TUGAS AKHIR – EE 184801

RANCANG BANGUN PARASUT OTOMATIS UNTUK KONDISI DARURAT UAV

Andre Fajar Nugroho
NRP 07111540000117

Dosen Pembimbing
Ronny Mardiyanto, ST., MT., Ph.D.

DEPATERMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2019

TUGAS AKHIR – EE 184801

RANCANG BANGUN PARASUT OTOMATIS UNTUK KONDISI DARURAT UAV

Andre Fajar Nugroho
NRP 07111540000117

Dosen Pembimbing
Ronny Mardiyanto, ST., MT., Ph.D.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2019

FINAL PROJECT – EE 184801

DESIGN OF AUTOMATIC PARACHUTE FOR UAV EMERGENCY CONDITION

Andre Fajar Nugroho
NRP 07111540000117

Supervisor
Ronny Mardiyanto, ST., MT., Ph.D.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Electrical Technology
Sepuluh Nopember Institute of Technology
Surabaya 2019

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "**Rancang Bangun Parasut Otomatis untuk Kondisi Darurat UAV**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 10 Juli 2019

Andre Fajar Nugroho
NRP. 0711 15 4000 0117

.....Halaman ini sengaja dikosongkan.....

RANCANG BANGUN PARASUT OTOMATIS UNTUK KONDISI DARURAT UAV

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Elektronika
Departemen Teknik Elektro
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing

Ronny Mardiyanto, ST., MT., Ph.D.
NIP. 198101182003121003

SURABAYA
JULI, 2019

.....Halaman ini sengaja dikosongkan.....

RANCANG BANGUN PARASUT OTOMATIS UNTUK KONDISI DARURAT UAV

Nama : Andre Fajar Nugroho
Pembimbing : Ronny Mardiyanto, ST., MT., Ph.D.

ABSTRAK

Dalam tugas akhir ini dibuat sebuah alat untuk mendeteksi kondisi darurat pada *Unmanned Aerial Vehicle* (UAV). Kondisi darurat yang dideteksi yaitu ketika pesawat dalam posisi roll atau pitch lebih dari 70 derajat atau kecepatan vertikalnya lebih dari 10 m/s. Setelah kondisi darurat terdeteksi, alat ini akan membukakan parasut sehingga pesawat tidak langsung jatuh. Alat ini dibuat untuk mengurangi dampak kerusakan ketika terjadi kerusakan UAV atau kehilangan kontrol saat sedang terbang. Yang sudah ada saat ini, parasut akan langsung terbuka ketika terjadi kerusakan pada sistem pesawat pada ketinggian berapapun. Sehingga pada alat ini, ditambah perancangan pada sistemnya untuk bisa memperkirakan pada ketinggian berapa parasut harus terbuka. Untuk itu digunakan rumus kecepatan terminal dan gerak lurus berubah beraturan sedemikian sehingga sistem bisa menghitung pada ketinggian berapa parasut terbuka. Kemudian untuk mendeteksi kondisi darurat UAV menggunakan sensor barometer BMP280 dan sensor akselerometer giroskop MPU6050. Sensor BMP280 digunakan untuk mengetahui ketinggian dan kecepatan vertikal UAV. Sensor MPU6050 digunakan untuk mengetahui posisi roll dan pitch UAV. Sistem akan mendeteksi terlebih dahulu UAV dalam kondisi darurat atau tidak. Jika sudah terdeteksi, sistem akan melakukan penghitungan pada ketinggian berapa parasut akan terbuka. Pada pengujian, alat ini dapat megukur ketinggian dengan error 0,2915 meter. Error roll dan pitch masing – masing sebesar 4,2 derajat dan 4,545 derajat. Rata – rata selisih waktu antara parasut keluar dari pesawat dan parasut terbuka adalah 3,348 detik.

Kata kunci: Kecepatan Terminal, Parasut, *Unmanned Aerial Vehicle* (UAV)

.....Halaman ini sengaja dikosongkan.....

DESIGN OF AUTOMATIC PARACHUTE FOR UAV EMERGENCY CONDITION

Name : Andre Fajar Nugroho
Supervisor : Ronny Mardiyanto, ST., MT., Ph.D.

ABSTRACT

In this final project, a device that can detection emergency condition UAV is created. Emergency conditions are detected when UAV is in roll position or pitch more than 70 degrees or its vertical speed is more than 10 m/s. After an emergency condition is detected, this device will open the parachute so that UAV does not immediately fall. The device is created to reduce the impact of damage when UAV lost control or crash on the air. In this time, used of parachute is still little to solve that problem. Which already exists today, parachute will opened when system of UAV are failure at any height. So that, this device is designed to estimate at how high the parachute opened. So, it is used terminal velocity equation and accelerated linear motion so that system can calculate at how high the parachute is open. Then, barometer sensor BMP280 and accelerometer gyroscope sensor MPU6050 is used to detect UAV emergency condition. BMP280 is used to find out the height and vertical velocity of UAV. MPU6050 is used to find out roll and pitch position of UAV. The system will detect UAV position, whether in an emergency condition or not. If it has been detected, the system will do the calculation at how high the parachute will opened. On the experiment, this device can measure the height, roll position and pitch position with each error 0,2915 meter, 4,2 degrees, and 4,545 degrees. The average difference time between the parachute out of the UAV and parachute opened is 3,348 s.

Keywords: Terminal Velocity, Parachute, Unmanned Aerial Vehicle (UAV)

.....Halaman ini sengaja dikosongkan.....

KATA PENGANTAR

Assalammualaikum Wr. Wb.

Puji syukur saya ucapkan kepada Allah SWT, karena berkat rahmat dan karunia-Nya saya dapat menjalani dan menyelesaikan Tugas Akhir yang berjudul **"Rancang Bangun Parasut Otomatis untuk Kondisi Darurat UAV"**, yang mana Tugas Akhir ini merupakan salah satu syarat yang harus dipenuhi dalam menyelesaikan program studi strata 1 (S1) di Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember Surabaya.

Dalam menyelesaikan Tugas Akhir ini, saya ingin mengucapkan banyak terima kasih kepada:

1. Kedua orang tua dan keluarga saya yang selalu memberikan dukungan dan do'a selama pengerjaan Tugas Akhir ini.
2. Bapak Ronny Mardiyanto, ST., MT., Ph.D. selaku dosen pembimbing yang selalu memberi bimbingan dan masukan demi terselesaikannya tugas akhir ini.
3. Teman – teman Elka yang selalu siap membantu dalam pengerjaan tugas akhir ini.
4. KITT yang selalu memberi dukungan dan semangat ketika mengalami kejenuhan.
5. Teman-teman seangkatan dan seperjuangan Fakultas Teknologi Elektro Angkatan 2015 (e55) yang senantiasa memberikan dukungan baik melalui diskusi atau moril.
6. Semua pihak yang bersangkutan selama pengerjaan Tugas Akhir ini, yang belum bisa saya sebutkan satu per satu.

Dengan terselesaikannya Tugas Akhir ini saya berharap hasilnya dapat bermanfaat dalam perkembangan teknologi di Indonesia khususnya di bidang UAV (*Unmanned Aerial* Vehicle). Saya menyadari bahwa dalam penulisan Tugas Akhir ini masih terdapat kekurangan. Saran dan kritik dari semua pihak akan sangat membantu saya untuk pengembangan yang lebih lanjut.

Wassalammualaikum Wr. Wb.

Surabaya, 2019

Andre Fajar Nugroho

.....Halaman ini sengaja dikosongkan.....

DAFTAR ISI

PERNYATAAN KEASLIAN ..... **i**
TUGAS AKHIR ..... **iii**
ABSTRAK ..... **v**
ABSTRACT ..... **vii**
KATA PENGANTAR ..... **ix**
DAFTAR ISI ..... **xi**
DAFTAR GAMBAR ..... **xv**
DAFTAR TABEL ..... **xvii**
BAB 1 PENDAHULUAN **1**
1.1. Latar Belakang 1
1.2. Perumusan Masalah 2
1.3. Tujuan Penelitian 2
1.4. Batasan Masalah 3
1.5. Metodologi Penelitian 3
1.6. Sistematika Penulisan 4
1.7. Relevansi 5
BAB 2 TEORI PENUNJANG DAN TINJAUAN PUSTAKA **7**
2.1. Unmanned Aerial Vehicle (UAV) 7
2.2. Arduino Nano 8
2.3. Kecepatan Terminal 8
2.4. BMP280 10
2.5. MPU6050 11
2.6. Filter Eksponensial 11
2.7. Matlab 12
2.8. MicroSD Card 12
2.9. Modul SD Card 13
2.10. Tinjauan Pustaka 13
2.10.1. *An Automatic Parachute Separation System for Dispersal Miniature Reconnaissance Robot* 13
2.10.2. *Parachute Decelerate Trajectory Optimization Design Based on Genetic Algorithm* 13
2.10.3. *Prediction of the Parachute Deploy for Landing at Desired Point* 14
2.10.4. *UAV Fault Detection based on GA-BP Neural Network* 14
2.10.5. *Advantages of Using Multi-Agent Principles in FAIL SAFE UAV System Design* 15

2.10.6. *A Simplex Architecture for Intelligent and Safe Unmanned Aerial Vehicles* 15
2.10.7. Mars Parachutes 16
2.10.8. SkyFallX 16
2.10.9. ParaZero 17
2.10.10. Skycat 17
2.10.11. Fruity Chutes 19
2.10.12. FLIRT (Flying Intelligent Robotic Tools) 19
BAB 3 PERANCANGAN SISTEM 21
3.1. Diagram Blok Sistem 22
3.2. Perancangan Perangkat Keras 23
3.2.1. Sensor BMP280 23
3.2.2. Sensor MPU6050 24
3.2.3. Modul SD Card 24
3.2.4. Desain Parasut 25
3.2.5. Pemasangan Mekanik Parasut pada Pesawat 27
3.3. Perancangan Perangkat Lunak 28
3.3.1. Pembacaan BMP280 29
3.3.2. Pembacaan MPU6050 30
3.3.3. Penyimpanan Data 31
3.3.4. Pendeteksian Kondisi Darurat UAV 32
3.3.5. Penghitungan Kecepatan Terminal 32
3.3.6. Perumusan Ketinggian Minimal untuk Membuka Parasut 34
BAB 4 PENGUJIAN DAN ANALISIS 37
4.1. Pengujian Pembacaan Ketinggian 37
4.2. Pengujian Pembacaan Roll dan Pitch 38
4.2.1. Pengujian Roll 39
4.2.2. Pengujian Pitch 40
4.3. Pengujian Pada Pesawat 41
4.3.1. Pengujian Pertama 42
4.3.2. Pengujian Kedua 43
4.3.3. Pengujian Ketiga 44
4.3.4. Pengujian Keempat 45
4.3.5. Pengujian Kelima 46
4.3.6. Pengujian Keenam 47
4.3.7. Pengujian Ketujuh 48
4.3.8. Pengujian Kedelapan 49
4.3.9. Pengujian Kesembilan 51

4.4. Analisa Keseluruhan Sistem 52
BAB 5 PENUTUP..... 53
5.1. Kesimpulan 53
5.2. Saran 53
DAFTAR PUSTAKA 55
LAMPIRAN A..... 57
LAMPIRAN B..... 65
LAMPIRAN C..... 67
BIODATA PENULIS 69

.....Halaman ini sengaja dikosongkan.....

