FINGER TRACKING UNTUK INTERAKSI PADA VIRTUAL KEYBOARD

Raga Mukti Alhaqqi
Jurusan Teknik Informatika, Nana Ramadijanti, Setiawardhana
Politeknik Elektronika Negeri Surabaya
Institut Teknologi Sepuluh Nopember Surabaya
Kampus PENS-ITS Keputih Sukolilo Surabaya 60111
Telp (+62)31-5947280, 5946114, Fax. (+62)31-5946114
Email: muktiraga_it07@yahoo.co.id

Abstrak

Secara konservatif, interaksi antara manusia dengan komputer saat ini masih menggunakan mouse, keyboard, dan layar monitor. Melalui Proyek Akhir ini, akan diberikan suatu alternatif substitusi keyboard konvensional dengan kamera video (webcam). Kamera digunakan sebagai sensor untuk menelusuri pergerakan atau perilaku jari tangan. Selanjutnya, perilaku jari tangan ini diterjemahkan dalam aksi keyboard. Sistem ini kemudian dinamakan dengan virtual keyboard.

Pendeteksian jari tangan menggunakan haar cascade pada library OpenCV, selanjutnya tracking pergerakan jari tangan diterapkan dengan metode Kalman Filter, yaitu digunakan untuk memprediksi posisi jari tangan pada frame selanjutnya.

Kata kunci : Open CV, Haar Cascade, Virtual Keyboard, Kalman Filter, Tracking**.**

1. Latar Belakang

Kemajuan teknologi informasi yang semakin pesat telah memaksa manusia untuk berusaha mengikutinya Teknologi tersebut dapat digunakan oleh semua kalangan yang dapat memanfaatkannya untuk berbagai keperluan. Teknologi tersebut memudahkan mereka dalam memenuhi kebutuhan dengan lebih cepat, lebih efisien, lebih mudah dan akurat.

Keyboard sebagai input device pada komputer merupakan alat yang sangat dibutuhkan pengguna komputer untuk berinteraksi. Dalam sebuah ruangan, pengguna ingin berinteraksi dengan komputer dengan layar yang lebar dan besar, sedangkan pengguna berada pada tempat yang jauh dari komputer dan keyboard tidak mungkin untuk dijangkau. Oleh karena itu dibutuhkan suatu aplikasi yang dapat menggantikan fungsi dari sebuah keyboard dan dapat di jangkau dimana saja, di depan camera webcam pengguna bisa menggunakan ujung jarinya atau tip pointer untuk mengetikkan beberapa teks atau berinteraksi dengan komputer.

2. Dasar Teori

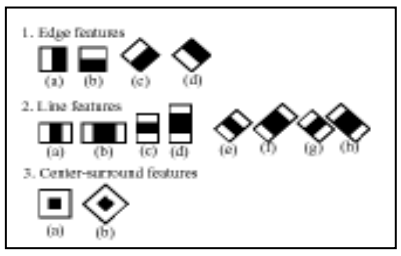
a. Object Detection

Metode yang digunakan untuk melakukan *object detection* pada proyek akhir ini adalah *Haar Classifier*, yaitu metode object detection yang membangun sebuah boosted rejection cascade, yang akan membuang data training negative sehingga didapat suatu keputusan untuk menentukan data positif.

Haar Classifier merupakan metode supervised learning, yaitu membutuhkan data training untuk dapat mendeteksi obyek-obyek tertentu. Untuk itu, *Haar Classifier* membutuhkan data positif (obyek yang akan dideteksi) dan data negatif (bukan obyek yang akan dideteksi).

b. Haar Training

Setiap haar fitur terdiri dari gabungan kotak-kotak hitam dan putih



Gambar Haar Fitur

Tiga (3) tipe kotak (rectangular) fitur:

- Tipe two-rectangle feature
- Tipe three-rectangle feature
- Tipe four-rectangle feature

Nilai haar-like feature adalah perbedaan antara jumlah nilai-nilai piksel gray level dalam daerah kotak hitam dan daerah kotak putih:

$f(x) = \text{sum}_{\text{black rectangle}}(\text{pixel gray level}) - \text{sum}_{\text{white rectangle}}(\text{pixel gray level})$

kotak haar-like feature dapat dihitung secara cepat menggunakan “integral image”.

c. Cascade Classifier

Cascade classifier adalah sebuah rantai stage classifier, dimana setiap stage classifier digunakan untuk mendeteksi apakah di dalam image sub window terdapat obyek yang diinginkan (object of interest).

