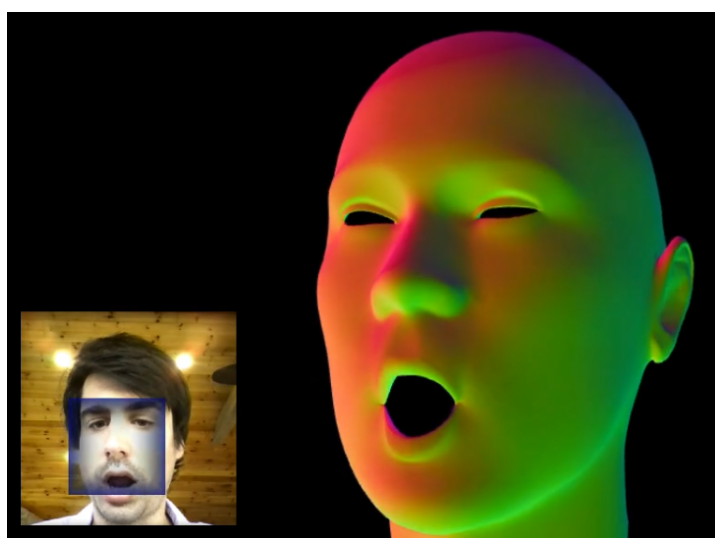


# JEEFACETRANSFERAPI

XAVIER BOURRY (JEELIZ)



## INTRODUCTION

*JEEFACETRANSFERAPI* is an API to detect the user emotions and the rotation of the head from a video stream. *JEEFACETRANSFERAPI* get the videostream, instantiate the neuron network and give in real-time user emotions. It is used by *WEBOJI* API. The 2 API have been separated because *JEEFACETRANSFER* could be used for other purpose than *webojis*. For example to detect if a client is satisfied or the mental state of a driver. *JEEFACETRANSFERAPI* does not require *THREE.js*.

## INTEGRATION

The application requires :

- The script **jeefacettransferNNC.js**,
- It is not launched directly, *WEBOJI* or other API should use *JEEFACE-TRANSFER*,
- A *<canvas>* element in the DOM. The video of the user with the detection window could be displayed on this canvas.

## ATTRIBUTES (READ ONLY)

- *JEEFACETRANSFERAPI.ready* : boolean, if the virtual fitter is ready or not

## METHODS

Pre-init methods

All these methods SHOULD be called before *JEEFACETRANSFERAPI.init()*.

- *JEEFACETRANSFERAPI.set\_size(<integer> widthPx, <integer> heightPx)* : This function SHOULD be called before every other call, even *JEEFACETRANSFERAPI.init()*. It sets the dimensions of the canvas in pixels,
- *JEEFACETRANSFERAPI.set\_audio(<boolean> isAudio)* : get the audio stream of the microphone. Default: *false*,
- *JEEFACETRANSFERAPI.onWebcamAsk(<function> callback)* : launch the callback just before asking for the webcam,
- *JEEFACETRANSFERAPI.onWebcamGet(<function> callback)* : launch the callback just after getting the webcam video stream,
- *JEEFACETRANSFERAPI.onContextLost(<function> callback)*: launch the callback if the webgl context is lost.

General methods

- *JEEFACETRANSFERAPI.init(<object> spec)*: Init the library. *spec* is a dictionary with the following properties :

