iOS双摄像头在直播中的应用

原创：花椒iOS 花椒技术 11月19日

如今随着短视频、直播应用的火爆，客户端应用中对摄像头的使用和音视频的处理成了一个必备技能。除了音视频采集、处理、编码等基础功能的应用，对一些摄像头新功能和新特性的探索和应用也是我们平时重点关注的方向。

WWDC2019上苹果iOS13系统对部分机型支持了多摄像头同时采集的功能，这也是一个大家期待已久的功能。

借用WWDC2019上的一张效果图

看到WWDC2019上对该功能的介绍后，我们第一反应就是这个功能简直就是为我们户外直播量身定做的。经常观看直播的用户都知道，户外主播在直播的时候往往都是用后置摄像头来给大家给大家呈现一些周边和美景，而在需要跟观众交流互动的时候又需要切换到前置摄像头，这就使得直播体验大打折扣。有了前后多摄像头同时采集的功能，我们就能将前后两个相机返回的视频画面合成后推流，大大提升户外主播的直播体验。下面我们详细介绍在应用内支持双摄像头需要做的工作。

机型限制：

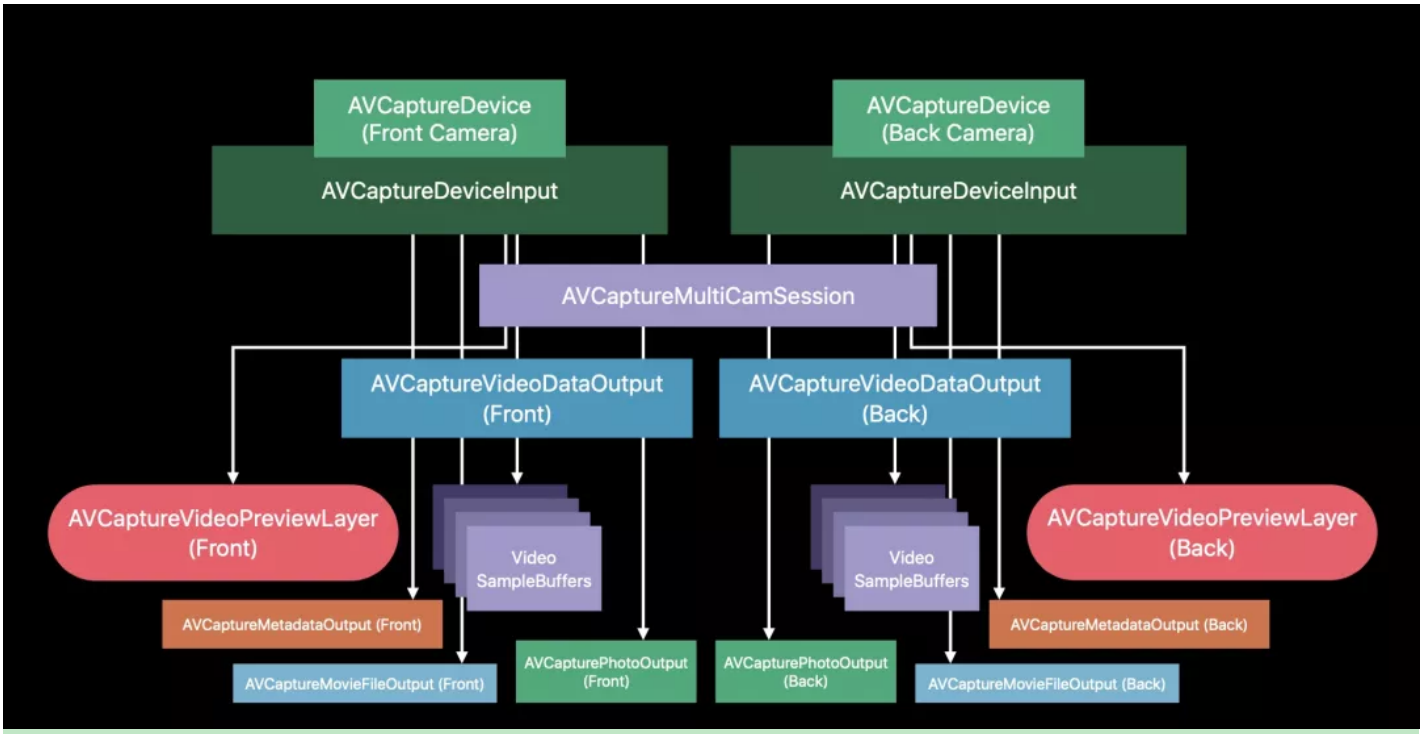
iPhone A12（iPhone XS, iPhone XS Max, iPhone XR）及以上机型