Untuk mendapatkan nilai cascade, perlu dilakukan training cascade. Classifier dengan banyak fitur akan mendapatkan tingkat pendeteksian yang lebih tinggi serta error yang rendah, namun akan berdampak pada waktu yang dibutuhkan untuk penghitungannya. Adapun atribut-atribut yang perlu diubah dan dicari nilai optimumnya adalah sebagai berikut:

- jumlah stage,
- jumlah feature, dan
- nilai threshold.

Tiap stage mengurangi tingkat false positif serta meningkatkan tingkat pendeteksian. Sedangkan

jumlah stage akan terus ditambahkan hingga target pendeteksian dan tingkat false positifnya ditemukan. Pada Gambar diatas, dapat kita lihat bahwa tiap sub-window yang bernilai negatif (bukan obyek yang dicari) akan langsung disisihkan dan secara otomatis tidak akan melanjutkan proses pencarian pada sub-window tersebut. Sedangkan untuk sub-window image yang bernilai positif akan terus dicek lagi pada stage berikutnya hingga ditemukan obyek yang dimaksud.

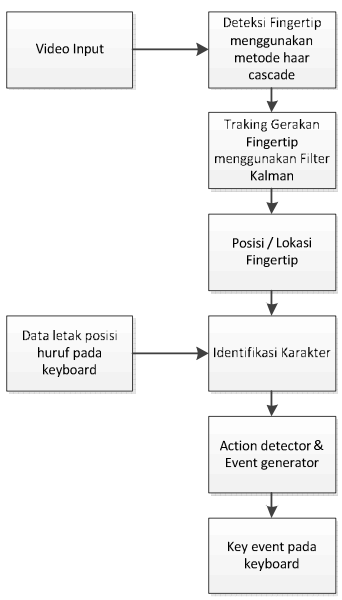
d. Objek Tracking

Metode filter kalman mangasumsikan bahwa informasi terbaik diperoleh dengan estimasi. Tujuannya adalah untuk menggunakan pengukuran yang diamati dari waktu ke waktu yang mengandung noise dan menghasilkan nilai-nilai yang cenderung lebih dekat dengan nilai sebenarnya.

Filter Kalman merupakan rekursif estimator, yang berarti hanya kondisi estimasi dari waktu langkah sebelumnya dan pengukuran pada saat ini yang diperlukan untuk menghitung perkiraan kondisi pada saat ini.

Filter Kalman memiliki dua tahap, yaitu **Predict** dan **Update**. Fase predict menggunakan perkiraan kondisi dari timestep sebelumnya untuk menghasilkan perkiraan kondisi di timestep saat ini. Pada tahap update prediksi pior saat ini dikombinasikan dengan informasi observasi saat ini untuk memperbaiki perkiraan kondisi.

3. Perancangan dan Pembuatan Sistem



Gambar Blok diagram sistem

Perancangan sistem ini dimulai dari training objek jari menggunakan algoritma Haar cascade yang di lakukan dengan menggunakan OpenCV, perancangan deteksi ini secara murni menggunakan tool yang telah disediakan oleh Haar Cascade.
Pengolahan citra dilakukan dengan training image pada algoritma haar training. Dengan keterangan sebagai berikut:

- Mempunyai 2400 data gambar positif masing-masing objek, yang dimaksud gambar positive adalah gambar yang didalamnya terdapat gambar suatu objek yang akan dideteksi. Seperti, gambar jari. Kemudian mempunyai 2500 gambar negatif ,yang dimaksud dengan gambar negative adalah suatu gambar yang didalamnya tidak terdapat suatu objek yang akan dideteksi dan gambar tersebut bertipe grayscale. Kemudian dilakukan crop gambar objek pada gambar positif sesuai dengan objek yang akan dideteksi.
- Setelah keseluruhan proses cropping selesai maka akan didapatkan nilai vector gambar tesebut, kemudian dilakukan proses training haar. Taining haar ini memerlukan waktu yang cukup lama, berdasarkan jumlah gambar dan jumlah stage yang terdapat pada training. Semakin banyak jumlah gambar dan semakin besar nilai stage, maka hasil deteksi objek yang di dapatkan juga semakin baik. Setelah training ini selesai, maka akan didapatkan nilai variable cascade. Nilai cascade tersebut kemudian dikonvert menjadi xml database. dengan menggunakan nilai Xml tesebut pada program. Maka program akan dengan mudah mendeteksi objek yang terdapat pada gambar.

