The Future of Communication on the Web

Adam Ullman
adam@tokbox.com
@aullman
github.com/aullman

tokbox

WebRTC

- WebRTC is a free, open project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs. The WebRTC components have been optimized to best serve this purpose.

- Our mission: To enable rich, high-quality RTC applications to be developed for the browser, mobile platforms, and IoT devices, and allow them all to communicate via a common set of protocols.

- The WebRTC initiative is a project supported by Google, Mozilla and Opera, amongst others.

Huh!?

Video Chat in the Browser without plugins! (and data channels)

The WebRTC project turns 5

Time to celebrate!

More info on WebRTC

- Samples https://webrtc.github.io/samples/
- Getting Started https://webrtc.org/start/
- OpenTok JS Getting Started: https://tokbox.com/developer/sdks/js/

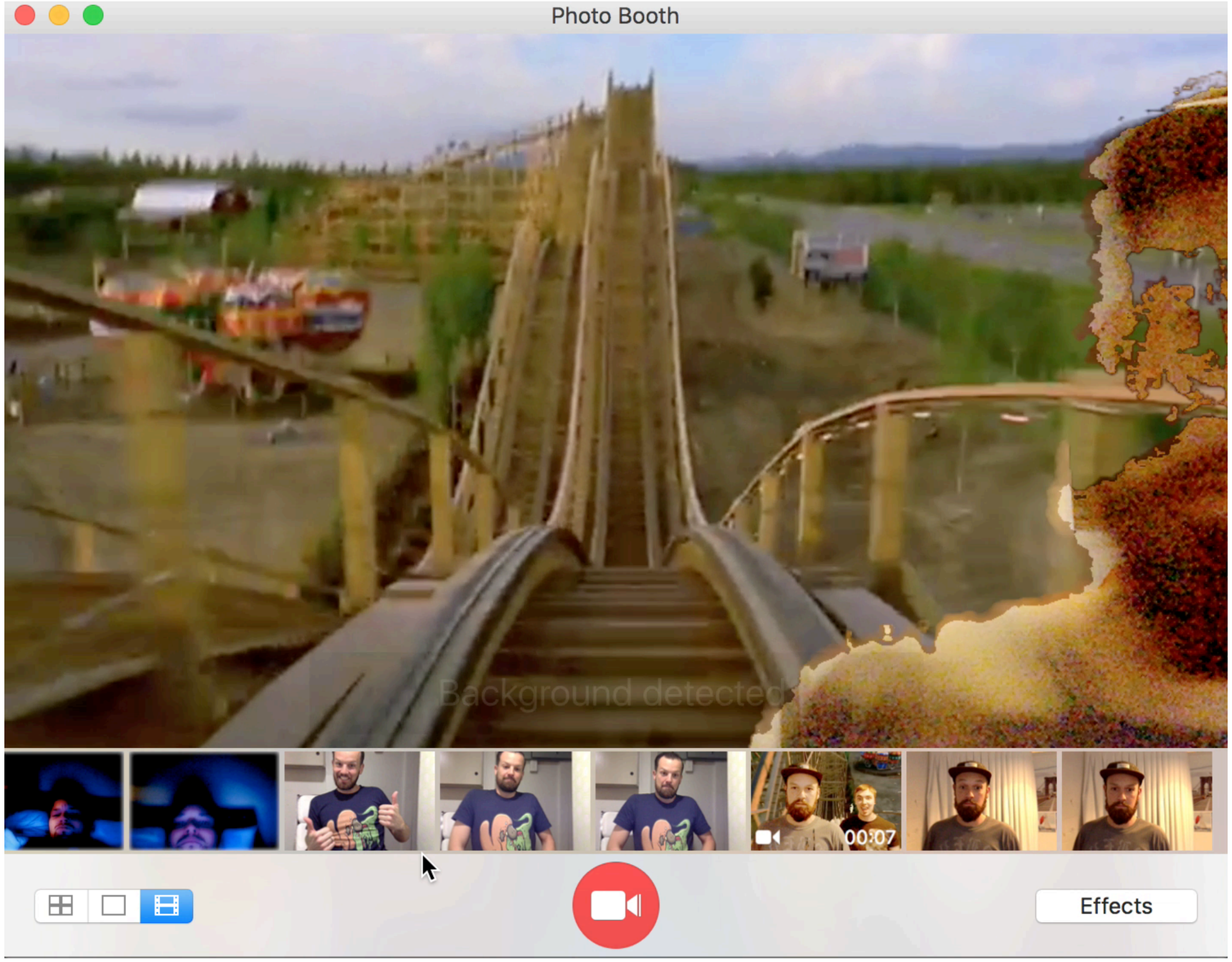
New Stuff

- **`VideoElement.captureStream()`** - stream the contents of a VideoElement
