Pelacakan dan Estimasi Pose Video Wajah 3 Dimensi

Resmana Lim[1], Davina[2], Silvia R.[2]
[1]Fakultas Teknologi Industri, Jurusan Teknik Elektro, Universitas Kristen Petra
[2]Fakultas Teknologi Industri, Jurusan Teknik Informatika, Universitas Kristen Petra
e-mail: resmana@petra.ac.id, davina0379@yahoo.com

Abstrak

Paper ini membahas sistem pelacakan dan estimasi pose wajah dengan menggunakan webcamera. Model wajah wireframe 3D digunakan disini bersama dengan metode feature tracking Lucas-Kanade. Titik fiture wajah dilacak pada setiap frame video yang masuk dan pose wajah diestimasi menggunakan metode Fully projective. Sistem ini dibangun menggunakan Microsoft Visual C++ 6.0®, Microsoft® DirectShow®, Intel Image Processing dan Open Source Computer Vision Library. Aplikasi ini telah diimplementasikan dan dapat melacak pergerakan wajah secara real-time yaitu 30 frames/s dengan menggunakan webcam resolusi 320×240 pixel/frame pada PC Pentium III/533 MHz. Sistem cukup prospektif digunakan sebagai salah satu metode interaksi manusia komputer maupun pada sistem pengenalan ekspresi wajah.

Kata kunci: pelacakan wajah 3D , estimasi pose, pelacakan Lucas-Kanade, interaksi manusia komputer.

Abstract

The paper describes a face tracking and pose estimation system by using a webcamera. A 3D wireframe face model was used in conjunction with a feature tracking method of Lucas-Kanade. A set of face feature points was tracked in every video frame by using Lucas-Kanade method. A fully projective method was depployed for face pose estimation. This application was built using Microsoft Visual C++ 6.0®, Microsoft® DirectShow®, Intel Performance Library and Open Source Computer Vision (OpenCV) Library. The application has been implemented and able to track the face movement in real time (30 frames/second), using webcam with the resolution of 320´240 pixel/frame on PC Pentium III/533 MHz. The system is prospective to be used for human-computer interaction applications as well as face expression recognition system.

***Keywords**: 3D face tracking, pose estimation, Lucas-Kanade tracking, human computer interaction.*

Pendahuluan

Pelacakan wajah merupakan bagian penting dalam memecahkan berbagai permasalahan termasuk dalam pengenalan wajah, analisa/ pengenalan ekspresi wajah, video konferensi, dan lain-lain. Kemampuan melacak wajah juga memberikan konstribusi yang cukup besar terhadap interaksi manusia dan komputer. Salah satu penerapannya adalah pelacakan wajah untuk menggerakkan mouse sehingga user tidak perlu menggunakan tangan, cukup dengan gerakan wajah saja misalnya. Tentu saja untuk keperluan ini diperlukan kamera sebagai input.

Pelacakan pergerakan wajah dengan wire frame 3 dimensi ini menggunakkan model wire frame wajah yang merupakan kumpulan titik koordinat dalam 3 dimensi untuk mengestimasi pose dari hasil pelacakan pergerakan wajah yang didapat dari kamera ataupun file video.

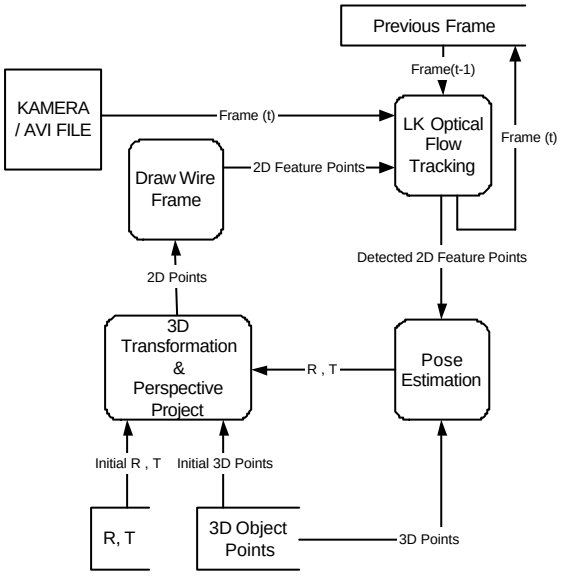
Catatan: Diskusi untuk makalah ini diterima sebelum tanggal 1 Mei 2002. Diskusi yang layak muat akan diterbitkan pada Jurnal Teknik Elektro volume 2, nomor 2, September 2002.