DAFTAR GAMBAR

Gambar 2.1 Pesawat UAV Fix Wing 7
Gambar 2.2 Arduino nano 8
Gambar 2.3 Besar Koefisien Hambat Udara pada Beberapa Bentuk Benda[11] 9
Gambar 2.4 Gambar BMP280 10
Gambar 2.5 Gambar MPU6050 11
Gambar 2.6 MicroSD card 12
Gambar 2.7 Modul SD card 13
Gambar 2.8 (a) Tabung Parasut, (b) Kontrol Manual, (c) Kontrol Otomatis dari Mars Parachutes[1] 16
Gambar 2.9 Tabung Parasut dan Kontroler dari SkyFallX[2] 17
Gambar 2.10 Parasut Pada DJI Phantom dari ParaZero[4] 17
Gambar 2.11 (a) Parasut Ganda Bersebelahan (b) Parasut Ganda Atas Bawah[3] 18
Gambar 2.12 Pesawat FLIRT[6] 19
Gambar 3.1 Ilustrasi Sistem 21
Gambar 3.2 Diagram Blok Sistem 22
Gambar 3.3 Antarmuka BMP280 dengan Arduino nano 23
Gambar 3.4 Rangkaian Skematik BMP280[22] 23
Gambar 3.5 Antarmuka MPU6050 dengan Arduino nano 24
Gambar 3.6 Rangkaian Skematik MPU6050[23] 24
Gambar 3.7 Antarmuka modul SD card dengan Arduino nano 25
Gambar 3.8 Rangkaian Skematik Modul SD Card[24] 25
Gambar 3.9 Desain Parasut dan Ukurannya 26
Gambar 3.10 Tempat Meletakkan Parasut dan Servo 27
Gambar 3.11 Parasut yang Dimasukkan di Tempat Parasut 27
Gambar 3.12 Tampak Bawah Pesawat, Tempat Parasut 28
Gambar 3.13 Grafik Hubungan Kecepatan Terminal dengan Massa (kecepatan awal 10) 33
Gambar 3.14 Grafik Jarak untuk Mencapai Kecepatan Terminal 35
Gambar 4.1 Realisasi Alat (Mikrokontroler, Sensor, dan Data Logger) 37
Gambar 4.2 Perbandingan Pembacaan Ketinggian Tanpa Filter dan Dengan Filter 37
Gambar 4.3 Pengujian Sensor MPU6050 39
Gambar 4.4 Percobaan Pertama, Parasut Terbuka 42
Gambar 4.5 Grafik Ketinggian, Kecepatan, dan Indikator Parasut

Pengujian Pertama.....43
Gambar 4.6 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Kedua 44
Gambar 4.7 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Kedua 44
Gambar 4.8 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Ketiga 45
Gambar 4.9 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Percobaan Ketiga 45
Gambar 4.10 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Keempat 46
Gambar 4.11 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Keempat 46
Gambar 4.12 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Kelima.....47
Gambar 4.13 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Kelima 47
Gambar 4.14 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Keenam 48
Gambar 4.15 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Keenam 48
Gambar 4.16 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Ketujuh.....49
Gambar 4.17 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Ketujuh.....49
Gambar 4.18 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Kedelapan.....50
Gambar 4.19 Grafik Ketinggian dan Indikator Parasut Pengujian Kedelapan.....50
Gambar 4.20 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Kedelapan.....51
Gambar 4.21 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Kesembilan.....51
Gambar 4.22 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Kesembilan.....52

DAFTAR TABEL

Tabel 4.1 Hasil pengujian BMP280 di Gedung B Teknik Elektro ITS terhadap lantai dasar Gedung 38
Tabel 4.2 Hasil Pengukuran Sudut Roll dan Error..... 39
Tabel 4.3 Pembacaan Nilai Pitch, Posisi 0o Menghadap ke Atas 40
Tabel 4.4 Pembacaan Nilai Pitch, Posisi 0o Menghadap ke Bawah 41

.....Halaman ini sengaja dikosongkan.....

BAB 1
PENDAHULUAN

Tugas akhir merupakan suatu penelitian yang dilakukan sebagai persyaratan akademik untuk mendapatkan gelar sarjana teknik di Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Topik yang akan dibahas pada tugas akhir ini mengenai parasut otomatis untuk kondisi darurat pesawat tanpa awak.

Pada bab ini membahas mengenai hal-hal yang mendahului pelaksanaan tugas akhir. Hal tersebut meliputi latar belakang, perumusan masalah, tujuan penulisan, batasan masalah, metodologi penelitian, sistematika penulisan, dan relevansi.

1.1. Latar Belakang

Parasut adalah suatu perangkat yang digunakan untuk memperlambat gerakan suatu benda yang jatuh dari atmosfer. Parasut dapat memperlambat gerakan jatuhnya karena adanya gaya hambat udara Penggunaan parasut yang paling umum adalah untuk terjung payung dari pesawat. Selain untuk memperlambat kecepatan vertikal, parasut juga bisa digunakan untuk memperlambat kecepatan horizontal suatu benda seperti yang digunakan pada pesawat jet atau mobil pada perlombaan *drag race*.

Pada perkembangannya, parasut digunakan untuk berbagai kebutuhan, seperti olahraga paralayang, *skydiving*, latihan terjun payung untuk militer, serta untuk kebutuhan penyelamatan. Untuk penyelamatan, parasut biasanya dipakai oleh pilot pesawat jet, sehingga ketika terjadi kerusakan pada pesawat jetnya, pilot bisa langsung menekan tombol untuk keluar dari pesawat dan secara otomatis parasut akan terbuka. Selain digunakan oleh manusia, parasut juga bisa digunakan untuk menjatuhkan suatu benda, misalnya ketika menjatuhkan barang – barang kebutuhan pada daerha tertentu yang sulit diakses

Tidak hanya itu, sekarang mulai dikembangkan parasut sebagai *safety flight* pada UAV atau kendaraan udara tanpa awak baik itu drone atau *quadcopter* maupun pesawat. Tujuannya untuk mengurangi resiko kerusakan ketika UAV terjatuh karena mengingat biaya untuk membuat UAV tidaklah murah sehingga dibutuhkan peralatan khusus untuk

mencegah kerusakan yang terjadi. Di luar negeri, sudah mulai dikembangkan parasut untuk mengatasi permasalahan tersebut, bahkan sudah ada beberapa yang diperjualbelikan Namun kebanyakan parasut tersebut dirancang pada drone. Sedangkan untuk pesawat *fixed wing* masih sedikit yang mengembangkannya. Hal tersebut mungkin dikarenakan drone lebih banyak diminati daripada *fixed wing* untuk saat ini. Beberapa contoh merek yang digunakan untuk drone adalah *Mars Parachutes*[1], *SkyFallX*[2], SkyCat[3], dan ParaZero[4], sedangkan untuk *fixed wing* adalah *Fruity Chutes*[5], dan FLIRT[6].

Dari beberapa produk yang telah disebutkan, ada kemiripan mekanisme untuk mendeteksi kondisi daruratnya yaitu, jika drone mengalami perubahan ketinggian yang cepat. Hal tersebut mengindikasikan bahwa drone sedang dalam kondisi jatuh bebas. Selain itu, juga ada yang menggunakan pendeteksian kemiringannya. Jika drone miring sebesar 90 derajat, makan parasut akan langsung terlontar dari dalam tempat parasut. Kemudian ada juga yang menambahkan jumlah parasut yang terpasang sehingga bisa lebih memperlambat lagi kecepatan turunnya.

Pada tugas akhir ini, akan ditambahkan pengembangan untuk memperkirakan pada ketinggian berapa parasut harus terbuka jika sudah terdeteksi kondisi daruratnya. Sehingga, parasut tidak akan terbuka jika posisi UAV masih terlalu tinggi meskipun sudah terdeteksi dalam kondisi darurat. Kondisi darurat dalam tugas akhir ini didefinisikan yaitu jika posisi roll atau pitch pesawat lebih dari 70 derajat. Selain itu, jika kecepatan jatuh pesawat lebih dari 10 m/s, atau perubahan ketinggian pesawat yang tinggi.

1.2. Perumusan Masalah

Permasalahan yang dibahas dalam tugas akhir ini adalah :

1. Bagaimana cara mendeteksi keadaan darurat pada UAV.
2. Bagaimana menentukan waktu yang tepat untuk membuka parasut.

1.3. Tujuan Penelitian

Penelitian pada tugas akhir ini bertujuan sebagai berikut :

1. Dapat membuat sistem yang bisa mengenali kondisi darurat pada UAV.
2. Dapat menentukan waktu yang tepat untuk membuka parasut.

1.4. Batasan Masalah

Batasan masalah dari tugas akhir ini adalah sebagai berikut :

1. Tidak ikut serta dalam pembuatan UAV.
2. Kondisi yang dideteksi adalah kemiringan pesawat dan kecepatan vertikal pesawat.