  - Firefox 47+
  - Chrome 53+ (with Experimental Web Platform Features enabled)
- **`Canvas.captureStream()`** - stream the contents of a Canvas
  - Firefox 47+
  - Chrome 52+
- **`MediaRecorder`** - record a stream to a file
  - Firefox 29+
  - Chrome 49+

Why?

Because I want to do this…

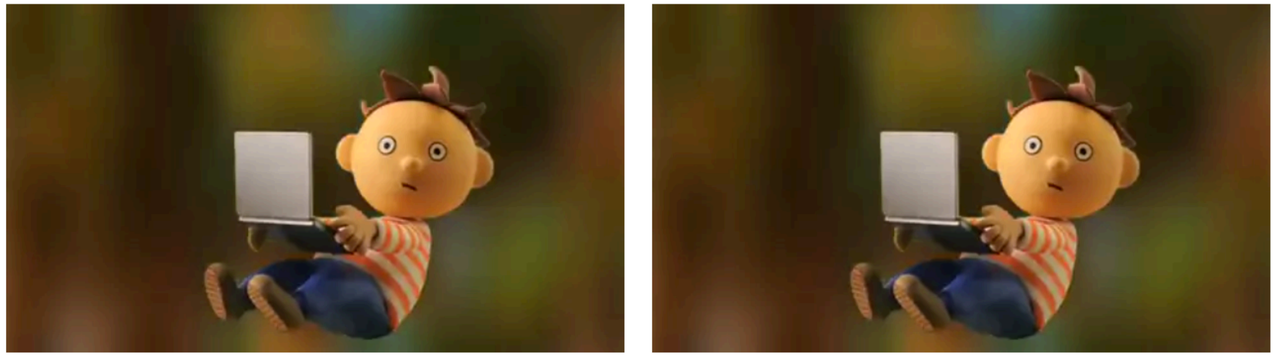
and this…



+ So much more

WebRTC samples

captureStream(): video to video



This demo requires Firefox 47, or Chrome 53 with **Experimental Web Platform features** enabled from chrome://flags.

`VideoElement.captureStream()`

webrtc.github.io/samples/src/content/capture/video-video/

VideoElement.captureStream()

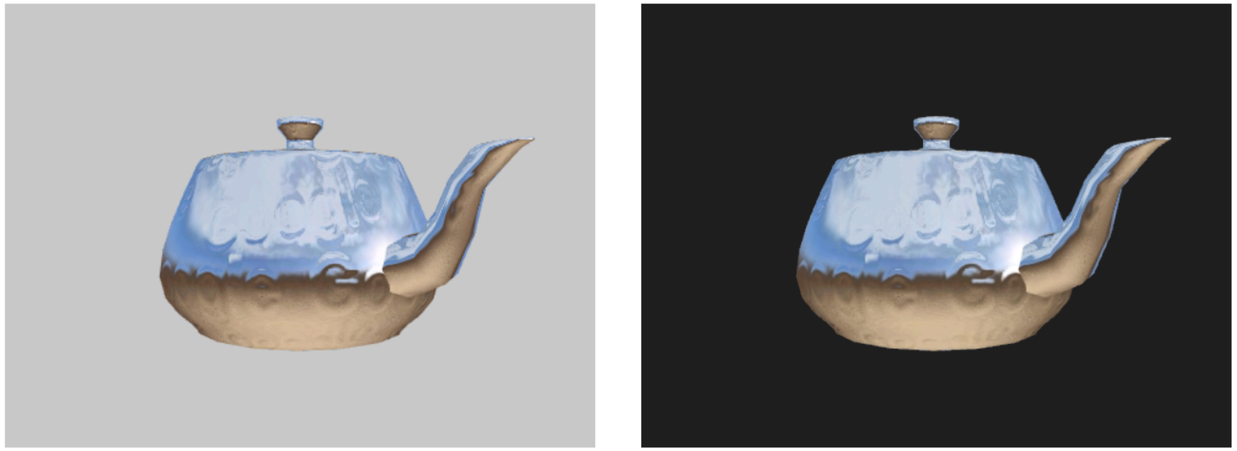
```
const leftVideo = document.querySelector('#leftVideo');
const rightVideo = document.querySelector('#rightVideo');

leftVideo.addEventListener('play', () => {
   rightVideo.srcObject = leftVideo.captureStream();
});

leftVideo.play();
```