Model wire frame wajah ini pertama-tama diproyeksikan ke layar sebagai inisial fitur yang akan dilacak. Sebelum proses pelacakan, maka model yang sudah di proyeksikan ini harus dicocokan dahulu ke wajah orang yang akan dilacak. Titik-titik dalam 2 dimensi ini selanjutnya yang akan digunakan dalam proses pelacakan dan disebut sebagai fitur. Pelacakan disini menggunakan metode Lucas-Kanade Optical Flow. Proses pelacakan disini adalah mendapatkan optical flow dari fitur-fitur image sequence (t dan t+1) yang koresponden. Sehingga diketahui perpindahan dari masing-masing fitur. Selanjutnya posisi baru dari masing-masing fitur yang informasinya masih dalam 2 dimensi ini digunakan untuk meng-estimasi pose. Untuk estimasi pose disini digunakan metode Fully Projective yang merupakan modifikasi dari metode Lowe. Melalui metode ini dengan menggunakan informasi posisi titik yang baru dan model wajah 3 dimensi maka akan didapatkan besar transformasi dari gerakan wajah yaitu translasi dan rotasi terhadap sumbu x,y, dan z. Selanjutnya setelah diketahui rotasi

dan translasinya maka model wajah 3D ini ditansformasikan dengan rotasi dan translasi yang telah didapat kemudian diproyeksikan kembali ke layar. Demikian proses ini terus berlanjut sehingga dapat diketahui besar transformasi dari pergerakan wajah.

Sistem Pelacakan & Estimasi Pose

Sistem keseluruhan dapat dilihat pada gambar 1 dan dapat dibagi menjadi 5 bagian utama yaitu:



Gambar 1. Blok Diagram Sistem Estimasi Pose Wajah

1. Pengambilan *frame* oleh kamera atau *file* AVI.
   Pengambilan *frame* gambar ini menggunakan Microsoft DirectShow untuk mendapatkan *frame* yang *real time* (± 30 *frame*/s).
2. Menampilkan obyek *wire frame* 3D ke layar.
   Obyek *wire frame* yang berupa kumpulan titik koordinat dalam 3D ditampilkan di layar untuk menunjukkan transformasi pergerakan wajah.
3. Pemilihan fitur untuk pelacakan.
   Pemilihan fitur yang dimaksud disini adalah menentukkan titik-titik pada wajah yang akan dilacak pergerakkannya (memilih daerah yang mudah di lacak yaitu daerah yang memiliki nilai eigen tinggi. Misal: daerah pojok).
4. Pelacakan pergerakan fitur.
   Melacak pergerakan dari titik-titik yang telah dipilih sehingga diketahui perubahan letaknya.
5. Perhitungan transformasi obyek/estimasi pose.
   Melakukan perhitungan terhadap perubahan gerakan wajah berdasarkan informasi titik-titik hasil pelacakan dengan obyek 3D sehingga dapat diketahui besar transformasinya (rotasi dan translasi terhadap sumbu x, y, dan z).

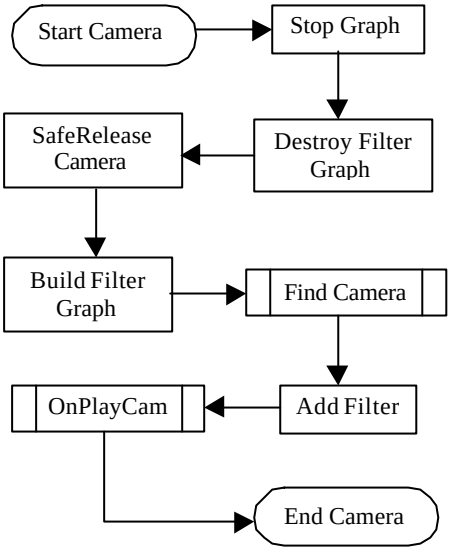
Pengambilan Frame Video – DirectShow

Modul ini bertujuan untuk melakukan pengambilan gambar yang berasal dari kamera atau dari file video. Dalam proses ini terdiri dari dua sub program, yang pertama adalah proses untuk menerima *input* baik dari kamera atau dari file video dan yang kedua adalah proses untuk melakukan pengambilan gambar (*grabbing*) dari *input* tersebut. Kedua proses tersebut menggunakan fasilitas *library* yang telah disediakan oleh Microsoft® DirectShow® yaitu *directshow.* Penggunaan *library* ini sangat bermanfaat karena proses *input* dapat berlangsung secara *realtime*.
Untuk proses ini, pertama kali adalah membuat beberapa *variable* dengan tipe data objek yang disediakan oleh *library* directshow, antara lain:

- IGraphBuilder* m_CamBuilder1
  Berfungsi sebagai *interface* dari Filter Graph Manager.
- IMediaControl* m_CamControl1
  Berfungsi untuk menangani kontrol terhadap *stream* yang sedang berlangsung.
- IVideoWindow* m_CamView1
  Berfungsi untuk membangun *interface output video window.*
- IBaseFilter* m_Cam1, m_CamTrans1
  Berfungsi untuk pengontrol filter dan pada aplikasi berfungsi untuk menspesifikasikan *pin* dan *query* informasi filter.
- IFilterGraph* m_CamGraph1
  Berfungsi untuk membangun filter. Pada aplikasi digunakan untuk menambahkan filter pada graph, menghubungkan dan memutuskan hubungan dengan filter, menghapus filter, dan melakukan beberapa operasi dasar lainnya.

Program aplikasi DirectShow yang dibuat dalam sistem ini, berdasarkan sumber *input*nya (*source filter*) secara garis besar dapat dibagi menjadi dua bagian, yaitu: *input* dari file berupa AVIFile, dan *input* dari kamera. Bagian *input* AVIFile dibangun dengan pendekatan pertama, sedangkan bagian *input* kamera dibangun dengan menggunakan pendekatan ketiga. Kedua bagian

ini secara garis besar memiliki prosedur utama yang hampir sama. Gambar 2 adalah diagram alir dari bagian *input* :



Gambar 2. Blok Diagram Bagian Input Video

Transformasi 3D dan Proyeksi 3D ke 2D

Pada transformasi 3D ini digunakan matrik yang merupakan gabungan dari rotasi dan translasi x, y, dan z. Aturan yang digunakan untuk menggabungkan transformasi ini disebut aturan Roll-Pitch-Yaw (RPY). Translasi XYZ untuk menempatkan koordinat *origin* yang kemudian diikuti rotasi terhadap 3 koordinat axis dengan urutan: Z, Y, X. Bagian rotasi dikombinasikan sebagai berikut:

$$[R] = [R(z)][R(y)][R(x)] \quad (1)$$

Dengan menggabungkan translasi T_x, T_y, T_z dengan rotasi R_x, R_y, R_z maka matriks tranformasi R adalah sebagai berikut:

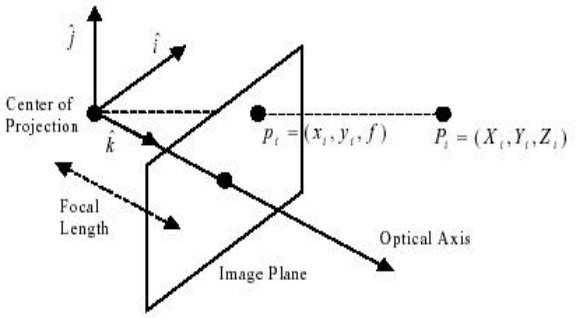
$$[R] = \begin{bmatrix} cosAcosB & cosAsinBsi\ nC - sinAcosC & cosAsinBco\ sC + sinAsinC & T_x \\ sinAcosB & sinAsinBsi\ nC + cosAcosC & sinAsinBco\ sC - cosAsinC & T_y \\ -sinB & cosBsinC & cosBcosC & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

Dengan demikian sebuah titik 3D dalam koordinat homogen $P_i = (X_i, Y_i, Z_i)^T$ dapat ditransformasikan dengan persamaan:

$$\overline{P_i} = [R] \cdot P_i \quad (3)$$

Hubungan antara titik koordinat pada ruang nyata (3 dimensi) dan titik koordinat koresponsensinya pada *image* (2 dimensi) merupakan hasil proyeksi dari ruang nyata ke bidang *image*. Titik-titik di ruang nyata yang diproyeksikan ke bidang *image* ini tergantung pada jarak dari pusat proyeksi f (*focal length*). Hubungan antara kordinat *image* $p_i=(x_i,y_i)$ dan koordinat nyata $P_i=(X_i,Y_i,Z_i)$ dapat dinyatakan sebagai berikut:

$$x_i = \frac{f}{z_i} X_i, \; y_i = \frac{f}{z_i} Y_i \quad (4)$$



Gambar 3. Proyeksi 3D ke 2D

Model 3D Wire Frame Wajah

Model *wire frame* wajah yang didapat dari Candide [3] terdiri dari 113 titik tetapi hanya digunakan 18 titik saja yang ada di daerah alis (6 titik), mata (4 titik), hidung (4 titik), dan mulut (4 titik). Pengurangan jumlah titik ini dimaksudkan untuk mempermudah pelacakan karena *wire frame* yang asli terlalu kompleks sehingga jarak antar titik sangat kecil bahkan jika dilihat secara 2 dimensi ada titik yang bertumpuk. Ke-18 titik ini dianggap dapat mewakili pergerakan wajah secara keseluruhan dan merupakan fitur dalam pelacakan. Sedangkan *wire frame* yang asli 113 titik digunakan untuk menampilkan animasi dari estimasi pose hasil pelacakan pergerakan wajah.

