once these issues have been addressed, the system will be used for a study where the learning of two groups of high school students will be measured and compared. One group will learn using the augmented books, and the other will learn using the books without the augmented content. Further studies with similar goals might be necessary to reinforce or broaden the conclusions from this study.

The required learning for a user wishing to design an AR book should be minimised, to encourage the use of the system. Documentation detailing the process of specifying a book in the configuration file and the process and available methods for creating new types of behaviours should be created. Also, the addition of more built-in behaviours and the improvement of the current ones would reduce and potentially eliminate the need for a user to learn how to programmatically create their own custom behaviours.

An AR authoring tool with animation and interaction capabilities would allow non-programmers to create content with minimal training. Currently, users can create a new book without special behaviours by changing the configuration files. If more built-in behaviours were provided then this might be sufficient for limited use. However this system is relatively technical, so some form of What You See Is What You Get (WYSIWYG) editor could mean that users with much lower technical skills could create their own books.

The concept visualisations could be improved. Principles for designing chemistry visualisation tools are described by Wu and Shah [1], but are not chemistry specific, and could be applied to any subject which suffers from poor visualisation. These principles could be used in the designing of content and the refinement of the framework. The principles are providing multiple representations and descriptions of the content, showing the connections between these different views, showing the transformation between 2D and 3D and integrating information into the visualisations.

4. CONCLUSIONS

An Augmented Reality framework was developed which allows users to create new AR books by specifying the content to use in a simple structured configuration file. To add interactive behaviour to a page, the user can either choose one of the built-in interactions or create their own, by extending the Page class. Model animations can be created in the user's 3D modelling program and then controlled programmatically. Texture animations are created by specifying a node to animate and the image files to animate it with. The markers used to track the pose of the pages can be used as visual elements on the pages.

A prototype book demonstrating the features of the framework has been created. This book, along with additional content, will be used to evaluate the effectiveness of AR for teaching Electromagnetism; an inherently spatially difficult topic. Augmented Reality has the potential to offer a new approach to learning with interactive animated 3D visualisations. These capabilities are especially useful for subjects with difficult 3D concepts. Beyond improved user engagement, digitally augmented education could allow a more customised and closely monitored learning environment, making education more efficient and stimulating.

5. REFERENCES

[1] H.-K. Wu and P. Shah, "Exploring visuospatial thinking in chemistry learning," Science Education, vol. 88, no. 3, pp. 465–492, 2004. [Online]. Available: http://dx.doi.org/10.1002/sce.10126

[2] M. Papastergiou, "Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation," Computers & Education, vol. 52, no. 1, pp. 1–12, 2009.

[3] F. J. Boster, G. S. Meyer, A. J. Roberto, C. Inge, and R. Strom, "Some effects of video streaming on educational achievement1," Communication Education, vol. 55, no. 1, pp. 46–62, 2006. [Online]. Available: http://www.informaworld.com/

[4] A. Dünser, K. Steinbügl, H. Kaufmann, and J. Glück, "Virtual and augmented reality as spatial ability training tools," in Proceedings of the 7th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: design centered HCI. ACM, 2006, pp. 125–132.

[5] A. Dünser, "Supporting low ability readers with interactive augmented reality," Annual Review of CyberTherapy and Telemedicine, p. 39, 2008.

[6] B. Field, "A course in spatial visualisation," Journal for Geometry and Graphics, vol. 3, no. 2, pp. 201–209, 1999.

[7] C. Oman, W. Shebilske, J. Richards, T. Tubr´e, A. Beall, and A. Natapoff, "Three dimensional spatial memory and learning in real and virtual environments," Spatial Cognition and Computation, vol. 2, no. 4, pp. 355–372, 2000.

[8] D. Passig and S. Eden, "Virtual reality as a tool for improving spatial rotation among deaf and hard-of-hearing children," CyberPsychology & Behavior, vol. 4, no. 6, pp. 681–686, 2001.

[9] M. Schnabel and T. Kvan, "Spatial understanding in immersive virtual environments," International Journal of Architectural Computing, vol. 1, no. 4, pp. 435–448, 2003.

[10] S. Chasteen and S. Pollock, "A research-based approach to assessing student learning issues in upper-division electricity & magnetism," in 2009 Physics Education Research Conference Proceedings, edited by M. Sabella, C. Henderson, and C. Singh (AIP Press, Melville, NY, 2009), pp. 7–10.

[11] A. Clark, "Opira: The optical-flow perspective invariant registration augmentation and other improvements for natural feature registration," (Ph.D. Thesis), 2009.

[12] K. H., "Artoolkit," http://www.hitl.washington.edu/artoolkit/. [Online]. Available: http://ci.nii.ac.jp/naid/10019957376/en/

[13] D. Bentall, R. Green, "Application Control Through Accurate Finger Tracking," (Course Project), 2011.

APPENDIX 1A: EXAMPLE CONFIGURATION FILE

```
markerDir = "markers/";
modelDir = "models/";

camWidth = 800;
camHeight = 600;

book = {
    pages = (
        {
            type = "SelectablePage";
            scenes = (
                {
                    @include "marker_b1p1-1.cfg"
                    @include "model_repellingMags.cfg"
                },
                {
                    @include "marker_b1p1-2.cfg"
                    @include "model_attractingMags.cfg"
                }
            );
        },
        {
            scenes = (
                {
                    @include "marker_b1p2-1.cfg"
                    @include "model_singleMagField.cfg"
                }
            );
        }
    );
};
```

APPENDIX 1B: model_singleMagField.cfg

```
model = {
    file = "singleMagField.osg";
    animNodes = ();
    movieNodes = (
        {
            name = "fieldMesh";
            texturePaths = [
                "Field Lines 1.png",
                "Field Lines 2.png",
                "Field Lines 3.png",
                "Field Lines 4.png",
                "Field Lines 5.png"
            ];
            autoPlay = true;
        }
    );
    scale = [3.0, 3.0, 3.0];
    translation = [0.0, 0.0, 20.0];
    rotation = [0.0, 0.0, 0.0];
}
```