WebRTC samples

Stream from canvas to video element



`Canvas.captureStream()`

`webrtc.github.io/samples/src/content/capture/canvas-video/`

Canvas.captureStream()

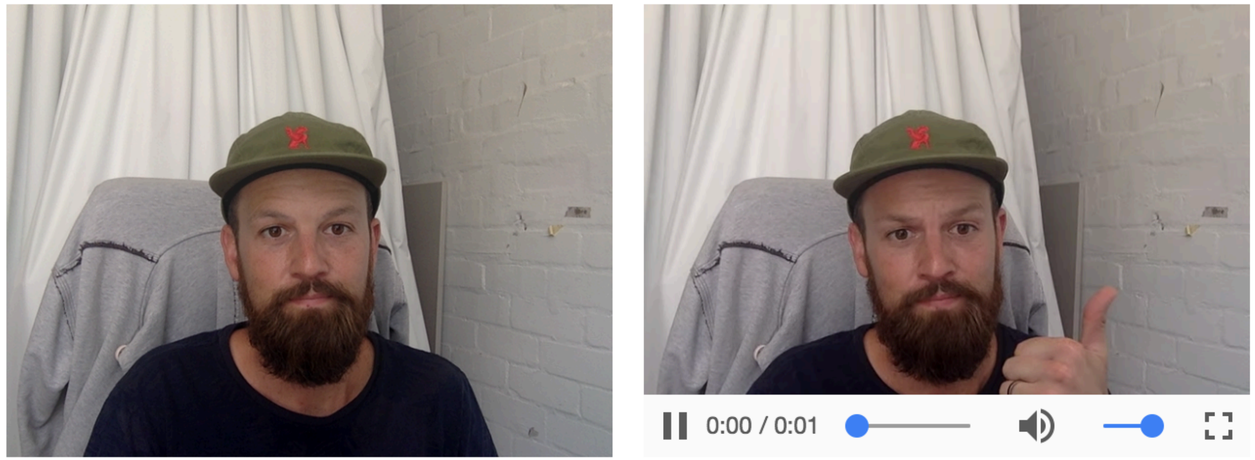
```
const canvas = document.querySelector('canvas');
const video = document.querySelector('video');

video.srcObject = canvas.captureStream();
```

WebRTC samples **MediaRecorder**

This demo requires Firefox 29 or later, or Chrome 47 or later with **Enable experimental Web Platform features** enabled from chrome://flags.

For more information see the MediaStream Recording API Editor's Draft.



MediaRecorder

webrtc.github.io/samples/src/content/getusermedia/record/

MediaRecorder

```
let mediaRecorder;
let recordedBlobs;

const gumVideo = document.querySelector('video#gum');
const recordedVideo = document.querySelector('video#recorded');

function startRecording() {
  recordedBlobs = [];
  const options = { mimeType: ‘video/webm;codecs=vp9' };
  mediaRecorder = new MediaRecorder(stream, options);
  mediaRecorder.addEventListener('dataavailable', event => {
    if (event.data && event.data.size > 0) {
      recordedBlobs.push(event.data);
    }
  });
  mediaRecorder.start(10); // collect 10ms of data
}

function stopRecording() {
  mediaRecorder.stop();
}

function play() {
  const superBuffer = new Blob(recordedBlobs, {type: 'video/webm'});
  recordedVideo.src = window.URL.createObjectURL(superBuffer);
}
```

Can I really make Snapchat in a browser?