Gambar 4. Model Wire Frame 113 titik (kiri) dan Model Wire Frame 18 titik (kanan)

Pelacakan Fitur Metode Lucas-Kanade [1]

Misal I dan J adalah *image* 1 *channel* (*gray*). I(x) = I(x,y) dan J(x) = J(x,y) adalah nilai *grayscale* dari kedua *image* pada lokasi $x = [x\ y]^T$, dimana x dan y adalah koordinat *pixel image. Image* I

disebut sebagai *image* yang pertama dan *image* J adalah *image* kedua.

Untuk memudahkan pemahaman, *image* I dan J adalah fungsi diskrit (*array*) dan koordinat vektor pojok kiri atas adalah $\begin{bmatrix}0 & 0\end{bmatrix}^T$. Misal n_x dan n_y adalah lebar dan tinggi dari kedua *image*, maka koordinat vektor *pixel* pada kanan bawah adalah $\begin{bmatrix}n_x - 1 & n_y - 1\end{bmatrix}^T$.

Misalkan titik $u = \begin{bmatrix}u_x & u_y\end{bmatrix}^T$ ada pada *image* pertama I. Tujuan dari pelacakan fitur adalah untuk menemukan lokasi $v = u + d = \begin{bmatrix}u_x + d_x & u_y + d_y\end{bmatrix}^T$ pada *image* kedua J agar I(u) dan J(v) mirip. Vektor $d = \begin{bmatrix}d_x & d_y\end{bmatrix}^T$ adalah kecepatan *image* yang disebut juga *optical flow*. Misal ω_x dan ω_y adalah dua nilai *integer*, maka dapat didefinisikan kecepatan *image* d sebagai vektor yang meminimisasi fungsi residu $\in$ adalah sebagai berikut:

$$\in(d) = \in(d_x, d_y)$$

$$= \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} (I(x,y) - J(x+d_x, y+d_y))^2 \quad (5)$$

Fungsi diatas dihitung pada daerah sekitar *image* yang berukuran $(2\omega_x + 1)\times(2\omega_y + 1)$. Daerah sekitar *image* ini kemudian disebut *window*. Nilai umum untuk ω_x dan ω_y adalah 2-7 *pixel*.

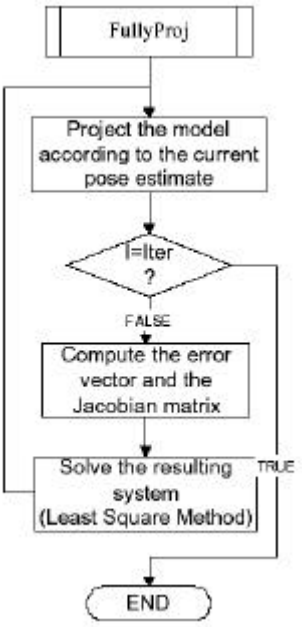
Dua komponen penting dalam pelacakan fitur adalah keakuratan dan ketangguhan. Keakuratan berhubungan dengan akurasi *subpixel* yang digunakan untuk pelacakan. Tentunya, ukuran *window* yang kecil akan lebih baik agar tidak kehilangan detail yang ada di *image* (nilai w_x dan w_y kecil). Hal ini sangat diperlukan pada area *occluding image* (daerah yang menggangu pelacakan) dimana dua area memiliki potensi untuk berpindah dengan kecepatan yang berbeda. Ketangguhan berhubungan dengan kesensitifan dari pelacakan terhadap perubahan cahaya, ukuran pergerakan *image*, dan lain-lain. Umumnya, untuk mengatasi gerakan-gerakan yang besar digunakan ukuran *window* yang besar.

Dengan melihat persamaan residu diatas maka lebih baik jika $d_x \le \omega_x$ dan $d_y \le \omega_y$. Karena itu harus ada satu yang dikorbankan antara akurasi dan ketangguhan dalam memilih ukuran *window*.

Untuk mengatasi masalah ini, maka digunakan implementasi piramidal pada algoritma Lucas-Kanade. karena algoritma *optical flow* Lucas-Kanade itu sendiri telah memenuhi kriteria akurasi.

