OpenCV# for Unity

by

NWH ::coding

USER MANUAL

INSTALLATION

Once downloaded from the Asset Store all the files will be inside "OpenCVSharpforUnity" folder. This is due to the Unity Asset Store policy. To make the asset work properly, "StreamingAssets" directory from "OpenCVSharpForUnity" needs to be copied to the root of the project.

Once this is done example scene named *Menu* can be opened from the "Examples" directory. After pressing play right at the bottom of the screen in grey letters there should be text saying "OpenCV Loaded". If there is some other message check the troubleshooting section. If the tips shown there do not help contact us at nwhcoding@gmail.com .

USING OPENCV# FOR UNITY

Once installation is done add following lines into your C# script:

```
#using OpenCvSharp;
#using OpenCvSharp.Aruco;
#using OpenCvSharp.Face;
#using OpenCvSharp.[some_other_OpenCV_module];
#using NWH;
```

OpenCvSharp namespace contains all the OpenCV functionality.

NWH namespace contains various utility functions such as conversion between OpenCV and Unity formats and you can omit it if you do not plan using them.

If building for Android check README_ANDROID.

Texture2D and Mat formats

Note that when using conversion functions you will have to take care to use corresponding Mat and Texture2D formats. E.g. you can not convert 8UC3 mat to RGBA32 (3 channels compared to 4 channels). Also, it is recommended to use 3-channel textures with Android.

Corresponding supported formats are:

- 8UC1 <==> R8
- 8UC2 <==> RG16
- 8UC3 <==> RGB24
- 8UC4 <==> RGBA32

Some of the utility or conversion functions return only specific formats and those will need to be converted to required format before being used. E.g. WebCamTextureToMat always returns 8UC4 Mat due to performance reasons and such Mat needs to be converted to 8UC3 before being used on Android. For example:

```
Texture2D tex = new Texture2D(webCamTexture.width, webCamTexture.height,
TextureFormat.RGB24, false);
Mat mat = new Mat(webCamTexture.height, webCamTexture.width, MatType.CV_8UC4);

CvUtil.GetWebCamMat(webCamTexture, ref mat); [This function always returns 8UC4]
Cv2.CvtColor(mat, mat, ColorConversionCodes.RGBA2RGB); [Mat is now 8UC3]

… do something with mat …

CvConvert.MatToTexture2D(mat, ref tex); [Convert 8UC3 mat to RGB24 texture]
```

CONVERSIONS

CvConvert.cs contains multiple conversion functions. Below is a table of avaliable conversions with measured execution time. Execution time is for dual core CPU running at 3.2GHz in editor.

Source format	Destination format	Function name	Execution time (s)	Equivalent FPS
Mat	Texture2D	MatToTexture2D	0.0021	477
	Color32[]	MatToColor32Array	0.0085	118
Texture2D	Mat	Texture2DToMat	0.0023	435
	Color32[]	Texture2DToColor32	0.0002	5000
	RenderTexture	Texture2DToRenderTexture	0.0019	526
Color32[]	Mat	Color32ArrayToMat	0.0010	1000
	Texture2D	Color32ArrayToTexture2D	0.0011	910
	byte[]	Color32ArrayToByteArray	0.0007	1429
byte[]	Color32[]	ByteArrayToColor32Array	0.0702	14
		ByteArrayToColor32ArrayFast	0.0028	357
RenderTexture	Texture2D	RenderTextureToTexture2D	0.0044	227
WebCamTexture	Mat	WebCamTextureToMat	0.0012	833
	Texture2D	WebCamTextureToTexture2D	0.0019	526
	RenderTexture	WebCamTextureToRenderTexture	0.0020	500

All the functions expect a reference to the output data structure - this is intended to reduce garbage collection by not creating new data structures all the time. In some cases passed texture or OpenCV Mat will be resized to fit input data in which case it will be mentioned in the function description.

UTILITIES

Currently there are three utility functions to make use of OpenCV in Unity easier:

- GetWebCamTexture2D - retrieves image from provided WebCamTexture (must be playing) and returns it in Texture2D format. Texture2D will have dimesions of image retrieved from WebCamTexture.
- GetWebCamMat - same as GetWebCamTexture2D just returns OpenCV Mat.
- GetStreamingAssetsPath() - returns absolute file path to the file in the StreamingAssets directory. Useful for OpenCV functions that require path instead of file contents (e.g. Imread).

COMPATIBILITY

OpenCV# for Unity is compatible with following platforms / architectures (architectures not compatible with Unity are not included):

OS	Windows	Mac OS X	Android	Linux	iOS
Supported architecture(s)	x86, x86_64	x86_64	ARMv7	x86_64	i386, x86_64, ARM64, ARMv7, ARMv7s
Library format	.dll	.bundle / .dylib	.so	.so	.framework
Minimum requirements	Windows 7	OS X 10.6	Android 4	Ubuntu 12.04, SteamOS	iOS 7
Unity Editor	Yes	Yes	-	Yes	-
Standalone	Yes	Yes	Yes	Yes	Yes