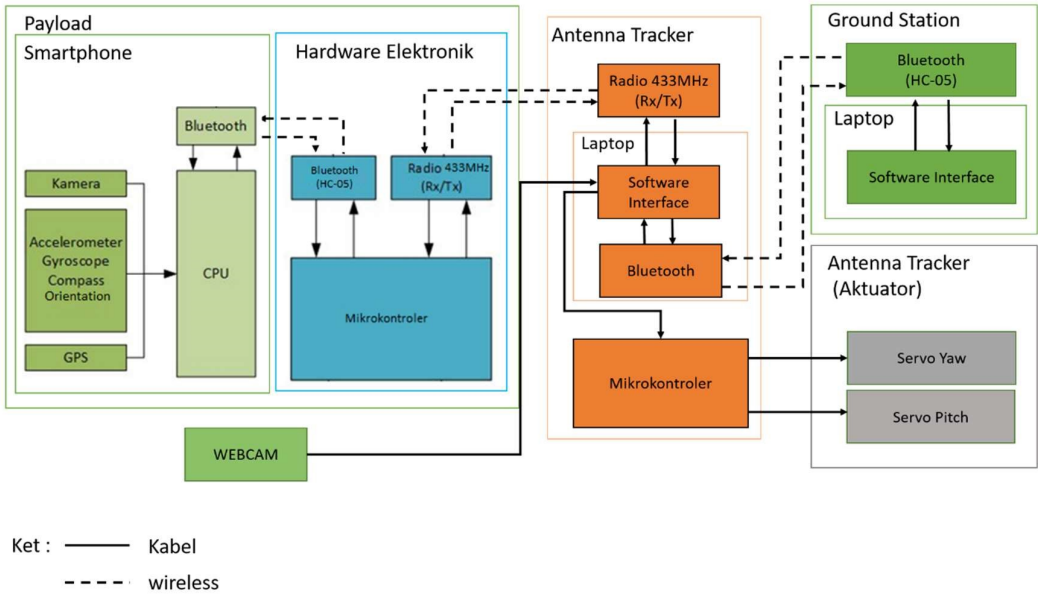
BAB III

PERANCANGAN SISTEM

Pada bab ini akan membahas mengenai perancangan sistem. Perancangan ini akan terbagi menjadi 4 bagian utama, yaitu: blok diagram sistem, strategi baru dari *antenna tracker*, pemilihan komponen, perancangan mekanik, perancangan *hardware* dan perancangan *software*.

3.1 Blok Diagram Sistem

Adapun gambaran dari blok diagram sistem yang akan dirancang adalah sebagai berikut.



Gambar 3.1 Blok diagram sistem

Keterangan blok diagram diatas adalah sebagai berikut:

1. *Payload*

   Berfungsi untuk mengirim data sensor *accelerometer, gyroscope, orientation,* dan GPS dari *smartphone*.

2. *Hardware* Elektronik

   Berfungsi sebagai perubah komunikasi dari *smartphone* yang asalnya menggunakan *bluetooth* diubah menjadi komunikasi radio 433MHz.

3. *Webcam*

   Berfungsi untuk menangkap objek warna.

4. Radio 433MHz

   Berfungsi untuk mengirimkan dan menerima data menggunakan frekuensi radio di 433 MHz.

5. Laptop (*Antenna tracker*)

   Berfungsi sebagai pemproses sistem pengolahan citra untuk penggenalan objek berwarna, sebagai pengolahan data sensor yang diterima kemudian dikirim kembali kepada *ground station* dan pemproses data GPS yang menjadi pergerakkan *antenna tracker* terhadap *payload.*

6. *Bluetooth* (pada Laptop *Antenna tracker*)

   Berfungsi untuk mengirimkan dan menerima data menggunakan frekuensi *bluetooth* yang ada dimiliki laptop.

7. Mikrokontroler

   Berfungsi sebagai media proses yang mengontrol pergerakkan servo *pitch* dan *yaw.*

8. Servo *Pitch*

   Berfungsi untuk pergerakkan *antenna tracker vertical*.

9. Servo *Yaw*

   Berfungsi untuk pergerakkan *antenna tracker horizontal*.

10. Laptop (*Ground Station*)

    Berfungsi untuk menerima data yang dikirim dari *payload* untuk di *monitoring*.

11. *Bluetooth Module*

    Berfungsi untuk mengirimkan dan menerima data menggunakan frekuensi *bluetooth*.

12. USB to TTL

    Berfungsi sebagai alat konverter dari USB menjadi serial, yang mana keluaranya berupa RX dan TX yang dipasangkan dengan *bluetooth module.*

3.2 Strategi Baru pada *Antenna tracker*

Pada sistem *antenna tracker* yang dibuat ditambahkan beberapa fitur dan penambahan algoritma untuk membantu dalam metode *tracking* pada *antenna tracker*. Berikut adalah beberapa fitur dan penambahan algoritma pada sistem yang dibuat:

3.2.1 Membuat Komunikasi antara *Ground Station* dan *Antenna Tracker* Menjadi *Wireless*

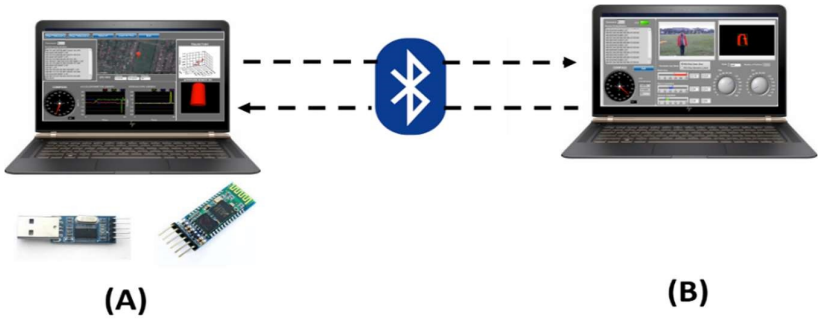
Komunikasi yang digunakan pada perancangan ini adalah menggunakan frekuensi *bluetooth*. Alasan menggunakan *bluetooth* adalah

supaya membuat komunikasi secara *wireless,* sebab pada KOMURINDO 2016 penggunaan kabel memiliki resiko banyak sekali seperti terinjak, ketarik dan membuat pergerakkan *antenna tracker* menjadi terhambat seperti Gambar 3.2.



Gambar 3.2 Penggunaan kabel pada sistem sebelumnya

Maka dari itu komunikasi menggunakan *wireless* diperlukan untuk mengurangi resiko tersebut. Pada komunikasi ini laptop *ground station* menggunakan *bluetooth module* yang dipasangkan dengan USB to TTL sedangkan pada laptop *antenna tracker* menggunakan *bluetooth* adapter bawaan pada laptop seperti pada Gambar 3.3



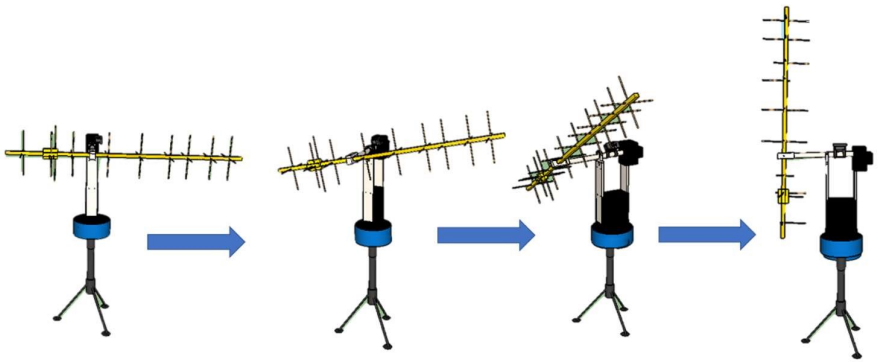
Gambar 3.3 Ilustrasi komunikasi antar laptop *antenna tracker* dan laptop *ground station*

Keterangan :(A) Laptop GS menggunakan *bluetooth* module dan USB to TL

(B) Laptop *antenna tracker* menggunakan *bluetooth* bawaan dari laptop

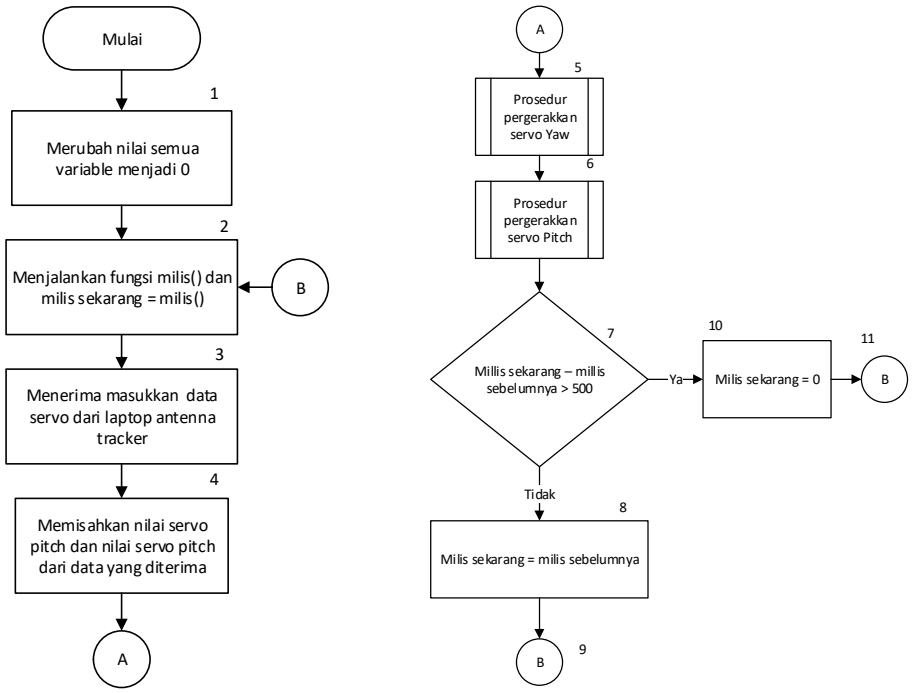
3.2.2 Memperbaiki Algoritma pada Mikrokontroler

