BAB II

LANDASAN TEORI

2.1 Keamanan Komputer

Keamanan komputer adalah perlindungan terhadap pencurian atau perusakan *hardware*, *software*, dan informasi yang terkandung di dalamnya (Gasser, 1988). Bidang keamanan komputer menjadi semakin penting karena meningkatnya ketergantungan akan sistem komputer di masyarakat kini. Keamanan komputer juga menjangkau keamanan berbagai teknologi pintar seperti *smarphone* dan *smart television* serta menjangkau pula keamanan jaringan seperti internet dan jaringan pribadi.

Menurut Hibbard (2009), keamanan komputer harus mampu memberikan *information assurance* atau jaminan informasi terhadap data-data yang disimpan. Terdapat beberapa jenis jaminan informasi yang harus diberikan oleh keamanan komputer, yakni:

1. *Confidentiality*, memastikan informasi hanya dapat dilihat oleh pihak yang memiliki otoritas untuk melihatnya.
2. *Integrity*, memastikan informasi yang disimpan tidak berubah-ubah isinya diluar kehendak dari penyimpan informasi tersebut.
3. *Availability*, memastikan informasi siap diakses oleh pengguna ketika diperlukan.
4. *Possession*, memastikan informasi tetap dipegang atau dibawah kendali pihak yang berotoritas.

5. *Authenticity*, memastikan informasi memang benar secara nyata dan bukan merupakan sesuatu yang tidak nyata atau bukan yang sebenarnya.
6. *Utility*, memastikan informasi layak dipakai dan dapat digunakan.
7. *Privacy*, memastikan informasi pribadi aman dari pengamatan pihak lain atau intrusi dari pihak lain.
8. *Authorized Use,* memastikan layanan yang memakan biaya hanya tersedia untuk pihak berotoritas.
9. *Nonrepudiation*, memastikan pengirim pesan atau pemulai transaksi tidak dapat menyangkal tindakan tersebut.

Authentication atau otentikasi adalah proses identifikasi dari sebuah pihak untuk memastikan bahwa identitas pihak tersebut benar sesuai dengan identitas yang disebutkan oleh pihak tersebut dan tidak dipalsukan. Pihak yang telah terotentikasi tidak membuat pihak tersebut otomatis mendapatkan akses. Akses hanya diberikan bila pihak yang telah terotentikasi memiliki otoritas untuk mendapatkan akses teresebut (McDaniel, 2006). Otentikasi bertujuan untuk memenuhi jaminan informasi *authenticity* dan memastikan bahwa identitas pengguna tidak dipalsukan dan benar secara nyata.

2.2 Time-of-flight Camera

Time-of-flight camera (Tof *camera*) adalah kamera yang mampu melakukan pemindaian tiga dimensi dari suatu objek dengan cara mengukur *time-of-flight*/waktu perjalanan cahaya dari antara kamera dengan objek tersebut untuk setiap titik gambar. Kamera ToF mendapatkan hasil seluruh gambar dengan setiap pancaran laser atau cahaya sedangkan sistem LIDAR (*Light Radar*) melakukan

pindaian satu per satu setiap titik gambar untuk setiap pancaran laser atau cahaya (Iddan dan Yahav, 2001).

Menurut Iddan dan Yahav(2001), sebuah kamera ToF harus memiliki komponen-komponen berikut:

1. *Illumination unit*, komponen yang melakukan pemancaran laser atau cahaya dalam frekuensi tertentu.
2. *Optics*, komponen lensa yang mengumpulkan cahaya yang terpantul dan memfokuskannya ke komponen *image sensor*. Terdapat filter optis untuk memastikan hanya cahaya yang memiliki panjang gelombang yang sama dengan panjang gelombang cahaya yang dipancarkan oleh *illumination unit* yang dapat lewat.
3. *Image sensor*, komponen yang mendeteksi cahaya yang masuk dan mengubahnya ke dalam bentuk *pixel*.
4. *Driver electronics*, komponen yang mengendalikan komponen-komponen lainnya dan memastikan bahwa komponen *image sensor* memiliki frekuensi deteksi cahaya yang sama dengan frekuensi pemancaran cahaya oleh komponen *illumination unit*.
5. *Computation/Interface*, komponen yang melakukan perhitungan jarak berdasarkan data yang diterima oleh komponen *image sensor*. Hasil perhitungan tersebut kemudian ditampilkan dalam bentuk gambar pada sebuah *interface*.

