Appendix 1 - Code

```
/*
Camera Controller.cs
This script is for controlling cameras, I use this script
to control both the First-Person camera and also a remote
camera for creating fly-through animations or to allow
viewing of the data from a different location
*/

using UnityEngine;
using System.Collections;

public class cameraController : MonoBehaviour {

    GameObject _cameraFP = null;
    GameObject _cameraWV = null;

    void Start ()
    {
        // Initiate cameras
        _cameraFP = GameObject.Find("Main Camera");
        if (_cameraFP == null)
            Debug.Log("Start(): First Person Camera not found");

        _cameraWV = GameObject.Find("GoT camera");
        if (_cameraWV == null)
            Debug.Log("Start(): GoT camera not found");
        //run the sub-routine to select the actual camera
        //(default camera is camera 1)

        SelectCamera(1);
    }

    //sub-routine to select the camera
    void SelectCamera(int cameraIndex)
    {
        if (_cameraFP != null)
            _cameraFP.camera.enabled = (cameraIndex == 0);
        if (_cameraWV != null)
            _cameraWV.camera.enabled = (cameraIndex == 1);

    }
}
```

```
/* cameraGyro.js */
#pragma strict
    // Gyroscope-controlled camera for iPhone & Android revised 2.26.12
    // This script has been adapted by Stuart Eve, for use in the Dead Men's Eyes system
    // Stereoskopix FOV2GO Copyright (c) 2011 Perry Hoberman
    // Perry Hoberman <hoberman@bway.net>
    //
    // Usage:
    // Attach this script to main camera.
    // Note: Unity Remote does not currently support gyroscope.
    // Use Landscape Left for correct orientation
    //
    // This script uses three techniques to get the correct orientation out of the avroscope attitude:
    // 1. creates a parent transform (camParent) and rotates it with eulerAngles
    // 2. for Android (Samsung Galaxy Nexus) only: remaps gyro.Attitude quaternion values from xvzw to wxyz (quatMap)
    // 3. multiplies attitude quaternion by quaternion quatMult

    // Also creates a grandparent (camGrandparent) which can be rotated to change heading
    // This node allows an arbitrary heading to be added to the gyroscope reading
    // so that the virtual camera can be facing any direction in the scene, no matter which way the phone is actually facing
    // Option for touch input - horizontal swipe controls heading

    static var gyroBool : boolean;
    private var gyro : Gyroscope;
    private var quatMult : Quaternion;
    private var quatMap : Quaternion;
    // camera grandparent node to rotate heading
    private var camGrandparent : GameObject;
    private var heading : float = 0;
    private var headingUpDwn : float = 0;

    public var joystick : Joystick;

    // mouse/touch input
    public var touchRotatesHeading : boolean = true;
    private var screenSize : Vector2;
    private var mouseStartPoint: Vector2;
    private var headingAtTouchStart : float = 0;
    private var headingUpDwnAtTouchStart : float = 0;
    @script AddComponentMenu ("stereoskopix/s3d Gyro Cam")

    //GUI variables

    public var stringToEdit = "Hello World";

    function Awake() {
        // find the current parent of the camera's transform
        var currentParent = transform.parent;
        // instantiate a new transform
        var camParent = new GameObject ("camParent");
        // match the transform to the camera position
        camParent.transform.position = transform.position;
        // make the new transform the parent of the camera transform
        transform.parent = camParent.transform;
        // instantiate a new transform
        camGrandparent = new GameObject ("camGrandParent");
        // match the transform to the camera position
        camGrandparent.transform.position = transform.position;
        // make the new transform the grandparent of the camera transform
        camParent.transform.parent = camGrandparent.transform;
        // make the original parent the great grandparent of the camera transform
        camGrandparent.transform.parent = currentParent;

```

```
64          // check whether device supports gyroscope