Dalam Algoritma pada mikrokontroler di tambahkan fungsi *milis()* atau penambahan *delay*. Fungsi *millis()* ini bekerja dengan menghitung waktu yang digunakan oleh mikrokontroler untuk mengeksekusi setiap program yang berjalan. Jadi, setiap mili detik mikrokontroler bekerja, maka setiap mili detik tersebut nilainya akan dibaca oleh fungsi *millis()* ini. Fungsi ini sebenarnya membaca nilai yang berasal dari *internal clock* yang ada pada ic mikrokontroler yang ada pada Arduino. Dengan menggunakan fungsi milis, mikrokontroler dapat bekarja secara paralel karena mikrokontroler Arduino tidak biasa bekerja secara paralel untuk mengontrol 2 servo secara bersamaan dan juga penggunaan milis ini juga dapat mengurangi momentum pada servo sehingga *gear* pada servo tidak cepat rusak atau kehilangan akurasinya.



Gambar 3.4 Ilustrasi pergerakkan *antenna tracker*

Berikut *Flowchart* dari algoritma yang telah dibuat:



Gambar 3.5 *Flowchart* program utama pada mikrokontroler

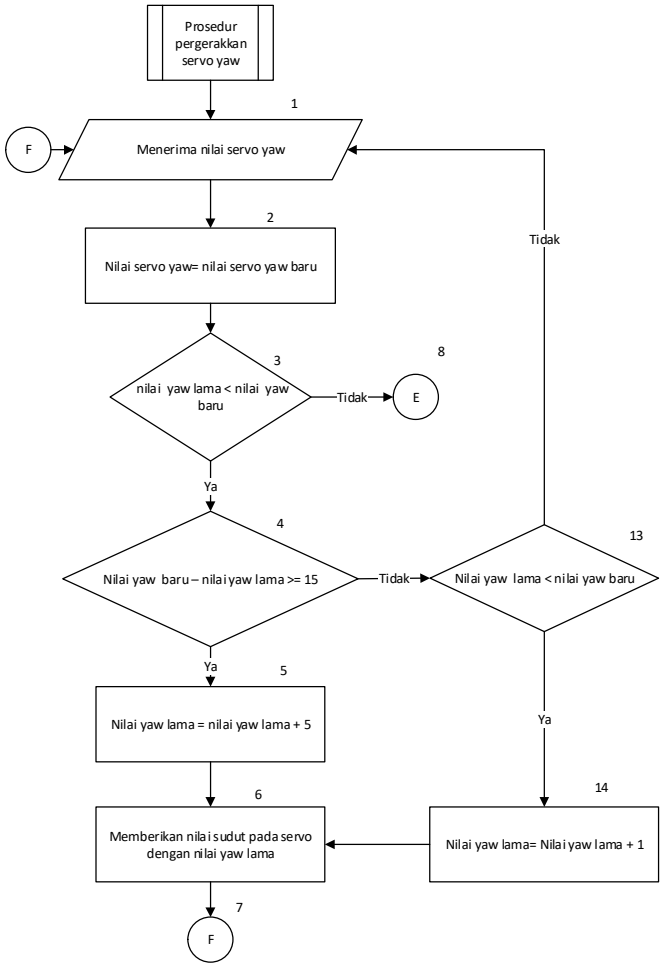
Berikut penjelasan *flowchart* pada Gambar 3.5:

1. Mengubah nilai semua variable menjadi 0 seperti nilai *yaw* lama dan *pitch* lama diubah menjadi 0.
2. Menjalankan fungsi milis atau menghitung setiap kali program berulang dan menjadikan milis sekarang = jumlah setiap kali program berulang.
3. Menerima masukkan data servo dari laptop *antenna tracker*. Contoh format data yang dikirim dari laptop *antenna tracker* ke mikrokontroler seperti tabel 3.1.

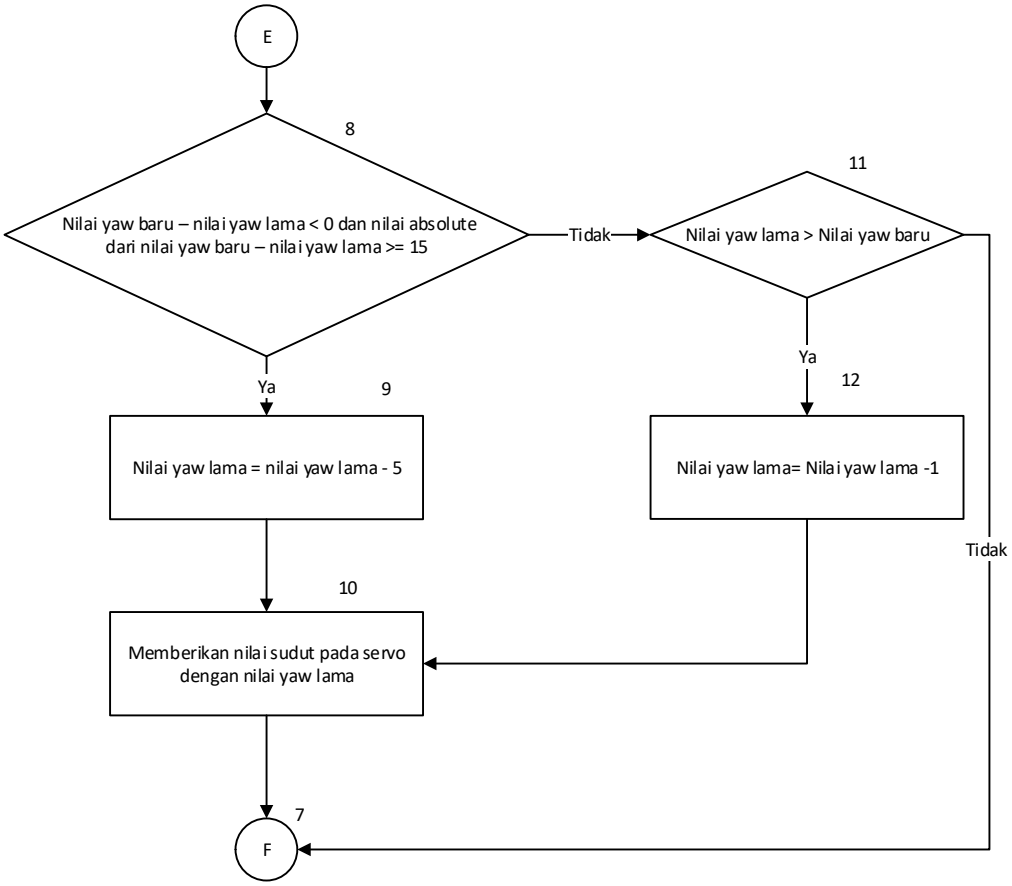
Tabel 3.1 Format pengiriman data AT

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
3	0		4	0
Servo *Yaw*			Servo *Pitch*	

4. Memisahkan nilai *pitch* dan nilai servo *yaw* dari data yang diterima.
5. Masuk pada prosedur untuk pergerakkan servo *yaw.*
6. Masuk pada prosedur untuk pergerakkan servo *pitch.*
7. Membandingakan hasil Milis sekarang – milis sebelumnya > 500
8. Jika benar, milis sekarang = 0 atau mengulang lagi jumlah loop yang sudah dihitung menjadi 0 dan masuk ke no 9. Jika salah, masuk ke no 10.
9. Kembali ke no 1.
10. Milis sekarang = nilai milis sebelumnya dan lanjut ke no 11.
11. Kembali ke no 1.