What about filters?

getUserMedia

```
navigator.mediaDevices.getUserMedia({
  audio: true,
  video: true
}).then(stream => {
  const video = document.createElement('video');
  video.srcObject = stream;
  video.play();
});
```

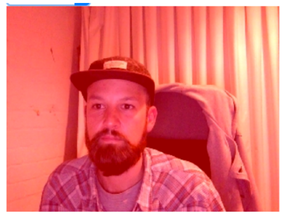
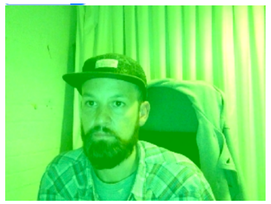
Render video in Canvas

```
const canvas = document.createElement('canvas');
const ctx = canvas.getContext('2d');
canvas.width = video.videoWidth;
canvas.height = video.videoHeight;
document.body.appendChild(canvas);

const drawLoop = () => {
  ctx.drawImage(video, 0, 0, 640, 480);
  requestAnimationFrame(drawLoop);
};
drawLoop();

const canvasStream = canvas.captureStream();
if (stream.getAudioTracks().length) {
  canvasStream.addTrack(stream.getAudioTracks()[0]);
}
```

colourShift

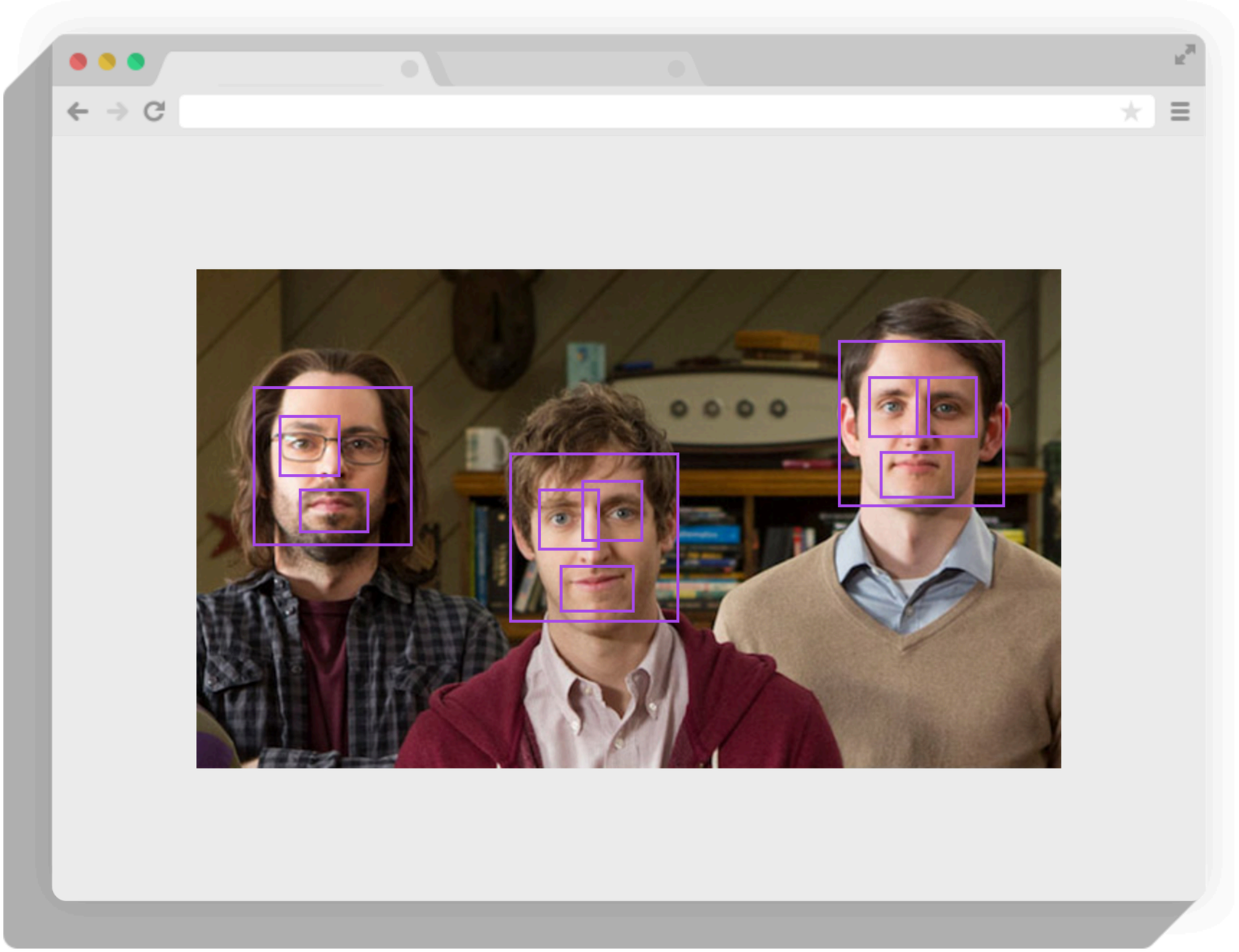

```
function redShift(imgData) {
  const res = new Uint8ClampedArray(imgData.data.length);
  for (let i = 0; i < imgData.data.length; i += 4) {
    res[i] = Math.min(255, imgData.data[i] + 50);
    res[i + 1] = imgData.data[i + 1];
    res[i + 2] = imgData.data[i + 2];
    res[i + 3] = imgData.data[i + 3];
  }
  const resData = new ImageData(res, imgData.width, imgData.height);
  return resData;
}
```