1.5. Metodologi Penelitian

Langkah-langkah yang dikerjakan pada tugas akhir ini adalah sebagai berikut :

1. Studi Literatur

Studi literatur bertujuan untuk mempelajari teori dasar atau penelitian tentang hal terkait yang sudah ada guna menunjang pengerjaan tugas akhir ini. Literatur yang dicari bisa diperoleh dari buku, jurnal, paper, atau artikel yang terpercaya.

2. Observasi dan Analisa Masalah

Pada tahap ini dilakukan pengkajian terhadap sistem kerja UAV dan alat pengukur ketinggian UAV. Analisa terhadap hubungan massa UAV dan kecepatan jatuh bebasnya untuk memperkirakan waktu pembukaan parasut. Observasi dan analisa masalah dilakukan dengan mengkaji paper, jurnal, dan buku – buku yang membahas tentang hal tersebut.

3. Persiapan Alat dan Bahan

Tahap ini merupakan tahap pencarian informasi tentang alat dan bahan yang akan dipakai untuk membuat keseluruhan sistem, yang bisa didapat dari studi literatur dan bimbingan dosen pembimbing. Kemudian dilakukan pengumpulan alat dan bahan yang akan digunakan. Alat dan bahan yang digunakan yaitu seperangkat laptop, Arduino nano, sensor barometer, sensor akselerometer dan giroskop, servo, modul microSD, microSD, serta alat dan bahan pendukung lainnya.

4. Perancangan dan Pembuatan Alat

Perancangan bertujuan untuk mendapatkan desain dan mekanisme yang optimal sehingga nantinya alat bisa berfungsi dengan baik. Perancangan disini ada hardware dan software.

Perancangan hardware berupa mendesain PCB, pembuatan parasut, dan penempatan parasut pada pesawat. Kemudian untuk software dilakukan pembuatan source code yang meliputi pembacaan sensor barometer, sensor akselerometer dan giroskop, penyimpanan data ke microSD, dan gerakan servo untuk membuka tempat penyimpanan parasut.

5. **Tahap Pengujian**

Pengujian dilakukan secara bertahap, dari pengujian source code pada setiap sensor apakah sudah berfungsi dengan baik atau belum. Kemudian setelah semua berfungsi secara terpisah, dilakukan pengujian secara keseluruhan sistem dengan memasang alat pada pesawat UAV kemudian diterbangkan dan diuji apakah parasut bisa berfungsi dengan baik atau tidak.

6. **Analisa dan Evaluasi**

Analisa dilakukan terhadap hasil pengujian sehingga karakteristik dari software dan hardware bisa diketahui. Analisa dilakukan pada pembacaan sensor barometer, sensor akselerometer dan giroskop, penyimpanan data di microSD, serta gerakan servo untuk melepaskan parasut. Apabila dari hasil yang didapat masih belum sesuai, dilakukan evaluasi pada sistem untuk dirancang dan diuji kembali.

7. **Penyusunan Laporan Tugas Akhir**

Tahap penyusunan laporan merupakan tahap terakhir dalam proses pengerjaan tugas akhir ini. Laporan berisi tentang seluruh kegiatan yang telah dilakukan dalam proses pembuatan tugas akhir, mulai dari pendahuluan, studi literatur, perancangan dan pembuatan sistem, pengujian dan analisa, dan penutup.

1.6. Sistematika Penulisan

Dalam buku tugas akhir ini, pembahasan mengenai sistem yang dibuat terbagi menjadi lima bab dengan sistematika penulisan sebagai berikut :

- **BAB I: Pendahuluan**

  Bab ini menjelaskan tentang latar belakang, rumusan masalah, tujuan masalah, metodologi, sistematika penulisan, dan relevansi.

- **BAB II: Tinjauan Pustaka**
  Bab ini berisi tentang teori yang mendasari penyusunan laporan tugas akhir secara umum, khususnya yang berhubungan dengan komponen yang akan digunakan.
- **BAB III: Perancangan Sistem**
  Bab ini berisi tentang penjelasan rancangan dari sistem yang akan dibuat, meliputi perangkat keras (*hardware*) dan perangkat lunak (*software*).
- **BAB IV: Pengujian dan Analisis**
  Bab ini menjelaskan tentang pengujian alat yang telah dibuat beserta analisanya.
- **BAB V: Penutup**
  Bab ini berisi tentang kesimpulan yang diperoleh dari pembuatan alat serta saran untuk pengembangan lebih lanjut.

1.7. Relevansi

Dengan dibuatnya tugas akhir ini akan membantu meminimalisir kerusakan pada pesawat UAV ketika jatuh dengan cara membuka parasut. Parasut akan terbuka secara otomatis tergantung dari kecepatan pesawat UAV saat jatuh dan pada ketinggian berapa, sehingga parasut tidak terbuka terlalu tinggi atau terlalu rendah.

.....Halaman ini sengaja dikosongkan.....

BAB 2
TEORI PENUNJANG DAN TINJAUAN PUSTAKA

Suatu penelitian memerlukan teori-teori yang sudah ada sebelumnya untuk dikaji lebih dalam untuk memperkuat argumen penulis. Teori tersebut digunakan untuk membantu penulis dan sebagai dasar dalam membuat suatu penelitian.

Pada bab ini terdapat teori dasar yang menjadi landasan untuk merumuskan dan menyelesaikan masalah yang akan dibahas pada penelitian ini. Pada bagian ini terdapat tinjauan pustaka tentang komponen yang akan digunakan untuk membuat alat pada penelitian ini.

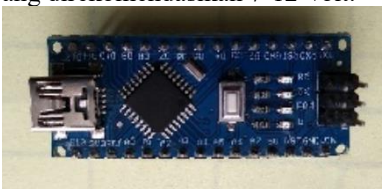
2.1. Unmanned Aerial Vehicle (UAV)

Gambar 2.1 Pesawat UAV Fix Wing

UAV (Unmanned Aerial Vehicle) atau biasa disingkat UAV adalah pesawat tanpa awak. UAV dapat dikendalikan dengan *remote* dari jarak jauh, diprogram dengan perintah tertentu, atau bahkan dengan sistem pengendali otomatis yang lebih kompleks. Aplikasi UAV sangat beragam tergantung kebutuhan. Dalam bidang militer, biasa digunakan untuk sejumlah misi, seperti pengintaian atau penyerangan[7]. Selain itu, UAV juga bisa digunakan untuk pemetaan suatu wilayah. Selain itu, UAV juga digunakan untuk pemetaan suatu wilayah dan masih banyak lagi[8]. Kontrol pesawat tanpa awak ada dua variasi utama, variasi pertama yaitu dikontrol melalui pengendali jarak jauh dan variasi kedua adalah pesawat yang terbang secara mandiri berdasarkan program yang dimasukan ke dalam pesawat sebelum terbang[7].

2.2. Arduino Nano

Arduino Nano adalah sebuah *mini board* berbasis mikrokontroler Atmega328. Arduino Nano mempunyai 14 pin digital input/output (pin 0-13) dimana 6 pin bisa digunakan untuk output analog (pin 3, 5, 6, 9, 10, 11) yang digunakan untuk pengaturan PWM (*Pulse Width Modulation*), 8 pin input analog (pin A0-A7) yang biasa digunakan untuk membaca tegangan dari sensor dan mengkonversikannya menjadi nilai 0 dan 1023, sebuah osilator Kristal 16 MHz, sebuah koneksi USB, sebuah ICSP *header*, dan sebuah tombol reset. Arduino nano dioperasikan dengan menggunakan port USB computer, USB *charger*, atau adaptor AC-DC dengan tegangan yang direkomendasikan 7-12 Volt.



Gambar 2.2 Arduino nano

2.3. Kecepatan Terminal

Ketika suatu benda terjatuh dari atmosfer, ada dua gaya yang mempengaruhi benda tersebut. Yang pertama adalah gaya gravitasi dari benda itu sendiri, yang bisa dirumuskan sebagai berikut:

$$W = m \times g$$

Kemudian yang kedua adalah gaya hambat udara, yang bisa dirumuskan sebagai berikut:

$$D = \frac{Cd \times \rho \times v^2 \times A}{2}$$

Kedua gaya tersebut bekerja pada benda dengan arah yang berlawanan. Gaya gravitasi memiliki arah ke bawah dan gaya hambat udara memiliki arah ke atas. Dari kedua gaya tersebut hanya gaya gravitasi yang memiliki nilai relatif konstan, sedangkan gaya hambat udara ada beberapa parameter yang bisa berubah – ubah. Sehingga pada titik tertentu, nilai gaya gravitasi dan gaya hambat udara akan bernilai sama, sehingga benda akan terjatuh dengan percepatan nol atau kecepatan

konstan. Nilai kecepatan konstan tersebutlah yang biasa dinamakan dengan kecepatan terminal[9]. Jika dituliskan dalam perhitungan, dapat ditulis sebagai berikut:

$$F = W - D = m \times a$$

F adalah resultan gaya dari gaya gravitasi dan gaya hambat udara yang akan bernilai nol ketika besar kedua gaya bernilai sama.