65          #if UNITY_3_4
66          gyroBool = Input.isGyroAvailable;
67          #endif
68          #if UNITY_3_5
69          gyroBool = SystemInfo.supportsGyroscope;
70          #endif
71
72          if (gyroBool) {
73              gyro = Input.gyro;
74              gyro.enabled = true;
75              #if UNITY_IPHONE
76                  camParent.transform.eulerAngles = Vector3(90,90,0);
77                  if (Screen.orientation == ScreenOrientation.LandscapeLeft) {
78                      quatMult = Quaternion(0,0,0.7071,0.7071);
79                  } else if (Screen.orientation == ScreenOrientation.LandscapeRight) {
80                      quatMult = Quaternion(0,0,-0.7071,0.7071);
81                  } else if (Screen.orientation == ScreenOrientation.Portrait) {
82                      quatMult = Quaternion(0,0,1,0);
83                  } else if (Screen.orientation == ScreenOrientation.PortraitUpsideDown) {
84                      quatMult = Quaternion(0,0,0,1);
85                  }
86              #endif
87              #if UNITY_ANDROID
88                  camParent.transform.eulerAngles = Vector3(-90,0,0);
89                  if (Screen.orientation == ScreenOrientation.LandscapeLeft) {
90                      quatMult = Quaternion(0,0,0.7071,-0.7071);
91                  } else if (Screen.orientation == ScreenOrientation.LandscapeRight) {
92                      quatMult = Quaternion(0,0,-0.7071,-0.7071);
93                  } else if (Screen.orientation == ScreenOrientation.Portrait) {
94                      quatMult = Quaternion(0,0,0,1);
95                  } else if (Screen.orientation == ScreenOrientation.PortraitUpsideDown) {
96                      quatMult = Quaternion(0,0,1,0);
97                  }
98              #endif
99              Screen.sleepTimeout = SleepTimeout.NeverSleep;
100         } else {
101             #if UNITY_EDITOR
102                 //print("NO GYRO");
103             #endif
104         }
105     }
106
107     function Start() {
108         screenSize.x = Screen.width;
109         screenSize.y = Screen.height;
110     }
111
112     function Update () {
113         if (gyroBool) {
114             #if UNITY_IPHONE
115                 quatMap = gyro.attitude;
116             #endif
117             #if UNITY_ANDROID
118                 quatMap =
Ouaternion(gyro.attitude.w,gyro.attitude.x,gyro.attitude.y,gyro.attitude.z);
119             #endif
120             transform.localRotation = quatMap * quatMult;
121         }
122         #if (UNITY_IPHONE || UNITY_ANDROID) && !UNITY_EDITOR
123             if (touchRotatesHeading) {
124                 GetTouchMouseInput();
125             }
126             camGrandparent.transform.localEulerAngles.y = heading;
127             camGrandparent.transform.localEulerAngles.z = headingUpDwn;
128         #endif
```

```
129
130         if (joystick.position.x > 0 || joystick.position.x < 0) {
131             camGrandparent.transform.position.z += joystick.position.x;
132         };
133
134         if (joystick.position.y > 0 || joystick.position.y < 0) {
135             camGrandparent.transform.position.x += joystick.position.y;
136         };
137
138     }
139
140     function GetTouchMouseInput() {
141         if(Input.GetMouseButtonDown(0)) {
142             mouseStartPoint = Input.mousePosition;
143             headingAtTouchStart = heading;
144             headingUpDwnAtTouchStart = headingUpDwn;
145         } else if (Input.GetMouseButton(0)) {
146             var delta : Vector2;
147             var mousePos = Input.mousePosition;
148             delta.x = (mousePos.x - mouseStartPoint.x)/screenSize.x;
149             delta.y = (mousePos.y - mouseStartPoint.y)/screenSize.y;
150             heading = (headingAtTouchStart+delta.x*100);
151             heading = heading%360;
152             headingUpDwn = (headingUpDwnAtTouchStart+delta.y*100);
153             headingUpDwn = headingUpDwn%360;
154         }
155     }
```

```
1  /*
2  chooseHut.js
3  This script is used by the database system ARK (ark.lparchaeology.com) to automatically
display the 3d model
4  and location of a chosen house.