Data Positif

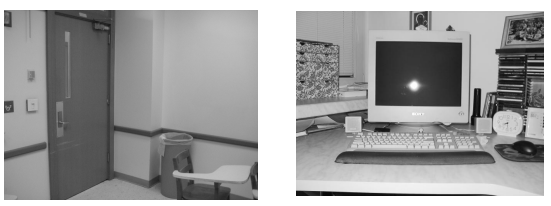
Data positif adalah sekumpulan data Gambar yang mengandung obyek yang akan dideteksi. Sebagai contoh, objek jari.

Selanjutnya dari sekumpulan gambar positif tersebut dilakukan data bounding atau croping terhadap objek spesifik yang akan dideteksi.

Objek yang telah di bounding menggunakan objectmaker (tools pada haar cascade), apabila keseluruhan gambar telah di bounding menggunakan objectmaker, maka akah menghasilkan data yang tersimpan di dalam file text, data tersebut merupakan komposisi dari titik – titik setiap gambar yang telah di bounding

Data Negatif

Data negative adalah sekumpulan gambar yang tidak mengandung obyek yang akan dideteksi, Sedangkan gambar di bawah ini merupakan beberapa contoh data yang dijadikan sample data negatif. Bisa dilihat dari beberapa gambar tersebut bahwa di dalam gambar tidak mengandung obyek positif (jari).



Sample Set Data Positif

Langkah selanjutnya adalah menciptakan sample data set positif untuk training data dengan menggunakan file createsample.cpp. Adapun parameter-parameter yang digunakan dalam createsample.cpp adalah sebagai berikut:

- ❖ Infoname – nama file text yang menampung data-data obyek image positif dan deskripsi dari obyek yang telah ditandai.
- ❖ Num – jumlah sample data positif
- ❖ Winwidth – lebar sample
- ❖ Winheight – tinggi sample

Pada tahap ini, dengan menggunakan file createsample.cpp kita akan membuat file vector yang merupakan hasil proses dari file text, untuk kemudian disimpan ke dalam file .vec.

Training Data

Tahap selanjutnya adalah training data untuk menghasilkan sebuah cascade classifier yang berisi sejumlah stage yang didalamnya terdapat kumpulan data classifier. Untuk melakukan proses training, gunakan file haartraining.cpp.

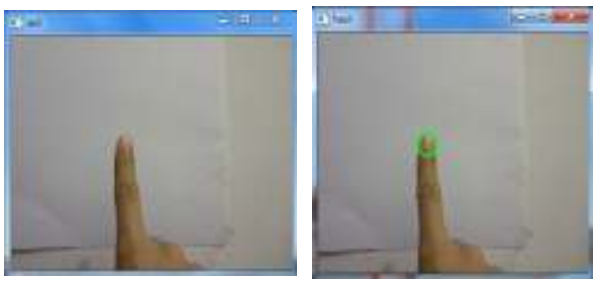
Convert Cascade ke file xml

Untuk memudahkan proses pembacaan pada program, openCv menggunakan file xml agar dapat membaca cascade classifier yang telah dihasilkan pada proses training. Oleh karena itu folder cascade classifier tersebut harus di ubah dahulu ke dalam bentuk xml menggunakan convert.exe. yang telah disediakan oleh openCv.

Setelah training selesai maka file xml akan menjadi suatu database objek deteksi. Kamera akan menangkap Gambar kemudian aplikasi melakukan load database pada xml yang sudah terbuat maka aplikasi akan mencari objek yang akan dideteksi .

Deteksi Objek

Langkah berikutnya adalah menerapkan cascade classifier yang telah di ubah ke bentuk xml pada program utama. Untuk melakukan pendeteksian objek pada tiap frame yang tertangkap kamera, OpenCv menyediakan fungsi cvHaarDetectObjects.



hasil deteksi objek ujung jari menggunakan Haar Cascade, sebelah kiri adalah video gambar asli sedangkan gambar yang kanan adalah video hasil deteksi terhadap objek ujung jari (lingkaran warna hijau pada ujung jari).

