# 纯前端实现人脸识别-提取-合成

2017/09/14 · [JavaScript](http://web.jobbole.com/category/javascript-2/) · [1 评论](http://web.jobbole.com/92408/" \l "article-comment)· [人脸识别](http://web.jobbole.com/tag/%e4%ba%ba%e8%84%b8%e8%af%86%e5%88%ab/)

分享到：**2**

原文出处： [雅X共赏](http://refined-x.com/2017/09/06/%E7%BA%AF%E5%89%8D%E7%AB%AF%E5%AE%9E%E7%8E%B0%E4%BA%BA%E8%84%B8%E8%AF%86%E5%88%AB-%E6%8F%90%E5%8F%96-%E5%90%88%E6%88%90/" \t "http://web.jobbole.com/92408/_blank)

最近火爆朋友圈的军装照H5大家一定还记忆犹新，其原理是先提取出照片中的面部，然后与模板进行合成，官方的合成处理据说由天天P图提供技术支持，后端合成后返回给前端展示，形式很新颖效果也非常好，整个流程涉及的人脸识别和图像合成两项核心技术在前端都有对应的解决方案，因此理论上前端也可以完成人脸识别-提取-合成整个流程，实现纯前端的军装照H5效果。

## **前端人脸识别**

首先需要的是人脸识别，这个一听就觉得高大上的东西原理并不深奥，无非是用人的面部特征规则对图像进行匹配和识别，这项工作前端虽然可以实现，但前端实现基本就只能依据内置规则库进行匹配，这个库的质量就决定了识别质量，而通常更成熟的方案是引入机器学习，让程序不断自我修正和提高，进一步提高识别率，机器学习的前端库倒是也有，但把这两者结合起来的还没发现，因此对前端人脸识别的准确率不要报太高期望。

现有的前端人脸识别库不算多，这里我们选择的是效果相对好点的[trackingjs](https://trackingjs.com/" \t "http://web.jobbole.com/92408/_blank)，这个类库功能非常强大，库如其名，它可以完成各种追踪类的图像处理任务，人脸识别只是其众多功能之一，而且通过选配插件，还可以精确识别眼睛、鼻子等五官的位置，貌似稍微折腾一下也可以实现美图秀秀的效果。

这里我们只用trackingjs实现面部识别，初始化一个面部识别任务的代码如下：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | *//实例化*  **var** tracker = **new** tracking.ObjectTracker(['face']);  *//识别回调*  tracker.on('track', **function**(event) {  **if** (!event.data.length) {  **return** console.log('画面中没有人脸');      }      event.data.**forEach**(**function**(rect, i) {          console.log(rect);*//单个面部数据*      })  })  *//配置参数*  ... |

这样一个面部识别任务就初始化完成了，调用方式如下：

JavaScript



|  |  |
| --- | --- |
| 1  2 | tracking.track('#img', tracker);  *//其中'#img'参数是目标图像的选择器* |

在识别回调中event.data就是数组格式的面部数据，如果长度为0则表示图像中没有人脸或者识别失败，如果识别成功，单个面部数据的格式如下：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6 | {      x: number,          *//面部位于原图x轴方向位置*      y: nuber,           *//面部位于原图y轴方向位置*      width:number,       *//面部区域宽度*      height:nubmer       *//面部区域高度*  } |

有了这个面部数据就可以很容易的将该区域从原图中提取出来，前端当然就用canvas啦，示例如下：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **var** img = document.getElementById("img");  **var** faceCtx = document.getElementById("mycanvas").getContext('2d');    **var** theFace = ...; *//假设我们识别到了theFace*    *//使用drawImage()方法将面部绘制出来*  faceCtx.drawImage(img, theFace.x, theFace.y, theFace.width, theFace.height, 0, 0, theFace.width, theFace.height); |

到这里我们已经实现了面部识别 + 提取，而且代码量也没多少，其实这里面有个小坑要在实践中才会发现，那就是trackingjs的配置，文档中能找到4个跟识别有关的配置，分别是：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4 | setClassifiers(classifiers)  setEdgesDensity(edgesDensity)  setScaleFactor(scaleFactor)  setStepSize(stepSize) |