Kamera ToF sudah memiliki banyak penerapan dalam berbagai aplikasi yang memenuhi berbagai kebutuhan pengguna. Dalam bidang otomotif, kamera ToF digunakan untuk pengamanan pejalan kaki, deteksi pra-tabrakan, dan deteksi posisi

yang tidak sesuai (Elkhalili dkk., 2006). Dalam industri *game*, kamera ToF dipakai oleh Kinect generasi kedua untuk memungkinkan *natural use interfaces* oleh pemain dengan cara penggunaan teknik *computer vision* dan teknik pengenalan gestur tubuh (Rubin, 2013).

Menurut Zhang dan Lu (2012) sebuah kamera ToF akan menghasilkan 3 jenis data untuk setiap *pixel* yang dihasilkan yakni *distance*, *amplitude*, dan *intensity*. *Distance* adalah jarak dari kamera dengan objek. *Amplitude* adalah tingkat keandalan dari nilai jarak yang didapatkan. Semakin reflektif permukaan dari objek, semakin tinggi pula nilai *amplitude* ini. Bila jarak objek di luar dari jarak pengamatan, maka nilai *amplitude* akan mendekati nol. *Intensity* adalah tingkat kecerahan dari suatu objek. Semakin banyak cahaya yang masuk ke kamera maka nilai *intensity* akan semakin tinggi.

2.3 Kinect Xbox One

Kinect Xbox One adalah perangkat interaksi yang diproduksi oleh Microsoft sebagai pelengkap perangkat Xbox One untuk menambahkan modalitas palet dari desainer *user interfaces: gestures* dan *speech* (Tashev, 2013). Kinect Xbox One merupakan perkembangan lebih lanjut dari Kinect Xbox 360 yang telah dikembangkan sebelumnya. Kinect Xbox One memanfaatkan teknologi *time-of-flight* (ToF) *camera* untuk mendapatkan data kedalaman yang lebih akurat sehingga pengenalan gestur tubuh dapat dilakukan dengan lebih akurat (Meisner, 2013). Meisner (2013) juga menyebutkan bahwa cahaya yang digunakan untuk dilakukan perhitungan jarak adalah cahaya inframerah sehingga dapat dilakukan perhitungan jarak dengan tingkat pencahayaan yang rendah atau tanpa pencahayaan.

Gambar 2.1 Contoh Gambar Kinect Xbox One

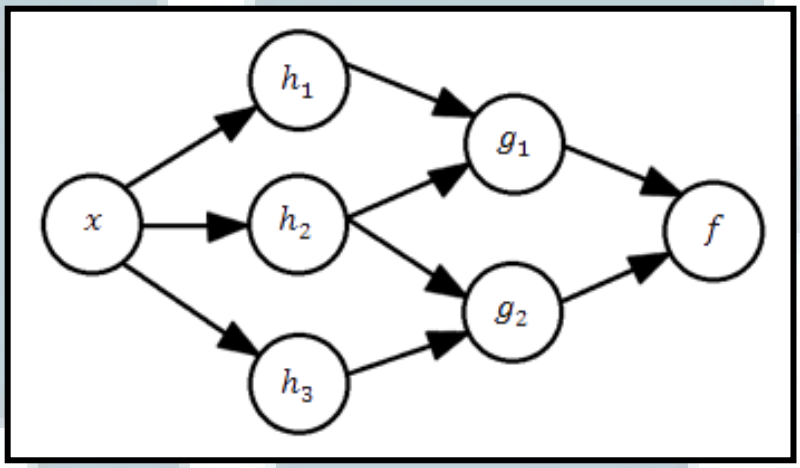
Kinect for Windows SDK dan *toolkit* memiliki *driver*, *APIs*, *device interfaces*, *tools*, dan juga contoh-contoh cara pemrograman untuk mempermudah pengembangan aplikasi yang akan menggunakan perangkat Kinect Xbox One. Hasil data yang didapatkan dari kamera milik Kinect Xbox One adalah data gambar berwarna dalam bentuk *Red Green Blue* (RGB), data kedalaman dalam bentuk sebelas angka biner, dan data inframerah dalam bentuk sebelas angka biner.