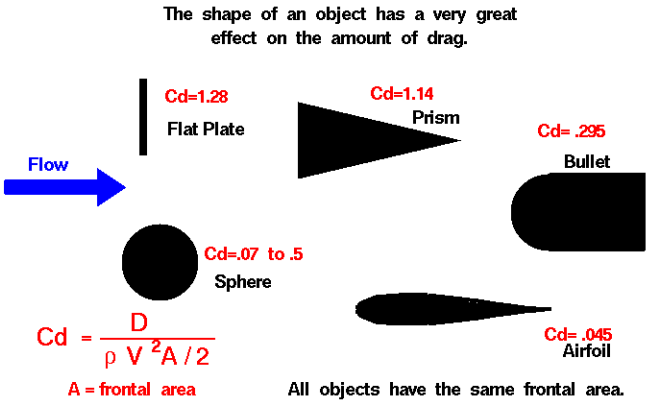
$$W = D$$

$$m \times g = \frac{Cd \times \rho \times v^2 \times A}{2}$$

Sehingga didapat rumus kecepatan terminal sebagai berikut[10] :

$$v = \sqrt{\frac{2 \times m \times g}{Cd \times \rho \times A}}$$

Keterangan:
W = gaya gravitasi (N)
D = gaya hambat udara (N)
m = massa benda (kg)
g = percepatan gravitasi (m/s^2)
Cd = koefisien hambat
ρ = massa jenis udara (kg/m^3)
A = luas permukaan benda (m^2)



Gambar 2.3 Besar Koefisien Hambat Udara pada Beberapa Bentuk Benda[11]

Terlihat bahwa kecepatan terminal dipengaruhi oleh beberapa hal, yaitu massa benda, percepatan gravitasi, koefisien hambat benda, massa jenis udara, dan luas permukaan benda yang tegak lurus terhadap arah gaya hambat udara. Untuk percepatan gravitasi dan massa jenis udara nilainya relatif tetap pada setiap tempat. Sedangkan untuk massa benda, koefisien hambat benda, dan luas permukaan bisa berubah – ubah tergantung dari benda yang dijatuhkan. Untuk massa benda relatif tetap ketika benda berada di udara, sedangkan koefisien hambat benda dan luas permukaan benda bisa berubah. Pada gambar 2.3 menunjukkan beberapa bentuk benda dengan besar koefisien hambat udaranya.

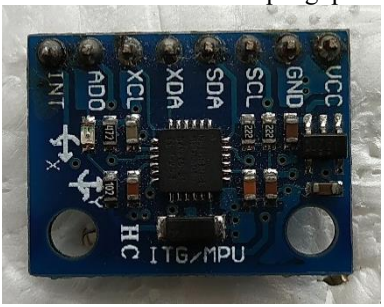
2.4. BMP280

BMP280 adalah sensor tekanan barometrik absolut yang dirancang khusus untuk aplikasi *mobile*. Modul sensor ini disimpan dalam sebuah *metal lid* 8-*pin* LGA yang sangat rapat dengan ukuran 2.0 x 2.5 mm2 dan tinggi 0.95 mm sehingga BMP 280 memiliki dimensi yang kecil dan konsumsi daya yang rendah. Hal ini memungkinkan BMP 280 untuk diimplementasikan dengan perangkat elektronik bertenaga baterai. BMP 280 dilengkapi dengan sensor suhu yang dapat digunakkan untuk mendeteksi suhu disekitar. Untuk mendapatkan tekanan *barometric* dengan cara menghubungkan SDA dan SCL, yang nantinya dikonversi menjadi satuan *pascal.* Dari tekanan *barometric* ini maka akan didapatkan ketinggian dari daerah yang diukur tersebut. BMP 280 sangat cocok untuk mengukur ketinggian dari suatu lantai karena memiliki akurasi yang relatif baik[12].

Gambar 2.4 Gambar BMP280

2.5. MPU6050

MPU6050 adalah perangkat pengenalan gerakan pertama di dunia yang dirancang dengan daya rendah, harga murah, dan mempunyai kinerja tinggi. Perangkat ini sering digunakan pada *smartphone dan tablet.* MPU6050 mengkombinasikan 3 axis gyroscope dan 3 axis accelerometer pada satu papan yang sama bersama dengan *Digital Motion Processor* yang memproses algoritma 6 axis MotionFusion. Perangkat ini dapat mengakses magnetometer eksternal atau sensor yang lain menggunakan *auxiliary* master I2C bus, yang memungkinkan perangkat untuk mengumpulkan satu set data sensor tanpa adanya intervensi dari prosesor sistem. Ukuran perangkat ini sebesar 4 mm x 4 mm x 0.9 mm. Untuk kepresisian pengenalan gerakan lambat dan cepat, untuk gyroscope memiliki rentang skala penuh, yaitu ±250, ±500, ±1000, dan ±2000o/sec. Sedangkan untuk accelerometer memiliki rentang skala penuh, yaitu ±2g, ±4g, ±8g, dan ±16g. Kemudian ada satu fitur tambahan lagi yaitu sensor suhu dengan variasi ±1% dari kisaran suhu pengoperasian[13].



Gambar 2.5 Gambar MPU6050

2.6. Filter Eksponensial

Filter eksponensial biasa digunakan untuk memprediksi dalam dunia statistika, namun juga bisa digunakan untuk memfilter sinyal dan dapat diproses secara cepat. Berikut ini persamaan filter eksponensial.

$$y[n] = a \times x[n] + (1 - a) \times y[n - 1]$$

$x[n]$ merupakan data dari sinyal input, $y[n]$ merupakan output dari sinyal setelah difilter, dengan a adalah parameter eksponensial. Nilai a berada pada rentang nilai 0 sampai 1. Semakin banyak noise maka nilai a yang optimal adalah mendekati nol, jika sebaliknya maka nilai a yang optimal adalah mendekati ke satu[14].

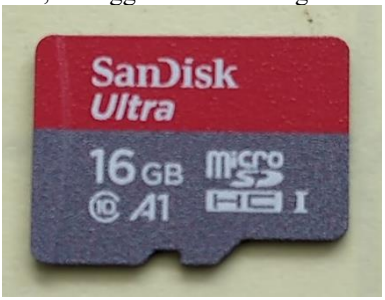
2.7. Matlab

Matlab adalah *platform* pemrograman yang dirancang khusus untuk insinyur dan ilmuwan. Inti dari matlab adalah bahasa Matlab, Bahasa berbasis matriks yang memungkinkan ekspresi komputasi matematika yang paling murni.

Matlab dapat digunakan untuk menganalisis data, pengembangan algoritma, dan membuat model dan aplikasi. Bahasa, aplikasi, dan fungsi matematika bawaan pada matlab memungkinkan untuk mengeksplorasi berbagai pendekatan untuk sampai pada solusi dengan cepat[15].

2.8. MicroSD Card

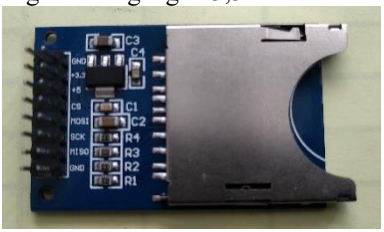
MicroSD merupakan kartu memori *non-volatile* yang dikembangkan oleh *SD Card Association.* MicroSD berfungsi untuk menyimpan berbagai macam file. MicroSD terbagi menjadi SDSC yang berkapasitas maksimum sekitar 2 GB, SDHC (*High Capacity*) memiliki kapasitas dari 4 GB sampai 8 GB, SDCX (*Extended Capacity*) memiliki kapasitas diatas 32 GB. MicroSD sudah terformat dengan system file sebagai FAT16, SDHC sebagai FAT32, sedangkan SDXC sebagai ExFAT. FAT16 dan FAT 32 memungkinkan dapat diakses melalui semua perangkat host *SD reader.* Perangkat yang menggunakan microSD mengidentifikasi kartu dengan string 128-*bit.* Untuk kartu standar, 12 dari *bit* digunakan untuk mengidentifikasi jumlah *cluster* memori (antara 1 sampai 4096) dan 3 *bit* digunkan untuk mengidentifikasi jumlah blok per *cluster.* Pada tugas akhir ini, microSD digunakan untuk menyimpan data yang didapat dari GPS, ketinggian dan sensor gas dalam format .csv.



Gambar 2.6 MicroSD card

2.9. Modul SD Card

SD card module adalah modul untuk pembaca kartu microSD dengan antarmuka menggunakan SPI. Pada Arduino sendiri sudah menyediakan *library* untuk bisa mengakses modul ini sehingga penggunaannya lebih mudah. *Power supply* untuk modul ini sebesar 5V dengan dilengkapi regulator tegangan 3,3V.



Gambar 2.7 Modul SD card

2.10. Tinjauan Pustaka

Tinjauan pustaka bertujuan untuk membandingkan perangkat yang telah ada dan dikembangkan sebelumnya dengan perangkat yang dirancang pada tugas akhir ini. Berikut merupakan judul paper atau proyek yang dibandingkan dengan proyek pada tugas akhir ini.

2.10.1. *An Automatic Parachute Separation System for Dispersal Miniature Reconnaissance Robot*

Pada penelitian ini dibuat sebuah sistem yang dapat mengenali secara presisi proses menjatuhkan barang dari modul peluncuran mulai dari robot miniature mendarat dan kemudian menentukan apakah robot akhirnya mendarat. Kontroler akan mengaktifkan motor untuk melakukan pemisahan parasut dengan robot yang telah mendarat. Digunakan sensor accelerometer ADXL345 untuk mengenali perubahan percepatan pada sumbu x, y, dan z[16].

2.10.2. *Parachute Decelerate Trajectory Optimization Design Based on Genetic Algorithm*

Dalam penelitian ini, topik yang diangkat adalah optimasi dari lintasan perlambatan parasut. Dilakukan analisa pengaruh dari hambatan parasut dan penurunan rumus batas luas kanopi. Untuk memastikan UAV memasuki tujuan kerja, diambil rentang dan waktu penerbangan sebagai fungsi kinerja, mempertimbangkan kecepatan

akhir dan ketinggian akhir sebagai kendala optimasi, dan pemilihan parameter dari parasut seperti luas kanopi dan waktu pembukaan sebagai rancangan variabel. Lintasan perlambatan yang optimal diselesaikan dengan metode pemrograman kuadratik sekuensial dan *genetic algorithm*. Dari kedua metode tadi akan dibandingkan hasilnya untuk aplikasi lintasan perlambatan ini. Mekanisme ketepatan waktu dari lintasan perlambatan parasut akan dianalisa secara singkat. Hasil contoh numerik mengindikasikan kebenaran dari rancangan lintasan perlambatan parasut, yang memiliki nilai praktis untuk memandu penelitian rancangan parasut[17].