Gambar 3.6 *Flowchart* sub program prosedur pengerak servo *yaw*

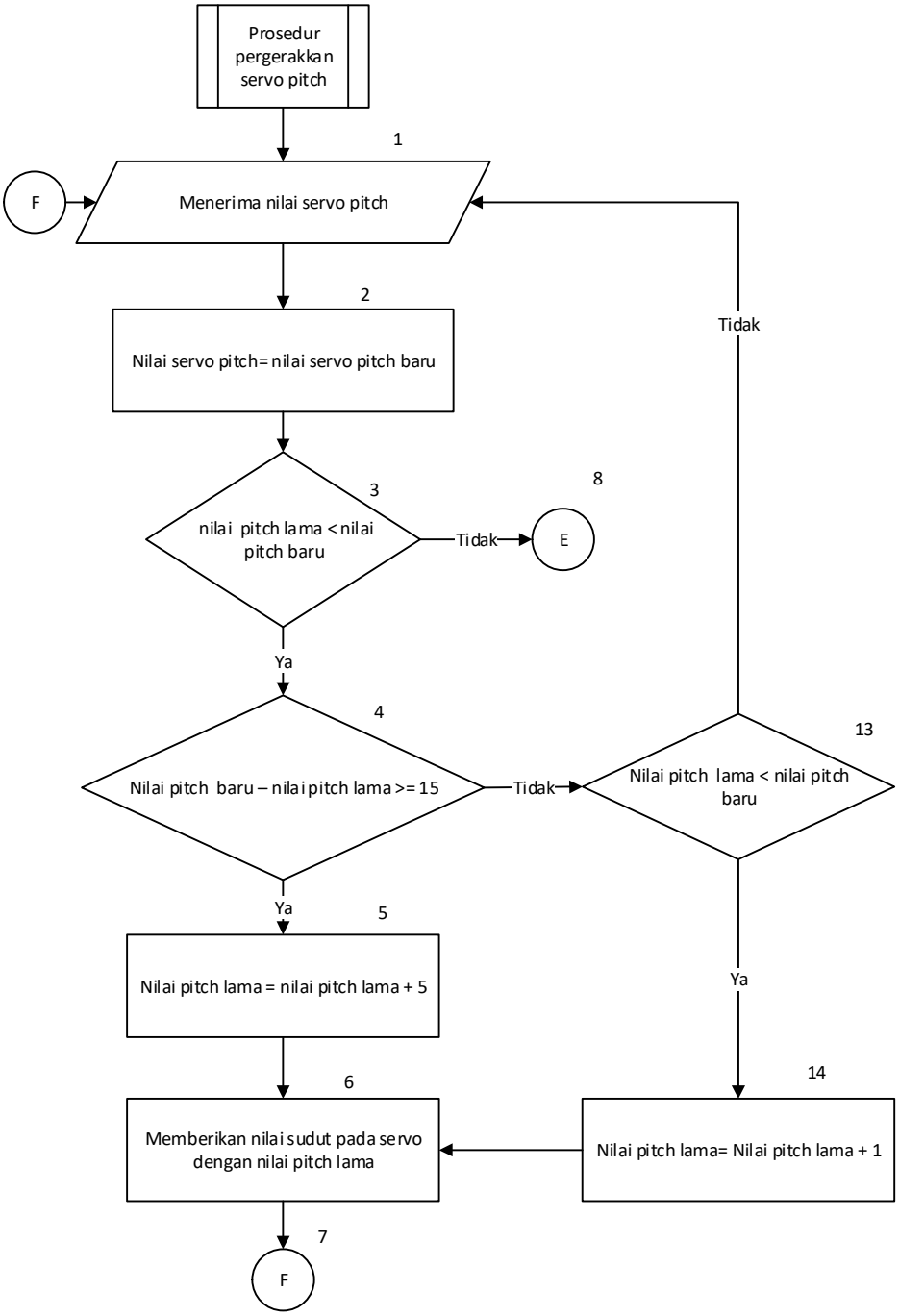


Gambar 3.7 *Flowchart* lanjutan sub program prosedur pengerak servo *yaw*

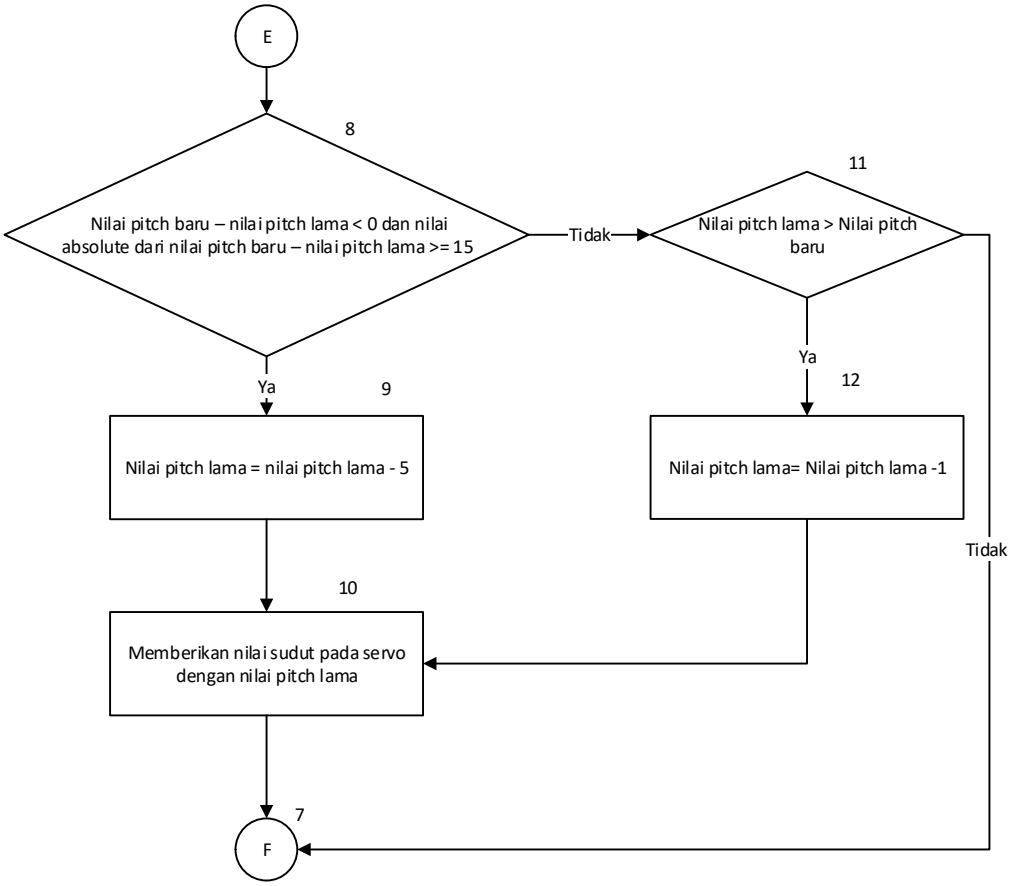
Penjelasan *flowchart* pada Gambar 3.6 dan Gambar 3.7

1. Menerima masukkan nilai servo *yaw*.
2. Menjadikannya variable nilai masukkan nilai *yaw* baru.
3. Membandingkan nilai *yaw* lama dengan nilai *yaw* baru apakah nilai *yaw* lama lebih besar dari nilai *yaw* baru. Jika benar, masuk ke no 4 atau jika salah, masuk ke no 8.
4. Membandingkan nilai *yaw* baru dikurang dengan nilai *yaw* lama lebih besar sama dengan 15. Jika benar, maka masuk ke no 5 dan jika salah, masuk ke no 13.
5. Menjadikan nilai *yaw* lama sama dengan nilai *yaw* lama ditambah 5.

6. Memberikkan nilai sudut pada servo dengan nilai *yaw* lama.
7. Kembali ke no 1.
8. Membandingkan nilai *yaw* baru – nilai lebih kecil dan nilai absolute dari nilai *yaw* baru – nilai *yaw* lama lebih besar sama dengan 15. Jika benar, masuk ke no 9. jika salah, masuk ke no 11.
9. Menjadikan nilai *yaw* lama dengan nilai *yaw* lama dikurang 5
10. Memberikkan nilai sudut pada servo dengan nilai *yaw* lama.
11. Membandingkan nilai *yaw* lama lebih besar dengan nilai *yaw* baru. Jika benar, masuk ke no 12. Jika salah, masuk ke no 7.
12. Menjadikan nilai *yaw* lama sama dengan nilai *yaw* lama dikurang 1 dan masuk ke no 10.
13. Membandingkan nilai *yaw* lama lebih kecil dari nilai *yaw* baru. Jika benar masuk ke no 14. Jika salah, masuk ke no 1
14. Menjadikan nilai *yaw* lama sama dengan nilai *yaw* lama ditambah 1 dan masuk ke no 6.



Gambar 3.8 *Flowchart* sub program prosedur pengerak servo *pitch*



Gambar 3.9 *Flowchart* lanjutan sub program prosedur pengerak servo *pitch*

Berikut penjelasan tentang *flowchart* pada Gambar 3.8 dan Gambar 3.9

1. Menerima masukkan nilai servo *pitch*.
2. Menjadikannya *variable* nilai masukkan nilai *pitch* baru.
3. Membandingkan nilai *pitch* lama dengan nilai *pitch* baru apakah nilai *pitch* lama lebih besar dari nilai *pitch* baru. Jika benar, masuk ke no 4 atau jika salah, masuk ke no 8.
4. Membandingkan nilai *pitch* baru dikurang dengan nilai *pitch* lama hasilnya lebih besar sama dengan 15. Jika benar, maka masuk ke no 5 dan jika salah, masuk ke no 13.
5. Menjadikan nilai *pitch* lama sama dengan nilai *pitch* lama ditambah 5.

