Project 1 - Around the World

<u>Application Due Tuesday November 21 at 11pm</u>

<u>Documentation Due Thursday November 28 at 11pm</u>

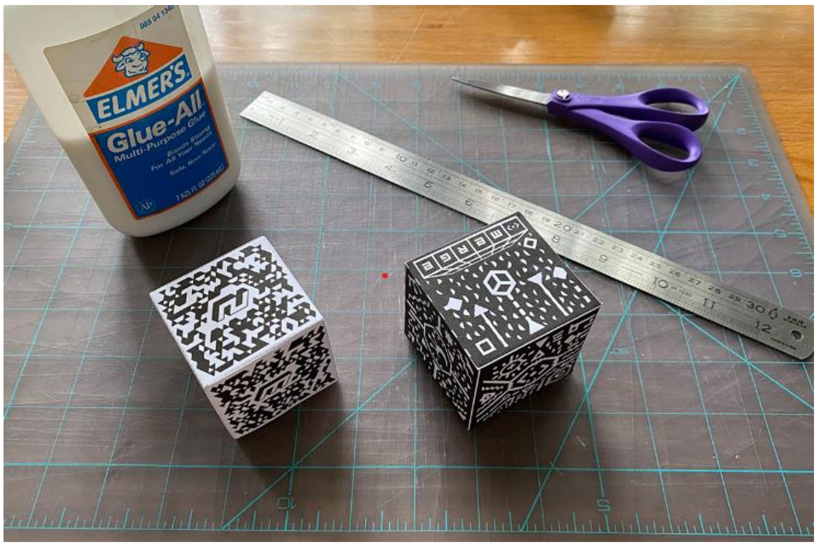
Project 1 is an individual project focused on Augmented Reality. In the future when everyone is wearing their AR enabled eyewear much of the time, what kind of future will it be? We are going to explore this a bit by looking into some AR knickknacks. A common thing to see in shops when you travel are small snow globes or other little trinkets that you can take home and place on a desk or a shelf to remind you of the place you visited. What if these were virtual instead of real so you could have one physical placeholder that could take on the form of many different knickknacks for different attractions. Since these are virtual, they could also be somewhat 'smarter' than their older siblings. In our case we are going to have these knickknacks be able to grab the current weather and time at the location of the attraction that they represent.

Implementing the Project

You will be implementing the project in Unity 2021.3.6f1 and Vuforia version 10.9 (10.9.3 at the moment).

I have created a sample starter project with a knickknack for Roswell, New Mexico. You can grab a zip file here (You have it). Inside the Assets/Scenes folder you will find RoswellScene. Drag that into your hierarchy and remove the default Untitled scene.

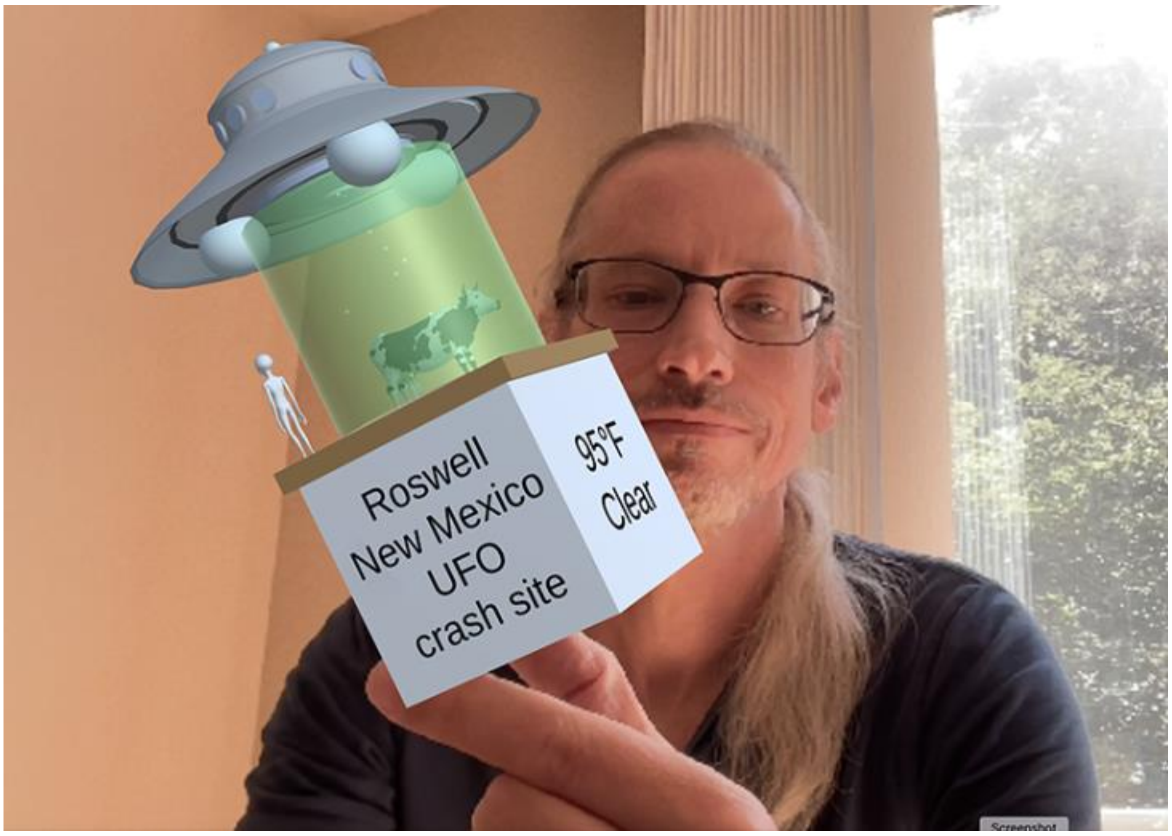
You will need to create the physical Merge cube so the application has something to map my AR knickknack onto. You will also need to create the physical Class cube for the second half of the assignment. Here is a zip file (You have it) containing images that you can print out and then cut out, fold, and re-assemble. I will also have some printed copies available in class.