Estimasi Pose

Garis besar dari metode Fully Projective [2] yang digunakan disini seperti pada gambar 5.



Gambar 5. Algoritma Fully Projective

Pertama-tama model wajah berupa titik-titik fitur 3D (vektor p) ditransformasikan dengan matrik transformasi R dengan persamaan 6:

$$\begin{bmatrix}x', y', z'\end{bmatrix}^T = R \cdot p. \quad (6)$$

$$\begin{bmatrix}d'_x, d'_y, d'_z\end{bmatrix} = -\begin{bmatrix}r_x \cdot t, r_y \cdot t, r_z \cdot t\end{bmatrix} \quad (7)$$

Setelah itu dilakukan proyeksi 3D ke 2D sehingga koordinat *image* dari setiap titik didapatkan dengan:

$$\begin{bmatrix}u, v\end{bmatrix} = f\left[\frac{x' + d'_x}{z' + d'_z}, \frac{y' + d'_y}{z' + d'_z}\right] \quad (8)$$

Langkah kedua adalah menghitung besarnya *error* antara koordinat hasil estimasi dengan titik 2D yang didapat dari hasil pelacakan Lucas-Kanade. Dengan vektor error *e* antara model dan *image*, maka nilai δ (parameter yang akan

diestimasi) dapat dicari untuk mengeliminasi nilai kesalahan:

$$J\boldsymbol{d} = e \text{, dimana } J_{ij} = \frac{\partial e_i}{\partial x_j} \quad (9)$$

$$\delta = \left[\Delta d_x, \Delta d_y, \Delta d_z, \Delta\Phi_x, \Delta\Phi_y, \Delta\Phi_z\right]^T \quad (10)$$

Dimana J adalah matrik Jacobian yang didapatkan dari hasil turunan parsial dari *u* dan *v* dan δ merupakan 6 parameter yang ingin dicari yaitu besar translasi dan rotasinya.

Tabel 1: Turunan Parsial dari u dan v

	U	*V*
x_t	*Fc*	*0*
y_t	*0*	*Fc*
z_t	$-fac^2$	$-fbc^2$
F_x	$-fac^2y'$	$-fc(z'+bcy')$
F_y	$fc(z'+acx')$	fbc^2x'
F_z	$-fcy'$	Fcx'
f	*Ac*	*Bc*

dimana $[a,b,c] = \left[x' + d'_x, y' + d'_y, \frac{1}{z' + d'_z}\right]$.

Matrik Jacobian untuk mencari 6 parameter transformasi δ adalah seperti persamaan 11 [2].

$$J = \begin{bmatrix} fc & 0 \\ 0 & fc \\ -fac^2 & -fbc^2 \\ -fac^2y' & -fc(z'+bcy') \\ fc(z'+acx') & fbc^2x' \\ -fcy' & fcx' \end{bmatrix} \quad (11)$$

Sehingga persamaan linier (9) dapat dinyatakan dalam bentuk lain sebagai berikut:

$$\frac{\partial u}{\partial d_x}\Delta d_x + \frac{\partial u}{\partial d_y}\Delta d_y + \frac{\partial u}{\partial d_z}\Delta d_z + \frac{\partial u}{\partial \Phi_x}\Delta\Phi_x + \frac{\partial u}{\partial \Phi_y}\Delta\Phi_y + \frac{\partial u}{\partial \Phi_z}\Delta\Phi_z = E_u$$

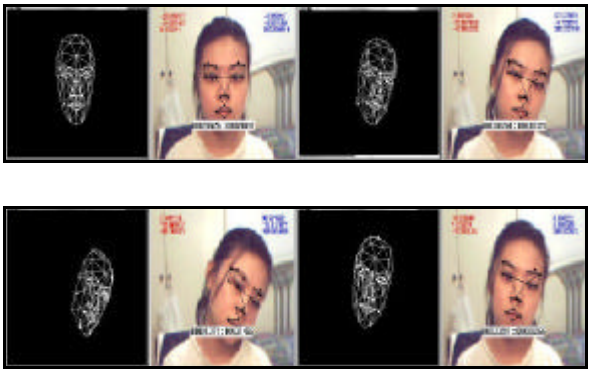
$$\frac{\partial v}{\partial d_x}\Delta d_x + \frac{\partial v}{\partial d_y}\Delta d_y + \frac{\partial v}{\partial d_z}\Delta d_z + \frac{\partial v}{\partial \Phi_x}\Delta\Phi_x + \frac{\partial v}{\partial \Phi_y}\Delta\Phi_y + \frac{\partial v}{\partial \Phi_z}\Delta\Phi_z = E_v \quad (12)$$

Selanjutnya persamaan linier (9) diatas diselesaikan dengan metode Least Square [4]. Setelah didapatkan nilai besaran transformasinya, maka matrik transformasi ini digunakan untuk iterasi berikutnya seperti pada langkah pertama. Proses ini akan diulang sebanyak jumlah iterasi yang kita inginkan. Hasil estimasi pose ditampilkan dilayar, serta digunakan pula untuk menggerakan animasi wajah 3D.