6. Memberikkan nilai sudut pada servo dengan nilai *pitch* lama
7. Kembali ke no 1
8. Membandingkan nilai *pitch* baru – nilai lebih kecil dan nilai absolute dari nilai *pitch* baru – nilai *pitch* lama lebih besar sama dengan 15. Jika benar, masuk ke no 9. jika salah, masuk ke no 11.
9. Menjadikan nilai *pitch* lama dengan nilai *pitch* lama dikurang 5
10. Memberikkan nilai sudut pada servo dengan nilai *pitch* lama
11. Membandingkan nilai *pitch* lama lebih besar dengan nilai *pitch* baru. Jika benar, masuk ke no 12. Jika salah, masuk ke no 7
12. Menjadikan nilai *pitch* lama sama dengan nilai *pitch* lama dikurang 1 dan masuk ke no 10.
13. Membandingkan nilai *pitch* lama lebih kecil dari nilai *pitch* baru. Jika benar masuk ke no 14. Jika salah, masuk ke no 1
14. Menjadikan nilai *pitch* lama sama dengan nilai *pitch* lama ditambah 1 dan masuk ke no 6.

3.2.3 Penambahan Sistem *Locking* Data GPS

Pada *software ground station* KOMURINDO 2016, belum ada sistem *locking* data GPS sehingga pergerakkan pada *antenna tracker* menjadi eror atau tidak mengarah kepada *payload* dan jalur yang terbentuk pada *map ground station* menjadi berantakkan dapat dilihat pada Gambar 3.10. Terjadinya eror dikarenakan data yang diterima dari *payload* tidak lengkap. Karena itu ditambahkan sistem *locking* data GPS pada saat diterimanya data dari *payload*.



Gambar 3.10 Contoh saat tidak menggunakan sistem *locking* data GPS



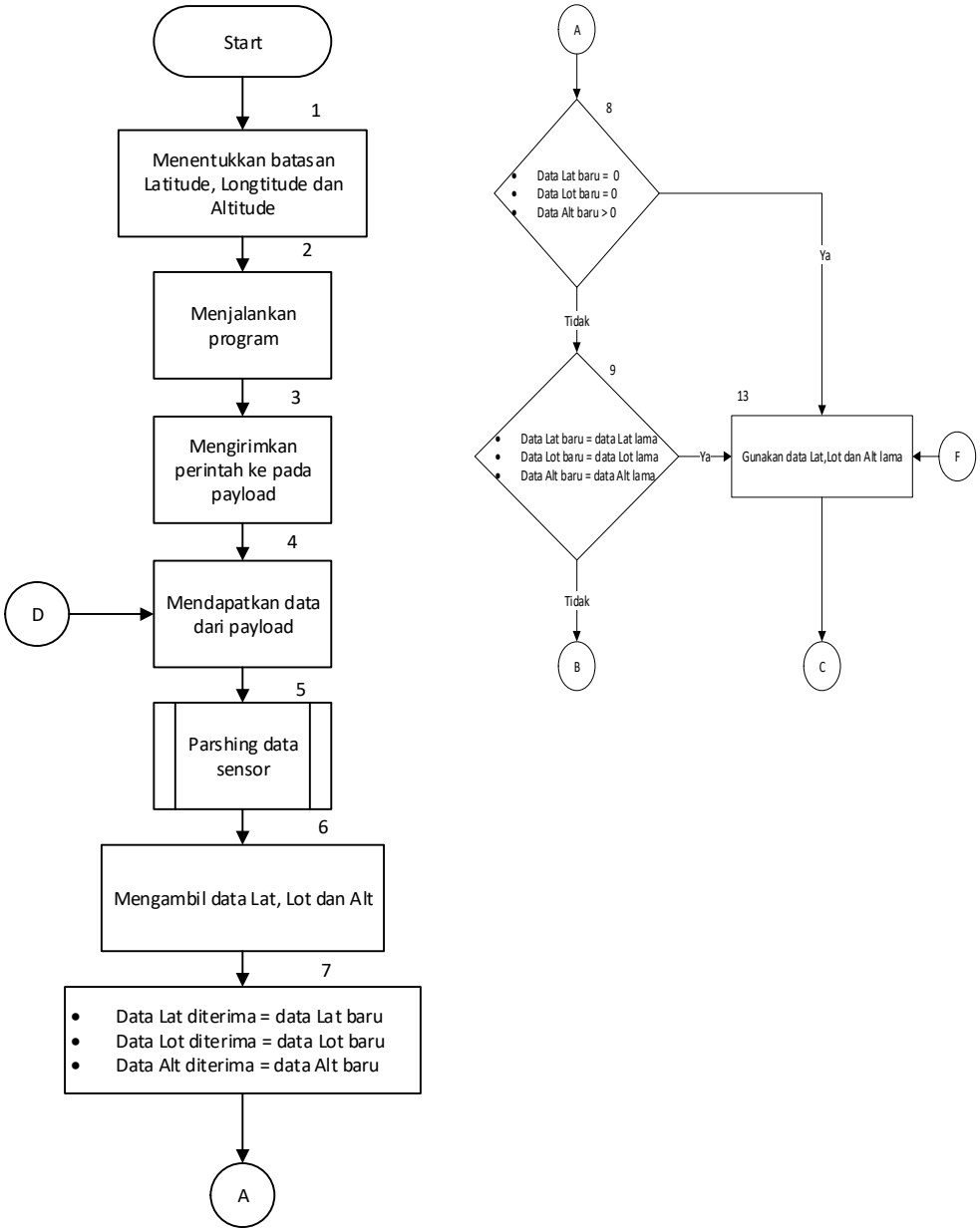
Gambar 3.11 Contoh saat menggunakan sistem *locking* data GPS

Data yang diterima dari *payload* dengan format berikut: *header code*, *accelerometer* sumbu X, *accelerometer* sumbu Y, *accelerometer* sumbu Z, *gyroscope* sumbu X, *gyroscope* sumbu Y, *gyroscope* sumbu Z, data *roll*, data *pitch*, data *yaw*, *latitude*, *longtitude* dan *altitude*. Sebelumnya data dikirim, pada *payload* semuanya ditambah 500 dan akan dikurangi 500 saat diterima. Contoh data yang dikirim dari *payload* seperti tabel 3.2:

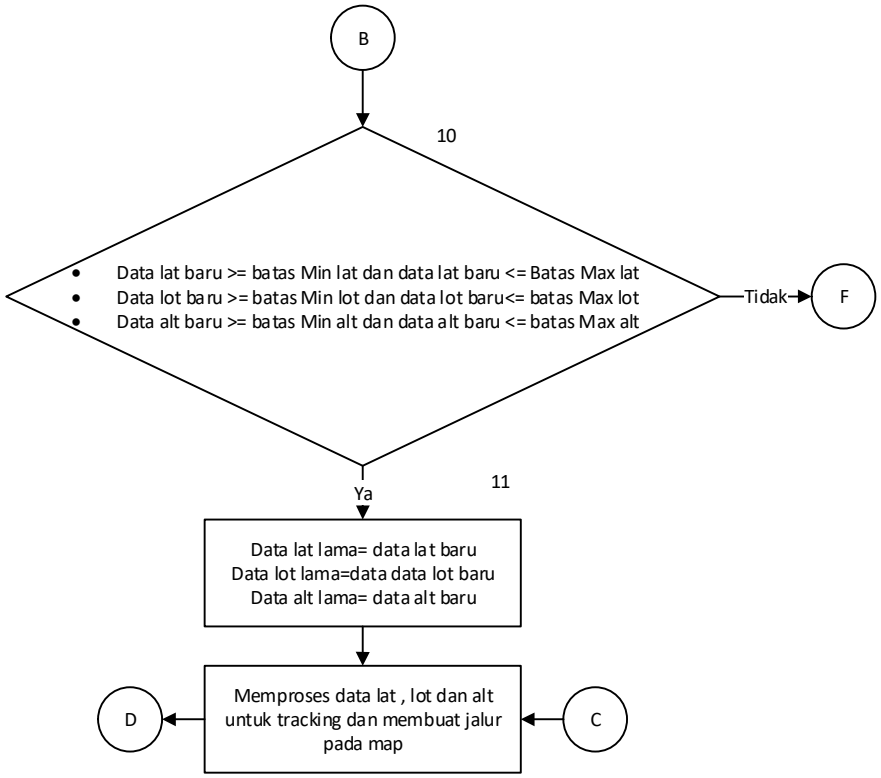
Tabel 3.2 Format data yang dikirim payload

Header code	004
Acc X	524
Acc Y	508
Acc Z	591
Gyro X	501
Gyro Y	500
Gyro Z	500
Roll	495
Pitch	515
Yaw	859
Latitude	492.34117
Longtitude	607.68959
Altitude	514

Berikut adalah *Flowchart* dari sistem *locking* data GPS:



Gambar 3.12 *Flowchart* sistem locking data GPS

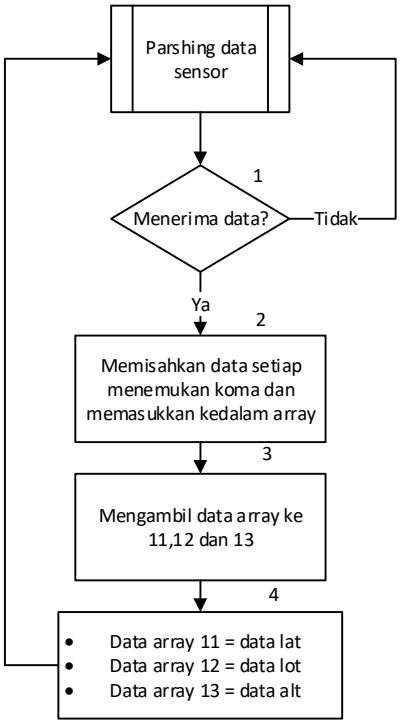


Gambar 3.13 *Flowchart* lanjutan dari sistem locking data GPS

Berikut penjelasan tentang *flowchart* dari Gambar 3.12 dan Gambar 3.13:

1. Menentukkan batasan untuk *latitude, longtitude* dan *altitude*.
2. Menjalankan program *antenna tracker* dan *ground station.*
3. Mengirimkan perintah kepada *payload*. perintah yang dikirim berupa angka contoh:
   - Perintah ‘1’ adalah perintah untuk mengirimkan sensor dari *payload.*
   - Perintah ‘5’ adalah perintah untuk menghentikan pengiriman data sensor dari *payload.*
4. Menerima data sensor dari *payload.*
5. Masuk ke sub program *parshing* data.
6. Mengambil data Lat, Lot dan Alt.

7. Menjadikan data Data Lat diterima = data Lat baru, Data Lot diterima = data Lot baru dan Data Alt diterima = data Alt baru.
8. Membandingkan Data Lat baru = 0, Data Lot baru = 0 dan Data Alt baru > 0. Jika benar, masuk ke no 13. jika salah, masuk ke no 9.
9. Membandingkan Data Lat baru = data lat lama, Data Lot baru = data lot lama dan Data Alt baru = data alt lama. Jika benar masuk ke no 13. Jika salah masuk ke no 10
10. Membandingkan Data lat baru >= batas Min lat dan data lat baru <= Batas Max lat, Data lot baru >= batas Min lot dan data lot baru<= batas Max lot, Data alt baru >= batas Min alt dan data alt baru <= batas Max alt. jika benar, masuk ke no 11. Jika salah, masuk ke no 13.
11. Menjadikan data lat lama = data lat baru, data lot lama= data lat baru dan data alt lama = data alt baru.
12. Memproses data lat, lot dan alt untuk *tracking* dan membuat jalur pada *map* dan masuk ke no 4.
13. Menggunakan data lat, lot dan alt lama dan masuk ke no 12.



Gambar 3.14 *Flowchart* sub program parshing

Berikut penjelasan *flowchart* dari Gambar 3.14:

1. Apakah menerima data dari *payload*? jika benar masuk ke no 2. Jika salah, mengulangi lagi dari awal.
2. Memisahkan data setiap menemukan koma dan memasukannya ke dalam *array*.
3. Mengambil data array ke 11, 12 dan 13.
4. Menjadikan data *array* 11 = data lat, data *array* 12 = data lot dan data *array* 13 = data alt.

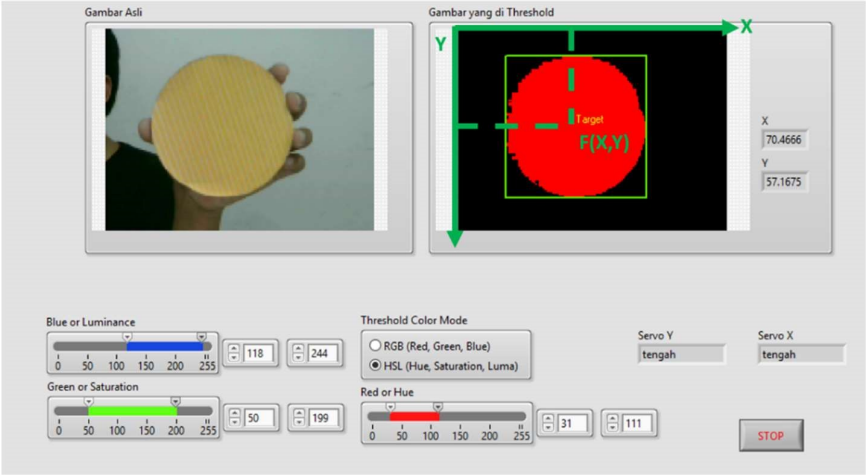
3.2.4 Algoritma Tambahan untuk mode *Tracking*

Untuk membantu mode *tracking* dalam *antenna tracker,* maka diperlukan algoritma tambahan pada sistem *antenna tracker* yaitu algoritma dengan pengenalan warna pada objek. Pada Gambar 3.15 adalah contoh dari program yang telah dibuat, dapat dilihat objek berwarna merah akan ditargetkan untuk referensi posisi dari *payload*.



Gambar 3.15 Contoh penggunaan pengenalan warna objek

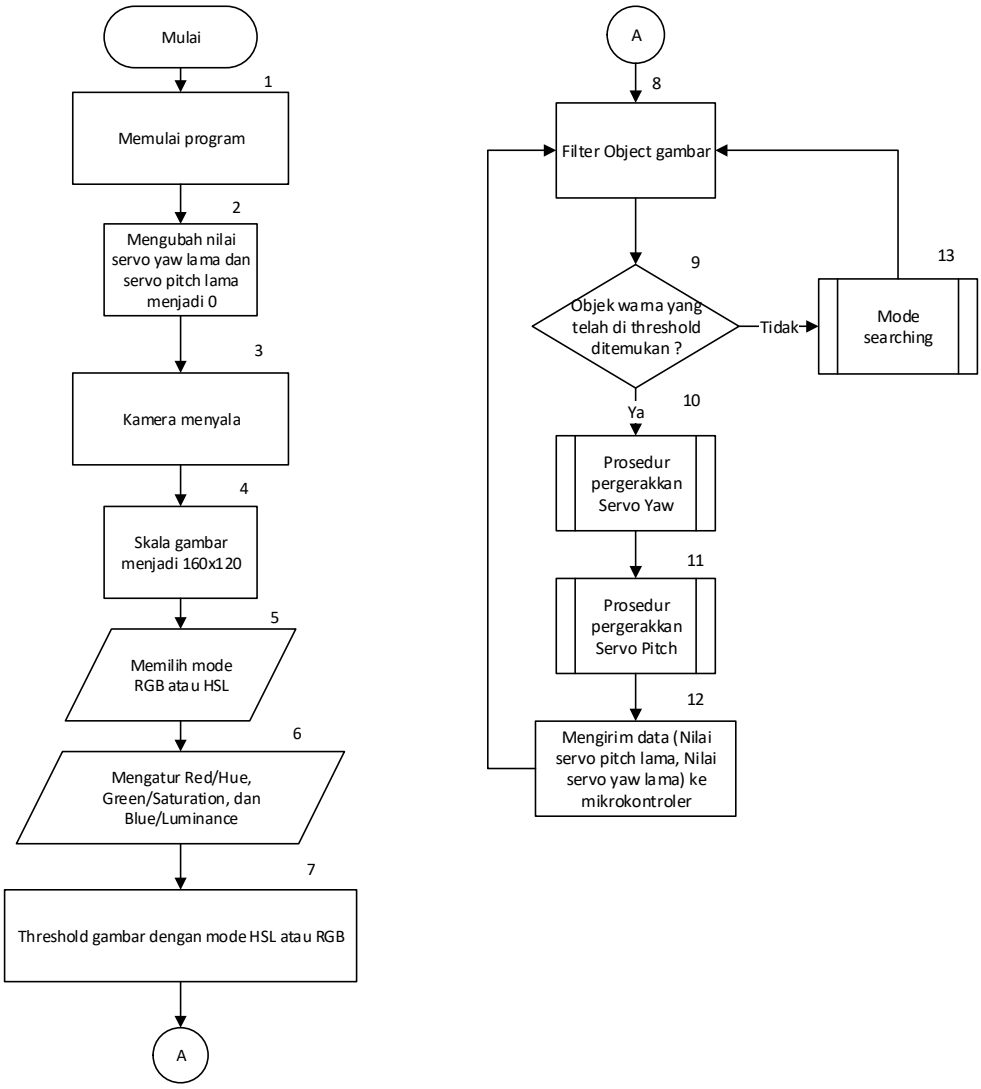
Penentuan posisi dapat didefinisikan sebagai fungsi dua variabel f(x,y), dimana x dan y adalah koordinat spasial dan nilai f(x,y) adalah *brightness level* dari citra pada koordinat tersebut, hal tersebut diilustrasikan pada Gambar 3.16.