看不懂吧，我也看不懂，而且文档中对他们没有任何有用的说明，在测试中我只使用了后两个配置，翻译过来分别是”比例因子”和”步长”，经过枯燥的人肉测试发现，这两个参数的有效取值范围分别在1 - 2和1.1 - 2，其中setStepSize不能为1，否则会浏览器会卡死，所以从1.1开始取值，取值超过2也可以，但识别成功的概率就很低了。通过调整这两个参数绝大多数图像都可以成功识别，唯独对面部大特写很难识别，这可能需要配合另外两个参数吧，我实在没耐心继续人肉测试下去了，感兴趣的自己回去玩吧。

## **前端图像处理**

经过上一步的识别+提取我们已经得到了面部图像，要实现合成军装照效果我们还需要对面部图像进行处理，使色调与模板一致，将来才能毫无违和感的融合在一起，具体到军装照这个例子我们需要将面部重新着色，并达到”做旧”的老照片效果，如果用PS想必大家都会，但在前端怎么实现呢？

这里我们需要借助腾讯前端团队出品的[AlloyImage](http://alloyteam.github.io/AlloyImage/" \t "http://web.jobbole.com/92408/_blank)，这是一个堪称前端PS的前端图像处理类库，比如要实现上述效果，我们只需要这样：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **var** faceImg = document.getElementById("theFace");  faceImg.loadOnce(**function**() {      AlloyImage(**this**).act("灰度处理").add(          AlloyImage(**this**.width, **this**.height, "#808080")          .act("高斯模糊", 4)          .act("色相/饱和度调节", 22, 45, 0, **true**),          "叠加"      ).replace(**this**);  } |

然后你就得到了一个做旧的人脸，还是非常简单的，AlloyImage的使用基本可以说是傻瓜化，感兴趣的就自己花个五分钟去看下官方文档吧，这里不再赘述。

然后就要说一下我们这个图像处理和人家天天P图的差距了，虽然我们得到了理想的色调，但要想把随便一张人脸与特定模板做合成，有两件事必不可少。首先是面部角度矫正，如果模板是正的而你的照片是歪的，直接暴力拼接肯定很违和，所以需要先识别出面部角度，并纠正到指定角度；然后是面部中心定位，因为人脸识别的结果提取出来后不一定是以面部中心为中心的，所以在合成之前要识别出面部中心线，并以此为依据与模板进行定位。然而这些我们都没有，所以我们只能对输入的图像的要求更高，如果输入了嘴歪眼斜的图片，结果就只能尴尬了。

最后的图片合成部分就更简陋了，先将处理好的面部画到画布指定位置，然后将抠好图的脸部透明png模板铺在上面，完成。实际过程中需要处理一些小问题，比如要根据模板的面部尺寸将面部图像缩放到合适的尺寸；抠模板时要将边缘模糊处理，而且尽量保留模板本来的面部轮廓，只将五官抠掉。即便这样，合成结果还是很容易穿帮，不过纯前端处理也没有更好的办法了。

## **效果展示**

好了，说的再多不如看个例子，示例提供三种图片输入源，分别是本地图片、远程图片、内置示例。其中内置的图片大部分是提前在PS中纠正过角度的，而且内置图片会自动匹配到我事先调校好的参数，不出意外可以直接识别出人脸；如果选择本地图片作为图片源，最好选择头部姿态垂直的正面照，同时参考内置图片的 参数设置调节参数，一次识别不成功很正常，需要多调几次；也可以使用远程图片识别，但因为canvas受到跨域策略影响，远程图片只能识别不能提取和合成。

示例：[纯前端军装照合成](http://refined-x.com/projects/codes/tracking.html)

## **后记**

最初是抱着好奇的心态开始捣鼓这个项目的，虽然最终的合成效果远远达不到生产要求，但整个示例撸下来后对人脸识别和图片处理技术都有了基本的认识，对canvas操作中一些细节问题的解决也略微补足了一下这方面的知识空白，算略有收获吧。