Augmented reality for Unity: the ARToolkit package

JM Vezien

Jan 2016

ARToolkit provides an easy way to develop AR applications in Unity via the ARTookit package for Unity, available for download at:

http://artoolkit.org/documentation/doku.php?id=6_Unity:unity_about

The installation is covered in:

http://artoolkit.org/documentation/doku.php?id=6_Unity:unity_getting_started

Basically, you import the package via Assets--> Import Package--> custom Package

Current version is 5.3.1

Once imported, the package provides assets in the ARToolkit5-Unity hierarchy. There is also an additional ARToolkit entry in the general menu of unity. A bunch of example scenes are available for trials in ARToolkit5-Unity-->Example scenes. The simplest is SimpleScene.unity, where a red cube is attached to a "Hiro" pattern, like the simpleTest example in the original ARToolkit tutorials.

The way ARToolkit for Unity organises information is simple:

- The mandatory component for ARToolkit is called a `ARController`. Aside from managing the camera parameters (for example, selecting the video input), it also records the complete list of markers your application will use, via `ARMarker` scripts (one per marker).
- A "Scene Root" will contain everything you need to AR display. It is a standard (usually empty) object with a `AR Origin` script. the rest of the scene will be children of it.
- The standard camera remains basically the same, but is now driven by a specific `ARCamera` script. By default, the camera is a video see-through (like a webcam).
- Objects to be tracked are usually "empty" geomeries (GameObject--> Create empty) to which one attaches a special `AR Tracked Object` script. AR objects are tagged with their pattern name, like "Hiro" or "Kanji". The corresponding ARToolkit marker files should be placed in the asset hierarchy (Resources-->ardata-->markers).
- Geometries attached to the AR objects (like the red cube in the simplest example) are then placed as children of the (tracked) AR objects. they can be rigidly attached to it and of course any script can then modify their behaviour, like a standard Unity object.

Additional infos:

- If you want to use your own patterns, create them with `mk_patt` (found in the ARToolkit distribution - you have used this during the AR courses).
- Note that AR objects are small because it is assumed that the default Unity unit is the meter. So a typical 1cm objects will be scaled 0.01 .
- Unity uses "Layers" to render the objects (just like Photoshop). The camera should be in the `default` layer, where the video is displayed, while AR objects (and any other visible object) should be in the "AR foreground" layer, which is the default mask for the camera render.
- There is only one camera in the scene. So, when multiple objects are tracked, the camera is positioned relative to the first marker found in the video. Then, the other AR objects are MOVED relative to the camera, based on their relative position with respect to the camera (computed with the corresponding marker).
- ARToolkit for unity supports several kinds of markers:

1. `Square` markers: these are the standard ARtoolkit markers
2. `Multimarker` : these are the standard multiple markers of ARToolkit.
3. `Square Barcode`: special markers designed for scenes where a lot of markers are needed in a automated fashion.
4. Natural Feature Tracking (NFT) markers: a marker made of a natural (planar) image. See http://artoolkit.org/documentation/doku.php?id=3_Marker_Training:marker_nft_training

Again, the documentation on the artoolkit.org site is pretty comprehensive. Internet tutorials are also available.

Android export:

This works like any android export in Unity: select the proper Android target in File-->Build settings, then build to generate a .apk. Download the app. on your android device and enjoy !