2.4 Neural Networks

Neural networks/jaringan syaraf tiruan adalah metode pemrosesan informasi yang mengambil inspirasi dari cara kerja sistem syaraf asli di dalam makhluk hidup (Aleksander dan Morton, 1990). *Neural networks* terdiri atas banyak elemen-elemen pemrosesan yang saling terkoneksi (neuron) yang bekerja secara bersama-sama untuk menyelesaikan suatu masalah. Jaringan syaraf tiruan (JST) seperti makhluk hidup pada umumnya belajar dengan contoh. JST dikonfigurasi untuk tujuan tertentu seperti deteksi pola dan klasifikasi data dengan melalui proses pelatihan. Proses pelatihan dilakukan dengan cara mengubah hubungan-hubungan antar elemen atau neuron (Shiffman, 2012).

Menurut Russell (2012), di dalam implementasinya ke sistem komputer sebuah metode pemrosesan informasi dapat disebut sebagai *neural networks* bila metode tersebut memiliki kedua karakteristik berikut:

1. Memiliki kumpulan *weights* atau beban yang dapat berubah-ubah dan diubah dengan menggunakan sebuah algoritma pelatihan.
2. Memiliki kemampuan untuk mengira-ngira fungsi-fungsi non-linear dari *input* yang dimasukkan.



Gambar 2.2 Model matematika dari sebuah neural networks

Model matematika dari suatu *neural networks* bila mengacu pada gambar 2.1 dapat dituliskan sebagai

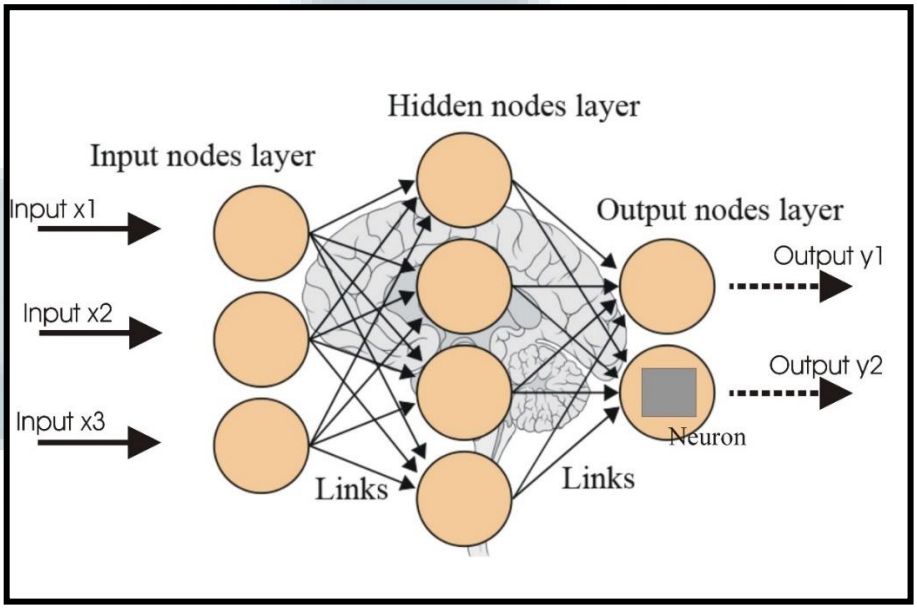
$$f(x) = K\left(\sum w_i g_i(x)\right) \tag{2.1}$$

dengan setiap elemen dijelaskan satu per satu seperti berikut:

1. *f(x)* adalah hasil *output* dari pemrosesan informasi menggunakan *neural networks*.
2. *x* adalah *input* atau masukan dari *neural networks* yang akan diproses.

3. K adalah fungsi aktivasi yang digunakan untuk menggabungkan berbagai *input* suatu *node* menjadi *output* dari *node* tersebut. Fungsi yang digunakan adalah fungsi baku seperti fungsi *hyperbolic* atau fungsi *sigmoid*.
4. w_i adalah beban atau *weight* dari sinaps atau hubungan antar *node*.
5. $g_i(x)$ adalah hasil pemrosesan dari *node* g terhadap *input* x yang diterimanya.
6. $w_i g_i(x)$ adalah pengalian *weight* terhadap *input* yang terjadi ketika *input* diteruskan melalui sinaps.
7. $\sum w_i g_i(x)$ adalah gabungan dari setiap hasil pengalian yang diterima oleh *node*.

Hasil dari perhitungan tersebut tidak akan menghasilkan *output* yang sesuai karena nilai beban atau *weight* pada awalnya ditentukan secara acak atau *random*. Perolehan *output* yang baik dapat dilakukan dengan melakukan pengubahan dan penyesuaian terhadap nilai-nilai beban tersebut dengan melakukan proses pelatihan terhadap *neural networks*.



Gambar 2.3 Bentuk Standar *Neural Networks* (Tadiou, 2010)

Menurut Shiffman (2012), dalam proses pelatihan tersebut terdapat beberapa strategi yang dapat digunakan untuk pelatihan yaitu:

- *Supervised Learning*, melibatkan seorang guru yang lebih pintar dari sistem itu sendiri. Guru memberi pertanyaan yang sudah guru tersebut ketahui jawabannya, sistem menebak jawaban dari pertanyaan tersebut, guru memberi tahu jawaban yang benar, dan kemudian sistem mengubah jaringan untuk mengurangi tingkat kesalahan.
- *Unsupervised Learning*, dilakukan ketika tidak ada jawaban yang jelas untuk pertanyaan-pertanyaan yang diberikan. Sistem menentukan sendiri pola yang mungkin terdapat di dalam pertanyaan yang diberikan dan menebak sendiri jawabannya.
- *Reinforcement Learning*, dilakukan dengan melakukan observasi. Sistem mengambil sebuah keputusan dari keputusan-keputusan yang dapat diambil dan mengamati lingkungannya. Apabila lingkungannya mengembalikan hasil yang negatif, sistem akan mengubah jaringannya untuk membuat keputusan yang berbeda. Strategi ini banyak dipakai di bidang robotik.

2.5 Backpropagation

Backpropagation adalah metode pelatihan yang umum digunakan untuk melatih *neural networks* dan digabungkan dengan metode optimasi seperti misalnya *gradient descent* (Alpaydin, 2010). Metode *gradient descent* menghitung gradien dari *loss function* dan menerapkan hasilnya ke dalam setiap *weight* atau beban yang terdapat di dalam jaringan. Metode *backpropagation* tergolong ke dalam metode

pelatihan *supervised learning* karena diperlukan informasi mengenai tingkat kebenaran *output* yang dihasilkan agar sistem *neural networks* dapat memperbaiki kesalahannya.