5  It takes a URL from the ARK database and then splits that URL to identify the house
number that is required,
6  it then matches that with the correct house attribute in Unity and moves the virtual
camera to the correct
7  location
8  */
9  #pragma strict
10
11 function Start () {
12
13     //we need to get the hut id - so we can choose the right camera to start from
14     Debug.Log("SRC: " + Application.srcValue);
15     //this gives us the src value - which we can now explode and parse
16     //first explode on the '?' to get the start of the params
17     var src = Application.srcValue;
18     // an example src = "data/unity_projects/leskernick.unity3d
/leskernick.unity3d?item_key=hut_cd&hut_cd=LK12_28";
19     var splitsrc = src.Split('?'[0]);
20
21     // iterate through the array - we should be in the querystring now
22      for (var value : String in splitsrc) {
23          var splitparam = value.Split('&'[0]);
24          for (var param : String in splitparam) {
25              var splitparam2 = param.Split('='[0]);
26              if (splitparam2[0] == 'hut_cd') {
27                  var id = splitparam2[1];
28                  var split_id = id.Split('_'[0]);
29                  SelectHut(parseInt(split_id[1]));
30              };
31          }
32      }
33 }
34
35 function Update () {
36
37 }
38
39 var hut : GameObject;
40 var player : GameObject;
41
42 //find the house and move the player (i.e. the first person camera) to its location
43 function SelectHut (index : int) {
44     hut = GameObject.Find("hut_" + index);
45     Debug.Log(hut);
46     player.transform.position = hut.transform.position;
47 }
48
49
```

```
/*Clipped.shader
* When attached to a render object this shader
* will cause the object to be occluded by any
* other render object that has the TransWall
* shader attached to it.
*/
Shader "Clipped" {
    Properties {
        _MainTex ("Base (RGB)", 2D) = "white" {}
    }
    SubShader {
        Tags { "RenderType"="Opaque" "Queue" = "Geometry+2" }
        LOD 200

        CGPROGRAM
        #pragma surface surf Lambert

        sampler2D _MainTex;

        struct Input {
            float2 uv_MainTex;
        };

        void surf (Input IN, inout SurfaceOutput o) {
            half4 c = tex2D (_MainTex, IN.uv_MainTex);
            o.Albedo = c.rgb;
            o.Alpha = c.a;
        }
        ENDCG
    }
    FallBack "Diffuse"
}

```

```
/*
convertToBng.cs
This script is used to convert GPS coordinates
from the WGS84 geographic projection (LatLongs) into
the OSGB36 British National Grid map projection.

It can also be used to convert from BNG into Unity
gamespace coordinates by using a false Easting and Northing

This script is used to automatically update the virtual
position of the AR device from the real reality (GPS) position

It also builds the Graphical User Interface for the Location-Based
AR application
*/
using UnityEngine;
using System.Collections;

[System.Serializable]
public class MapCoordinate {
    public float latitude;
    public float longitude;
    public float altitude;
    public float heading;

    public float x {
        get {
            return longitude;
        }

        set {
            longitude = value;
        }
    }

    public float y {
        get {
            return latitude;
        }

        set {
            latitude = value;
        }
    }

    public float z {
        get {
            return altitude;
        }

        set {
            altitude = value;
        }
    }

    public float direction {
        get {
            return heading;
        }

        set {
            heading = value;
        }
    }

    public MapCoordinate(float lon, float lat, float alt, float dir) {
```

```
    longitude = lon;
    latitude = lat;
    altitude = alt;
    direction = dir;
  }
}

public class convertToBNG : MonoBehaviour {

  public float gpsAccuracy = 5;
  public float gpsUpdateDistance = 1;
  public float falseEasting = 212500;
  public float falseNorthing = 75000;
  private GameObject GyroCam;
  bool gpsRunning = false;
  private bool ready = false;
  public MapCoordinate globalPos;
  public MapCoordinate prevGlobalPos;
  public MapCoordinate BNGPos;

  private float BNG_E;
  private float BNG_N;
  private float BNG_alt;
  private float refresh_time = 2.0f;

  //GUI Variables
  private string BNG_E_input = "218178";
  private string BNG_N_input = "80101.90";
  private string BNG_alt_input = "292";
  private string BNG_heading_input = "180";

  public Shader diffuse = Shader.Find("Diffuse");
  public Shader transwalls = Shader.Find("TransWalls");
  public GameObject landscape = GameObject.Find("blender_dtm_10k_lores");

  public GameObject huts = GameObject.Find("huts");
  public GameObject spheres = GameObject.Find("spheres");


  void Awake() {
    prevGlobalPos = new MapCoordinate(0,0,0,0);
    globalPos = new MapCoordinate(0,0,0,0);
    BNGPos = new MapCoordinate(0,0,0,0);
  }

  // Use this for initialization
  void Start () {

    StartGPS();
    LatLongToEastNorth(globalPos.latitude, globalPos.longitude, globalPos.altitude,
true):

  }

  // Update is called once per frame
  void Update () {

  }
  private double toRad(double val)
  {
    return val * (System.Math.PI / 180);
  }

  public void LatLongToEastNorth(double latitude, double longitude, double
altitude, bool move_camera = false)
  {
```

```
131             //This will not work unless you have your lats and longs in decimal degrees.
