# *FACE DETECTION* MENGGUNAKAN JAVASCRIPT SEBAGAI FILTER AWAL PRESENSI BERBASIS WEB

**Anggy Trisnawan Putra**
Jurusan Ilmu Komputer, FMIPA, Universitas Negeri Semarang
Email: anggy.trisnawan@mail.unnes.ac.id

***Abstrak***

*Presensi berbasis web merupakan salah satu alternatif praktis diantara berbagai macam metode aplikasi presensi pegawai. Foto presensi yang diambil menggunakan webcam komputer banyak yang tidak standar, misalnya wajah tidak lurus (tampak samping), hanya tampak sebagian saja, bahkan ada yang sengaja tidak menampakkan wajah. Fakta semacam ini menyebabkan pemrosesan lanjutan berupa pengenalan wajah (face recognition) maupun monitoring pimpinan menjadi tidak efisien. Pemanfaatan javascript sebagai gerbang untuk menjaga input foto pegawai supaya lebih tertib dapat dilakukan dengan cara menerapkan algoritma face detection sebelum pengambilan foto. Foto tidak akan diambil jika posisi wajah tidak sesuai dengan ketentuan, sehingga kualitas foto pegawai yang dikirim ke server menjadi lebih baik.*

***Kata kunci** : Face detection, Javascript, Presensi web*

## 1. Pendahuluan

Presensi merupakan salah satu cara untuk mendisiplinkan pegawai. Metode presensi dapat dilakukan dengan berbagai cara, mulai dari cara manual (tanda tangan di lembar presensi) sampai dengan otomasi via teknologi informasi. Beberapa teknologi yang dapat digunakan diantaranya adalah *fingerprint, webcam,* mesin absensi retina, dsb. Dengan adanya pemanfaatan teknologi informasi terbukti bahwa kecurangan-kecurangan yang terjadi pada metode presensi manual dapat dikurangi (Angraini, 2009).

Kecanggihan dari teknologi informasi ini rupanya tidak luput dari kelemahan. Misalnya saja untuk mesin *fingerprint,* alat pemindai sidik jari ini tidaklah tahan banting (mudah rusak) (Ginting, 2010). Selain itu, di beberapa kasus juga ditemui bahwa alat ini tidak dapat mendeteksi/mengenali sidik jari seseorang (Septiyaning, 2015). Selain itu, hasil penelitian pada (Angraini, 2009) mengatakan bahwa terdapat faktor-faktor yang menghambat penerapan teknologi *fingerprint* yaitu motivasi pengguna, instruksi penggunaan sistem yang minim, dan kecepatan waktu penggunaan sistem yang tidak efisien.

Presensi berbasis web merupakan salah satu alternatif yang murah dan praktis untuk memenuhi kebutuhan presensi pegawai. Selain dari fakta bahwa perangkatnya yang sangat terjangkau dan tidak mudah rusak, implementasinya pun

tidak serumit metode yang sudah disebutkan sebelumnya. Presensi ini hanya membutuhkan *webcam*, komputer yang terkoneksi dengan jaringan internet / intranet, dan browser yang akan menampilkan antarmuka aplikasi presensi untuk selanjutnya mengirim foto pegawai ke server setiap kali presensi dilakukan. Setelah itu, sistem dapat melakukan proses *face recognition* untuk menentukan apakah foto tersebut benar-benar foto pegawai yang ada. Proses ini dapat dilakukan secara manual dengan melibatkan atasan langsung dari pegawai yang bersangkutan tanpa bantuan sistem.

Proses *face recognition* baik manual maupun terkomputerisasi tentunya membutuhkan sumber foto yang bagus dan jelas. Padahal, pada kenyataannya di lapangan, tidak semua pegawai memposisikan wajahnya secara benar di depan *webcam.* Banyak diantara foto yang terkirim ke server tidak memadai, misalnya saja tertutup helm, menggunakan masker, blur/kabur, atau bahkan tidak ada wajah sama sekali. Tentu saja ini menghambat proses selanjutnya karena presensi dengan cara ini sangat bergantung pada input foto pegawai. Tanpa foto yang bagus, proses verifikasi sistem tidak akan berjalan dengan baik.


Gambar 1. Contoh foto presensi yang tidak baik (sumber : simpeg.unnes.ac.id)

Algoritma *face recognition* secara *realtime* seperti yang sudah diteliti di (Walgamage, 2014) menggunakan server *backend* yang melakukan proses secara otomatis dengan perintah antarmuka aplikasi pengambil foto. Di dalamnya juga dijelaskan bahwa proses *face detection* sangatlah penting dilakukan sebelum foto dikirim ke server untuk menjamin bahwa proses *face recognition* tidak berjalan sia-sia. Pada platform perangkat bergerak, misal android, *face detection* dilakukan menggunakan *native face detection API* yang sudah tersedia sejak android versi 4.0 (Walgamage, 2014).