You will need to create a Vuforia developer account and get a free Vuforia App License key that you can paste into the Vuforia Configuration window in Unity in order for Vuforia to track your cubes. On the Vuforia developer site you can click on Develop / License Manager and then Get Development Key to create a new license key. You can then go to the project in Unity and bring up Window / Vuforia Configuration and look near the top in the Global section and add in your App License Key.

You will also need to create an account at OpenWeather to get your API key to download weather information for the knickknack to display. You will need to add this key to the URL variable at the top of the WeatherAPIScript.cs in the Assets folder. The Week 2 notes discuss this.

At this point you should be able to run the project in Unity, hold the Merge Cube up in front of the camera and see and hear the knickknack appear.



This application should give you a good starting point for creating your own AR knickknack.

The attraction(s) you choose could be a traditional one like Eiffel Tower, or the Sydney Opera House, or Old Faithful in Yellowstone, but could also be a location that has personal significance to you like the Music Box Theatre or Manny's Deli, or Margie's Candies, or Soldier Field. You need to pick an attraction location other than Roswell. The location should be a real place that you can visit, and should be small enough that it doesn't have other attractions within it (e.g. it cant be a city like Chicago, or a national park like Yellowstone). Locations in other countries are fine. Your knickknack should be an appropriate, and preferably obvious, representation of that place. The models you use should be reasonably detailed but do not need to be photo-realistic.

You will be able to grab the local time for different locations using http://worldtimeapi.org

60% of the points on the project is for creating a basic knickknack using the merge cube marker with

- 3 unique models from the web (remember to cite the creators)
- 2 unique models that you create on your own (more on this below)
- at least one relevant, and not annoying, ambient sound
- at least one light
- the frame rate should remain high with smooth motion and no stuttering
- one side of the cube should give the name of the attraction
- one side of the cube should give the current weather for the attraction along with the temperature in degrees F. The temperature should update at regular intervals (e.g. once every 10 minutes)
- one side of the cube should give the local time for the attraction in 12 hour am/pm style
- the content of the 4th side is up to you, but it should be relevant to the attraction
- turning the cube upside down and then back right side up should change the lighting to a different but still useful style. Turning the cube upside down again should change the scene back to the original lighting
- allowing the user to view this through a webcam on their computer (or on a smart phone)
- all of the text should be readable and fit within its side of the cube and not overflow

40% of the points on the project are for adding a second knickknack using the class cube marker for a different attraction where both knickknacks are visible and function simultaneously. The knickknack for the second attraction should have

- 3 unique models from the web (remember to cite the creators)
- 2 unique models that you create on your own (more on this below)
- at least one relevant and not annoying ambient sound
- at least one light
- the frame rate should remain high with smooth motion and no stuttering with both visible at the same time
- one side of the cube should give the name of the attraction
- one side of the cube should give the weather for the attraction along with the temperature in degrees C. The temperature should update at regular intervals (e.g. once every 10 minutes)
- one side of the cube should give the local time for the attraction in 24 hour style
- the content of the 4th side is up to you, but it should be relevant to the attraction

- turning the cube upside down and then back right side up should change the lighting to a different but still useful style. Turning the cube upside down again should change the scene back to the original lighting
- deploying the app to an iOS or Android phone (a user should be able to use their phone to view both of the fully functional knickknacks at the same time)
- all of the text should be readable and fit within its side of the cube and not overflow

Graduate students in the class also need to be able to ...

Use the Vuforia Astronaut ImageTarget to create a dynamic laptop sticker. Laptop stickers are pretty popular these days to personalize your devices, and with AR these kinds of stickers can come to life. You should create an animated laptop sticker designed to work with the laptop screen up, i.e. the 3D models you add should make sense on a vertical surface, and the scale should make sense for a laptop. You should use 3 unique models from the web (remember to cite the creators) and 2 unique models that you create on your own. As usual the frame rate should remain high. The animated sticker does not have to have anything to do with the astronaut ImageTarget, we are just going to standardize on that. In our hypothetical future world there would be a 2D sticker that would look cool on your laptop and become a cooler 3D version of itself. The topic of the sticker is pretty open, as long as it is G rated.