Hasil-Hasil Percobaan

Sistem dibangun menggunakan bahasa pemrograman Microsoft Visual C++, dengan memanfaatkan software library Intel OpenSource Computer Vision (OpenCV). Sistem diuji coba dengan video yang diambil dari webcam. Pengujian sistem disini terdiri dari pengujian terhadap akurasi dan ketangguhan pelacakan, ketepatan dalam mengestimasi pose, dan pengaruh jumlah fitur terhadap pelacakan dan estimasi pose. Akurasi disini berhubungan dengan ketepatan dalam pelacakan. Ketangguhan berhubungan dengan kemampuan melacak dalam berbagai intensitas cahaya dan kecepatan gerakan.

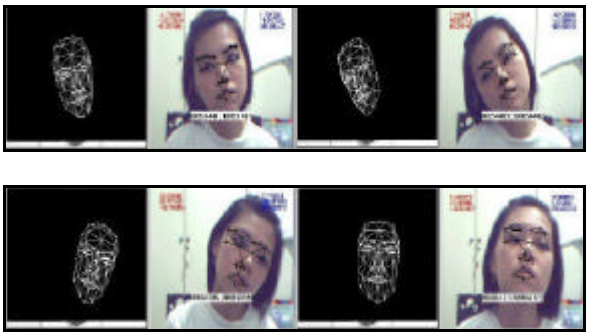
Gambar 7 dibawah menunjukkan sekuen pelacakan wajah dengan cahaya menggunakan cahaya lampu ruangan ditambah dengan lampu sorot yang menerangi daerah wajah.



Gambar 7. Sekuen Pelacakan Wajah dengan Cahaya Ruangan dan Lampu Sorot

Dapat dilihat pelacakan disini cukup baik juga estimasi pose yang didapat untuk menggerakkan *wire frame* 3D pada layar animasi mampu menunjukkan pose yang hampir sama. Letak fitur mulai awal proses pelacakan hingga akhir pelacakan masih baik. Sistem cukup akurat karena fitur- fitur yang dilacak jelas sehingga memudahkan proses pelacakan. Kemungkinan fitur hilang kecil.

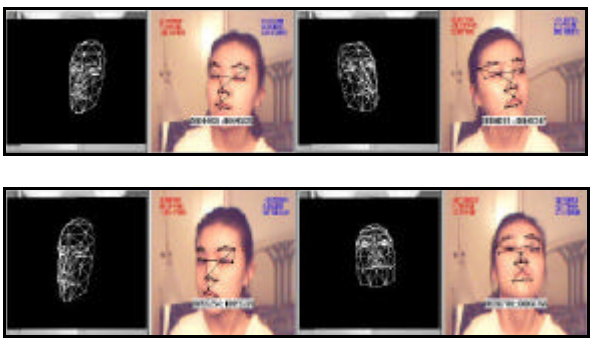
Gambar 8 dibawah menujukkan sekuen pelacakan wajah dengan cahaya lampu ruangan saja.



Gambar 8. Sekuen Pelacakan Wajah dengan Cahaya Ruangan

Disini pelacakan cukup baik juga estimasi pose yang didapat untuk menggerakkan *wire frame* 3D pada layar animasi mampu menunjukkan pose yang hampir sama. Letak fitur saat pelacakan ada beberapa yang mengalami pergerseran (tidak sesuai dengan posisi fitur yang seharusnya) walaupun hal ini tidak terlalu mempengaruhi hasil estimasi pose karena jumlah fitur yang tidak tepat masih jauh lebih sedikit dibanding yang tepat. Sistem urang akurat karena fitur-fitur yang dilacak kurang jelas walaupun lingkungan cukup terang karena adanya cahaya lampu ruangan tetapi daerah bagian wajah kurang terang sehingga mengurangi tingkat akurasi pelacakan. Kemungkinan fitur hilang cukup besar.

Gambar 9 dibawah menujukkan sekuen pelacakan wajah dengan cahaya lampu sorot yang menerangi daerah wajah saja.



