Design an AR Application Using The Tiled Display Walls

Jidapa Kongsakoonwong[1], **Boonsit Yimwadsana**[1], **Jason H. Haga**[2]
jidapa.kog@student.mahidol.ac.th, boonsit.yim@mahidol.ac.th, jh.haga@aist.go.jp, jh.haga@aist.go.jp
Faculty of Information and Communication TechnologyMahidol University, Bangkok, Thailand [1]
Information Technology Research Institute, National Institute of Advanced Industrial Science and Technology, Tsukuba, Japan[2]

ABSTACT

Data visualisation plays an important role in understanding data through different graphical representations of the data. However, newer technologies have expanded data visualization beyond a single computer screen and paper printouts. Often these traditional technologies lack context and can reduce the efficiency of understanding the data by the user. Hence, it is necessary to develop visualization approaches that facilitate understanding and exploration of data in an easy manner by the user. One of technologies that can help us to add context to the information is augmented reality (AR). This technology has become very popular recently as a means to add context to data objects and allow users to interact spatially with the information. This paper uses the new ARKit tool to develop an AR application that creates virtual information layers that are superimposed over data objects in the physical world. The user can interact with the layers and manipulate the data to generate better understanding. Our proof-of-concept AR application was demonstrated with a tiled display wall. In the future, the concept of our application can be applied to developing a dynamic AR application for data intensive science.

BACKGROUND

ARKit is a set of software development tools for developers to build augmented reality app for iOS (1). It is an integration of digital information with the user environment in real time. With augmented reality, it uses the existing environment and overlays new information on top of it.

SceneKit combines a high-performance rendering engine with a descriptive API for import, manipulation and rendering of 3D assets. It requires only descriptions of a scene's content and any actions or animations that content should perform (2).

UI gesture is used to observe the gesture of the user. It is a gesture-recognizer object that decouples the logic for recognizing a sequence of touches and acts on that recognition (3). Since, one of the objects recognizes a common gesture, it sends an action message to each designated target object.

Recognizing Image in an AR Experience detects known 2D images in the user's environment, and use the object's positions to place AR content (4).

TECHNICAL TOOLS

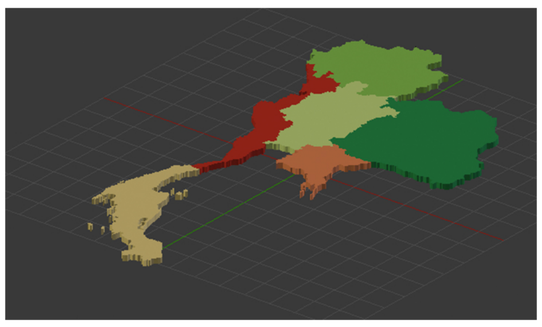
Blender SceneKit ARKit xCode

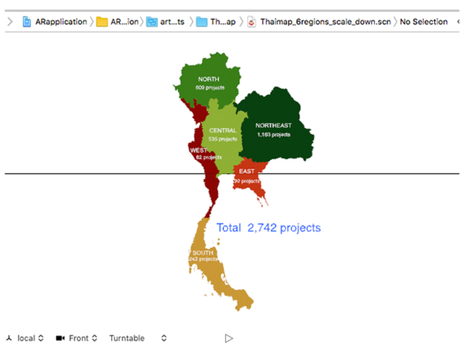
iOS11 or later
iPhone6s or later

UI Gestures

Image recognition

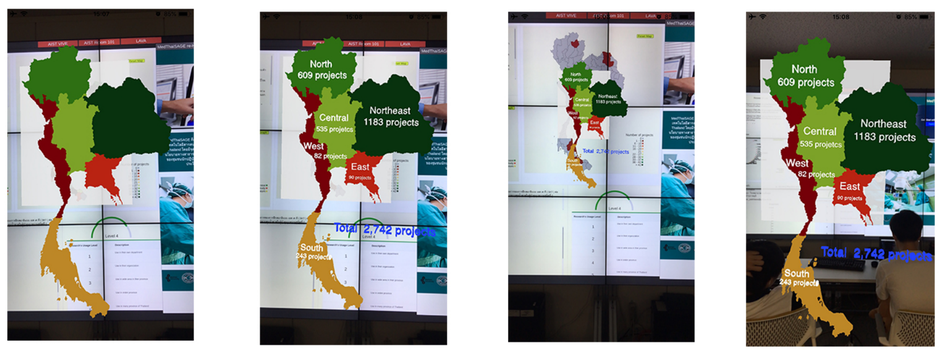
IMPLEMENTATION AND RESULTS





1.Prepare 3D object and import to xCode
Created 3D object for image recognition with Blender using Hard-surface modelling technique and import into xCode using Collada(.dae).

2. Preparing the image and adding as an image reference into xCode
The image is a Thai map from the mednacea website(5) that shows the number of research projects in each province. After creating the image, it is imported to xCode using the Assets.xcassets group.
However,the physical image size must be known before entering into xCode



AR image recognition | Adding information (Tap/Long-press gesture) | Scale down (Pinch gesture) | Scale up (Pinch gesture)

3. UI Gesture for user interaction
We used the UIGestureRecognizer class for concrete gesture recognition. There are three main gestures: (1) long-press for adding information, (2) tap to remove information, and (3) pinch gesture for zooming in our out.

CONCLUSIONS

We created an interactive AR application that interacts with a tiled display wall using image recognition and gesture recognition. It is an embedded application since it will recognize an image by adding an image reference into the application. This application is a prrof-of-concept AR prototype that can add 3D data objects in layers over the physical world.

ACKNOWLEDGEMENTS

This work was done during an internship at AIST, Tsukuba, Japan and supported by the ICT International Team Grant. The work was also supported by Faculty of Information and Communication Technology, Mahidol University.

REFERENCES

(1) ARKkit,https://developer.apple.com/arkit/
(2) SceneKit, https://developer.apple.com/documentation/scenekit
(3) UI Gestures, https://developer.apple.com/documentation/uikit/uigesturerecognizer
(4) Imgae Recognition, https://developer.apple.com/documentation/arkit/recognizing_images_in_an_ar_experience
(5) mednacea, http://mednacea.ict.mahidol.ac.th/