Tracking objek

Tracking *object* menggunakan metode Kalman Filter, filter kalman merupakan estimator rekursif. Untuk menggunakan Kalman Filter pada proses pelacakan, diasumsikan pergerakan *object* antar*frame* adalah konstan. Proses pelacakan dengan Kalman Filter ini sangat erat kaitannya dengan tahap deteksi. Setelah setiap *object* dalam *frame-frame* menghubungkan urutan citra yang telah terdeteksi di setiap *frame*. Ini bertujuan untuk menentukan identifikasi dan lokasi dari suatu *object* di titik-titik berbeda dalam suatu urutan citra.

Dengan Kalman Filter, setiap *object* yang dilacak dinyatakan dengan atribut-atributnya seperti posisi, kecepatan, atau ukurannya yang secara kolektif disebut sebagai status dari *object* tersebut. Metode Kalman Filter ini menggunakan informasi dari *object* terdeteksi di suatu *frame* dan status *object* tersebut di *frame* sebelumnya untuk membuat suatu perkiraan dari status baru *object* tersebut. Sebagai perumpamaan dalam proyek akhir ini, posisi ujung jari pada *frame* sebelumnya bisa digunakan untuk mendapatkan perkiraan kasar dari posisi ujung jari di *frame* setelahnya.

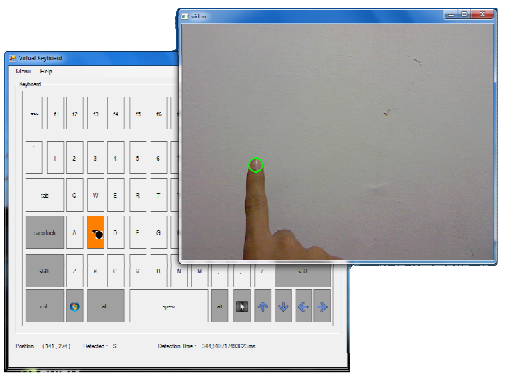
Action Detection

Setelah obejek fingertip terdeteksi dan pointer keyboard telah berada di saatu area huruf tertentu maka jika user ingin menekan huruf tersebut di simulasikan dengan cara menggerakkan finger seperti halnya menggerakkan finger pada waktu mengetik di keyboard konvnsional. Pada saat menggerakkan finger terdapat perubahan posisi x,y di area huruf tertentu, kemudian dilakukan penghitungan terhadap jarak antara posisi finger sebelum bergerak dan posisi finger setelah bergerak, penghitungan jarak menggunakan rumus *Euckidean Distance* $\sqrt{(x2-x1)^2+(y2-y1)^2}$.

4. Uji Coba dan Analisa

4.1 Hasil Running Program

Pada aplikasi ini terdapat dua window yang ditampilkan, yaitu menampilkan video hasil tracking terhadap objek jari dan gambar layout virtual keyboard beserta pointernya yang secara real time mengikuti gerak jari seorang user. Jika posisi ujung jari user terletak pada posisi area tombol pada keyboard maka system akan mendeteksinya sebagai sebuah karakter yang diindikasikan dengan berubahnya warna background menjadi warna orange.



Pengujian Terhadap Cahaya

Pengujian ini menitik beratkan pada aspek cahaya pada saat proses image processing. Faktor cahaya sangat berpengaruh besar pada tingkat keberhasilan deteksi. Pada percobaan kali ini cahaya di bagi menjadi tiga bagian waktu, yakni gelap, normal, terang. Dan jumlah percobaan yang akan dilakukan sebanyak 10 kali percobaan.

Tabel 4.1. Pengujian terhadap cahaya

Percobaan	Gelap	Normal	Terang
1	X	V	V
2	X	V	V
3	X	V	V
4	X	V	X
5	X	V	V
6	X	V	V
7	X	V	X
8	X	V	X
9	X	V	V
10	X	V	V
Rata-rata	**0%**	**100%**	**80%**

Dari tabel pengujian cahaya di atas dapat dilihat bahwa untuk menghasilkan deteksi yang baik di butuhkan pencahayaan yang cukup.

Pengujian Terhadap Jarak Objek dengan Kamera

Pengujian ini mengacu pada jarak antara jari dengan kamera. Dimana jarak sangat berpengaruh terhadap pendeteksian objek, setelah melakukan beberapa pengujian, diharapkan dapat memperoleh jarak yang efektif antara objek dengan kamera sehingga proses pendeteksian dapat dilakukan dengan cepat.

