

PART ONE: 3D RECONSTRUCTION

SET UP THE PROJECT

1. Create a new Unity project named Tango Workshop.
2. File > Build Settings > change **Platform** to Android.
 - a. Player Settings > change **Bundle Identifier** to `com.TangoWorkshop.Example`.
3. Import Tango package (Wasat release).
4. Import workshop package.
5. Delete **Main Camera**.

ADD TANGO PREFABS AND RECONSTRUCTION SCRIPTS

6. Assets > TangoPrefabs > add **Tango Manager** to the scene.
 - a. Check **Enable Depth**.
 - b. Check **Enable Video Overlay**.
 - i. Change **Method** to `Texture and Raw Bytes`.
 - c. Check **Enable 3D Reconstruction (Experimental)**.
 - i. Change **Resolution (meters)** to `0.05`.
 - ii. Check **Generate Color**.
7. Assets > TangoPrefabs > add **Tango Delta Camera** to the scene.
 - a. Disable Character Controller.
8. Scene hierarchy > Create > Create Empty.
 - a. Rename to **Dynamic Mesh**.
 - b. Assets > TangoSDK > Examples > Common > Scripts > add **TangoDynamicMesh**.
 - c. Add Component > Mesh > Mesh Renderer.
 - i. Materials > change **Element 0** to `unlit_vertex_color`.
 - d. Add Component > Physics > Mesh Collider.
9. Scene hierarchy > Create > Create Empty.
 - a. Rename to **GUI**.
 - b. TangoSDK > Examples > ExperimentalMeshBuilderWithColor > Scripts > add **MeshBuilderWithColorGUIController**.

ADD THE WORKSHOP GAME MANAGER PREFAB

10. Edit > Project Settings > Physics > Gravity > change **Y** component to `-1`.
11. Assets > TangoWorkshop > Prefabs > add **Game Manager** to the scene.
12. File > Save Scene > choose a filename for the scene.
13. File > Build & Run > (change to Android again if needed) choose a filename for the APK.

1.1 - A COLOR MESH IS GENERATED AND SIMPLE SHAPES CAN BE PLACED OR THROWN.

PART TWO: AUGMENTED REALITY

ADD TANGO AUGMENTED REALITY CAMERA PREFAB

1. Delete **Tango Delta Camera**.
2. Assets > TangoPrefabs > add **Tango AR Camera**.
 - a. Camera > Clipping Planes > increase **Far** to **10** (or larger).
3. File > Build & Run.

2.1 - THE CAMERA'S RGB VIDEO NOW APPEARS BEHIND THE LESS ACCURATE DYNAMIC MESH.

EXTRACT RGB VIDEO AS A RENDER TEXTURE

4. **Tango Manager** > uncheck **Generate Color**.
5. Assets > Tango Workshop > Textures > Create > Render Texture.
 - a. Rename to **VideoRGBTexture**.
 - b. Change **Size** to **1920 x 1200**.
6. **Tango AR Camera** > Tango AR Screen (Script) > Edit Script.
 - a. Line 37: declare a **RenderTarget** reference:

```
34 [RequireComponent(typeof(Camera))]
35 public class TangoARScreen : MonoBehaviour, ITangoLifecycle, ITangoCameraTexture
36 {
37     /// <summary>
38     /// TANGO WORKSHOP - If set, this RenderTexture will receive the RGB video
39     /// instead of the camera rendering it to the screen.
40     /// </summary>
41     public RenderTexture videoRGB;
42
43     /// <summary>
44     /// If set, m_updatePointsMesh in PointCloud also gets set. Then PointCloud
```

- b. Line 136: insert a custom **CommandBuffer** when the **RenderTarget** is set:

```
134 CommandBuffer buf = VideoOverlayProvider.CreateARScreenCommandBuffer();
135 m_camera.AddCommandBuffer(CameraEvent.BeforeForwardOpaque, buf);
136
137 // TANGO WORKSHOP - Copy to videoRGB and clear if videoRGB is set.
138 if (videoRGB)
139 {
140     CommandBuffer extract = new CommandBuffer();
141     extract.Blit((Texture)null, videoRGB);
142     extract.ClearRenderTarget(true, true, Color.black);
143     m_camera.AddCommandBuffer(CameraEvent.BeforeForwardOpaque, extract);
144 }
145
146 m_camera.AddCommandBuffer(CameraEvent.BeforeGBuffer, buf);
147 }
```

7. **Tango AR Camera** > Tango AR Screen (Script) > change **Video RGB** to **VideoRGBTexture**.
8. With **VideoRGBTexture** selected, press the editor play button.