132             latitude = toRad(latitude);
133             longitude = toRad(longitude);
134
135             double a = 6377563.396, b = 6356256.910; // Airy 1830 major &amp;amp; minor
semi-axes
136             //double a = 6378137.0, b = 6356752.314245; WGS84 major &amp;amp; minor semi-axes
137
138             double F0 = 0.9996012717; // NatGrid scale factor on central meridian
139             double lat0 = toRad(49);
140             double lon0 = toRad(-2); // NatGrid true origin
141             double N0 = -100000, E0 = 400000; // northing &amp;amp; easting of true origin,
metres
142             double e2 = 1 - (b * b) / (a * a); // eccentricity squared
143             double n = (a - b) / (a + b), n2 = n * n, n3 = n * n * n;
144
145             double cosLat = System.Math.Cos(latitude), sinLat = System.Math.Sin(latitude);
146             double nu = a * F0 / System.Math.Sqrt(1 - e2 * sinLat * sinLat) ; //
transverse radius of curvature
147             double rho = a * F0 * (1 - e2) / System.Math.Pow(1 - e2 * sinLat * sinLat,
1.5): // meridional radius of curvature
148
149             double eta2 = nu / rho - 1;
150
151             double Ma = (1 + n + (5 / 4) * n2 + (5 / 4) * n3) * (latitude - lat0);
152             double Mb = (3 * n + 3 * n * n + (21/8)*n3) * System.Math.Sin(latitude -
lat0) * System.Math.Cos(latitude + lat0);
153             double Mc = ((15/8)*n2 + (15/8)*n3) * System.Math.Sin(2 * (latitude - lat0))
* Svstem.Math.Cos(2 * (latitude + lat0));
154             double Md = (35 / 24) * n3 * System.Math.Sin(3 * (latitude - lat0)) *
Svstem.Math.Cos(3 * (latitude + lat0));
155             double M = b * F0 * (Ma - Mb + Mc - Md); // meridional arc
156
157             double cos3lat = cosLat * cosLat * cosLat;
158             double cos5lat = cos3lat * cosLat * cosLat;
159             double tan2lat = System.Math.Tan(latitude) * System.Math.Tan(latitude);
160             double tan4lat = tan2lat * tan2lat;
161
162             double I = M + N0;
163             double II = (nu / 2) * sinLat * cosLat;
164             double III = (nu / 24) * sinLat * cos3lat * (5 - tan2lat + 9 * eta2);
165             double IIIA = (nu / 720) * sinLat * cos5lat * (61 - 58 * tan2lat + tan4lat);
166             double IV = nu * cosLat;
167             double V = (nu / 6) * cos3lat * (nu / rho - tan2lat);
168             double VI = (nu / 120) * cos5lat * (5 - 18 * tan2lat + tan4lat + 14 * eta2 -
58 * tan2lat * eta2);
169
170             double dLon = longitude - lon0;
171             double dLon2 = dLon * dLon, dLon3 = dLon2 * dLon, dLon4 = dLon3 * dLon, dLon5
= dLon4 * dLon, dLon6 = dLon5 * dLon;
172
173             double N = I + II * dLon2 + III * dLon4 + IIIA * dLon6; //This is the northing
174             double E = E0 + IV * dLon + V * dLon3 + VI * dLon5; //This is the easting
175             BNGPos.x = (float)E;
176             BNGPos.z = (float)N;
177             BNGPos.y = (float)altitude;
178             Debug.Log("BNG E: " + E + " BNG N: " + N + " Alt: " + altitude);
179             if (move_camera) {
180                 MoveCameraToGameSpace(E, N, altitude);
181             }
182
183         }
184
185     public void MoveCameraToGameSpace(double raw_BNG_E, double raw_BNG_N, double
raw BNG_alt, double heading = 9999.99){
186         GyroCam = GameObject.Find("camGrandParent");
```

```
187             Debug.Log("I should be moving");
188             //now put the GyroCamera into the right place
189             BNG_E = (float)raw_BNG_E - falseEasting;
190             BNG_N = (float)raw_BNG_N - falseNorthing;
191             BNG_alt = (float)raw_BNG_alt;
192             //Vector3 pos = new Vector3(BNG_E, BNG_alt, BNG_N);
193             GyroCam.transform.position = new Vector3(BNG_E, BNG_alt, BNG_N);
194             if (heading != 9999.9) {
195                 GyroCam.transform.localEulerAngles = new Vector3(0.0f, (float)heading,
0.0f):
196 
197             } else {
198                 GyroCam.transform.localEulerAngles = new Vector3(0.0f, 0.0f, 0.0f);
199             }
200         }
201 
202         IEnumerator ActivateGPS() {
203             gpsRunning = true;
204             Input.location.Start(gpsAccuracy, gpsUpdateDistance);
205 
206             float duration = 0;