iPad Pro A12X （2019款iPad Pro）及以上机型

系统限制：

iOS13及以上

多路视频采集：



在创建单摄像头时我们会使用AVCaptureSession来管理AVCaptureInput的音视频输入和AVCaptureOutput的音视频输出。

创建多摄像头我们则需要将AVCaptureSession替换成AVCaptureMultiCamSession，然后配置前后摄像头并添加输入输出。

关键代码：

```
- (void)configSession {
///判断是否支持多摄像头
    if (AVCaptureMultiCamSession.isMultiCamSupported == NO) {
        return;
    }
///创建多摄像头会话
    self.cameraSession = [[AVCaptureMultiCamSession alloc] init];
    [self.cameraSession beginConfiguration];
    if ([self configBackCamera] == NO) {
        [self.cameraSession commitConfiguration];
        return;
    }
    if ([self configFrontCamera] == NO) {
        [self.cameraSession commitConfiguration];
```

```
        return;
    }
    if ([self configMicrophone] == NO) {
        [self.cameraSession commitConfiguration];
        return;
    }
    [self.cameraSession commitConfiguration];
}
```

```
- (BOOL)configFrontCamera {
    AVCaptureDevice *frontCamera = [self.class getCaptureDeviceWithPosition:AVCa
    if (frontCamera == nil) {
        return NO;
    }

    NSError *error = nil;
    self.frontDeviceInput = [[AVCaptureDeviceInput alloc] initWithDevice:frontCa
    if (![self.cameraSession canAddInput:self.frontDeviceInput]) {
        return NO;
    }
///这里需要注意只添加input不绑定connections
    [self.cameraSession addInputWithNoConnections:self.frontDeviceInput];

    self.frontVideoDataOutput = [[AVCaptureVideoDataOutput alloc] init];
    self.frontVideoDataOutput.videoSettings = @{(__bridge NSString *)kCVPixelBuf
    [self.frontVideoDataOutput setSampleBufferDelegate:self queue:self.dataOutpu

    if (![self.cameraSession canAddOutput:self.frontVideoDataOutput]) {
        return NO;
    }
///这里需要注意只添加output不绑定connections
    [self.cameraSession addOutputWithNoConnections:self.frontVideoDataOutput];

    AVCaptureInputPort *port = [[self.frontDeviceInput portsWithMediaType:AVMedi
                                                        sourceDeviceType:frontC
                                                    sourceDevicePosition:frontC
    AVCaptureConnection *frontConnection = [[AVCaptureConnection alloc] initWith
```

```
29
30      if (![self.cameraSession canAddConnection:frontConnection]) {
31          return NO;
32      }
33  ///手动添加connection
34      [self.cameraSession addConnection:frontConnection];
35      [frontConnection setVideoOrientation:AVCaptureVideoOrientationPortrait];
36      [frontConnection setAutomaticallyAdjustsVideoMirroring:NO];
37      [frontConnection setVideoMirrored:YES];
38
39      self.frontPreviewLayer = [[AVCaptureVideoPreviewLayer alloc] initWithSessionI
40      AVCaptureConnection *frontPreviewLayerConnection = [[AVCaptureConnection all
41      [frontPreviewLayerConnection setAutomaticallyAdjustsVideoMirroring:NO];
42      [frontPreviewLayerConnection setVideoMirrored:YES];
43      if (![self.cameraSession canAddConnection:frontPreviewLayerConnection]) {
44          return NO;
45      }
46      self.frontPreviewLayer.frame = CGRectMake(30, 30, 180, 320);
47      [self.containerView.layer addSublayer:self.frontPreviewLayer];
48  ///手动添加connection
49      [self.cameraSession addConnection:frontPreviewLayerConnection];
50
51      return YES;
52  }
```