2.2 - THE RENDER TEXTURE SHOWS THE EMULATED ROOM AND THE CAMERA DISPLAY IS BLACK.

WRITE A SHADER TO PROJECT THE RGB VIDEO ON TO THE DYNAMIC MESH

9. Assets > TangoWorkshop > Shaders > Create > Shader > Standard Surface Shader.
 - a. Rename to **ARProjectionShader**.
 - b. Emulate the shader seen below:

```
1 // original work credited to Deniz Cetinalp: https://github.com/DenizTC/YorkUResearch
2
3 Shader "Tango Workshop/AR Projection" {
4     Properties {
5         _MainTex("Video RGB Texture", 2D) = "white" {}
6     }
7     SubShader {
8         Tags{ "RenderType"="Opaque" }
9         LOD 200
10
11         CGPROGRAM
12         #pragma surface surf ShadowOnly fullforwardshadows
13         #pragma target 3.0
14
15         sampler2D _MainTex;
16
17         struct Input {
18             float2 uv_MainTex : TEXCOORD0;
19             float4 screenPos;
20         };
21
22         inline fixed4 LightingShadowOnly(SurfaceOutput s, half3 lightDir, half atten) {
23             fixed4 c;
24             c.rgb = s.Albedo * atten * _LightColor0.rgb;
25             c.a = s.Alpha;
26             return c;
27         }
28
29         void surf(Input IN, inout SurfaceOutput o) {
30             o.Albedo = tex2D(_MainTex, IN.screenPos.xy / IN.screenPos.w);
31             o.Alpha = 1.0f;
32         }
33     } ENDCG
34 }
35 FallBack "Diffuse"
36 }
```

10. Assets > TangoWorkshop > Materials > Create > Material.
 - a. Rename to **ARProjectionMaterial**.
 - b. Change **Shader** to Tango Workshop > **AR Projection** (matches line 3 above).
 - c. Change **Video RGB Texture** (matches line 5 above) to **VideoRGBTexture**.
11. Dynamic Mesh > Mesh Renderer > Materials > change **Element 0** to **ARProjectionMaterial**.
12. File > Build & Run.

2.3 - THE RENDER TEXTURE IS PROJECTED ON TO THE DYNAMIC MESH.

PART THREE: LIGHTING AND POINT CLOUD

EXTEND THE WORKSHOP GAME MANAGER TO CREATE AND MOVE A POINT LIGHT

1. Game Manager > Game Manager (Script) > Edit Script.

- a. Line 19: declare a `Light` reference:

```
17  
18     private GameObject marker;  
19     private Light pointLight;  
20  
21     void Start()
```

- b. Line 85: write GUI code for creating and moving a point light:

```
82     // move position up for the next row of buttons  
83     height -= 128f;  
84 }  
85  
86 // if the marker is active, create a "light" button and the code for when it's pressed  
87 if (marker.activeSelf &&  
88     GUI.Button(new Rect(Screen.width / 2f - 128f, Screen.height - 128f, 256f, 96f),  
89         "<size=30>Place a:\nLight</size>"))  
90 {  
91     // get position similarly to line 75  
92     Vector3 position = marker.transform.position  
93         + marker.transform.forward * (0.5f - POS_OFFSET);  
94  
95     if (!pointLight)  
96     {  
97         // create a new GameObject with the Light component  
98         GameObject newObject = new GameObject("Light", new System.Type[] { typeof(Light) });  
99  
100        // store a reference to the Light component and set some initial characteristics  
101        pointLight = newObject.GetComponent<Light>();  
102        pointLight.type = LightType.Point;  
103        pointLight.shadows = LightShadows.Soft;  
104        pointLight.intensity = 5f;  
105        pointLight.range = 1f;  
106    }  
107  
108    // set the light's position and color  
109    pointLight.transform.position = position;  
110    pointLight.color = new Color(Random.value, Random.value, Random.value);  
111 }
```