2.10.3. *Prediction of the Parachute Deploy for Landing at Desired Point*

Pada penelitian ini, topik yang diangkat tentang bagaimana memprediksi pembukaan parasut untuk pendaratan pada titik yang diinginkan. Sistem parasut UAV membutuhkan pemodelan 9 DOF, sehingga dibuat persamaan gerak untuk sistem ini. Kemudian data input dan output di-*training* menggunakan *neural network*. Dataset input yang dipakai adalah kondisi terbang UAV seperti, kecepatan UAV, posisi pembukaan parasut, dan kecepatan angin. Sedangkan untuk dataset output yang dipakai adalah titik pendaratannya seperti, posisi rentang silang dan rentang bawah yang disimulasikan dengan pemodelan dinamis 9 DOF. Dengan menggunakan *training* input dan output dataset, dapat dibuat fungsi perkiraan nonlinier untuk *neural network*. Sehingga dapat diprediksi waktu pembukaan parasut kondisinya[18].

2.10.4. *UAV Fault Detection based on GA-BP Neural Network*

UAV mempunyai sistem kontrol yang nonlinier dan kompleksitas yang tinggi, sensor merupakan perangkat yang sangat penting untuk menyediakan data penerbangan untuk sistem, sehingga performanya menjadi penentu UAV saat terbang. Sensor biasanya rusak secara tiba – tiba. Dalam penelitian ini digunakan paket wavelet untuk mencari fitur kesalahan dan merancang sistem deteksi kegagalan, yaitu Back Propagation neural network yang dioptimasi dengan *Genetic Algorithm.* Algoritmanya tidak hanya mengatasi kekurangan dari *training* Back Propagation neural network yang lambat dan mudah jatuh ke minimum lokal, tapi juga meningkatkan kinerja sistem dan pengenalan kegagalan. Studi simulasi

menunjukkan sistem bahwa, yang memiliki prospek aplikasi yang baik, dapat dengan efisien mendeteksi kegagalan sensor UAV[19].

2.10.5. *Advantages of Using Multi-Agent Principles in FAIL SAFE UAV System Design*

Pada penelitian ini disajikan penggunaan *multi-agent* sebagai *fail-safe* ketika terjadi kegagalan pada sistem UAV. *Multi-agent* digunakan untuk mendeteksi keadaan UAV jika terjadi kerusakan pada sistem dan kemudian akan mengatasinya. Sistem tersebut harus mampu merespon kerusakan dalam kasus apapun, sehingga sistem mampu melanjutkan fungsi kerjanya kembali. Beberapa kemungkinan kerusakannya yaitu kerusakan pada sensor utama yang sangat berperan penting dalam pergerakan UAV atau ketika baterai habis ketika masih terbang. Harapannya sistem tersebut mampu merespon secara tepat dan jika memungkinkan untuk mencegah kerusakan fisik pada komponen yang ada pada UAV. Pencegahannya dapat berupa alarm peringatan bahwa telah terjadi kegagalan sistem atau yang lainnya sehingga ketika kegagalan terjadi di udara, UAV dapat mendarat dengan aman dan mengurangi resiko kerusakan komponen[20].

2.10.6. *A Simplex Architecture for Intelligent and Safe Unmanned Aerial Vehicles*

Pada penelitian ini, dirancang desain sistem toleransi kesalahan pada UAV. Pada pengerjaannya menggunakan dua platform perangkat lunak dan perangkat keras yang heterogen dengan karakteristik keandalan dan performa yang berbeda, yaitu *High-Assurance* (HA) dan *High-Performance* (HP). HA berfokus pada kesederhanaan dan verifikasi pada perangkat lunak dan menggunakan prosesor toleran yang mudah dan sementara. Sedangkan HP berfokus pada kecerdasan dan fungsionalitas pada perangkat lunak dan menggunakan prosesor yang kompleks dan berkinerja tinggi. Selama operasi normal, platform HP bertanggung jawab untuk mengendalikan UAV. Namun, jika terjadi kegagalan karena perangkat sementara atau bug perangkat lunak, platform HA akan mengambil alih sampai platform HP kembali normal. Untuk platform HA menggunakan Arduino, sedangkan untuk platform HP menggunakan Tegra TK1[21].

2.10.7. Mars Parachutes

Mars Parachutes merupakan salah satu perusahaan yang mengembangkan parasut untuk digunakan sebagai *safety flight* pada drone. Peralatan yang dibuat terdiri dari dua bagian, yaitu parasut beserta tempat penyimpanannya dan kontroler. Tempat penyimpanan parasutnya berbentuk tabung. Di dalamnya terdapat parasut dan pegas, yang digunakan untuk melontarkan parasut keluar dari tabung. Untuk menahan penutup tabungnya digunakan servo yang akan dikontrol oleh kontroler. Sehingga tabung tempat parasut bisa lebih fleksibel diletakkan untuk jenis drone yang berbeda – beda. Namun, Mars Parachutes juga menyediakan drone dan perangkat parasutnya dalam satu paket. Untuk kontrolernya terdiri dari dua, yaitu kontroler manual dan kontroler otomatis. Kontroler otomatis bekerja dengan mendeteksi kecepatan jatuh drone. Kontroler manual digunakan untuk mengantisipasi jika kontroler otomatis tidak dapat berfungsi dengan baik[1].

(a) (b) (c)

Gambar 2.8 (a) Tabung Parasut, (b) Kontrol Manual, (c) Kontrol Otomatis dari Mars Parachutes[1]

2.10.8. SkyFallX

SkyFallX sejenis dengan Mars Parachutes, yaitu parasut pengaman yang dipasang pada drone. Jika pada produk sebelumnya yang dideteksi hanya kondisi jatuh bebas atau kecepatan jatuh yang tinggi, maka pada SkyFallX digunakan juga sensor akselerometer dan giroskop untuk mendeteksi posisi kemiringan drone. Sehingga jika posisi drone pada posisi kemiringan sekitar 90 derajat, maka akan dideteksi juga sebagai kondisi tidak normal dan parasut terbuka. Kemudian pada sistem juga diberi tambahan suara alarm ketika terdeteksi kondisi yang telah disebutkan tadi sebelum akhirnya parasut terbuka, sehingga pencarian drone ketika sudah sampai tanah lebih

mudah semisal jatuhnya jauh dari pilot ketika mengoperasikan. Sama seperti Mars Parachutes, SkyFallX juga menyediakan beberapa ukuran parasut tergantung dari beban drone yang akan digunakan[2].

Gambar 2.9 Tabung Parasut dan Kontroler dari SkyFallX[2]

2.10.9. ParaZero

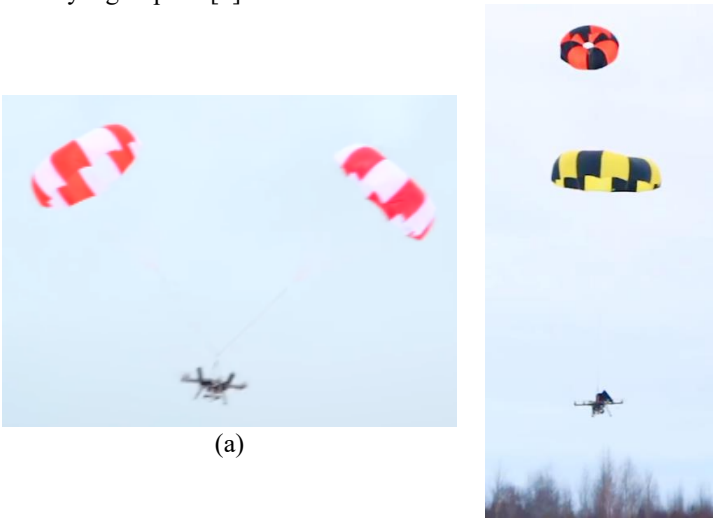
Seperti dua produk sebelumnya, ParaZero juga parasut pengaman yang dipasang pada drone. Namun jika pada perusahaan sebelumnya perangkat parasut yang diproduksi terpisah dengan dronenya atau dengan kata lain bisa dipasang untuk berbagai jenis drone, untuk ParaZero mereka memproduksi perangkat parasutnya didesain hanya untuk beberapa jenis drone saja. Namun mereka juga menyediakan pembuatan perangkat parasut dengan desain drone yang dimiliki konsumen. Sistem pada ParaZero sama seperti Mars Parachutes hanya mendeteksi kondisi jatuh bebas drone. Selain itu, sistem juga dilengkapi dengan alarm seperti pada SkyFallX. Yang membedakan ParaZero dengan dua produk sebelumnya ialah ParaZero menambahkan memori pada sistemnya sehingga data sensor saat terbang bisa disimpan dan bisa digunakan untuk menganalisa keadaan drone ketika jatuh[4].

Gambar 2.10 Parasut Pada DJI Phantom dari ParaZero[4]

2.10.10. Skycat

Skycat mirip dengan produk – produk sebelumnya yaitu parasut yang dipasang pada drone, *quadcopter*, atau sejenisnya. Mereka juga

menyediakan berbagai maca ukuran parasut tergantung berat dari drone atau *quadcopter*. Parasut yang paling besar bisa menahan beban drone hingga 40 kg dengan berat perangkat parasutnya mencapai 0,9 kg. Kemudian jumlah parasut yang digunakan ada yang menggunakan parasut ganda bersebelahan dan parasut ganda atas bawah. Hal tersebut dilakukan untuk memberikan tambahan gaya hambat ke atasnya. Untuk sensornya, Skycat didesain untuk mendeteksi perubahan penurunan ketinggian selama 3-10 meter. Jika terdeteksi penurunan ketinggian, perangkat akan melontarkan parasut. Selain menggunakan kendali otomatis, untuk melontarkan parasut dari dalam tabung juga bisa menggunakan *remote* yang digunakan untuk mengontrol UAV terbang. Ketika parasut sudah terlontar, secara otomatis putaran motor akan langsung berhenti supaya tali parasut tidak terlilit pada motor. Kemudian baterai untuk mensuplai perangkat tersebut bisa dari baterai yang digunakan untuk UAV atau dari sumber baterai yang terpisah[3].