*Face detection* pada platform PC (*personal computer*) menggunakan browser dapat dilakukan dengan menggunakan perangkat lunak openCV melalui *web service*. Foto dikirimkan dari

web browser menuju server yang terinstall openCV, lalu diproses untuk dikembalikan ke web browser dengan informasi koordinat wajah yang ditemukan (Rosebrock, 2015). Proses ini tentu membutuhkan *resource* yang tidak sedikit karena terdapat proses pengiriman ke server, proses *face detection*, lalu dikirim kembali untuk proses selanjutnya. Walaupun dapat diteruskan ke proses *face recognition* yang dapat juga dilakukan oleh open CV tanpa kembali ke web browser secara langsung, proses ini akan menjadi sia-sia ketika foto yang dikirim ternyata tidak memenuhi standar foto yang baik.

Javascript telah dikenal sebagai bahasa pemrograman yang tertanam di setiap browser. Bersama HTML dan CSS, Javascript merupakan salah satu komponen penting di dalam teknologi web saat ini (Rosebrock, 2015). Dengan kemampuannya yang dapat berjalan di sisi *client,* Javascript memiliki keunggulan yaitu dapat memanfaatkan *resource* masing-masing komputer *client* tanpa membebani server. Dengan memanfaatkan Javascript, algoritma *face detection* dapat diterapkan tanpa melibatkan *resource* server sama sekali. Dengan kemampuannya yang interaktif, Javascript dapat diatur supaya foto hanya dikirimkan pada saat Javascript dapat menemukan wajah pegawai yang melakukan presensi. Dengan begitu, dapat dipastikan bahwa foto yang terkirim ke server berkualitas baik.

**2. Kajian Pustaka**

Pada (Nash, 2013) dijelaskan algoritma *face detection* menggunakan web browser yang mengakses server terinstall openCV.


Gambar 2. Diagram algoritma *face detection* menggunakan openCV (Nash, 2013)

Langkah-langkah dalam proses tersebut di atas adalah sebagai berikut (Nash, 2013).

1. Pengguna mengunggah gambar.
2. Face-detect.php mengirim gambar yang terunggah ke face-detect.exe.
3. Face-detect.exe mendeteksi wajah dan menulis pesan log.
4. Face-detect.php membaca pesan log yang dihasilkan dan mengirim kembali dengan bentuk JSON untuk dibaca di antarmuka web.

Sama halnya dengan langkah di atas, proses serupa dilakukan di (Kurniawan, 2014), dengan menggunakan openCV, server mengolah data foto yang

masuk untuk diteliti apakah foto pegawai benar-benar sesuai dengan ID pegawai yang melakukan foto. Alangkah sayangnya jika proses yang sedemikian berat ini tidak didukung dengan kualitas foto yang memadai.

Tulisan ini fokus pada langkah sebelum aplikasi mengirim foto (mengunggah gambar) ke server. Library yang digunakan dalam tulisan ini adalah jQuery Face Detection Plugin (JFDP) yang ditulis oleh Jay Salvat (https://github.com/jaysalvat/jquery.facedetection).

### 2.1 *jQuery Face Detection Plugin* (JFDP)

JFDP ditulis oleh Jay Salvat dengan memanfaatkan algoritma *face detection* dari Liu-Liu (https://github.com/liuliu/ccv). JFDP mampu mendeteksi wajah yang ada pada gambar dan video lalu memberitahu koordinat dari wajah yang ditemukan. JFDP memiliki beberapa *callback*, salah satunya yang akan dimanfaatkan adalah *callback complete*. *Callback* ini akan berikan ketika JFDP selesai melakukan proses deteksi wajah pada foto.

```
<script>
  $('#picture').faceDetection({
    complete: function (faces) {
      console.log(faces);
    }
  });
</script>
```

Setelah JFDP selesai melakukan proses, *callback complete* memiliki satu variabel yaitu object wajah yang ditemukan. Apabila memang ditemukan wajah, variabel ini akan berisi informasi sebagai berikut.