Gambar 3.16 Contoh penentuan posisi

Kamera *webcam* yang diatur menjadi resolusi 160x120 pixel, sehingga batasan koordinat x bernilai 0-160 sedangkan nilai batasan koordinat y bernilai 0-120. Untuk menentukan posisi maka diberi batasan setiap posisi, seperti kiri koordinat x harus lebih besar sama dengan 1 dan harus kurang dari 53 maka benda akan berada di posisi kiri pada layar. Dalam Gambar 3.16 adalah contoh jika benda ada di posisi tengah.

Berikut adalah *flowchart* dari program pengenalan warna objek yang telah dibuat:

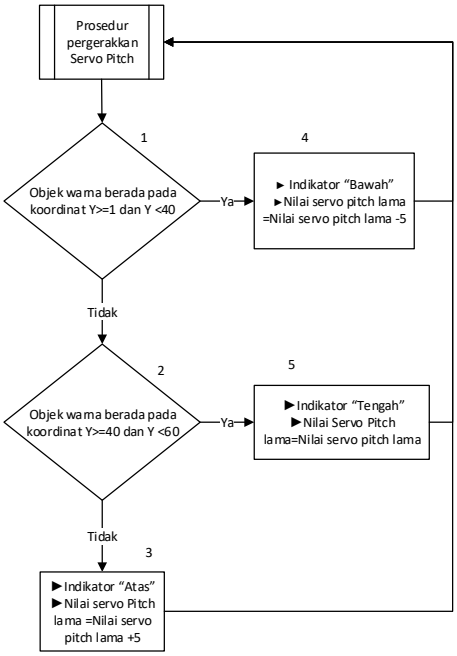


Gambar 3.17 *Flowchart* program utama pengenalan warna objek

Berikut penjelasan *flowchart* pada Gambar 3.17:

1. Memulai menjalankan program *color tracking.*
2. Mengubah nilai variable servo *yaw* lama dan servo *pitch* lama menjadi 0.
3. Menyalakan kamera *webcam.*
4. Merubah resolusi kamera menjadi 160x120.
5. Memilih mode RGB atau HSL untuk mode pengenalan warna.

6. Mengatur batasan nilai *Red* atau *Hue*, *Green* atau *Saturation* dan *Blue* atau *Luminance*.
7. Threshold gambar dengan mode HSL atau RGB.
8. Memfilter objek gambar.
9. Apakah objek warna ditemukan? jika benar masuk ke no 9 dan jika salah masuk ke no 13.
10. Masuk ke prosedur pergerakkan servo *yaw.*
11. Masuk ke prosedur pergerakkan servo *pitch.*
12. Mengirim data nilai servo dengan nilai servo *yaw* lama dan *pitch* lama.
13. Masuk ke prosedur mode *searching*.

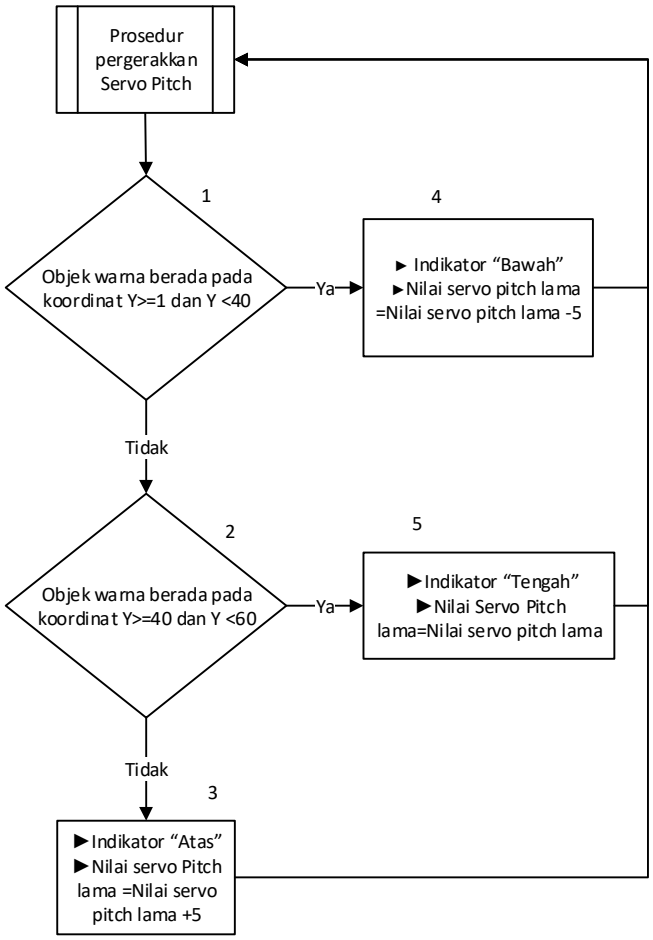


Gambar 3.18 *Flowchart* sub program pergerakkan servo *pitch*

Berikut penjelasan *flowchart* pada Gambar 3.18:

1. Membandingkan objek warna pada koordinat Y lebih besar sama dengan 1 dan Y kurang dari 40. Jiak benar masuk ke no 4 dan jika salah masuk ke no 2.

2. Membandingkan objek warna berada pada koordinat Y lebih besar sama dengan 40 dan Y kuran dari 60. Jika benar masuk ke no 5 dan jika salah masuk ke no 3.
3. Menjadi kan *Indikator* posisi “Atas” dan nilai servo *pitch* lama = nilai servo *pitch* lama +5.
4. Menjadikan *Indikator* posisi “Bawah “dan nilai servo *pitch* lama = nilai *pitch* lama -5.
5. Menjadikan *Indikator* posisi “Tengah “dan nilai servo *pitch* lama = nilai *pitch* lama.



Gambar 3.19 *Flowchart* sub program pergerakkan Servo *yaw*

Berikut penjelasan *flowchart* pada Gambar 3.19:

1. Membandingkan objek warna pada koordinat X lebih besar sama dengan 1 dan X kurang dari 53. Jika benar masuk ke no 4 dan jika salah masuk ke no 2.
2. Membandingkan objek warna berada pada koordinat lebih besar sama dengan 53 dan x kurang dari 90. Jika benar masuk ke no 5 dan jika salah masuk ke no 3.
3. Menjadi kan *Indikator* posisi "Kanan" dan nilai servo *yaw* lama = nilai servo *yaw* lama +5.
4. Menjadikan *Indikator* posisi "Kiri "dan nilai servo *yaw* lama = nilai *pitch* lama -5.
5. Menjadikan *Indikator* posisi "Tengah"dan nilai servo *yaw* lama = nilai *yaw* lama.

3.3 Pemilihan Komponen

Pemilihan komponen merupakan suatu proses awal untuk mengetahui komponen yang akan digunakan pada sistem elektroniknya. Dengan topik bahasan mengenai komponen-komponen yang digunakan, studi komponen dan perancangan sistem.

3.3.1 Mikrokontroler

Mikrokontroler merupakan sebuah chip yang dapat diprogram untuk menghasilkan keluaran yang diinginkan dengan memasukkan program yang digunakan di dalam chip tersebut. Arduino Nano adalah mikrokontroler yang menggunakan chip berjenis AVR atmega 328 yang diproduksi oleh perusahaan atmel, dalam penggunaannya arduino ini mudah digunakan dikarenakan sudah terdapat *bootloader* untuk menangani proses upload program dari komputer.

Tabel 3.3 Perbandingan jenis mikrokontroler

Spesifikasi	Arduino Nano	Arduino UNO	Arduino Mega
Tegangan operasi	5V	5V	5V
Masukkan Tegangan	7-12V	7-12V	7-12V
Digital I/O Pins	12	14	54
Clock Speed	16 MHz	16 MHz	16 MHz
Flash Memory	32 KB *of which* 2 KB *used by bootloader*	32 KB (ATmega328P) *of which* 0.5 KB *used by bootloader*	256 KB of *which* 8 KB *used by bootloader*
Panjang	45 mm	68.6 mm	101.52 mm
Lebar	18 mm	53.4 mm	53.3 mm
SRAM	2 KB	2 KB (ATmega328P)	8 KB

Berdasarkan data perbandingan pada tabel 3.3, mikrokontroler arduino nano dipilih karena dalam penggunaannya tidak banyak membutuhkan pin I/O dan memiliki kapasitas memori yang cukup sesuai dengan kebutuhan.

3.3.2 Kamera

Kamera merupakan alat yang digunakan untuk menangkap objek berupa Gambar /citra. Kamera *webcam* C310 merupakan modul kamera yang dapat dijalankan menggunakan perangkat bantu berupa PC/Laptop.

Tabel 3.4 Perbandingan jenis kamera *webcam*

Spesifikasi	**Jenis Komponen**	
	***Webcam* C310**	***Webcam* C270**
Tegangan masukan	USB 5 Vdc	USB 5 Vdc
Video (HD 720p)	Ya	Ya
Automatic Light	Ya (*Premium*)	Ya (*Standard*)
Photo Quality	5 Mpx	3 Mpx

Berdasarkan data perbandingan tabel 3.4 diatas, kamera *webcam* C310 memiliki resolusi kamera yang besar yaitu 5 Mpx dengan *automatic light premium* tampilan gambar gelap dapat diatur kecerahannya dengan kualitas *premium*.