这里需要注意，在使用AVCaptureSession的时候，我们要修改相机的分辨率等通常会使用setSessionPreset:来设置。但是AVCaptureMultiCamSession不支持setSessionPreset:，所以我们需要单独对每个AVCaptureDevice进行设置。

```
1   ///修改分辨率
2   - (BOOL)reduceResolutionForCamera:(AVCaptureDevicePosition)position {
3       for (AVCaptureConnection *connect in self.cameraSession.connections) {
4           for (AVCaptureInputPort *inputPort in connect.inputPorts) {
5           ///找到需要操作的输入设备
6               if (inputPort.mediaType == AVMediaTypeVideo && inputPort.sourceDevic
7                   AVCaptureDeviceInput *videoDeviceInput = (AVCaptureDeviceInput *
```

```
                NSArray *formats = videoDeviceInput.device.formats;
                ///遍历该设备支持的formats然后设置对应的format
                for (AVCaptureDeviceFormat *format in formats) {
                    if (format.isMultiCamSupported) {
                        CMVideoDimensions dimensions = CMVideoFormatDescriptionG
                        ///1280*720分辨率的format
                        if (dimensions.width == 1280 && dimensions.height == 720
                            NSError *error = nil;
                            ///这句需要添加，否则可能修改不成功
                            [self.cameraSession beginConfiguration];
                            if ([videoDeviceInput.device lockForConfiguration:&e
                                videoDeviceInput.device.activeFormat = format;
                                ///修改帧率
                                [videoDeviceInput.device setActiveVideoMinFrameD
                                [videoDeviceInput.device setActiveVideoMaxFrameD
                                [videoDeviceInput.device unlockForConfiguration]
                                [self.cameraSession commitConfiguration];
                                return YES;
                            }
                            [self.cameraSession commitConfiguration];
                        }
                    }
                }
            }
        }
    }
    return NO;
}
```