Gambar 9. Sekuen Pelacakan Wajah dengan Lampu Sorot

Hasil pelacakan dengan lampu sorot saja menunjukkan hasil yang lebih baik daripada pelacakan menggunakan cahaya ruangan saja tetapi masih kalah dengan pelacakan yang menggunakan baik cahaya ruangan maupun lampu sorot. Letak fitur ada beberapa yang mengalami pergerseran terhadap letak fitur yang seharusnya. Keadaan sekeliling yang gelap mengurangi kejelasan dari fitur yang hendak dilacak sehingga mengurangi akurasi dari pelacakan disini. Kemungkinan fitur hilang ada.
Selanjutnya dilakukan pengujian terhadap nilai translasi z. Pengujian disini melakukan 2 macam gerakan yaitu maju (mendekati kamera) dan mundur (menjauhi kamera). Posisi awal disini diinisialkan sebagai 0 cm dan Tz (Translasi z) inisial adalah 400. Dari pengujian disini diharapkan dapat terlihat linieritas dari translasi hasil estimasi dengan kenyataan sesungguhnya.

Tabel 2. Hasil Estimasi Translasi terhadap Sumbu z

Gerakan Maju	T_z	Relatif T_z	Gerakan Mundur	T_z	Relatif T_z
0 cm	400	0	0 cm	400	0
5 cm	370	30	5 cm	430	30
10 cm	340	60	10 cm	470	70
15 cm	320	80	15 cm	500	100
20 cm	300	100	20 cm	520	120
25 cm	285	115	25 cm	535	135
30 cm	250	150	30 cm	560	160

Dari hasil pengujian diatas dapat dilihat adanya nilai beda yang cukup konstan setiap penambahan atau pengurangan jarak sebesar 5 cm sehingga dapat membentuk suatu fungsi linier Tz=400+5×Jarak(cm). Hasil pengujian disini tidak benar-benar akurat karena nilai Tz terus di-*update* dan model kesulitan untuk diam tanpa gerakan saat pengambilan nilai ditambah lagi dengan adanya distorsi kamera. Meskipun demikian masih dapat diambil nilai perkiraannya. Selanjunya pengujian terhadap rotasi terhadap sumbu x,y,z yang sesungguhnya dengan rotasi hasil pose estimasi.

Tabel 3: Hasil Estimasi Rotasi terhadap Sumbu X, Y & Z

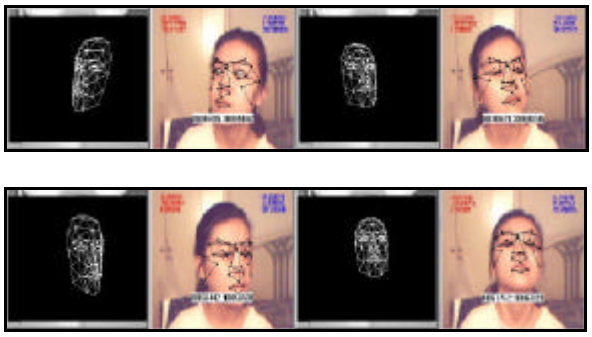
Rotasi Rx	**Estimasi Rx**	**Kesalahan estimasi**
30°	20°	-10°
45°	35°	-10°
60°	50°	-10°
90°	100°	10°
Rotasi Ry	**Estimasi Ry**	**Kesalahan estimasi**
30°	35°	5°
45°	50°	5°
60°	70°	10°
90°	100°	10°
Rotasi Rz	**Estimsi Rz**	**Kesalahan estimasi**
30°	30°	0°
45°	50°	5°
60°	60°	0°
90°	100°	10°

Hasil estimasi dari rotasi x, y, dan z menunjukkan hasil yang hampir sama dengan kenyataannya. Sama dengan translasi, pengujian disini nilainya tidak bisa benar-benar akurat karena adanya distorsi dan kesulitan pengukuran dalam derajat. Dari hasil estimasi diatas didapatkan tingkat kesalahan rata-rata sebesar 4°.

Pengujian hasil estimasi disini ingin mengetahui seberapa tinggi akurasi dari algoritma Fully Projective dalam mengestimasi pose. Dan setelah dilihat dari hasil pengujian disini dapat dilihat bahwa algoritma ini cukup akurat.