2. File > Build & Run.

3.1 - A POINT LIGHT CAN NOW BE CREATED AND RELOCATED IN AUGMENTED REALITY

CREATE A DARTBOARD GAME THAT DETECTS WALLS USING TANGO POINT CLOUD

3. Assets > TangoPrefabs > add **Tango Point Cloud** to the scene.
4. **Game Manager** > Game Manager (Script) > Edit Script.
 - a. Line 17: declare a **GameObject** reference for the dartboard prefab:

```
15 [Tooltip("Drag & drop shape prefabs here to use them in the game.  
16 public GameObject[] shapePrefabs;  
17 [Tooltip("The dartboard prefab.")]  
18 public GameObject dartboardPrefab;  
19  
20 private GameObject marker;
```

- b. Lines 21 and 23: declare a **GameObject** reference for instantiating a dartboard and a **TangoPointCloud** reference:

```
19  
20 private GameObject marker;  
21 private GameObject dartboard;  
22 private Light pointLight;  
23 private TangoPointCloud pointCloud;  
24  
25 void Start()
```

- c. Line 27: grab a reference to the **TangoPointCloud** attached to **Tango Point Cloud**:

```
25 void Start()  
26 {  
27     // grab a reference to the Tango Point Cloud  
28     pointCloud = FindObjectOfType<TangoPointCloud>();  
29     // make an instance of the marker prefab  
30     marker = Instantiate(markerPrefab);
```

- d. Line 25 (before **void Start()**): write a method for detecting a wall surface:

```
23 private TangoPointCloud pointCloud;
24
25 // returns true if the device is pointed at a surface that is approximately vertical
26 private bool DeviceIsPointedAtWall(out Vector3 worldPosition, out Plane plane)
27 {
28     bool result = false;
29     Vector2 screenCenter = new Vector2(Screen.width / 2f, Screen.height / 2f);
30
31     // use TangoPointCloud to determine the best-fit plane at the screen center
32     if (pointCloud.FindPlane(Camera.main, screenCenter, out worldPosition, out plane))
33     {
34         // if the dot product of the plane's normal and the world up vector is
35         // between -0.1f and 0.1f, we'll agree it roughly resembles a wall
36         result = Mathf.Abs(Vector3.Dot(plane.normal, Vector3.up)) < 0.1f;
37     }
38
39     return result;
40 }
```

- e. Line 135: write GUI code for creating, moving and rotating the dartboard:

```
133         pointLight.color = new Color(Random.value, Random.value, Random.value);
134     }
135
136     // create the "dartboard" button and the code for when it's pressed
137     if (GUI.Button(new Rect(Screen.width - 288f, Screen.height - 128f, 256f, 96f),
138         "<size=30>Place a:\nDartboard</size>"))
139     {
140         Vector3 worldPosition;
141         Plane plane;
142
143         // instantiate the dartboard if necessary
144         if (!dartboard) dartboard = Instantiate(dartboardPrefab);
145
146         // if the device is pointed at a wall, put the dartboard there
147         // and rotate it. otherwise, hide the dartboard
148         if (DeviceIsPointedAtWall(out worldPosition, out plane))
149         {
150             dartboard.SetActive(true);
151             dartboard.transform.position = worldPosition;
152             dartboard.transform.LookAt(worldPosition + plane.normal);
153         }
154         else dartboard.SetActive(false);
155     }
156 }
157
158 }
```

5. **Game Manager** > Game Manager (Script) > change **Dartboard** to **Dartboard**.
6. File > Build & Run.

3.2 – A DARTBOARD CAN NOW BE CREATED WHEN A WALL IS DETECTED AT THE CENTER OF THE SCREEN