BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Berikut ini adalah beberapa penelitian tentang *gesture control* menggunakan *leap motion*.

Menurut penelitian Pallas, Setyawan & Prasetio (2017) menjelaskan tentang sistem kendali navigasi *quadcopter* menggunakan suara melalui *smartphone* dan *arduino* dengan metode *text processing*. *Input* suara akan diubah menjadi *text* dan dipisah menjadi satu kata. Lalu kata tersebut akan dicocokan dengan *database* yang telah dibuat menggunakan algoritme *stopword removal wordlist*. Hasil dari pencocokan akan dikirim melalui *arduino* dan dilakukan pengolahan data dengan *output* gerakan *quadcopter*. Waktu dalam melakukan *text preprocessing* dan pengolahan suara akan bertambah terus seiring dengan banyaknya kata yang diberikan. Untuk masukan 1 kata pengolahan suara menghasilkan *delay* 3 detik. Lalu untuk masukan 2 kata pengolahan suara menghasilkan *delay* 4 detik.

Menurut penelitian Pribadi, Jonemaro & Setyawan (2017) yaitu tentang pengendalian *quadcopter* dengan prinsip *virtual reality* menggunakan *google cardboard. User* memberikan instruksi sebuah gerakan kepala. *Input* gerakan kepala dikirim menuju aplikasi kendali *quadcopter* untuk dilakukan proses klasifikasi jenis gerakan kepala. Setelah itu aplikasi kendali *quadcopter* membuat *stereoscopic display* yaitu sebuah tampilan *virtual reality*. Dilanjutkan dengan aplikasi membaca data *gyroscope* berdasarkan gerakan kepala untuk menghasilkan *output* berupa gerakan *quadcopter*. Terdapat keterbatasan gerakan kepala untuk implementasi program kendali quadcopter dengan *VR*. Penulis hanya dapat mengimplementasikan 4 dari 6 gerakan kepala untuk kendali *quadcopter* dikarenakan keterbatasan implementasi pada *trinus VR* tidak bisa membaca gerakan kepala miring ke kanan maupun miring ke kiri.

Menurut penelitian Hadi, Setyawan & Maulana (2017) tentang sistem kendali navigasi *AR Drone quadcopter* dengan prinsip *natural user interface* menggunakan *microsoft kinect.* Gerakan tubuh pengguna akan diubah menjadi *sceleton tracking* dengan *microsoft kinect*. Data *sceleton tracking* akan diolah dengan komputer dengan pemrograman *javascript* yang nantinya diubah menjadi instruksi gerakan *quadcopter*. *Delay* yang dihasilkan dari proses deteksi gerakan tubuh sebesar 0,05 detik.

Menurut Pititeeraphab, et al. (2016) tentang sistem kendali lengan robot menunjukkan *leap motion.* Dalam penelitian ini menerapkan prinsip *leap motion* dan kontrol servo motor. Penelitian ini dirancang dan dibangun yang terdiri dari 3 bagian utama yaitu detektor, sinyal kontrol dan pengolahan data menggunakan *microcontroller* yang terdapat pada *arduino UNO R3*, tampilan dari lengan robot. Pada bagian pertama komponen internal dari *leap motion* yaitu sensor kamera digital akan mendeteksi gerakan tangan pengguna untuk mengontrol lengan robot. *CPU* yang terdapat pada *leap motion* akan memproses visual lalu

mengirimkan data posisi koordinat gerakan tangan dan jari-jari ke komputer. Data tersebut akan ditampilkan secara *3D* dikomputer. Setelah itu data dikirim menuju *microcontroller* untuk pengolahan sinyal dan mengirim instruksi kontrol untuk motor servo sebagai kontrol lengan robot.