Pengujian variasi jumlah titik fitur untuk mengetahui pengaruh jumlah titik fitur terhadap hasil pelacakan dilakukan dengan menambah jumlah titik fitur yang digunakan untuk pelacakan. Pengujian disini dilakukan terhadap titik fitur sebanyak:

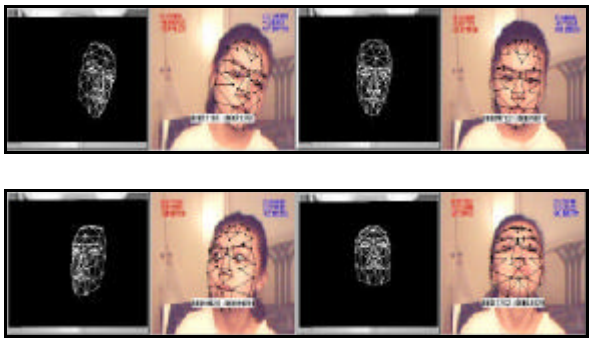
- 18 titik
- 33 titik



Gambar 10. Sekuen Pelacakan Wajah dengan 33 Titik Fitur

Dari sekuen pelacakan wajah dengan menggunakan 33 titik ini dihasilkan hasil yang cukup baik dan hampir sama dengan 18 titik fitur.

- 51 titik



Gambar 11. Sekuen Pelacakan Wajah dengan 51 Titik Fitur

Pada pelacakan wajah dengan menggunakan 51 titik fitur ini terlihat beberapa titik yang ada terutama pada titik-titik yang melingkari daerah wajah mengalami pergeseran dari posisi yang seharusnya.

Dari hasil pengujian terhadap 18, 33, dan 51 titik fitur dapat dilihat bahwa pelacakan disini cukup baik untuk daerah fitur yang memiliki nilai pojok yang tinggi. Sedangkan titik fitur yang berada di pinggiran wajah memiliki potensi yang besar untuk lepas karena pada sistem pelacakan disini tidak ada penanganan terhadap daerah yang tak terlihat oleh kamera. Jumlah titik fitur disini tidak akan mempengaruhi hasil pelacakan jika fitur baik dan berada di dalam daerah wajah.

Kesimpulan

1. Sistem pelacakan pergerakan wajah dengan wire frame 3D dapat berjalan dengan baik selama fitur inisial memiliki nilai pojok yaitu suatu daerah yang memiliki tingkat lekukan yang jika dikalikan dengan gradiennya akan meningkat menjadi pangkat 3 dari maximum nilai daerah sekitarnya.
2. Sistem pelacakan pergerakan wajah dengan wire frame 3D tidak menangani *occluding area* (daerah yang menggangu pelacakan) sehingga menyebabkan keterbatasan dalam gerakan untuk mendapatkan hasil pelacakan yang baik.
3. Sistem pelacakan pergerakan wajah dengan wire frame 3D dapat dilakukan secara *real-time* (30 *frames/second*) dengan menggunakan Microsoft® DirectShow®, *Intel Performance Library* dan *Open Source Computer Vision Library*.
4. Algoritma pelacakan Lucas-Kanade dari *Open Source Computer Vision Library* disini melacak berdasarkan *image brightness.* Jika daerah yang dilacak terkena sinar yang terlalu tinggi atau terlalu gelap sehingga daerah yang dilacak tidak jelas maka algoritma pelacakan tidak akan mendapatkan hasil yang baik. Oleh karena itu dipilih fitur yang merupakan daerah pojok untuk memudahkan pelacakan. Demikian pula jika gerakan cepat maka image akan terlihat kabur sehingga fitur yang dilacak hilang.
5. Algoritma Fully Projective yang mengestimasi pose dari hasil pelacakan berjalan sesuai dengan harapan selama hasil pelacakan baik. Ini dapat dilihat dari hasil animasi yang menampilkan posisi yang hampir sama

dengan sebenarnya dengan tingkat kesalahan rotasi rata-rata sebesar 4°.

Daftar Pustaka

[1] Bouguet, Jean-Yves. "Pyramidal Implementation of the Lucas-Kanade Feature Tracker Description of the Algorithm", Intel Technology Journal, pp. 1-9.

[2] Araújo, Helder, et al. "A Fully Projective Formulation to Improve the Accuracy of Lowe's Pose-Estimation Algorithm", pp. 1-16.

[3] Ahlberg, Jörgen. [http://www.icg.isy.liu.se/candide]. Linkoping University. December 2001.

[4] Intel® Image Processing Library Reference Manual. [PDF], Intel Corporation, 2000.

[5] Open SourceComputer Vision Library Reference Manual. [PDF], Intel Corporation, 2001.

[6] Press, William H, et al. "Numerical Recipes in C: The Art of Scientific Computing ", 2nd ed. Cambridge University Press, 1992.

[7] Krishnan, Radha, et al. "Monocular Pose of a Rigid Body Using Point Landmarks", Academic Press Inc, 1992.