Menurut Russel dan Norvig (2009), *backpropagation* dapat dibagai menjadi 2 fase yang berbeda seperti berikut:

1. Fase *propagation* atau fase penyebaran data. Langkah-langkah yang termasuk dalam fase ini adalah:
    a. Memasukkan data *input* untuk diproses oleh *neural networks* sehingga menghasilkan *output activations*. Nilai dari *output activations* didapatkan dengan menggunakan rumus (2.1). Data hanya dapat bergerak maju dari *input layer* menuju *output layer* atau disebut juga *forward propagation*.
    b. Melakukan pembandingan antara *output activations* dengan target latihan sehingga dapat dihasilkan *delta* atau selisih antara *input* dengan *output* dari semua *nodes* yang digunakan. Nilai *E* total atau *Mean Squared Error* (MSE) dari keseluruhan sistem yang meliputi setiap *node* dituliskan sebagai

$$E = \frac{1}{2}\sum(t - y)^2 \tag{2.2}$$

    Pembandingan dilakukan secara mundur dari *output layer* menuju *input layer* atau disebut juga *backward propagation*.
2. Fase *weight update* atau fase pengubahan nilai beban. Langkah-langkah yang termasuk dalam fase ini adalah:
    a. Melakukan penurunan parsial dari nilai *delta* yang diperoleh terhadap setiap *weight* untuk mendapatkan nilai *gradient* yang

dibutuhkan untuk setiap *weight*. Nilai dari *gradient* ini dituliskan sebagai

$$\frac{\partial E}{\partial w_{ij}} \tag{2.3}$$

b. Melakukan pengurangan *weight* sebesar sebagian (dalam persentase) dari gradien yang diperoleh sebelumnya. Semakin tinggi persentase yang dipakai, semakin cepat proses pelatihannya. Semakin rendah persentase yang dipakai, semakin akurat proses pelatihannya. Jumlah pengurangan *weight* ini dituliskan sebagai

$$\Delta w_{ij} = -\alpha \frac{\partial E}{\partial w_{ij}} \tag{2.4}$$

Kedua fase yang telah disebutkan tersebut kemudian diulangi secara terus-menerus hingga sistem kemudian menghasilkan nilai-nilai *output* yang diinginkan.

Penggunaan metode *gradient descent* di dalam *backpropagation* memerlukan turunan dari pencarian hasil *error* atau jumlah kesalahan. Jumlah dari *error* itu adalah selisih atau *delta* antara nilai *output* yang didapatkan dengan nilai *output* latihan. Jumlah *error* tersebut dituliskan sebagai rumus (2.2). Setiap elemen dijelaskan sebagai berikut:

1. *E* adalah nilai *error* dari keseluruhan sistem.
2. *t* adalah target dari nilai *output* yang dihasilkan.
3. *y* adalah hasil dari nilai *output* latihan atau nilai yang benar.
4. $1/2$ adalah konstanta yang dipakai untuk memudahkan proses turunan nantinya.

Setelah dilakukan penghitungan jumlah *error*, maka berikutnya dapat dihitung jumlah perubahan *weight* yang perlu dilakukan terhadap suatu *weight*

dengan menggunakan rumus (2.4). Rumus tersebut dijelaskan seperti berikut:

1. Δw_{ij} adalah jumlah pengubahan *weight* yang diperlukan.
2. w_{ij} adalah jumlah *weight* dari matriks baris ke-i dan kolom ke-j.
3. (2.3) adalah turunan parsial dari *error* terhadap *weight* dari matriks baris ke-i dan kolom ke-j. Ini merupakan *gradient* perubahan yang diperlukan oleh suatu *weight*.
4. -1 diperlukan untuk mengubah nilai *weight* menjadi ke arah minimum *error*, bukan maksimum. Ini digunakan agar nilai *error* mengecil bukan membesar.
5. α adalah tingkat pelatihan atau *learn rate* dari sistem *neural networks* dalam bentuk persentase. Ini dipakai agar pengubahan *weight* dilakukan hanya sebesar sebagian dari *gradient* yang diperlukan.