Menurut Sarkar, et al. (2016) gerak *controller* untuk mengontrol gerak *quadcopter* melalui gerakan manusia sederhana dengan menggunakan *leap motion* sebagai gerak *controller* dan *Parrot AR Drone 2.0* sebagai implementasinya. *Parrot AR Drone* terhubung dengan *ground station* melalui *Wi-Fi* dan *leap motion* terhubung dengan *ground station* dengan *USB port*. Kontrol gerak *Leap motion* menggunakan gerakan tangan dan *relays* ke *ground station*. *Ground station* menjalankan *ROS (Robot Operating System)* di *linux* yang digunakan sebagai *platform* dalam implementasi. *Python* sebagai bahasa program yang digunakan untuk menghubungkan antara *AR Drone* dengan gerakan tangan yang sederhana.

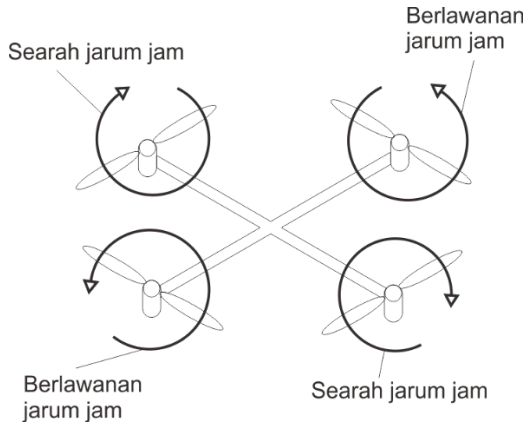
Berdasarkan penelitian-penelitian tersebut dapat disimpulkan bahwa banyak cara dalam mengimplementasikan *NUI* dalam *quadcopter*. Pada penelitian Hadi, Setyawan & Maulana (2017), *NUI* diimplementasikan dengan menggunakan *microsoft kinect* sebagai alat pendeteksi gerakan manusia namun kekurangannya *microsoft kinect* yang terlalu besar masih dirasa kurang praktis untuk penggunaan secara terus menerus. Dengan *leap motion* yang ukurannya lebih kecil dari *microsoft kinect* sehingga menjadi lebih praktis untuk penggunaan kendali *quadcopter*. Selanjutnya pada penelitian Pribadi, Jonemaro & Setyawan (2017) hanya 4 dari 6 gerakan kepala yang bisa diimplementasikan untuk menggerakkan *quadcopter* sehingga ada beberapa gerakan *quadcopter* yang tidak bisa dilakukan, hal ini tentunya akan sangat berpengaruh pada performa sistem, sehingga dengan penggunaan *leap motion* diharap seluruh gerakan *quadcopter* bisa diimplementasikan. Implementasi NUI pada *quadcopter* juga pernah dilakukan pada penelitian Pallas, Setyawan & Prasetio (2017) mengggunakan suara. Namun pada sistem ini memiliki kelemahan yaitu *delay* yang sangat lama, seiring dengan bertambahnya jumlah kata yang diberikan oleh pengguna, dengan *leap motion delay* yang dihasilkan kurang dari 3 detik dari pada penggunaan suara sebagai sistem kendali *quadcopter* karena proses deteksi gerakan tangan pada *leap motion* hampir sama dengan proses deteksi gerakan tubuh pada *microsoft kinect* dengan *delay* dibawah 1 detik sehingga relative cepat dari pada menggunakan kontrol suara yang menghasilkan *delay* 3 detik. Pada penelitian ini dilakukan implementasi *NUI* pada robot. Robot yang digunakan berbeda dengan yang digunakan oleh Pititeeraphab, et al. (2016), yang menggunakan *leap motion* sebagai pengendali lengan robot, pada penelitian ini digunakan *quadcopter* sebagai objek penelitian. Sebagai pembeda dari penelitian Sarkar, et al. (2016), yang meneliti pengendalian *quadcopter* menggunakan *leap motion* dengan *ROS* pada *platform Linux* sedangkan penelitian ini menggunakan pemrograman *javascript* sebagai bahasa pemrograman untuk mengendalikan *quadcopter* dengan *platform Windows*.