3.3.3 Catu Daya

Catu daya merupakan sebuah piranti penting dalam hal memberi *input* sumber daya pada suatu sistem. Pemilihan catu daya yang tepat akan mempengaruhi kinerja dari suatu sistem yang akan dibuat. Banyak beragam jenis catu daya yang ada di pasaran, salah satunya catu daya LiPo (*Lithium Polimer*). Catu daya *Litium Polimer* (LiPo) merupakan jenis catu daya yang dapat di isi ulang (*rechargeable*), di dalamnya tersusun beberapa inti sel yang disusun seri, yang bertujuan untuk meningkatkan kapasitas arus listrik yang dihasilkan. LiPo yang digunakan di perancangan ini menggunakan LiPo 3300 mAh.

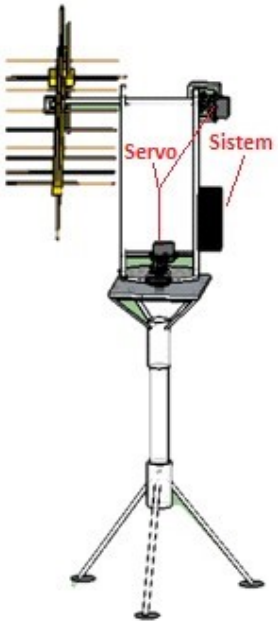
Tabel 3.5 Perbandingan jenis catu daya

Spesifikasi	Jenis Komponen		
	LiPo	Li Ion	NiMH
Dapat Di Charging	Ya	Ya	Ya
Tegangan	3.7 Vdc	3.7 Vdc	1.2 Vdc
Inti Sel	3 Sel	1 Sel	1 Sel
Memori efek	No	No	Yes

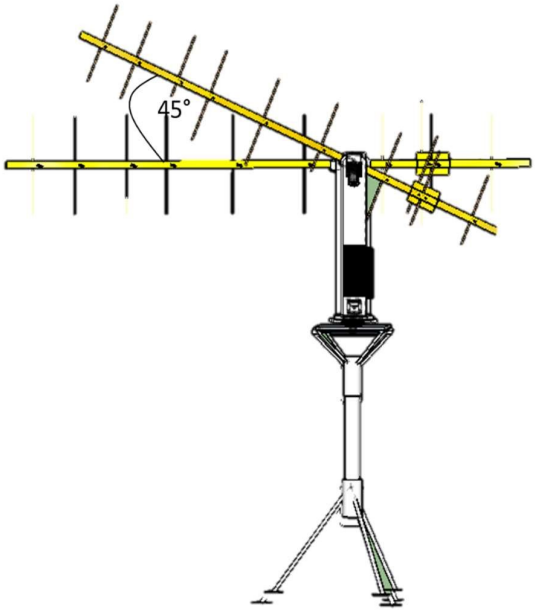
Berdasarkan data perbandingan tabel 3.5 di atas, catu daya *Lithium Polimer* dipilih karena memilki kapasitas cell yang banyak, selain itu *Lithium Polimer* tidak memiliki memori efek yang di hasilkan, sehingga saat di *charging* ulang tidak harus menunggu kosong.

3.4 Perancangan *Antenna Tracker* (AT)

Antenna tracker (AT) sebagai alat bantu modul radio 3DR 433MHz yang dipakai, agar dapat menerima sinyal dengan maksimal. Dengan metode perhitungan koordinat posisi *payload* dan posisi *ground station*, dengan demikian *antenna tracker* akan mendapatkan nilai *pitch* dan *yaw* agar dapat mengarah tepat dan mengikuti arah gerak *payload*. *Antenna tracker* juga dirancang agar tetap kokoh saat membawa beban antena yang dipakai. Berikut perbandingan desain antara yang akan dirancang dengan desain *antenna tracker* KOMURINDO 2016:



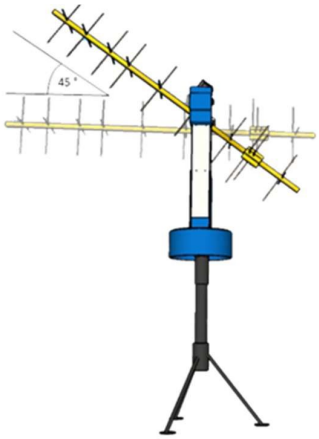
Gambar 3.20 Ilustrasi *antenna tracker* pad KOMURINDO 2016 tampak depan



Gambar 3.21 Ilustrasi *antenna tracker* pada KOMURINDO 2016 tampak samping



Gambar 3.22 Ilustrasi *antenna tracker* yang akan dirancang tampak depan



Gambar 3.23 Ilustrasi *antenna tracker* yang akan dirancang tampak depan

Dapat dilihat pada Gambar 3.20 dan Gambar 3.22, desain yang dibuat lebih rapih dan letak sistem tidak menggangu dari *antenna tracker*. Komponen *antenna tracker* dipilih berdasarkan kebutuhan perancangan, kualitas, dimensi, dan harga. Komponen utama dalam pembuatan *antenna tracker* ditampilkan pada tabel 3.6 berikut:

Tabel 3.6 Daftar bahan dan komponen penyusun *antenna tracker*

No	Bagian Antena Tracker	Nama Komponen	Keterangan
1	Kerangka Utama	Alumunium, Batang Baja	Ringan dan Kuat
2	Prosesor	Arduino Nano	Cukup untuk mengoperasikan sistem *antenna tracker*
3	Aktuator	Servo HS-645MG	Untuk menggerakkan *antenna tracker* dengan torsi yang kuat dan akurasi yang baik
4	Catu daya	Baterai Lipo 3300 mAh 11.1 V	Cukup untuk men-*supply* daya sistem AT

Berikut adalah perbandingan dari bentuk *antenna* yang telah dibuat dengan *antenna tracker* pada KOMURINDO 2016 .



Gambar 3.24 AT yang sudah di rancang tampak depan



Gambar 3.25 AT yang sudah dirancang tampak diagonal



Gambar 3.26 AT yang sudah dirancang tampak samping



Gambar 3.27 AT pada KOMURINDO 2016 tampak depan



Gambar 3.28 AT pada KOMURINDO 2016 tampak diagonal



Gambar 3.29 AT pada KOMURINDO 2016 tampak samping

3.5 Perancangan *Hardware*

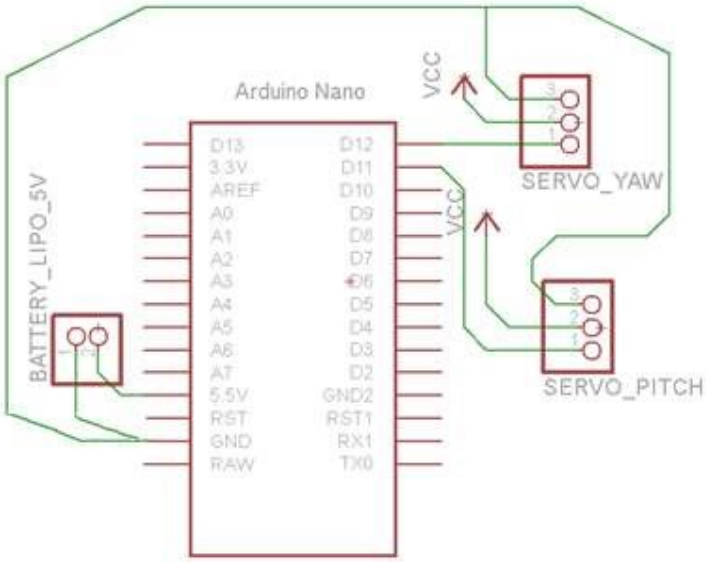
Pada perancangan *Hardware* (perangkat keras) ini meliputi pembahasan mengenai: mikrokontroler, *webcam*, catu daya dan servo.

3.5.1 Mikrokontroler Arduino Nano

Arduino Nano adalah *development board* (papan pengembangan) mikrokontroler yang berbasis chip ATmega328P dengan bentuk yang sangat mungil. Secara fungsi tidak ada bedanya dengan Arduino Uno. Perbedaan utama terletak pada ketiadaan *jack power* DC dan penggunaan konektor *Mini*-B USB.

Disebut sebagai papan pengembangan karena board ini memang berfungsi sebagai *arena prototyping* sirkuit mikrokontroler. Dengan menggunakan papan pengembangan, akan lebih mudah merangkai rangkaian elektronika mikrokontroler dibanding jika anda memulai merakit ATMega328 dari awal di *protoboard*.

Berikut skematik rangkaian dari modul Arduino nano:



Gambar 3.30 Skematik rangkaian pada Arduino nano

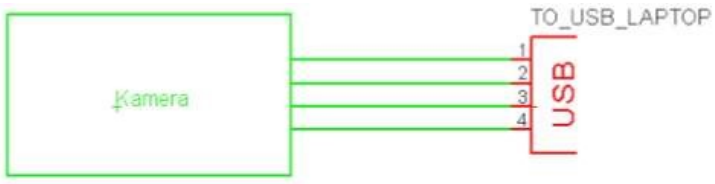
Berikut konfigurasi pin mikrokontroler arduino nano yang digunakan dalam perancangan.