数据的回调方法跟之前版本的一致，我们只需要在回调方法中获取相应数据即可。

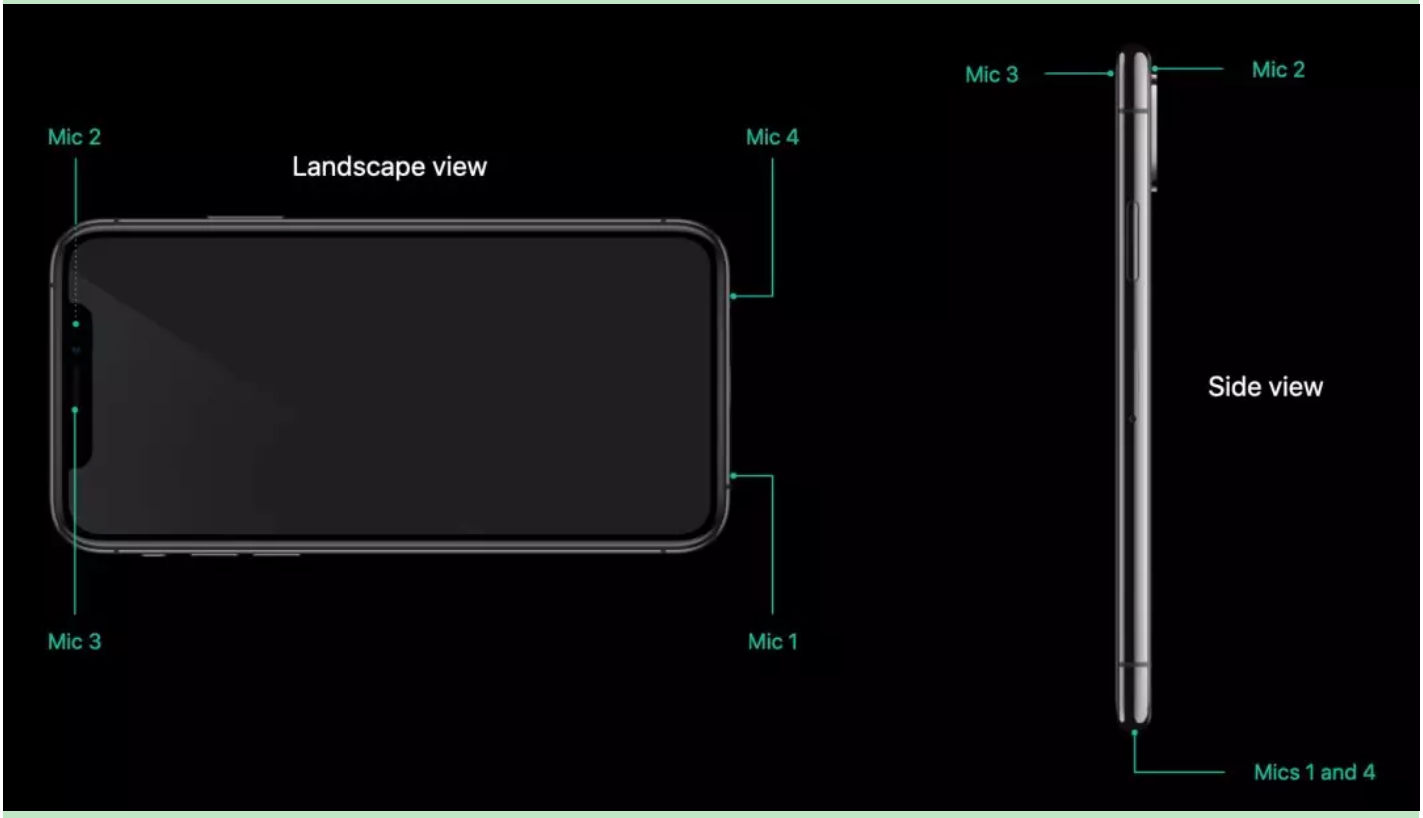
```
- (void)captureOutput:(AVCaptureOutput *)output didOutputSampleBuffer:(CMSampleBu
    if (output == self.frontVideoDataOutput) {
        CVImageBufferRef imageBuffer = CMSampleBufferGetImageBuffer(sampleBuffer)
    } else if (output == self.backVideoDataOutput) {
        CVImageBufferRef imageBuffer = CMSampleBufferGetImageBuffer(sampleBuffer)
```

```
    }
}
```

多路音频采集：

- 前置摄像头默认使用前置麦克风
- 后置摄像头默认使用后置麦克风
- 仅音频时默认使用环绕麦克风
- 可以使用AVAudioSession来使用各个麦克风的增强功能
- 使用方式同视频采集，这里就不过多介绍

麦克风分布图



注意事项：

1.add inputs 和 add outputs时不关联Connections

```
[self.cameraSession addInputWithNoConnections:self.frontDeviceInput]
[self.cameraSession addOutputWithNoConnections:self.frontVideoDataOutput]
```