2.2 Dasar Teori

Pada dasar teori akan membahas tentang teori-teori yang berkaitan dengan penelitian yang dibuat diantaranya *quadcopter, natural user interface, leap motion,* serta *node js.*

2.2.1 Quadcopter

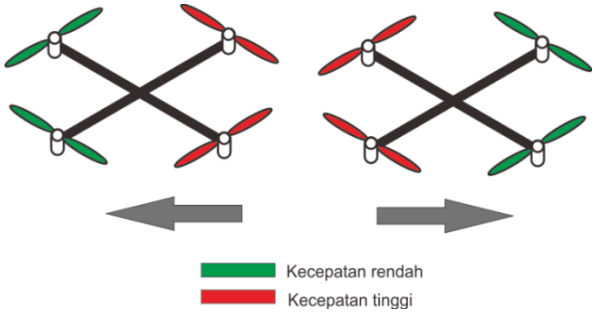
Quadcopter juga dikenal sebagai *quadrotor* merupakan sebuah helikopter yang terdapat empat *rotor* di bagian sisinya. *Rotor* menghadap ke atas dan ditempatkan di formasi persegi dengan jarak yang sama dari pusat massa dari *quadcopter* (Piskorski, et al., 2012). *Quadcopter* dikontrol dengan penyesuaian kecepatan sudut dari *rotor* yang berputar oleh motor listrik. Pada masing-masing motor terpasang baling-baling *(propeller)* untuk membuat aliran udara yang menghasilkan tekanan kearah bawah sehingga timbul gaya angkat pada *quadcopter* sehingga *quadcopter* dapat terbang. Terdapat dua baling-baling berputar searah jarum jam dan dua baling-baling lainnya berputar berlawanan dengan jarum jam seperti terlihat pada Gambar 2.1.



Gambar 2.1 Sistem kerja *rotor quadcopter*

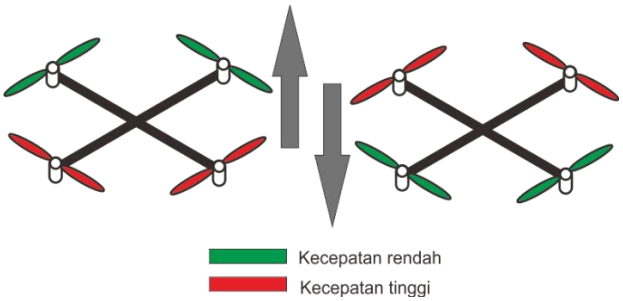
Kecepatan tiap *rotor*nya dapat dikendalikan sesuai keinginan, sehingga *quadcopter* dapat melakukan manuver dan berbagai jenis gerakan, selain itu *quadcopter* dapat melakukan *takeoff* dan mendarat secara vertikal. *Quadcopter* meruapak desain bentuk kecil dari *Unmanned Aerial Vehicles (UAV)* karena struktur yang dimiliki *quadcopter* yang sederhana. *Quadcopter*s digunakan dalam pengawasan, pencarian, penyelamatan, inspeksi konstruksi dan beberapa aplikasi lain. *Quadcopter* memiliki beberapa istilah dalam operasi gerakannya, antara lain:

1. *Roll* yaitu gerakan *quadcopter* yang bertumpu pada sumbu x dan bergerak ke arah kiri dan kanan. Untuk menghasilkan gerakan *roll* terdapat perubahan kecepatan pada dua motor bagian kiri dan dua motor bagian kanan. Sebagai contoh seperti pada Gambar 2.2, saat *quadcopter* ingin bergerak ke kiri maka kecepatan dua motor sebelah kiri akan dikurangi dan kecepatan dua motor bagian kanan akan ditambah, begitu pula sebaliknya (Piskorski, et al., 2012).



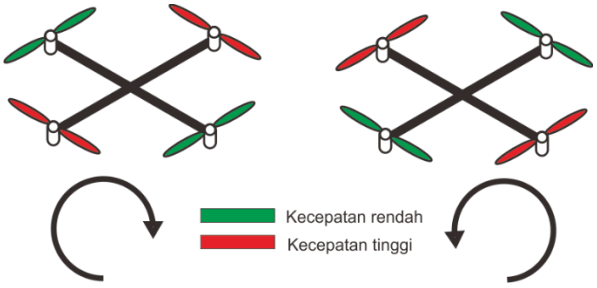
Gambar 2.2 Gerakan *roll quadcopter*

2. *Pitch* yaitu gerakan *quadcopter* yang bertumpu pada sumbu y dan bergerak ke depan dan belakang. Gambar 2.3 mengilustrasikan bahwa untuk menghasilkan gerakan *pitch* terdapat perubahan kecepatan pada dua motor bagian depan dan dua bagian motor belakang. Sebagai contoh saat ingin bergerak ke mundur maka kecepatan dua motor belakang akan dikurangi dan kecepatan dua motor bagian depan akan ditambah, begitu pula sebaliknya (Piskorski, et al., 2012).