Tabel 4.2. Pengujian terhadap jarak

Percobaan	10 cm	20 cm	30 cm	40 cm	50 cm
1	V	V	V	X	X
2	V	V	V	V	X
3	V	V	V	X	X
4	V	V	V	X	X
5	V	V	V	V	X
6	V	V	V	X	X
7	V	V	V	V	X
8	V	V	V	X	X
9	V	V	V	X	X
10	V	V	V	V	X
Rata-rata	**100%**	**100%**	**100%**	**40%**	**0%**

Dari tabel pengujian jarak di atas dapat disimpulkan bahwa jarak terbaik antara objek dengan kamera adalah berjarak 10 – 30 cm.

Pengujian Terhadap Kecepatan Gerakan Jari

Kecepatan user saat menggerakkan jari sangat berpengaruh terhadap baik buruknya deteksi. Sehingga perlu dilakukan pengujian terlebih dahulu agar menghasilkan kesimpulan kecepatan gerak jari yang ideal untuk menggunakan aplikasi ini.

Tabel 4.3. Pengujian terhadap kecepatan gerakan.

Percobaan	Diam	Sedang	Cepat
1	V	V	X
2	V	V	X
3	V	V	V
4	V	V	X
5	V	V	X
6	V	V	V
7	V	X	X
8	V	V	X
9	V	V	V
10	V	V	V
Rata-rata	**100%**	**90%**	**40%**

Dari tabel 4.3. terlihat bahwa semakin cepat user menggerakkan jari, maka nilai keberhasilan deteksi semakin kecil.

Pengujian Terhadap Kemiringan Posisi Jari

Gambar 4.2: a) 0^o b) 45^o c) 90^o d) 180^o

Dari gambar 4.2 terlihat bahwa pada sudut 0 dan 45 objek dapat dikenali, pada sudut 90 atau 180 (posisi terbalik dengan posisi webcam) objek tidak dapat dikenali lagi.

Pengujian Terhadap Jumlah Jari lebih dari Satu

Gambar 4.3: Percobaan dengan jari berjumlah lebih dari 1.

Dari gambar 4.3 dapat dilihat bahwa untuk jari yang lebih dari 1 tidak semuanya objek dapat dideteksi, jumlahnya bervariasi. Hal ini disebabkan oleh bermacam-macam faktor diantaranya sudut kemiringan jari, intensitas cahaya dll.

Tracking Objek

Pelacakan objek berupa posisi koordinat fingertip yang telah terdeteksi pada sebelumnya, pelacakan menggunakan metode filter kalman adalah

melakukan estimasi terhadap posisi fingertip di frame sesudahnya dengan melihat dari frame dan frame-1. Hasil pelacakan seperti pada gambar 4.4 dibawah ini :

Gambar 4.4: Contoh hasil pelacakan fingertip

Pada gambar 4.4 diatas terlihat bahwa warna biru adalah hasil prediksi dan warna merah adalah hasil nilai measurement sedangkan garis dengan warna kuning adalah hasil pelacakan pergerakan fingertip.

Keberhasilan pelacakan diukur dari seberapa dekat nilai prediksi yang didapat dari proses *prediction* pada *frame* tn dengan nilai *measurement* (nilai sebenarnya pada proses *frame* tn+1). Kedua proses tersebut adalah perilaku *prediction*-update dalam Kalman Filter.

Dalam tabel 4.4 berikut adalah hasil percobaan dengan mengambil 10 sample untuk mengukur ketepatan metode kalman filter dalam melakukan tracking terhadap objek.

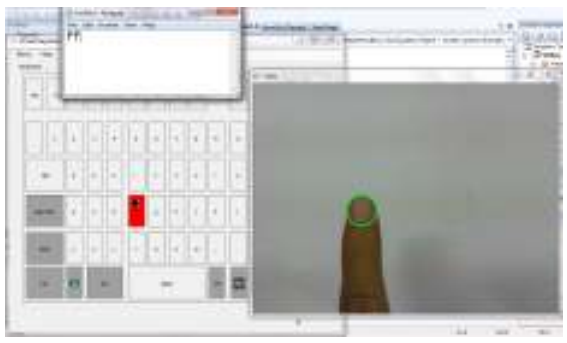
Tabel 4.4. Perbandingan hasil pelacakan dengan Filter Kalman