1. x — koordinat X wajah pada foto
2. y — koordinat Y wajah pada foto
3. width — lebar wajah
4. height — tinggi wajah
5. positionX — posisi X relatif terhadap halaman web
6. positionY — posisi Y relatif terhadap halaman web
7. offsetX — posisi X relatif terhadap induk
8. offsetY — posisi Y relatif terhadap induk
9. scaleX — rasio antara lebar foto asli dengan lebar yang ditampilkan
10. scaleY — rasio antara tinggi foto asli dengan tinggi yang ditampilkan
11. confidence — tingkat kepercayaan diri

### 2.2 jQuery

jQuery digunakan untuk membantu dalam proses pengiriman foto ke server secara *asynchronous*. Contoh pengiriman data menggunakan jQuery adalah sebagai berikut (JQuery Community Experts, 2010):

```
$.ajax({
            type: ‘POST’,
            url: ‘hello-ajax.html’,
            dataType: 'html',
            success: function(html, textStatus) {
               $('body').append(html);
            },
            error: function(xhr, textStatus, errorThrown) {
               alert('An error occurred! ' + errorThrown);
            }
});
```

**2.3 HTML5 *Canvas***

Javascript dapat digunakan untuk mengakses kamera (*webcam*) untuk ditampilkan secara terus menerus (*streaming*) (Bidelman, 2012) di tag HTML5 video dengan cara sebagai berikut :

```
<video autoplay></video>
<script>
 var errorCallback = function(e) {
  console.log('Rejected!', e);
 };

 // Not showing vendor prefixes.
 navigator.getUserMedia({video: true, audio: true},          function(localMediaStream) {
  var video = document.querySelector('video');
  video.src = window.URL.createObjectURL(localMediaStream);
  // Note: onloadedmetadata doesn't fire in Chrome when using it with getUserMedia.
  // See crbug.com/110938.
  video.onloadedmetadata = function(e) {
   // Ready to go. Do some stuff.
  };
```

```
  }, errorCallback);
</script>
```

Tampilan pada kanvas HTML5 dapat diambil dengan fungsi toDataURL(). Contoh kodenya adalah sebagai berikut (Jenkov, 2014) :

```
var canvas  = document.getElementById("ex1");
var dataUrl = canvas.toDataURL();
```

**3. Metode**

Tulisan ini menggunakan pendekatan metode penelitian dan pengembangan (*research and development*) dalam hal pemanfaatan javascript sebagai filter foto presensi pegawai berbasis web. Javascript digunakan untuk menjalankan algoritma *face detection* dengan manipulasi/penggabungan berbagai macam teknik yang sudah ada.

**4. Hasil dan Pembahasan**

Penggabungan berbagai teknik di dalam HTML5, jQuery, dan Javascript telah berhasil dilakukan dan menghasilkan aplikasi presensi berbasis web yang telah mengandung filter *face detection.* Cara kerja aplikasi web presensi dapat dijabarkan dalam langkah-langkah sebagai berikut:

Javascript digunakan untuk mengakses *webcam* sehingga dapat menangkap gambar dan ditampilkan secara terus-menerus pada tag HTML5 video. Menggunakan teknik menggambar pada kanvas HTML5, salin dan tempel gambar yang ada di tag video ke dalam tag HTML5 *canvas* secara terus menerus (dalam tulisan ini dibuat setiap 33 milidetik). Pada saat pegawai menekan tombol "presensi" atau semacamnya, sebuah kode javascript dijalankan. Kode ini mengambil gambar pada kanvas lalu diletakkan pada tag HTML img (dibuat *hidden*), selanjutnya dibaca oleh JFDP untuk dilakukan proses *face detection.*

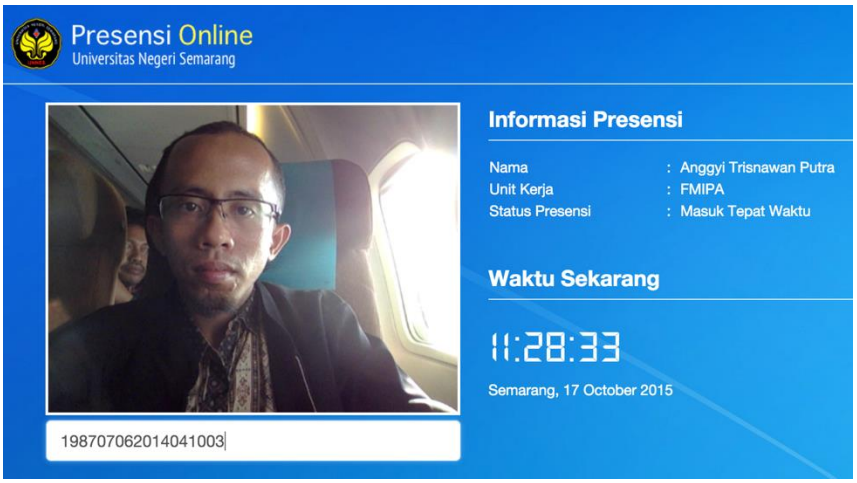
Apabila proses *face detection* selesai, cek apakah wajah ditemukan. Jika iya, lanjutkan proses presensi dengan mengirim gambar/foto ke server menggunakan jQuery. Apabila wajah tidak ditemukan, aplikasi presensi akan menampilkan informasi supaya pegawai memposisikan wajahnya dengan benar.