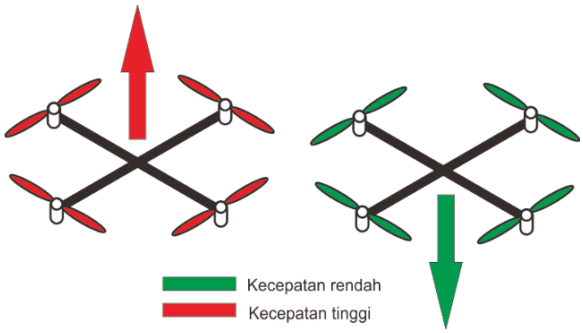
Gambar 2.3 Gerakan *pitch quadcopter*

3. *Yaw* yaitu gerakan *quadcopter* yang bertumpu pada sumbu z dan bergerak berputar ke kanan dan ke kiri. Gerakan ini dilakukan dengan melakukan perubahan kecepatan pada dua motor secara silang seperti pada Gambar 2.4 (Piskorski, et al., 2012).



Gambar 2.4 Gerakan *yaw quadcopter*

4. *Throttle* atau *gaz* yaitu gerakan *quadcopter* yang bertumpu pada sumbu z dan bergerak untuk turun atau naik. Seperti pada Gambar 2.5, saat *quadcopter* turun maka keseluruhan motor akan dikurangi kecepatannya. Sedangkan jika naik maka seluruh kecepatan motor akan naik juga (Piskorski, et al., 2012).



Gambar 2.5 Gerakan *throttle quadcopter*

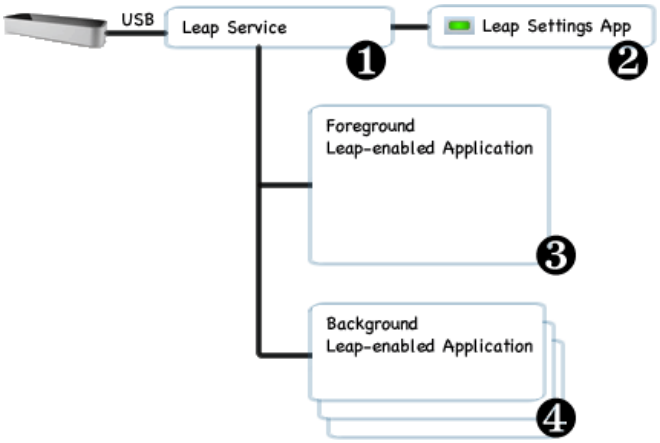
2.2.2 Natural User Interface

NUI (Natural User Interface) merupakan cara baru bagi manusia untuk melakukan komunikasi dengan komputer. Orang biasanya tertarik dengan *interface* yang dapat berinteraksi dengan cara yang sama seperti mereka lakukan. *NUI* dirancang untuk menggunakan kembali kemampuan yang ada untuk saling berinteraksi dengan secara tepat melalui sebuah konten (Ferreira, 2014). Menurut Fernández, et al., (2016) tentang *natural user interfaces* untuk interaksi manusia dengan *quadcopter*. Objek yang diteliti tentang sebuah *quadcopter* yang dapat dikontrol dengan *NUI (Natural User Interfaces)*. *NUI* merupakan pendekatan baru dalam dunia interaksi manusia dengan komputer yang memiliki *input* alami. Didalam *NUI* terdapat beberapa kategori antara lain:

1. *Visual Body Interaction* merupakan sebuah *quadcopter* berinteraksi dengan pilot menggunakan tampilan tubuh sebagai kontrol *quadcopter. Quadcopter* dilengkapi dengan kamera *on-board* lalu menggunakan algoritme citra digital untuk mendeteksi dan melacak.
2. *Visual Marker Interaction* merupakan sebuah *quadcopter* berinteraksi dengan pilot menggunakan sebuah isyarat visual atau sebuah tanda sebagai kontrol *quadcopter.*
3. *Hand Gesture Interaction* merupakan sebuah *quadcopter* berinteraksi dengan pilot menggunakan sebuah gerakan tangan sebagai kontrol *quadcopter* menggunakan *leap motion.*
4. *Speech Command Interaction* merupakan sebuah *quadcopter* berinteraksi dengan pilot menggunakan sebuah suara sebagai kontrol *quadcopter.*

2.2.3 Leap Motion

Leap motion dapat digunakan pada *OS Windows*, *Mac* dan *Linux*. *Leap motion SDK* menyediakan data dari *Leap motion* dengan cara *native application interface*. *Native application interface* menyediakan *library* dinamis. *Library* akan terhubung dengan *Leap motion* dan menyediakan data *tracking* dari aplikasi. *Library* ini dapat diguakan dalam bentuk aplikasi berbasis bahasa pemrograman C++ atau melalui bahasa pemrograman *Javascript*, *C#* dan *Python*. Berikut merupakan skema sistem kerja *leap motion* pada sebuah aplikasi terlihat pada Gambar 2.6 (Inc, 2013).



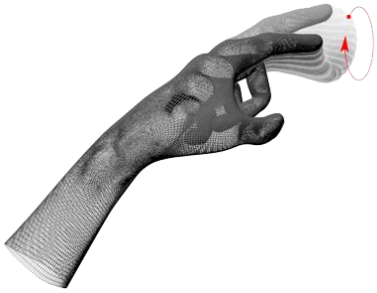
Gambar 2.6 Skema *leap motion* dalam sebuah aplikasi
Sumber : (Inc, 2013)

1. Layanan *leap motion* menerima data dari *leap motion controller* melewati *USB*. Proses ini mengirim informasi bahwa Leap-enabled applications sedang berjalan. Secara *default*, layanan akan mengirim data *tracking* untuk *foreground application*. Namun, aplikasi dapat meminta mereka menerima data di *background* (permintaan yang dapat ditolak oleh pengguna).
2. Aplikasi *leap motion* memungkinkan pengguna komputer untuk mengkonfigurasi instalasi *leap motion* untuk berjalan secara terpisah dari layanan. Aplikasi *leap motion* adalah sebuah *control panel* dalam aplikasi menu bar di *mac OS X* dan *windows*.
3. Ketika *foreground leap-enabled applications* menerima data gerakan *tracking* dari layanan. Sebuah *leap-enabled applications* dapat terhubung dengan layanan *leap motion* menggunakan *leap motion* native *library*. Aplikasi dapat terhubung langsung dengan *leap motion* native *library* dengan bahasa pemrograman C++, *javascript*, *python* dan *C#*.
4. Ketika *background leap-enabled application* kehilangan koneksi dengan sistem operasi, layanan *leap motion* akan menghentikan pengiriman data yang sedang dilakuakan. Aplikasi yang dimaksudkan dapat bekerja secara *background* dan dapat meminta izin untuk menerima data bahkan ketika bekerja secara *background*.

2.2.3.1 *Hand Tracking*

Pada *Leap motion* menetapkan pola gerakan tertentu sebagai gerakan yang bisa menunjukkan maksud pengguna atau instruksi. Perangkat lunak *leap motion* menampilkan gerak-gerik tangan yang diamati dalam sebuah *frame* dan sesuai dengan gerak-gerik tangan secara nyata seperti data tangan atau jari. Pola gerakan berikut merupakan gerakan yang ditetapkan oleh *leap motion* (Inc, 2013).

a. *Circle* merupakan sebuah gerakan dengan satu jari yang bergerak membentuk lingkaran. Dapat menggunakan jari mana saja atau peralatan yang lain untuk membuat gerakan lingkaran dapat dilihat pada Gambar 2.7.