Percobaan	Real X	Real Y	Prediksi X	Prediksi Y	Error X %	Error Y %
1	164	113	172	109	4,65	3,54
2	162	114	169	109	4,14	4,39
3	158	114	167	109	5,39	4,39
4	126	115	128	115	1,56	0
5	190	94	190	95	0	1,05
6	98	68	97	67	1,02	1,47
7	91	66	93	66	2,15	0
8	130	80	130	86	0	6,98
9	134	91	131	89	2,24	2,2
10	144	86	154	84	6,49	2,33

Keakuratan prediksi Kalman Filter cukup memuaskan dengan nilai kesalahan terbesar hanya mencapai 6,98% dan nilai terkecil adalah 0%.

Action Detection

Pada aplikasi virtual keyboard, user dapat menggerakkan fingertip untuk menekan suatu huruf. Dalam gambar 4.9 adalah tampilan program virtual keyboard saat dijalankan, terdapat gambar layout keyboard sebagai petunjuk bagi user dan video yang menampilkam objek fingertip yang berhasil dideteksi.



Gambar 4.9: Program Virtual Keyoard dijalankan

5 Kesimpulan dan Saran

KESIMPULAN

Pada bagian ini akan di ulas tentang kesimpulan dan analisa dari seluruh percobaan proyek akhir Finger Tracking untuk Interaksi pada Virtual Keyboard.
Berikut beberapa kesimpulan yang dapat diambil dari percobaan dan pengujian proyek akhir ini :

1. Keberhasilan pada proses training data akan menentukan tingkat keberhasilan dalam deteksi objek.
2. Semakin banyak jumlah data training dan jumlah stages dalam cascade clasifier maka akan semakin akurat dalam pendeteksian objek tetapi akan membutuhkan waktu training yang lebih lama, dalam proyek akhir ini digunakan 2400 gambar positif dan 2500 gambar negatif.
3. Hasil pengujian deteksi objek sangat dipengaruhi oleh kondisi cahaya dan jarak objek terhadap webcam.
4. Jarak ideal posisi objek dengan webcam yang dapat terdeteksi adalah antara 10 – 40 cm.
5. Kecepatan gerakan objek berpengaruh terhadap keberhasilan dalam deteksi objek, gerakan yang terlalu cepat menghasilkan deteksi yang sangat lemah.
6. Pegerakan objek sangat berpengaruh dalam proses pelacakan karena metode yang digunakan hanya dapat bekerja baik pada benda dengan kecepatan (*velocity*) yang statis.

SARAN

Dalam penyempurnaan proyek akhir ini disarankan untuk perbaikan pada beberapa bagian. Diharapkan proyek akhir ini dapat diteruskan agar didapatkan hasil yang maksimal. Perbaikan dalam penyempurnaan yang dimaksudkan diantaranya adalah :

1. Untuk membuat aplikasi yang lebih aplikatif, diharapkan dalam proyek akhir ini, system dapat mendeteksi 10 jari ujung jari tangan.
2. Proyek akhir ini diharapkan dapat mendeteksi objek dalam berbagai jarak dan posisi terhadap kamera agar menjadi aplikasi yang lebih dinamis.

6 Daftar Pustaka

1. Zhengyou Zhang, Ying Wu, Ying Shan, Steven Shafer, *Visual Panel : Virtual Mouse, Keyboard and 3D Controller with an Ordinary Piece of Paper*..
2. Zhengyou Zhang, *Vision-based Interaction with Fingers and Papers* , Proc. International Symposium on the CREST Digital Archiving Project, Tokyo, Japan, 2003.
3. Wijewantha N. S., *VistaKey: A Keyboard Without A Keyboard – A Type of Virtual Keyboard* , Final year project thesis, Informatics Institute of Technology, Wellawatta, Sri Lanka, April 2004.
4. Kenji Oka, Yoichi Sato, Hideki Koike, *Real-Time Fingertip Tracking and Gesture Recognition* , IEEE Computer Graphics and Applications, University of Tokyo, Tokyo, Japan, 2002.
5. Shahzad Malik, *Real-Time Hand Tracking and Finger Tracking for Interaction*, Project Report, 2003.
6. Gary Bradski & Adrian Kaehler, *Learning OpenCV Computer Vision with OpenCV Library,* O'REILLY.