(a) (b)

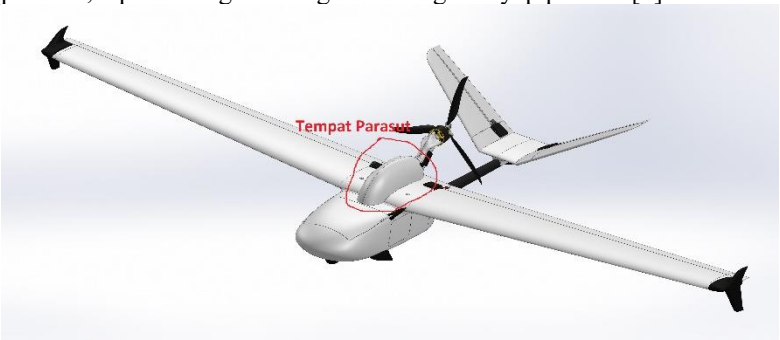
Gambar 2.11 (a) Parasut Ganda Bersebelahan (b) Parasut Ganda Atas Bawah[3]

2.10.11. Fruity Chutes

Berbeda dengan produk – produk sebelumnya yang mengembangkan parasut pengaman hanya untuk drone, Fruity Chutes juga mengembangkan parasut pengaman untuk pesawat *fixed wing*. Selain itu, mereka juga membuat beberapa produk lain yang masih berkaitan juga tentang UAV. Fruity Chutes juga menyediakan berbagai macam ukuran parasut tergantung dari berat pesawat yang akan digunakan. Perusahaan ini memang berfokus pada *recovery system* pada UAV, baik drone maupun pesawat dan dari berbagai macam jenis drone atau pesawat yang ada[5].

2.10.12. FLIRT (Flying Intelligent Robotic Tools)

FLIRT sebenarnya bukan hanya perangkat parasut untuk pengaman UAV, namun FLIRT menjadi satu kesatuan dengan sistem pada pesawat *fixed wing*. Sehingga perangkat parasut tidak bisa dilepas pasang untuk digunakan pada pesawat lain. FLIRT biasanya digunakan untuk kegiatan monitoring dan pemetaan. Rangkaian komponen pada FLIRT tidak hanya digunakan untuk *safety flight*, namun juga untuk keperluan yang telah disebutkan tadi. Sehingga bisa dikatakan FLIRT adalah sistem pada pesawat *fixed wing* yang telah dilengkapi dengan *safety flight*. Parasut diletakkan di atas badan pesawat, tepat di tengah – tengah bentangan sayap pesawat[6].

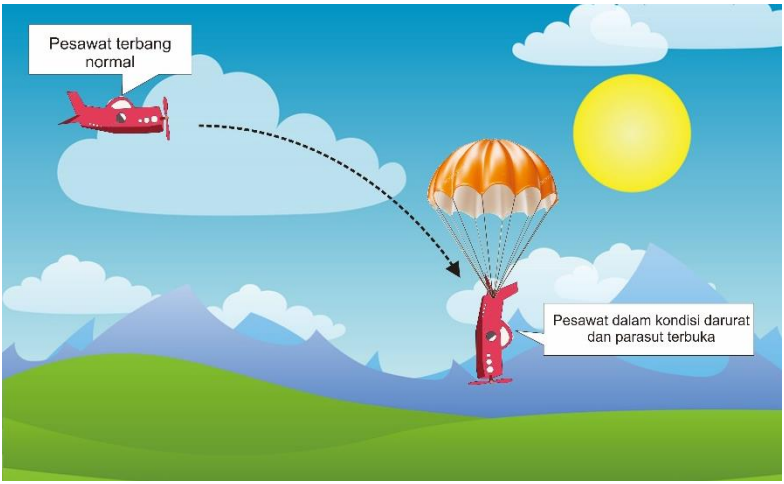


Gambar 2.12 Pesawat FLIRT[6]

.....Halaman ini sengaja dikosongkan.....

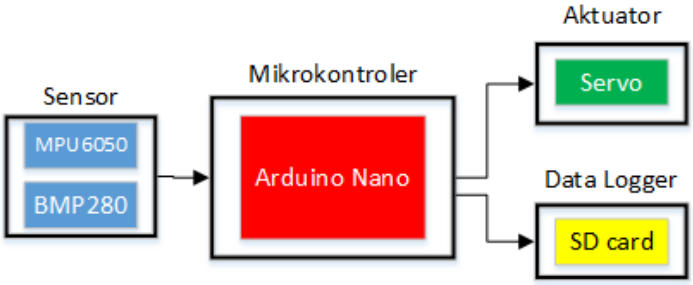
BAB 3
PERANCANGAN SISTEM

Pada bab ini dijelaskan perancangan sistem secara keseluruhan. Perancangan sistem terdiri dari dua bagian, yaitu perancangan perangkat keras dan perancangan perangkat lunak. Perancangan perangkat keras meliputi perancangan komponen elektronik, pembuatan parasut, dan pemasangan parasut pada pesawat. Komponen yang digunakan antara lain, Arduino Nano, sensor BMP280, sensor MPU6050, modul SD card, serta servo. Sensor BMP280 digunakan untuk mengetahui ketinggian serta kecepatan vertikal pesawat, sensor MPU6050 digunakan untuk mengidentifikasi arah gerak dan orientasi pesawat seperti roll, pitch, dan yaw. Modul SD card digunakan untuk menyimpan data – data dari sensor layaknya *blackbox*. Servo digunakan untuk menahan penutup parasut. Sedangkan untuk perancangan perangkat lunak meliputi program Arduino Nano dalam pembacaan sensor – sensor yang digunakan, serta penghitungan kecepatan terminal dan penghitungan ketinggian minimal untuk membuka parasut. Penghitungan yang disebutkan tadi akan disimulasikan terlebih dahulu di matlab untuk mengetahui bentuk grafiknya.



Gambar 3.1 Ilustrasi Sistem

3.1. Diagram Blok Sistem



Gambar 3.2 Diagram Blok Sistem

Pada tugas akhir ini mikrokontroler yang digunakan adalah Arduino Nano sebagai unit pengolahan data, penyimpanan data, dan pengeksekusi gerakan servo. Awalnya, Arduino Nano akan membaca data – data sensor berupa ketinggian, kecepatan vertikal pesawat, serta pergerakan roll, pitch dan yaw pesawat. Dari data tersebut, akan diidentifikasi apakah pesawat dalam kondisi darurat atau tidak. Kondisi yang pertama yaitu, ketika pesawat tiba – tiba kehilangan ketinggian atau kecepatan vertikalnya tinggi. Kemudian yang kedua, ketika sudut roll atau pitch nya mencapai mendekati 90 derajat. Ketika pesawat sudah teridentifikasi dalam kondisi darurat tersebut, kemudian sistem akan melakukan penghitungan untuk menentukan pada ketinggian berapa parasut akan terbuka.

Namun, pada percobaan belum tentu sistem yang telah dirancang bisa sesuai dengan yang diinginkan, sehingga perlu adanya analisa data yang dilakukan jika terjadi ketidaksesuaian. Oleh karena itu, dibutuhkan SD card sebagai tempat penyimpanan data yang telah didapat dari sensor. Sehingga data pembacaan sensor selama penerbangan dapat dilihat kembali, kemudian dianalisa dan mengevaluasi sistem yang telah dibuat.

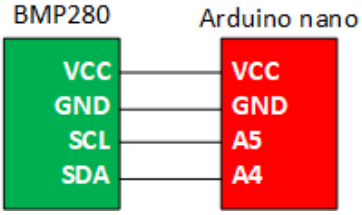
Dalam perencanaannya, sistem bisa memperkirakan kapan parasut harus terbuka sehingga parasut tidak terbuka pada ketinggian yang terlalu tinggi atau terlalu rendah. Jika parasut sudah terbuka pada ketinggian yang masih tinggi, dikhawatirkan pesawat akan terbawa angin dan malah membuat pesawat dan parasut tidak stabil. Tetapi, jika parasut terbuka pada ketinggian yang terlalu rendah, dikhawatirkan pesawat terlebih dahulu mencapai tanah sebelum parasut terbuka dengan sempurna dan memperlambat kecepatan jatuhnya pesawat. Selain itu, ketinggian minimal untuk membuka parasut juga tergantung pada seberapa berat

massa pesawat. Semakin berat massa pesawat, semakin tinggi juga ketinggian minimal untuk membuka parasut.

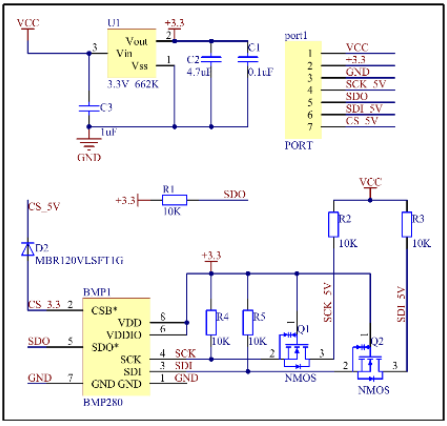
3.2. Perancangan Perangkat Keras

3.2.1. Sensor BMP280

Sensor barometer yang digunakan pada tugas akhir ini yaitu BMP280. Sensor barometer yaitu sensor yang digunakan untuk mengetahui tekanan udara dipermukaan. Sensor ini berupa modul dengan dimensi 19,2mm x 17,9mm dan tinggi 2,9 mm. Tegangan masukan pada sensor ini berkisar antara 3-5V. Sensor ini mampu mendeteksi tekanan udara antara 300 – 1100 hPa dengan akurasi pembacaan ±1 hPa. Pada tugas akhir ini, sensor ini digunakan untuk mengetahui ketinggian pesawat yang didapat dari mengkonversi pembacaan nilai tekanan udara. Antarmuka BMP280 dengan Arduino nano menggunakan pin I2C yaitu pin SDA dan SCL atau pin A5 dan A4 pada Arduino nano.