Gambar 2.7 Gerakan lingkaran dengan jari telunjuk
Sumber : (Inc, 2013)

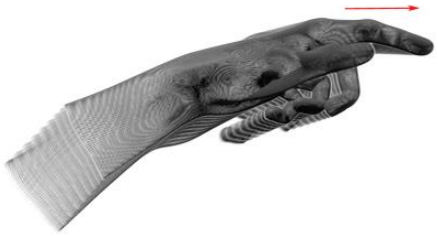
b. *Swipe* merupakan sebuah gerakan dengan jari berpindah secara linier dan lama. Gerakan *swipe* dapat menggunakan jari apa saja dan secara terus menerus dapat dilihat pada Gambar 2.8.

Gambar 2.8 Gerakan menggesek secara horisontal
Sumber : (Inc, 2013)

c. *Key Tap* merupakan sebuah gerakan menekan seperti menekan sebuah *keyboard*. Hanya satu gerakan objek yang dapat ditambahkan setiap gerakan dapat dilihat pada Gambar 2.9.

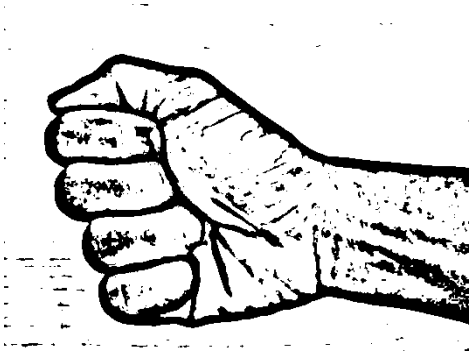
Gambar 2.9 Gerakan menekan tombol dengan jari telunjuk
Sumber : (Inc, 2013)

d. *Screen Tap* merupakan sebuah gerakan menekan seperti menekan layar komputer secara vertikal. Hanya satu gerakan objek yang dapat ditambahkan setiap gerakan dapat dilihat seperti pada Gambar 2.10.



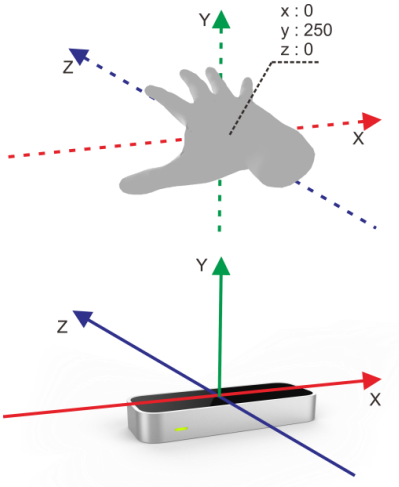
Gambar 2.10 Gerakan ketuk layar dengan jari telunjuk
Sumber : (Inc, 2013)

e. *Handheld* merupakan sebuah gerakan tangan menggenggam dengan posisi ibu jari menempel sejajar dengan jari telunjuk seperti terlihat pada Gambar 2.11.



Gambar 2.11 Gerakan *handheld*

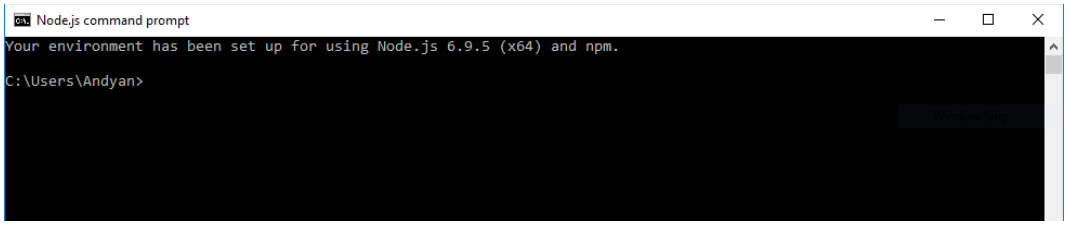
f. *Position* merupakan merupakan sebuah nilai yang berisi nilai sumbu x, y, z dari perpindahan tangan dari suatu titik menuju ke titik yang lain seperti pada Gambar 2.12.