207             while (duration < 20.0f) {
208                 if (Input.location.status == LocationServiceStatus.Running
209                     || Input.location.status == LocationServiceStatus.Failed) break;
210                 yield return new WaitForSeconds(0.1f);
211 
212                 duration += 0.1f;
213             }
214 
215             if (duration >= 20.0f) {
216                 Debug.Log("**** LocationService Timed out");
217             }
218 
219             if (Input.location.status == LocationServiceStatus.Failed) {
220                 Debug.Log("**** User declined LocationService?");
221                 gpsRunning = false;
222             }
223 
224             ready = true;
225             Input.compass.enabled = true;
226 
227             while (Input.location.status == LocationServiceStatus.Running) {
228                 globalPos.longitude = Input.location.lastData.longitude;
229                 globalPos.latitude = Input.location.lastData.latitude;
230                 globalPos.altitude = Input.location.lastData.altitude;
231                 //Debug.Log("Lat:" + globalPos.latitude + " Lon: " + globalPos.longitude + "
Alt: " + globalPos.altitude);
232                 LatLongToEastNorth(globalPos.latitude, globalPos.longitude,
alobalPos.altitude);
233                 /*
234                 if (globalPos.x != prevGlobalPos.x
235                     || globalPos.y != prevGlobalPos.y) Debug.Log("iphone gps: (" +
alobalPos.x + "," + globalPos.y + ")");
236                 */
237 
238                 yield return new WaitForSeconds(refresh_time);
239             }
240 
241             gpsRunning = false;
242         }
243 
244         public void StartGPS() {
245             if (Application.isEditor) {
246                 ready = true;
247                 return;
248             }
```

```
  249
  250           if (!gpsRunning) StartCoroutine(ActivateGPS());
  251       }
  252
  253       public void OnGUI() {
  254           BNG_E_input = GUI.TextField(new Rect(10, 10, 200, 20),BNG_E_input, 50);
  255           BNG_N_input = GUI.TextField(new Rect(10, 40, 200, 20),BNG_N_input, 50);
  256           BNG_alt_input = GUI.TextField(new Rect(10, 70, 200, 20),BNG_alt_input, 50);
  257           BNG_heading_input = GUI.TextField(new Rect(10, 110, 200, 20),BNG_heading_input,
50):
  258
  259           if (GUI.Button (new Rect (10,140,300,100), "Reset Position and Heading")) {
  260               MoveCameraToGameSpace(double.Parse(BNG_E_input), double.Parse(BNG_N_input),
double.Parse(BNG_alt_input), double.Parse(BNG_heading_input));
  261           }
  262
  263           if (GUI.Button (new Rect (10,240,300,100), "Show/Hide Landscape")) {
  264               if (landscape.renderer.material.shader == diffuse)
  265                           landscape.renderer.material.shader = transwalls;
  266                       else
  267                           landscape.renderer.material.shader = diffuse;
  268           }
  269           if (GUI.Button (new Rect (10,450,300,100), "Set position via GPS")) {
  270               LatLongToEastNorth(globalPos.latitude, globalPos.longitude,
alobalPos.altitude, false);
  271               BNG_E_input = GUI.TextField(new Rect(10, 10, 200, 20),"" + BNGPos.x, 50);
  272               BNG_N_input = GUI.TextField(new Rect(10, 40, 200, 20),"" + BNGPos.z, 50);
  273               BNG_alt_input = GUI.TextField(new Rect(10, 70, 200, 20),"" + BNGPos.y, 50);
  274           }
  275
  276           //now put in the buttons for the test scenarios
  277           if (GUI.Button (new Rect (1350,140,50,50), "Test 1")) {
  278               huts.SetActiveRecursively(false);
  279               spheres.SetActiveRecursively(false);
  280           }
  281           if (GUI.Button (new Rect (1350,210,50,50), "Test 2")) {
  282               huts.SetActiveRecursively(false);
  283               spheres.SetActiveRecursively(true);
  284           }
  285           if (GUI.Button (new Rect (1350,280,50,50), "Test 3")) {
  286               huts.SetActiveRecursively(true);
  287               spheres.SetActiveRecursively(false);
  288           }
  289       }
  290
  291   }
  292
  293
  294
```

```
/*
  Dead Man's Nose

 A simple web server that fires a digital fan
 using a WiFi shield.