Note that there is a very big difference between getting something working and getting it working well. The first is not that hard. The second takes much more time. You are expected to have things working well.

Turning in the Project

You should create a GitHub page for your project. You can integrate Unity with git so that it will track all your changes, or just regularly push files to git. The final project will need to be turned in via git so we know the timestamp on the files, but it can also be very helpful to have regular commits in case something goes wrong so you can get partial credit for your work. Initially this repository should be private to yourself, and then you can make it public for turning it in.

There are two due dates for the project.

The unity source and application is due first. This will be turned in via GitHub by making your repository public to at least everyone at UIC. Be sure to email the location of your repository to Andy and the TA before the deadline.

The second deadline is for the documentation.

You should create a public web page with multiple sections (visible to anyone for at least the duration of the course) that describes your work on the project. You can host your web page at UIC (http://people.uic.edu), GitHub, or the provider of your choice, as long as it remains publicly available to at least everyone in the class. You can use any publicly available web templates as long as you cite them, or create your own.

This page should have several sections including:

introduction and description of how to use your application and the things you can do with it written for someone who is not taking the class and does not know what the project is about, including showing your app running on a smartphone (if it runs on a smartphone) or running through a webcam in unity (if it doesn't run on a smartphone)

link to your git page that allows someone starting with a web browser to easily download the source code to your project, build it and run it. This page should have instructions on how to build your application and list the supported version numbers of all relevant software (Unity, VRTK, etc.), and how to install that software. This should be very detailed so that a person who is not in the class could follow them and get your app running in Unity.

listing of the source for any assets (models, textures, sounds, music) that you used that you didn't create yourself and how they relate to the requirements (e.g. if 15 models are required then please give us a numbered list of them including showing them in the scene). Also give a numbered list of the models you created yourself with an image showing them in the scene.

explanation of how your models and sounds represent that attraction, and if it is a non-traditional attraction why it is an important to you.

link to a 5 minute video showing off your project (see below)

at least a one page / 500 word discussion of whether you think things like this would become popular in several years when we have eye wear that slows AR objects like this to always be around us. What kinds of small objects do you think would benefit from being AR enhanced physical objects rather than being purely physical.

all of which should have plenty of screenshots with meaningful captions. Web pages like this can be very helpful later on in helping you build up a portfolio of your work when you start looking for a job, giving you something to point potential employers at, so please put some effort into it.

Regarding citing models and sounds that you used but did not create, my sample scene included these resources:

UFO model by hana
https://3dwarehouse.sketchup.com/model/273a556e078133532296d88107d065f6/UFO

UFO sound by Daniel Simion - https://soundbible.com/2213-Alien-Spaceship-UFO.html

Cow Model by Max M - https://3dwarehouse.sketchup.com/model/288b4047-1c6b-43e5-9b76-eb14ff9152fa/Cow

Grey alien model by Tainted Angel - https://3dwarehouse.sketchup.com/model/c3c00f04-75b2-4500-a3e8-d0664c2be100/Alien-Grey

Roswell Logo from the Roswell New Mexico site - https://roswell-nm.gov/FormCenter/Convention-Civic-Center-9/Submit-a-Community-Event-53

You should also create a 5 minute YouTube video showing the use of your application including narration with decent audio quality. That video should be in a very obvious place on your main project web page. Capturing video from a smartphone is probably best, but you can also capture video from a webcam. You can try to narrate while interacting but you will most likely find its useful to do some editing afterwards to tighten the video up. This video is helpful for us to know which parts of your project work, and may be useful in the future when you want to show off some of the projects you created, but you can't get the code to compile anymore or you don't have the hardware handy. Note that for a 5 minute video your video should have 4 minutes and 55 seconds to 5 minutes of useful content - more or less is bad.

Once you have your web page done, send the URL to Andy and Ashwini before the deadline. We will respond to this email as your 'receipt'.

We will be linking your web page to the course notes so please send Andy a nice representative jpg image/photo of your knickknack running, highlighting your chosen location for the web. This should be named p1.<your_last_name>.jpg and be roughly 1024 x 768 in size.

Presenting the Project

An important part of creating AR applications is getting feedback and using it to improve your design. This also allows you to see how others approached the problem, and maybe they solved some problems that you had issues with, and vice-versa, so it helps everyone get better for the next project.

We will be spending time in class for each person to show off their work in a live demo either through streaming your phone to the classroom wall (preferred) or streaming your laptop viewing your knickknack through a webcam to the classroom wall. See the Week 5 and 6 notes for details.

PS:

e.g., project (https://github.com/Contection/Shyan.Connor.UIC.CS428.Project1)

e.g., Web page(https://sites.google.com/view/connorshyan/uic-projects/cs428/project1)

e.g., Vidio (https://www.youtube.com/watch?v=G4C3VZFObk0)