invert

```
const filter = imgData => {
  const res = new Uint8ClampedArray(imgData.data.length);
  for (let i = 0; i < imgData.data.length; i += 4) {
    res[i] = 255 - imgData.data[i];
    res[i + 1] = 255 - imgData.data[i + 1];
    res[i + 2] = 255 - imgData.data[i + 2];
    res[i + 3] = imgData.data[i + 3];
  }
  const resData = new ImageData(res, imgData.width, imgData.height);
  return resData;
};
```

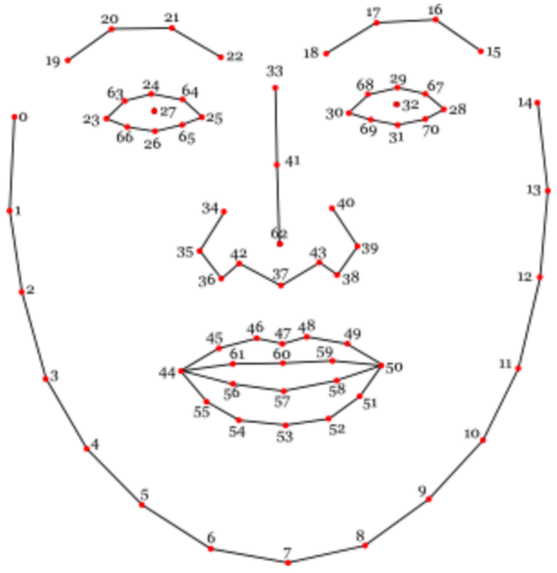
What about those cool face effects?

tracking.js – detect faces, eyes and mouths in a image



clmtrackr

clmtrackr is a javascript library for fitting facial models to faces in videos or images. It currently is an implementation of *constrained local models* fitted by *regularized landmark mean-shift*, as described in Jason M. Saragih's paper. **clmtrackr** tracks a face and outputs the coordinate positions of the face model as an array, following the numbering of the model below:



Comedy glasses

```
    if (!image) {
      image = document.createElement('img');
      image.src = 'http://localhost:8080/images/comedy-glasses.png';
    }
    ctx.drawImage(videoElement, 0, 0, canvas.width, canvas.height);
    const positions = ctracker.getCurrentPosition();
    if (positions && positions.length > 20) {
      const width = (positions[13][0] - positions[1][0]) * 1.1;
      const height = (positions[53][1] - positions[20][1]) * 1.15;
      const y = positions[20][1] - (0.2 * height);
      const x = positions[0][0];
      ctx.drawImage(image, x, y, width, height);
    }
```

Face deformation

Unwell

Facedeform preset

start

Face Deformation

Thanks!

Adam Ullman
adam@tokbox.com
@aullman
github.com/aullman

tokbox