Proses presensi berbasis web yang disajikan dalam tulisan ini berfokus pada proses *face detection* sebagai filter sebelum foto pegawai yang akan dikirim ke server. Dengan adanya filter ini, foto pegawai yang terkirim ke server dapat dipastikan berkualitas baik. Berikut ini

adalah contoh pengambilan foto pegawai yang tidak baik.


Gambar 3. Presensi yang gagal karena wajah tidak tampak seluruhnya

Pada gambar di atas jelas terlihat bahwa wajah yang tidak tampak secara keseluruhan tidak dapat diterima oleh algoritma *face detection.* Jadi, pegawai yang melakukan presensi akan berusaha menempatkan/memposisikan wajahnya sampai dapat diterima oleh sistem. Berikut ini adalah contoh presensi yang berhasil.




Gambar 4. Presensi yang berhasil dilakukan karena wajah tampak jelas

Pada gambar di atas terlihat bahwa presensi yang dilakukan dengan wajah yang lurus dan jelas dapat diterima oleh sistem. Selanjutnya, foto akan dikirimkan ke server presensi dengan kualitas yang baik untuk diproses sesuai dengan kebutuhan sistem.

## 5. Kesimpulan

Javascript telah berhasil diterapkan sebagai media filter foto presensi pegawai berbasis web dengan mengaplikasikan

algoritma *face detection.* Dengan menerapkan metode ini, terbukti bahwa pegawai yang presensi tidak bisa lagi semaunya sendiri. Pegawai harus menempatkan wajahnya sedemikian rupa sehingga *webcam* dapat menangkap wajah secara utuh dan jelas. Dengan begitu, harapannya pegawai dapat lebih tertib di dalam melakukan presensi setiap harinya.

## 6. Daftar Pustaka


Angraini. 2009. Audit Implementasi Biometrics Fingerprint (Studi Kasus Sistem Presensi Stmik Amikom Yogyakarta). Prosiding Seminar Nasional Aplikasi Teknologi Informasi 2009 (SNATI 2009). Yogyakarta, 20 Juni 2009

Bidelman, E. 2012. *Capturing Audio & Video in HTML5*. http://www.html5rocks.com/en/tutorials/getusermedia/intro/, diakses 16 Oktober 2015

Ginting, K. 2010. Mesin Absensi "Fingerprint" Rusak. http://www.kompasiana.com/kornelius ginting/mesin-absensi-finger-print-rusak_550058478133111918fa75e6, diakses 16 Oktober 2015

Jenkov, J. 2014. HTML5 Canvas: toDataURL(). http://tutorials.jenkov.com/html5-canvas/todataurl.html, diakses 16 Oktober 2015

jQuery Community Experts. 2010. jQuery Cookbook. O'Reilly Media, Inc., California

Kurniawan, L.M. 2014. Metode Face Recognition untuk Identifikasi Personil Berdasar Citra Wajah bagi Kebutuhan Presensi Online Universitas Negeri Semarang. Scientific Journal of Informatics. Vol 1(2): 210-220.

Nash. 2013. Web based interface for face detection with OpenCV. http://opencv-code.com/projects/web-based-interface-for-face-detection-with-opencv/, diakses 16 Oktober 2015.

Rosebrock, A. 2015. Creating a face detection API with Python and OpenCV (in just 5 minutes). http://www.pyimagesearch.com/2015/05/11/creating-a-face-detection-api-with-python-and-opencv-in-just-5-minutes/, diakses 16 Oktober 2015.

-------. 2015. Javascript. https://en.wikipedia.org/wiki/JavaScript, diakses 16 Oktober 2015

Septiyaning W., I. 2015. Mesin Presensi Minim, PNS Solo Mengeluh ke BKD. http://www.solopos.com/2015/04/10/disiplin-pns-mesin-presensi- minim-pns-solo-mengeluh-ke-bkd-593435, diakses 16 Oktober 2015

Walgamage, T., Farook, C. 2014. A Real-time Hybrid Approch for Mobile Face Recognition. 2014 Fifth International Conference on Intelligent Systems, Modelling and Simulation. Malaysia, 27 Jan-29 Jan 20