 This script is adapted by Stuart Eve from:

 This example is written for a network using WPA encryption. For
 WEP or WPA, change the Wifi.begin() call accordingly.

 Circuit:
 * WiFi shield attached
 * Analog inputs attached to pins A0 through A5 (optional)

 created 13 July 2010
 by dlf (Metodo2 srl)
 modified 31 May 2012
 by Tom Igoe
 */
#include <SPI.h>
#include <WiFi.h>

//Servo myservo;  // create servo object to control a servo
                // a maximum of eight servo objects can be created

//int pos = 0;    // variable to store the servo position


char ssid[] = "*******"; //  your network SSID (name)
char pass[] = "*******";    // your network password (use for WPA, or use as key for WEP)
int keyIndex = 0;                 // your network key Index number (needed only for WEP)

int status = WL_IDLE_STATUS;

WiFiServer server(80);

int fanPin = 9;
boolean fanIsOn = 0;

String currentLine = "";
String reqPin = "";
String power = "";


void setup() {
  // reserve space for the strings:
  currentLine.reserve(256);
  //Initialize serial and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }

  // check for the presence of the shield:
  if (WiFi.status() == WL_NO_SHIELD) {
    Serial.println("WiFi shield not present");
    // don't continue:
    while(true);
  }

  // attempt to connect to Wifi network:
  while ( status != WL_CONNECTED) {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);
    // Connect to WPA/WPA2 network. Change this line if using open or WEP network:
```

```
    status = WiFi.begin(ssid, pass);

    // wait 10 seconds for connection:
    delay(10000);
  }
  server.begin();
  // you're connected now, so print out the status:
  printWifiStatus();
  digitalWrite(2, HIGH);

  //myservo.attach(9);  // attaches the servo on pin 9 to the servo object

}


void loop() {
  // listen for incoming clients
  WiFiClient client = server.available();
  if (client) {
    Serial.println("new client");
    if (fanIsOn == 0){
     //   analogWrite(fanPin,250);
        fanIsOn = 1;
    } else {
      //  analogWrite(fanPin,150);
        fanIsOn = 0;
    }
    // an http request ends with a blank line
    boolean currentLineIsBlank = true;

    int charcounter = 0;
    while (client.connected()) {
      if (client.available()) {
       char c = client.read();
       // add incoming byte to end of line:
        currentLine += c;
        // if you've gotten to the end of the line (received a newline
        // character) and the line is blank, the http request has ended,
        // so you can send a reply
        if (c == '\n' && currentLineIsBlank) {
          // send a standard http response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connnection: close");
          client.println();
          client.println("<!DOCTYPE HTML>");
          client.println("<html>");
          // add a meta refresh tag, so the browser pulls again every 5 seconds:
         // client.println("<meta http-equiv=\"refresh\" content=\"5\">");
          // output the value of each analog input pin
          for (int analogChannel = 0; analogChannel < 6; analogChannel++) {
            int sensorReading = analogRead(analogChannel);
            client.print("analog input ");
            client.print(analogChannel);
            client.print(" is ");
            client.print(sensorReading);
            client.println("<br />");
          }
          client.println("</html>");
           break;
        }
        if (c == '\n') {
          // you're starting a new line
          currentLineIsBlank = true;
          //check to see if this line is the GET request
          if (currentLine.startsWith("GET")) {
```

```
133               Serial.println(currentLine);
134               reqPin = String(currentLine[10]) + String(currentLine[11]);
135               Serial.println("Pin Req = " + reqPin);
136               power = String(currentLine[19]) + String(currentLine[20]) +
String(currentLine[21]);
137               Serial.println("Power = " + power);
138             }
139
140             currentLine = "";
141           }
142           else if (c != '\r') {
143             // you've gotten a character on the current line
144             currentLineIsBlank = false;
145           }
146         }
147       }
148       digitalWrite(reqPin.toInt(), HIGH);
149       delay(5000);
150       digitalWrite(reqPin.toInt(), LOW);
151       delay(25);
152       // give the web browser time to receive the data
153       delay(1);
154         // close the connection:
155         client.stop();
156         Serial.println("client disonnected");
157     }
158   }
159
160
161
162   void printWifiStatus() {
163     // print the SSID of the network you're attached to:
164     Serial.print("SSID: ");
165     Serial.println(WiFi.SSID());
166
167     // print your WiFi shield's IP address:
168     IPAddress ip = WiFi.localIP();
169     Serial.print("IP Address: ");
170     Serial.println(ip);
171
172     // print the received signal strength:
173     long rssi = WiFi.RSSI();
174     Serial.print("signal strength (RSSI):");
175     Serial.print(rssi);
176     Serial.println(" dBm");
177   }
178
179
```

```
1  /*
2  drawGISLine.cs
3  This script creates 3D lines from a file extracted by
4  the GRASS GIS function v.out.ascii to represent GIS
5  polyline data within Unity
6  The ASCII file can be exported on-demand to allow
7  virtually real-time access to the GIS data
8
9  */
10
11 using UnityEngine;
12 using System.Collections;
13
14 //use the Vectrosity library to create the 3D tubes
15 using Vectrosity;
16
17 public class drawGISLine : MonoBehaviour {
18
19     public TextAsset coord_file = new TextAsset();
20     public float false_easting = new float();
21     public float false_northing = new float();
22     public float drop_height = new float();
23
24     // Use this for initialization
25     void Start () {
26         bool new_line = false;
27         //each ASCII file created by GRASS v.out.ascii is a standard format
28         //the first thing we need to do is get an array filled with the different line
elements within the ascii file
29         string[] dataLines = coord_file.text.Split('\n');
30         string[] dataPairs = new string[dataLines.Length];
31
32         //note we have set this to start at line 10 as that begins after the standard
ASCII header
33         int lineNum = 0;
34         for (int key = 0; key < dataLines.Length; ++key) {
35             if (key > 9) {
36                 dataPairs[lineNum++] = dataLines[key];
37             }
38         }
39         //create two Array holders to hold the collections of lines
40         //these will be filled with coordinate pairs which make up the lines
41         //once filled the arrays will be looped through to actual render the
42         //lines themselves
43         ArrayList lines = new ArrayList();
44         ArrayList line = new ArrayList();
45
46         bool new_line_bool = true;
47
48         int i = 0;
49
50         //loops through each coordinate pair (a line is made up of a set of coordinate
pairs)
51         //each Line in the ASCII file starts with an 'L' therefore each line can be
identified
52
53         foreach (string pair in dataPairs) {
54             if (pair != null) {
55                 if (pair.Contains("L")) {
56                     new_line_bool = true;
57                 } else {
58                     string[] coords = pair.Split(' ');
59                     ArrayList coord_clean = new ArrayList();
60
61                     foreach (string coord in coords ) {
62                         if (coord != "") {
```

```
                            coord_clean.Add(float.Parse(coord));
                        }
                    }
                    //check after cleaning we have exactly 2 coordinates
                    //then apply the false easting and northings to align them from real
                    //world coordinates into the Unity gamespace
                    if (coord_clean.Count == 2) {
                        float easting = (float)coord_clean[0] - false_easting;
                        float northing = (float)coord_clean[1] - false_northing;


                        //check they are valid coordinates if so either add them to a
                        //line or use them to start a new line
                        if (easting > 0 && northing > 0) {
                            if (new_line_bool == true) {
                                if (line.Count>1) {
                                    lines.Add(line.Clone());
                                }
                                line.Clear();
                                line.Add(new Vector3(easting,drop_height,northing));
                                new_line_bool = false;
                            } else {
                                line.Add(new Vector3(easting,drop_height,northing));
                            }
                        }
                    }
                }

            }
        }
        //once the line array is created - actually go through and draw the lines in Unity
        foreach (ArrayList line_seg in lines) {
            Vector3[] linePoints = new Vector3[line_seg.Count];
            int j = 0;
            foreach (Vector3 line_vec in line_seg) {
              linePoints[j] = line_vec;
              j++;
            }
            VectorLine myLine = new VectorLine("MyLine", linePoints, Color.red, null,
1.0f.LineType.Continuous);
            myLine.Draw3D();
        }
    }

    // Update is called once per frame
    void Update () {

    }
}

```

```
1  /*
2  OccludableAudio.cs
3  This script creates an audio source that is occluded by the geometry
4  within the scene.