Tabel 3.7 Alokasi pin mikrokontroler arduino nano yang digunakan

Nama Pin Arduino	No Pin	Fungsi	Keterangan
PD11(I/O)	11	OUTPUT	SERVO *PITCH* (DATA)
PD12(I/O)	12	OUTPUT	SERVO *YAW* (DATA)
POWER	5V	*INPUT*	5V dari BATTERY LIPO yang sudah di step down
GND	GND	OUTPUT	SERVO dan SISTEM

3.5.2 Kamera *Webcam* C310

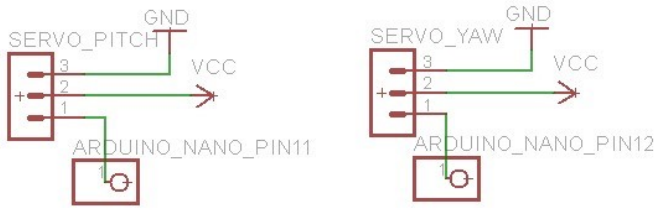
Modul kamera yang digunakan di perancangan ini adalah *webcam* C310 untuk mengambil data berupa gambar /citra analog yang diolah menjadi citra digital. Dimana *webcam* ini memerlukan perangkat PC/laptop yang dihubungkan melalui USB port agar dapat bekerja. *Webcam* C310 ini bekerja sesuai dengan perintah yang dibuat di laptop dengan menggunakan software LabVIEW. Berikut konfigurasi dari kamera *webcam* C310:



Gambar 3.31 Skematik konektor USB kamera *webcam* C310

3.5.3 Motor Servo HS-645MG

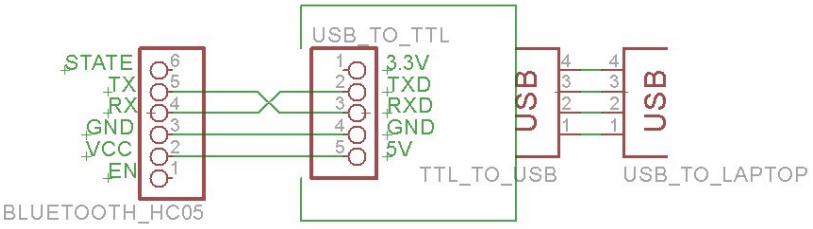
Motor Servo pada perancangan ini menggunakan servo HS-645MG yang berfungsi sebagai penggerak *antenna tracker* yang digunakan untuk pergerakkan *pitch* dan pergerakkan *yaw*. Berikut konfigurasi dari motor servo HS-645MG:



Gambar 3.32 Skematik konektor motor servo *pitch* dan *yaw*

3.5.4 *Bluetooth* dan *USB to TTL*

USB to TTL adalah konverter *USB to TTL* menggunakan *chip* konverter. Konverter ini bisa digunakan untuk segala macam aplikasi elektronik lainnya yang membutuhkan komunikasi serial melalui USB. Disini *USB to TTL* merubah *module bluetooth* yang keluaran serial menjadi USB untuk di lanjutkan pada laptop.



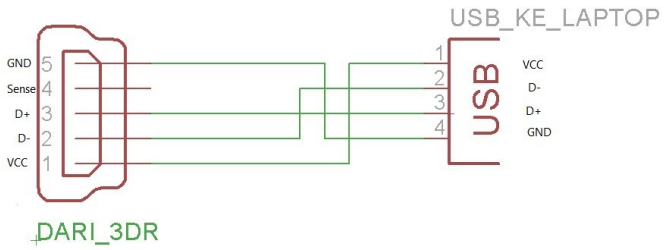
Gambar 3.33 Skematik konektor USB to TTL dengan *module bluetooth* HC-05

3.5.5 Module Radio 3DR 433Mhz

Module radio 3DR 433Mhz digunakan untuk menerima data dari *payload* kepada GS. Module radio ini dapat langsung dikoneksikan kepada laptop *antenna tracker*.



Gambar 3.34 Konfigurasi dari radio 3DR 433Mhz [16]



Gambar 3.35 Skematik konektor radio module 3DR 433MHz

3.6 Perancangan *Software*

Pada perancangan perangkat lunak (*software*) ini di bagi 2:

a. Software *Antenna Tracker*

Pada software ini ditempatkan pada *antenna tracker* yang memiliki fitur komputer *vision* yang dapat mendeteksi warna pada sebuah objek dan *tracking* menggunakan metode GPS. *Software* ini akan mendeteksi adanya objek berwarna menggunakan 1 kamera *webcam* yang terpasang di bagian depan *antenna tracker* lalu akan diproses dan datanya akan dikirim ke mikrokontroler dan juga dapat menentukkan posisi yang dituju dengan menggunakan metode *tracking* GPS.



Gambar 3.36 Tampilan *front panel* program *antenna tracker*

Berikut adalah penjelasan dari keterangan pada tampilan gambar diatas:

1. Kode perintah yang diterima dari *Ground Station* dan kemudian dikirim ke *payload.*
2. *Indikator* mode *tracking* yang dipilih dari *Ground Station,* yaitu:
   - Mode *manual* digunakan untuk mengatur secara *manual* pada software kepada ke 2 servo.
   - Mode *color* digunakan jika menggunakan *color tracking* saja.
   - Mode GPS digunakan jika hanya mau menggunakan *tracking* dengan metode GPS.
   - Mode gabungan digunakan jika mau menggunakan gabungan metode *tracking* dari mode GPS dan mode *color tracking.*
3. Kotak data yang dikirim ke *antenna tracker.*
4. *Indikator* untuk program jalan dan *bluetooth* telah tersambung.
5. Kotak data sebagai *Indikator* dari data yang diterima dari *payload.*

6. *Indikator* kompas *payload*.

7. *Indikator* nilai GPS *payload.*

8. Kontrol untuk memasukkan data GPS *antenna tracker.*

9. Pengaturan untuk mengatur pemilihan menggunakan mode RGB atau HSL.

10. Control untuk merubah RGB/HSL untuk batasan warna yang

- "*Hue*" atau "*Red*" merupakan panel untuk mengatur nilai *Hue*/*Red* dengan rentang nilai dari 0-255.
- "*Saturation*" atau "*Green*" merupakan panel untuk mengatur nilai *Saturasi/Green* dengan rentang nilai dari 0-255.
- "*Luminance*" atau "*Blue*" merupakan panel untuk mengatur nilai *Luminance/Blue* dengan rentang nilai dari 0-255.

11. *Indikator* jika objek warna ditemukan.

12. *Indikator* untuk informasi nilai servo *pitch* dan *yaw*.

13. Kontrol *knob* untuk pergerakkan *antenna tracker* pada mode *manual*.

14. *Picture box* gambar asli untuk menampilkan gambar asli dari kamera *webcam*.

15. *Picture box* gambar yang di *threshold* untuk menampilkan gambar yang sudah dirubah ke citra biner dari kamera *webcam.*

16. Tombol untuk menghentikan program.

b. *Software Ground Station*

Pada *software* ini hanya sebagai pemberi informasi yang didapatkan dari *payload* yang dikirim lewat laptop *antenna tracker.*



Gambar 3.37 Tampilan *front panel* program *ground station*

Berikut adalah penjelasan dari keterangan pada tampilan Gambar 3.37:

1. Tombol yang terdiri dari
   - Tombol start untuk memulai telemetri yang dikirim kepada *Antenna tracker* dan dilanjutkan ke *Payload.*
   - Tombol untuk menghentikan *telemetry* yang dikirim kepada *antenna tracker* dan dilanjutkan ke *payload.*
   - Tombol untuk memulai mode *tracking* sesuai dengan mode *tracking* yang dipilih.
2. Kode perintah yang akan dikirim ke *antenna tracker* dan dilanjutkan dikirim ke *payload.*

3. *Indikator* untuk informasi mode *tracking* telah dipilih.
4. Kontrol untuk merubah mode *tracking*, yaitu:
    - Mode *manual* digunakan untuk mengatur secara *manual* pada software kepada ke 2 servo.
    - Mode *color* digunakan jika menggunakan *color tracking* saja.
    - Mode GPS digunakan jika hanya mau menggunakan *tracking* dengan metode GPS.
    - Mode gabungan digunakan jika mau menggunakan gabungan metode *tracking* dari mode GPS dan mode color *tracking*.
5. *Indikator* untuk informasi program telah berjalan.
6. *Text box* untuk informasi data yang diterima dari laptop *antenna tracker*.
7. *Indikator* tentang informasi kompas pada *payload.*
8. Informasi tentang letak *payload* yang di visualkan dengan *map offline*.
9. Informasi tentang GPS *payload.*
10. Grafik tentang data *Accelerometer* X, Y dan Z.
11. Grafik tentang data *Gyroscope* X, Y dan Z.
12. Gambaran visual 3D tentang *payload.*
13. *Trajectory* untuk melihat posisi *payload* dalam bentuk 3D graph.