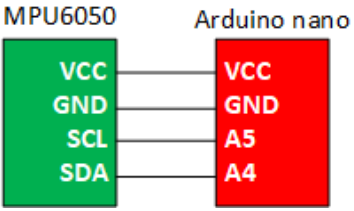
Gambar 3.3 Antarmuka BMP280 dengan Arduino nano



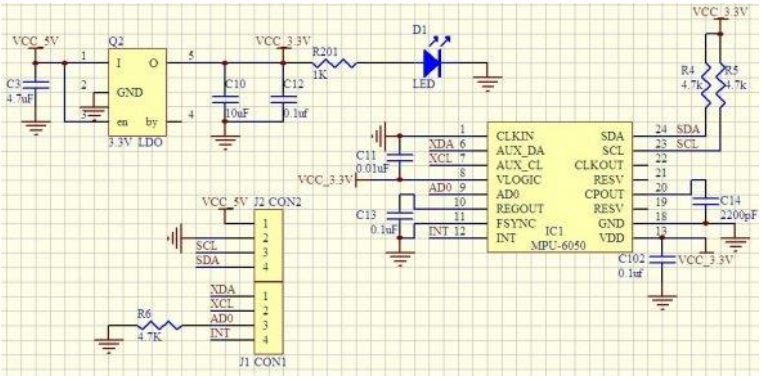
Gambar 3.4 Rangkaian Skematik BMP280[22]

3.2.2. Sensor MPU6050

Sensor akselerometer dan giroksop yang digunakan pada tugas akhir ini yaitu MPU6050. Sensor accelerometer yaitu sensor yang bisa mendeteksi orientasi berdasarkan gerakan ke arah sumbu x, y, dan z dengan membaca nilai percepatannya. Kemudian sensor gyroscope yaitu sensor yang digunakan untuk mendeteksi rotasi suatu benda. Sensor ini berupa modul dengan dimensi 21,2mm x 16,4mm x 3,3mm. Tegangan masukan sensor ini berkisar antara 3-5V. Pada tugas akhir ini, sensor ini digunakan untuk mengetahui orientasi dan kemiringan dari pesawat pada saat terbang. Antarmuka MPU6050 dengan Arduino nano menggunakan I2C juga, sama seperti BMP280.



Gambar 3.5 Antarmuka MPU6050 dengan Arduino nano

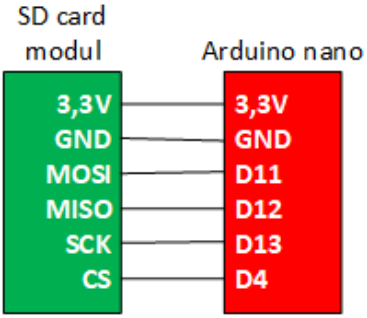


Gambar 3.6 Rangkaian Skematik MPU6050[23]

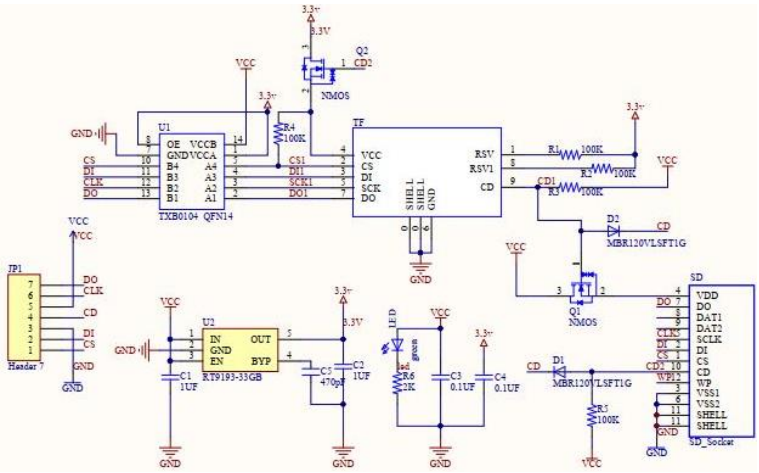
3.2.3. Modul SD Card

Modul SD card berguna untuk menyimpan data – data sensor ketika pesawat sedang terbang. Antarmuka modul SD card dengan

Arduino nano menggunakan pin SPI yaitu MISO, MOSI, dan SCK (pin 11, 12, dan 13). Selain itu ada tambahan satu pin lagi untuk *chip select* pada pin 4. Untuk tegangan masukannya, modul ini menyediakan dua pin dengan tegangan yang berbeda yaitu 3,3V dan 5V.



Gambar 3.7 Antarmuka modul SD card dengan Arduino nano

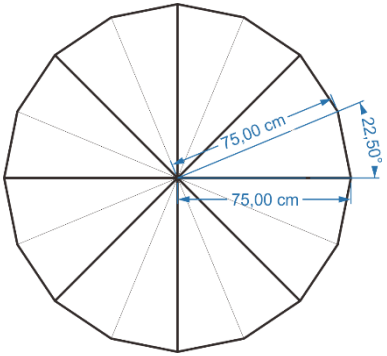


Gambar 3.8 Rangkaian Skematik Modul SD Card[24]

3.2.4. Desain Parasut

Pembuatan parasut menggunakan jenis kain parasut yang tipis dan ringan, yang biasa digunakan sebagai lapisan dalam jaket parasut. Pemilihan kain yang tipis dan ringan agar kain bisa mudah

mengembang ketika parasut terbuka di udara. Kemudian untuk tali parasut menggunakan benang wol, yang tidak terlalu tipis sehingga bisa merusak parasut serta tidak terlalu tebal dan berat sehingga tidak membebani parasut. Parasut ini memiliki bentuk segienambelas dengan panjang jari – jari 75 cm. Namun untuk jumlah tali yang digunakan hanya sebanyak 8 buah yang terikat pada tepi sudut parasut setiap dua sudut. Hal ini supaya tali tidak mudah melilit satu sama lain mengingat ukuran parasut yang tidak terlalu besar juga. Berikut gambar desain parasut beserta ukurannya.



Gambar 3.9 Desain Parasut dan Ukurannya

Garis jari – jari yang lebih tebal pada gambar 3.9 menunjukkan posisi untuk mengikat tali parasut. Sehingga tali yang digunakan sebanyak 8 dengan panjang kira – kira 1,6 meter setiap tali. Kemudian digunakan satu tali lagi sebagai pengikat antara badan pesawat dengan ujung setiap tali yang terikat pada parasut.

Berdasarkan gambar 3.9 dapat dihitung luas permukaan parasut dengan menggunakan rumus luas segitiga karena bentuk parasutnya seperti terdiri dari 16 buah segitiga yang disatukan. Rumus luas segitiga sebagai berikut.

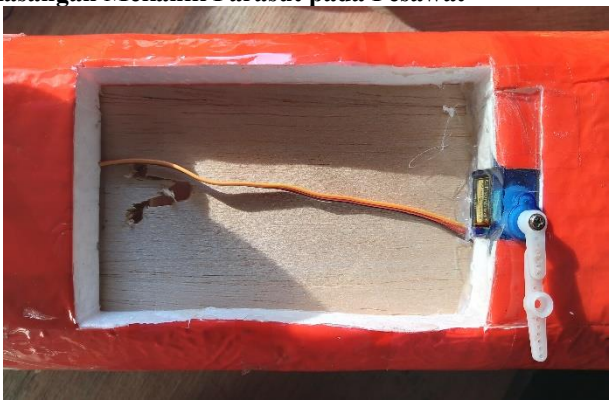
$$Luas\ \Delta = \frac{1}{2} \times r \times r \times sin\ \alpha \quad (3.1)$$

$$Luas\ \Delta = \frac{1}{2} \times 0{,}75m \times 0{,}75m \times sin\ 22{,}5 = 0{,}108\ m^2 \quad (3.2)$$

$$Luas\ Segi16 = 16 \times 0{,}108 = 1{,}722\ m^2 \quad (3.3)$$

Luas permukaan parasut tersebut nantinya akan digunakan dalam perhitungan untuk mencari nilai kecepatan terminal pesawat ketika parasut terbuka.

3.2.5. Pemasangan Mekanik Parasut pada Pesawat



Gambar 3.10 Tempat Meletakkan Parasut dan Servo

Gambar 3.11 Parasut yang Dimasukkan di Tempat Parasut

Setelah pembuatan parasut selesai, kemudian parasut akan

dirakit ke pesawat. Parasut akan diletakkan di bagian bawah pesawat seperti yang ditunjukkan pada gambar 3.11. Badan pesawat diberi lubang berbentuk persegi panjang dengan ukuran 12,5 x 8 cm. Kemudian diberi tambahan bahan sebagai penutup atau penahan dari bahan yang ringan agar parasut tetap berada di badan pesawat sebelum waktunya terbuka. Untuk penahan penutup tadi digunakan servo sebagai penahannya yang nantinya gerakan servo akan dikontrol oleh Arduino. Untuk bisa meletakkan parasut ke tempat yang telah dibuat, parasut harus dilipat sedemikian rupa sehingga bisa masuk ke dalam kotak yang telah dibuat. Biasanya parasut dilipat hingga berukuran sekitar 12x7,5 cm.

Gambar 3.12 Tampak Bawah Pesawat, Tempat Parasut

3.3. Perancangan Perangkat Lunak

Perancangan perangkat lunak disini meliputi algoritma pemrograman untuk melakukan beberapa proses, seperti pembacaan sensor barometer BMP280, pembacaan sensor akselerometer dan giroksop MPU6050, dan *data logger*.