5  This script should be attached to an audio source and the Listener
6  variable assigned to the player.
7  */
8  using UnityEngine;
9  using System.Collections;
10
11 public class OccludableAudio : MonoBehaviour {
12
13     private Transform m_MyTrans;
14     private AudioSource m_Source;
15
16     private float m_MaxDistance;
17
18     //set the level of occlusion (distance and fade)
19
20     public Transform Listener;
21     public float OccludedDistance = 5.0f;
22     public float FadeSpeed = 10.0f;
23     public LayerMask Mask;
24
25     void Start(){
26         m_MyTrans = transform;
27         m_Source = audio;
28         m_MaxDistance = m_Source.maxDistance;
29     }
30     void Update(){
31         float target;
32         //use the physics engine to raycast to the nearest occluding geometry
33         if (Physics.Linecast(Listener.position, m_MyTrans.position, Mask.value)){
34             target = OccludedDistance;
35         } else{
36             target = m_MaxDistance;
37         }
38         //fade the audio if necessary
39         m_Source.maxDistance = Mathf.MoveTowards(m_Source.maxDistance, target,
Time.deltaTime * FadeSpeed);
40         Debug.Log("maxdistance: " + m_Source.maxDistance);
41     }
42 }
43
```

```
/*
placeHuts.js
This script creates 3D points from a file extracted by
the GRASS GIS function v.out.ascii to represent GIS
point data within Unity
The ASCII file can be exported on-demand to allow
virtually real-time access to the GIS data
*/
#pragma downcast
var coord_file : TextAsset;
var false_easting : float;
var false_northing : float;
var drop_height : float;
var hut : Transform;
var prefix : String = 'hut';
var player : GameObject;

function Start () {

    //read in the object locations from the ascii file
    var returnChar = "\n"[0];
    var commaChar = "|"[0];

    //parse the file extracting the coordinate pairs

    var dataLines = coord_file.text.Split(returnChar);
    var buildDataPairs = new ArrayList();

    for (var dataLine in dataLines) {
        var dataPair = dataLine.Split(commaChar);
        buildDataPairs.Add(dataPair);
    }
    //put the coordinate pairs into an array and apply the false Easting
    //and Northings
    var dataPairs = buildDataPairs.ToArray();
    for (var i=0; i < dataPairs.length - 1; i++) {
        var easting = parseFloat(dataPairs[i][0]) - false_easting;
        var northing = parseFloat(dataPairs[i][1]) - false_northing;
        //in order for the 3D models to align properly with the landscape, they need
        //to be 'dropped' from a small height and their physics colliders take care of the
        //rest. Therefore place the model a little way above the 'ground' and then
instantiate
        //it. As soon as it hits the ground its rotation is frozen.
        var hit : RaycastHit;
        if (Physics.Raycast (Vector3(easting, drop_height, northing), -Vector3.up, hit,
100.0)) {
            var distanceToGround = hit.point.y + 5;
        }
        var newhut = Instantiate (hut, Vector3(easting, distanceToGround, northing),
Quaternion.identity);
        //finally name the model by using the attributes in the GIS ascii file
        newhut.name = prefix + "_" + dataPairs[i][2];

    };
}

function Update () {

}
```

```
/*
smellyFan.js
This script is used by the Dead Men's Nose Arduino
application. When the player reaches a certain
area in the Unity/AR world (a smellzone) - a
signal is sent to the webpage interface that
controls the Arduino microcontroller. The webpage is
sent an instruction via the querystring, to tell it
which pin to send power to. This then starts the
fan - which in turn wafts the approriate smell
from the Dead Man's Nose.
*/
#pragma strict

// Fire off Arduino pins
public var url = "http://192.168.0.4/?pin=";
public var smell1 = "09";
public var smell2 = "10";
function Start () {
    url = url + smell1;
     // Start a download of the given URL
    var www : WWW = new WWW (url);

    // Wait for download to complete
    yield www;

}
```

```
/*
* Transwalls.shader
* Any render object that has this shader
* attached will occlude any part of
* another object that has the Clipped
* shader attached to it
*
*/
Shader "TransWalls" {

    SubShader{
        Tags {"Queue" = "Geometry+1"}
        ColorMask 0
        Pass{}
    }

}

```