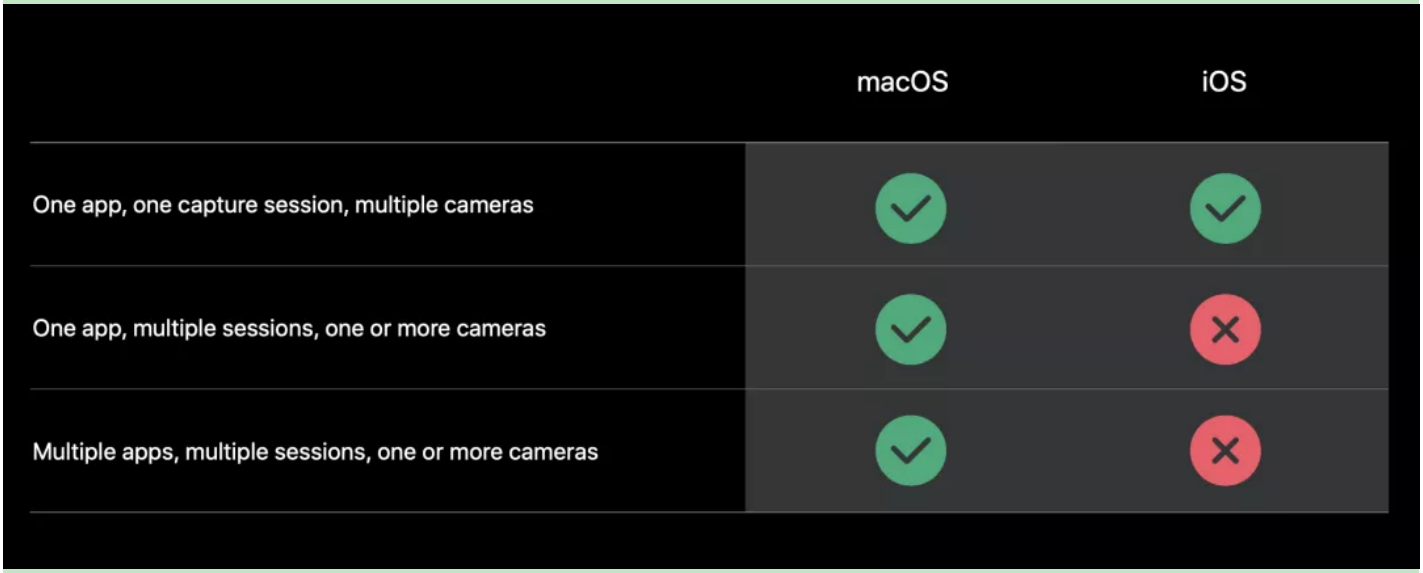
2.初始化AVCaptureVideoPreviewLayer时不关联Connections

```
[[AVCaptureVideoPreviewLayer alloc] initWithSessionWithNoConnection:self.cameraSe
```

3.手动添加AVCaptureConnection

```
[self.cameraSession addConnection:frontConnection]
```

4.iOS系统只能支持单会话多摄像头，Mac系统则可以支持多会话多摄像头



系统压力：

“There’s no such thing as a free lunch.”

多摄像头输出就意味着多传感器同时工作，这将带来更多的性能消耗和电池消耗。当系统消耗过大时会对我们应用的正常运行带来影响，最严重可能会导致应用被系统杀死或者设备发热警告。系统将压力级别分为五个级别：

```
AVCaptureSystemPressureLevelNominal     ///System pressure level is normal
AVCaptureSystemPressureLevelFair        ///System pressure is slightly elevated
AVCaptureSystemPressureLevelSerious     ///System pressure is highly elevated
AVCaptureSystemPressureLevelCritical    ///System pressure is critically elevated
AVCaptureSystemPressureLevelShutdown    ///System pressure is beyond critical
```

检测系统压力采用KVO的方式监听，代码如下：

```
- (void)addObserver {
    if (@available(iOS 11.0, *)) {
```

```
        [self.inputCamera addObserver:self forKeyPath:@"systemPressureState" c
    }
}
- (void)observeValueForKeyPath:(NSString *)keyPath ofObject:(id)object change:
    if ([keyPath isEqualToString:@"systemPressureState"]) {
        if (@available(iOS 11.0, *)) {
            AVCaptureSystemPressureState *state = change[NSKeyValueChangeNewKe
            NSDictionary *dict = @{ @"AVCaptureSystemPressureLevel" : state.le
        }
    }
}
```

所以在检测到系统消耗过度时，我们可以采取以下几种方式来降低系统的负载以保证应用正常使用

1. 降低帧率、分辨率
2. 降低应用其他场景对GPU、CPU的消耗
3. 禁用一个摄像头(禁用一个摄像头不需要重新配置Session，只需要设置对应的cameraInputVideoPort为禁用状态即可)

```
frontCameraInputVideoPort.enabled = false
```

Talk is cheap. Show me the code.

代码奉上：MulitCameraTest （ https://github.com/TideZhang/Demo.git ）

参考文档：

1. Introducing Multi-Camera Capture for iOS

https://developer.apple.com/videos/play/wwdc2019/249/

2. AVMultiCamPiP

https://developer.apple.com/documentation/avfoundation/cameras_and_media_capture/avmulticampip_capturing_from_multiple_cameras?language=objc