Unity facial landmarks height

Assign	
Status	In Progress
Priority	Medium
Date Created	@March 25, 2022 2:26 PM
Due Date	

Onnx

- Testing the `onnx` model from the mediapipe heightmap landmark detection project in unity
- Downloaded the old unity project, it is running
    - Now modifying it for new project

Steps

- Displaying image as texture
    - https://forum.unity.com/threads/how-to-load-a-texture2d-from-path-without-resources-load.914783/

    ```
    using Image<Rgb24> image = SixLabors.ImageSharp.Image.Load<Rgb24>(image_list[i]);
    var rawData = System.IO.File.ReadAllBytes(image_list[i]);
    Texture2D tex = new Texture2D(2, 2);
    tex.LoadImage(rawData);
    GameObject rawImage = GameObject.Find ("RawImage");
    rawImage.GetComponent<RawImage> ().texture = tex;
    ```

    - https://gyanendushekhar.com/2017/07/08/load-image-runtime-unity/

    ```
    myTexture = Resources.Load ("Images/SampleImage") as Texture2D;
    GameObject rawImage = GameObject.Find ("RawImage");
    ```

```
rawImage.GetComponent<RawImage> ().texture = myTexture;
```

- Working now!
- But it is only displaying last image of the loop?
    - Maybe images are being replaced!

- I can manually make more `raw image` components in UI
- or I can also make them in C# interface like

```
using UnityEngine;

public class Example : MonoBehaviour
{
    private void Start()
    {
        GameObject go1 = new GameObject();
        go1.name = "go1";
        go1.AddComponent<Rigidbody>();

        GameObject go2 = new GameObject("go2");
        go2.AddComponent<Rigidbody>();

        GameObject go3 = new GameObject("go3", typeof(Rigidbody), typeof(BoxCollider));
    }
}
```

- Make `canvas` object manually from UI
- Then find it from C# interface and add `raw image` components in it

```
GameObject canvas = GameObject.Find("Canvas");
canvas.AddComponent<RawImage>();

// Add texture to raw images
var rawData = System.IO.File.ReadAllBytes(image_list[i]);
Texture2D tex = new Texture2D(2, 2);
tex.LoadImage(rawData);
rawImage.GetComponent<RawImage> ().texture = tex;
```

    - https://docs.unity3d.com/ScriptReference/GameObject-ctor.html

Issues

- Model operation is not supported
    - Tried different `opset`
        - not working
    - Posted on their github

TFLite

Links

- https://github.com/asus4/tf-lite-unity-sample
    - https://github.com/asus4/tf-lite-unity-sample/blob/master/Assets/Samples/FaceMesh/FaceMeshSample.cs
    - https://github.com/asus4/tf-lite-unity-sample/blob/master/Assets/Samples/FaceMesh/FaceMesh.cs
- https://www.youtube.com/watch?v=s42RTgKSXis&t=550s
- https://github.com/homuler/MediaPipeUnityPlugin
- https://medium.com/@asus4/tensorflow-lite-on-unity-4a134e43cbc6
- https://github.com/keijiro/FaceMeshBarracuda
- https://github.com/keijiro/FaceLandmarkBarracuda

Ideas

- make button to load image
    - Show image
    - pass to code
    - display results on image?

Notes

- https://github.com/keijiro/FaceLandmarkBarracuda is working
    - Need to check more and understand

- https://github.com/asus4/tf-lite-unity-sample is with webcam but hard to convert to images?

- https://github.com/homuler/MediaPipeUnityPlugin is also an option but `installation` procedure might be long
    - Still not sure if it will resolve all the issues
    - https://www.youtube.com/watch?v=kMJOmzmMxZw

Method 1: https://github.com/keijiro/FaceLandmarkBarracuda

- In `Assets/Script/Visualizer.cs`

```
void OnRenderObject()
    {
        // Wireframe mesh rendering
        _material.SetBuffer("_Vertices", _detector.VertexBuffer);
        _material.SetPass(0);
        Graphics.DrawMeshNow(_template, Matrix4x4.identity);
        // Keypoint marking
        _material.SetBuffer("_Vertices", _detector.VertexBuffer);
        _material.SetPass(1);
        Graphics.DrawProceduralNow(MeshTopology.Lines, 400, 1);
    }
```

This part displays **Tessellation** and **Key points.**

- How to include more options in it?
    - Like just `contours` or `landmarks` without tessellation as in python code?
    - Ask on git?
    - Figured it out
        - Depends on `SetPass` somehow
            - Changing it to `_material.SetPass(0);` draws all points

- He has this

```
public Texture Texture
      => _dummyImage != null ? (Texture)_dummyImage : (Texture)_buffer;
```

to choose between `webcam input` and `image input`.