Gambar 2.12 Nilai *position*

2.2.4 Node js

Node js merupakan suatu *platform* yang pembuatannya menggunakan *javascript engine*. *Node js* dibuat dengan tujuan memperluas penggunaan *javascript* tidak hanya dipakai pada *platform website* saja, akan tetapi dapat digunakan berbagai *platform*.



Gambar 2.13 *Node js Command prompt*

Pada Gambar 2.13 merupakan untuk menjalankan instruksi *node* digunakan *node js command prompt*. Pada penelitian ini *node js* digunakan sebagai tempat program yang bertujuan untuk mengendalikan *quadcopter*. Pada *node js* terdapat kumpulan *node* yang disebut dengan *Node Package Manager* atau yang biasa di singkat dengan nama *NPM.* Komponen *NPM* terintegrasi satu sama lain dan menghasilkan berbagai instruksi *quadcopter* seperti *hover*, mendarat, atas, bawah, ke depan, dan lain-lain. Dalam setiap instruksi tersebut dapat dijalankan secara bersama-sama sehingga mempermudah *user* dalam pengoperasiannya.

2.2.5 AT Command

AT Command atau yang juga dikenal sebagai *Hayes Standard AT Command*, merupakan sebuah standar petunjuk untuk mengendalikan dan mengonfigurasi modem. Hayes Microcomputer Products yang merupakan pelopor pengembangan di bidang modem, dan digunakan secara keseluruhan atau sebagian oleh hampir setiap produsen modem untuk digunakan dengan *PC* (Linux, 2005).

AT Command adalah sekelompok instruksi yang menggunakan *serial port* untuk komunikasinya. Penggunaan *AT Command* umumnya terdapat pada penggunaan *handphone* dan modem. Dengan menggunakan *AT Command,* maka dapat dijalankan beberapa fungsi seperti membaca kekuatan sinyal, melihat vendor dari modem atau *handphone* yang digunakan, mengirim dan memperbarui SMS, dan sebagainya. *AT Command* pada dasarnya mirip dengan instruksi *prompt* pada *DOS* yaitu instruksi-instruksi yang digunakan untuk penulisan ke *port* komputer (Hadi, et al., 2017).

Format penulisan *AT Command* selalu dilakukan dengan kata *AT* yang merupakan singkatan dari *attention*, lalu diikuti dengan karakter lainnya yang memiliki fungsi masing-masing. Selain digunakan untuk penulisan ke *port*, *AT Command* juga dapat digunakan untuk penulisan instruksi ke modem. Beberapa contoh penulisan *AT Command* adalah sebagai berikut:

1. AT berfungsi sebagai kondisi *port* apakah *port* siap untuk berkomunikasi.

2. AT+CGMI sebagai instruksi untuk mengetahui merek *handphone.*
3. AT+CMGR sebagai instruksi untuk membaca isi dari SMS.

2.2.6 Node Package Manager

Node package manager adalah sekumpulan instruksi yang ada pada *node js*, fungsinya sebagai mempermudah para pengembang *javascript* dalam membangun suatu sistem dengan berbagai *code* yang disediakan, lalu *code* tersebut dapat digunakan untuk suatu penelitian. Penggunaan *NPM* dalam kontrol *quadcopter* berupa *Node AR Drone* 2.0 (Foundation, 2017).

2.2.6.1 *Node AR Drone 2.0*

Node AR Drone 2.0 merupakan sekumpulan instruksi yang digunakan untuk mengeksekusi atau memberikan instruksi pada *quadcopter*, seperti *takeoff*, mendarat, ataupun mendapatkan data sensor yang terdapat pada *quadcopter*. *Node AR Drone* ditulis menggunakan bahasa *javascript*, serta dapat membantu *user* untuk mengembangkan penelitian tentang *AR Drone 2.0*. Dalam pengembangannya *node AR Drone* mampu menyediakan berbagai macam instruksi yang ditujukan untuk pergerakan *quadcopter* dengan mengintegrasikan pergerakan utama yaitu *pitch, roll*, *yaw dan gaz*.