- *<string> canvasId*: ID of the *<canvas>* element,
- *<string> NNCpath*: Where to find the neuron network model,
- *<string | object> NNC*: If *NNCpath* is not defined, JSON string content or parsed of the neuron network file,
- *<function> callbackReady(<string> errCode)* : callback function. If there is no error, *errCode* is set to *false*. See next section for the error codes,
- *<object> videoSettings*: dictionnary which overrides WebRTC MediaConstraints. It has the following properties:
  - \* *videoElement*: *<video>* element used. Not set by default. Useful to use a custom video. If specified, all other video settings are not used. A *<canvas>* or *<img>* element can also be provided,
  - \* *deviceId*: ID of the device, not set by default,
  - \* *facingMode*: default: *'user'*. to use the rear camera, set to *'environment'*,
  - \* *idealWidth*: ideal video width in pixels. default: *320*,
  - \* *idealHeight*: ideal video height in pixels. default: *240*,
  - \* *minWidth*: min video width in pixels. default: *240*,
  - \* *maxWidth*: max video width in pixels. default: *1280*,
  - \* *minHeight*: min video height in pixels. default: *240*,
  - \* *maxHeight*: max video height in pixels. default: *1280*.
- *JEEFACETRANSFERAPI.onLoad(<function> callback)*: launch the callback function if JEEFACETRANSFERAPI is ready, otherwise wait until it is ready,
- *JEEFACETRANSFERAPI.switch\_sleep(<bool> isSleep)*: Stop the detection and the rendering loop, to save resources. It should be called when the fitter/viewer is not displayed,
- *JEEFACETRANSFERAPI.get\_cv()*: return the DOM element of the video canvas,
- *JEEFACETRANSFERAPI.get\_videoStream()*: return the WebRTC raw video stream, useful to record the audio track,
- *JEEFACETRANSFERAPI.switch\_displayVideo(<boolean> isDisplayVideo)*: if true, display the video of the user on the DOM canvas element, with a marker to delimit the face. Can be used before the initialization of the API.
- *JEEFACETRANSFERAPI.on\_detect(<function> callback)*: Launch the callback function if the face is detected or when the detection is lost. The callback is called with 1 argument, *true* if the face is detected, *false* if the detection is lost,
- *JEEFACETRANSFERAPI.set\_animateDelay(<integer> delay)* : Change the delay between 2 detections. it can be helpful to free up some resources to speed up DOM transition or video encoding. The value is given in milliseconds,

- `JEEFACETRANSFERAPI.set_color([<float> R,<float> G,<float> B])`: set the color of the frame (if displaying the debug view only). R,G,B are between 0 and 1. Default is [0.0,0.5,1.0] (light blue).

#### Initialization error codes

These error codes can be returned as first argument of `callbackReady` :

- `false`: no error occurs,
- `ALREADY_INITIALIZED`: the API has been already initialized,
- `NO_CANVASID`: no canvas ID was specified,
- `INVALID_CANVASID`: cannot found the `<canvas>` element in the DOM,
- `WEBCAM_UNAVAILABLE`: cannot get access to the webcam (the user has no webcam, or it has not accepted to share the device, or the webcam is already busy),
- `GL_INCOMPATIBLE`: WebGL is not available, or this WebGL configuration is not enough (there is no WebGL2, or there is WebGL1 without `OES_TEXTURE_FLOAT` or `OES_TEXTURE_HALF_FLOAT` extension).

#### Morph and rotation

These methods are used by higher level scripts (for example `WEBOJI`) to get the morph coefficients and the rotation from the neural network.

- `JEEFACETRANSFERAPI.get_nMorphs()`: returns the number of morphs,
- `JEEFACETRANSFERAPI.get_morphTargetInfluences()`: returns the array with the morph coefficients,
- `JEEFACETRANSFERAPI.get_morphTargetInfluencesStabilized()`: returns the array with the morph coefficients stabilized,
- `JEEFACETRANSFERAPI.get_morphUpdateCallback()`: function launched when the morph coefficients array is updated. The function is called with these arguments :
  - `<float> quality`: quality of the detection, between 0 (bad quality) and 1 (high quality),
  - `<float> benchmarkCoeff`: higher it is, less powerful is the computer of the user. Can be directly used as a factor of the blending coefficients for morphing amortization,
- `JEEFACETRANSFERAPI.get_rotation()`: return an array with the 3 euler angles which characterize the head rotation,
- `JEEFACETRANSFERAPI.get_rotationStabilized()`: same than
- `get_rotation()` method but with more stabilization and automatic rotation if the head is not found,
- `JEEFACETRANSFERAPI.get_positionScale()`: return a 3 floats array,  $[P_x, P_y, s]$ .  $P_x$  and  $P_y$  are the 2D position coordinates (relative to viewport size, each between 0 and 1).  $s$  is the scale relative to the width (1 for full width).  $P_x$  and  $P_y$  are oriented respectively from left to right (in mirrored view) and from bottom to top.

- *JEEFACETRANSFERAPI.is\_detected()*: returns if the face is detected or not.