- How to change `thickness` and `color` of points??
    - Managed to change color.
    - Option is in the `shader` code
    - Issue link
    - Thickness not easy.

Results

- Model is working

- Now trying to access the output results
    - Got the results in the `outputBuffer`
        - Length is 1404 because `468*3 = 1404`
    - Made some slight modifications in the code

```
// Run the BlazeFace model.
// using (var tensor = new Tensor(1, ImageSize, ImageSize, 3, _preBuffer))
//     _worker.Execute(tensor);
var tensor = new Tensor(1, ImageSize, ImageSize, 3, _preBuffer);
var output = _worker.Execute(tensor).PeekOutput();
float[] outputBuffer = output.ToReadOnlyArray();
Debug.Log(outputBuffer.Length);  // 1404 because 468*3 = 1404
```

- Accessing results

```
// Accessing results
float[] outputBuffer = output.ToReadOnlyArray();
Debug.Log(outputBuffer.Length);  // 1404 because 468*3 = 1404
for (int i = 0; i < outputBuffer.Length; i+=3){
  var result = (outputBuffer[i] + "," + outputBuffer[i+1] + "," + outputBuffer[i+2] + "\n");
  Debug.Log(result);
  }
```

- Output keeps updating continuously → why?
    - Because he updates visualizer in each frame because he is using webcam

    ```
    void LateUpdate()
        {
            // Face landmark detection
            _detector.ProcessImage(_webcam.Texture);

            // UI update
            _previewUI.texture = _webcam.Texture;
        }
    ```

    - I am using images, so same results are printed continuously!

Alternate way

- In case, I still wanted to print only once, copied the code from `LateUpdate()` function to `Start()` function.
- **Start()** → means running only once, just the first time.

Original Functions:

```
void Start()
    {
        _detector = new FaceLandmarkDetector(_resources);
        _material = new Material(_shader);
    }

void LateUpdate()
     {
         // Face landmark detection
         _detector.ProcessImage(_webcam.Texture);

         // UI update
         _previewUI.texture = _webcam.Texture;
     }
```

Updated Start() function:

```
void Start()
    {
        _detector = new FaceLandmarkDetector(_resources);
        _material = new Material(_shader);

        // Face landmark detection
        _detector.ProcessImage(_webcam.Texture);
        // UI update
        _previewUI.texture = _webcam.Texture;
    }
```

Method 2:

- It is working with `camera` feed.
- Not sure how to make it work with image input.
    - His reply on my issue post is not really helpful

- Ok it is working with image input now. Replace `webcaminput` lines and pass image texture to the update function. Also add `RawImage` in the scene inside `Canvas`.

```
// string imagePath = Application.streamingAssetsPath + "/data/enhanced/enhanced_100.png";
var rawData = System.IO.File.ReadAllBytes(imagePath);
Texture2D image = new Texture2D(2, 2);
image.LoadImage(rawData);
image = Resize(image, 256, 256);
OnTextureUpdate(image);
//FindObjectOfType<UnityEngine.UI.RawImage>().texture = image;
GameObject rawImage = GameObject.Find("RawImage");
rawImage.GetComponent<RawImage>().texture = image;
```

- Image was displayed on top of the results. Instead of trying to draw results on top of the image, I just decreased `opacity` of the image.
- Use the following to create a field for image instead of code-based input.

```
[SerializeField]
public Texture2D image = null;
```

Issue

Detection is not working on all the *height maps*. Not sure about the issue yet.

Maybe it is related to image size. For `enhanced_100`, detection fails below 256x256, but works for higher size. However, this is not the case with other images..

Note

- Both methods lack *documentation* on how they did it.
    - This makes it hard for me to understand anything or make modifications.