3.3.1. Pembacaan BMP280

Pembacaan BMP280 menggunakan *library* "BME280I2C.h" dan "wire.h" pada software Arduino IDE. Dengan menggunakan *library* ini dapat diperoleh nilai tekanan udara. Untuk mencari nilai ketinggian, dibutuhkan nilai tekanan udara permukaan laut sehingga nilai ketinggian bisa terukur dari permukaan laut. Sedangkan untuk nilai tekanan udara permukaan laut setiap saat berbeda dan harus mengecek tekanan udara permukaan laut saat itu di tempat itu. Maka dari itu, dilakukan kalibrasi tekanan udara dahulu yang akan digunakan sebagai acuan dan dimasukkan ke dalam program untuk mendapatkan ketinggian. Sehingga hasil yang didapat berupa ketinggian dari tempat sensor tersebut melakukan kalibrasi, bukan dari permukaan laut. Berikut program pembacaan tekanan udara dan kalibrasinya

```
void baca_tekanan() {
  BME280::PresUnit presUnit(BME280::PresUnit_hPa);
  tekanan = bmp.pres(presUnit);
}
void kalibrasi_bmp() {
  Serial.println("kalibrasi tekanan");
  for (int i=0; i<JML_KALIBRASI; i++) {
    baca_tekanan();
    seaLevelhPa += tekanan;
  }
  seaLevelhPa /= JML_KALIBRASI;
}
```

Jumlah kalibrasi sebanyak 1000 kali. Namun, data ketinggian yang didapat masih kurang stabil, sehingga digunakan sebuah filter agar hasilnya lebih stabil yaitu filter eksponensial. Data ketinggian tadi difilter sebanyak dua kali. Berikut program filter eksponensial yang digunakan

```
float expFilter(float input, float smoothBefore, float
alpha){
  return smoothBefore + alpha*(input-smoothBefore);
}
alt_filter = expFilter(ketinggian, alt_before, 0.1);
alt_before = alt_filter;
alt_filter2 = expFilter(alt_filter, alt_before2, 0.1);
alt_before2 = alt_filter2;
```

3.3.2. Pembacaan MPU6050

Pembacaan MPU6050 hanya menggunakan *library* "wire.h" kemudian diambil data akselerometer dan giroskop menggunakan antarmuka I2C. Sebelum mengambil data dari sensor, dilakukan pengaturan terlebih dahulu terhadap sensor seperti, pengaturan rentang skala penuh yang dipakai dan filter yang akan dipakai. Berikut program yang digunakan untuk pengaturan MPU6050:

```
Wire.beginTransmission(I2C_ADDRESS);
Wire.write(ADDRESS_REG);
Wire.write(NILAI_REGISTER);
Wire.endTransmission();
```

I2C_ADDRESS adalah alamat perangkat yang akan digunakan dan harus didefinisikan ketika memulai pengiriman atau penerimaan data. ADDRESS_REG adalah alamat register yang akan diatur nilai didalamnya. NILAI_REGISTER adalah nilai yang akan dimasukkan ke dalam alamat register yang berisi pengaturan dari setiap alamat register. Semua nilai tadi ditulis dengan banyak data sebanyak 16 bit. Untuk membaca data akselerometer dan giroskop perintahnya mirip dengan perintah diatas, namun ada sedikit perbedaan yaitu perintah Wire.write hanya dilakukan sekali karena alamat register digunakan untuk membaca, tidak untuk menulis data. Dan setelah mengakhiri transmisi, ditambah perintah untuk melakukan pembacaan nilai alamat register sebanyak berapa alamat setelahnya.

Kemudian data dari akselerometer digunakan untuk menghitung nilai roll dan pitch menggunakan rumus sebagai berikut:

$$roll = \tan^{-1}\left(\frac{aksel\_Y}{aksel\_Z}\right)$$

$$pitch = -\tan^{-1}\left(\frac{aksel\_X}{\sqrt{aksel\_Y^2 + aksel\_Z^2}}\right)$$

Berdasarkan rumus tersebut, nilai pembacaan roll akan memiliki rentang dari -180 sampai 180 derajat. Sedangkan untuk nilai pitch akan memiliki rentang dari -90 sampai 90 derajat. Sehingga untuk mengetahui posisi pitch nantinya akan diperhatikan juga nilai rollnya karena nilai pitchnya hanya

memiliki lebar nilai sebesar 180 derajat. Sedangkan untuk posisi roll tidak perlu memperhatikan nilai pitch karena nilai roll sudah memiliki lebar nilai sebesar 360 derajat.

Kemudian dari nilai sudut dan giroskop akan difilter menggunakan Kalman filter agar pembacaan sudutnya lebih akurat dan presisi. Untuk menggunakan Kalman filter memakai *library* yang sudah ada sehingga cukup memasukkan beberapa nilai yang dibutuhkan. Nilai yang digunakan yaitu nilai sudut, nilai giroskop, dan nilai *time sampling*. Berikut program dari pembacaan sudutnya hingga menggunakan Kalman Filter:

```
X = atan2(accelY, accelZ)*180/PI;
Y = -atan2(accelX, sqrt(pow(accelY, 2) + pow(accelZ,
2)))*180/PI;
double dt1 = (double)(micros() - timer1) / 1000000;
timer1 = micros();
roll = kalmanX.getAngle(X, gyroX, dt1); //kalman filter
pitch = kalmanY.getAngle(Y, gyroY, dt1);
```

3.3.3. Penyimpanan Data

Data yang telah didapat dari beberapa sensor akan disimpan di dalam microSD. Proses penyimpanan ini disebut *data logging*. Untuk bisa mengakses microSD digunakan *library* “SD.h” dan “SPI.h”. Dengan menggunakan *library* tersebut, Arduino nano dapat membaca maupun menulis file pada microSD. Data yang sudah didapat akan disimpan di dalam file dengan ekstensi .csv. Berikut program untuk menyimpan data pada microSD:

```
data = SD.open("data.csv", FILE_WRITE);
if (data) {
  data.print(roll); data.print("\t");
  data.print(pitch); data.print("\t");
  data.print(alt_filter2); data.print("\t");
  data.print(kecepatan_filter); data.print("\t" );
  data.print(parasut); data.print("\t");
  data.print(v_before); data.print("\t");
  data.print(h_total); data.println();
  data.close();
} else {
    Serial.println("error opening data.csv");
}
```

3.3.4. Pendeteksian Kondisi Darurat UAV

Ada beberapa kondisi darurat yang biasanya terjadi pada UAV yaitu baterai habis ketika di udara, terjadi error kelistrikan, dan terjadi error mekanik. Dari beberapa kondisi tersebut ditentukan beberapa parameter yang digunakan sebagai penanda bahwa UAV sedang dalam kondisi darurat.

Yang pertama adalah dengan mendeteksi kecepatan vertikalnya. Kecepatan vertikal UAV didapat dari mendiferensialkan nilai ketinggian yang didapat dari pembacaan sensor. Berikut penghitungan untuk mencari kecepatan pada program Arduino.

```
if(millis()- last >= SAMPLE_RATE){
  last = millis();
  dt2 = SAMPLE_RATE/1000.0;
  x2 = alt_filter2;
  kecepatan = (x2 - x1)/dt2;
  kecepatan_filter = expFilter(kecepatan, v_sebelum,
0.1);
  v_sebelum = kecepatan_filter;
  x1 = x2;
}
```

Yang kedua dengan mendeteksi posisi kemiringan dari UAV. Posisi kemiringan yang dideteksi yaitu posisi roll dan pitch. UAV biasanya dianggap pada kondisi darurat ketika posisi kemiringannya mendekati 90 derajat. Namun ada beberapa sebab yang kadang kemiringan UAV tidak sampai mendekati nilai 90 derajat. Hal itu disebabkan oleh berat pesawat. Jika berat pesawat lebih condong ke depan, maka kemungkinan kemiringan dari pesawat ketika jatuh juga akan semakin mendekati sudut 90 derajat.

3.3.5. Penghitungan Kecepatan Terminal

Kecepatan terminal disini untuk mengetahui kecepatan maksimum pesawat ketika sedang jatuh dan parasut terbuka. Sehingga dari kecepatan terminal tersebut bisa digunakan untuk memperkirakan kapan harus terbuka.

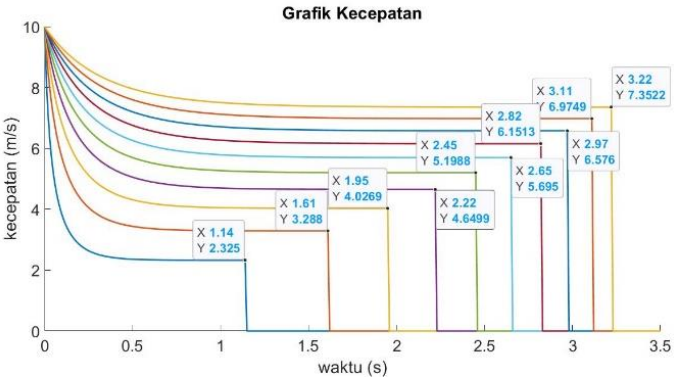
$$v = \sqrt{\frac{2 * m * g}{Cd * \rho * A}}$$

m = massa benda (kg)
g = percepatan gravitasi (m/s^2)
Cd = koefisien hambat
ρ = massa jenis udara (kg/m^3)
A = luas permukaan benda (m^2)

Dari gambar 3.5, gambar desain parasut, didapatkan luas parasut sebagai berikut. Luas segitiga = $^1/_2 * 0{,}75m * 0{,}75m * \sin 22{,}5 = 0{,}108\ m^2$. Luas keseluruhan parasut = $16 * 0{,}199m^2 = 1{,}728\ m^2$. Untuk nilai Cd pada parasut, secara umum mempunyai nilai 1,75, sedangkan untuk nilai ρ adalah 1,2 kg/m3 [25]. Sehingga didapat nilai kecepatan terminalnya

$$v = \sqrt{\frac{2 \times 1{,}2 \times 9{,}8}{1{,}75 \times 1{,}2 \times 1{,}722}} = 2{,}546\ {}^m/_s$$

Dari rumus tersebut juga bisa diketahui bahwa semakin berat massa pesawat, semakin tinggi juga kecepatan terminal yang dicapai pesawat ketika parasut terbuka. Selain itu juga diketahui berapa lama waktu yang dibutuhkan untuk mencapai kecepatan terminal dengan inisialisasi kecepatan awal adalah nol atau ketika pesawat dijatuhkan. Berikut gambar grafik tersebut, dengan dimisalkan kecepatan awal ketika jatuh adalah 10 m/s.



Gambar 3.13 Grafik Hubungan Kecepatan Terminal dengan Massa (kecepatan awal 10)

Dari gambar 3.7 terlihat bahwa semakin berat massa benda, semakin lama waktu yang dibutuhkan untuk mencapai kecepatan terminalnya. Kemudian semakin berat massa benda, semakin tinggi juga nilai kecepatan terminalnya.

3.3.6. Perumusan Ketinggian Minimal untuk Membuka Parasut

Dengan menggunakan Hukum Newton II, akan didapat juga percepatan pesawat saat mulai terjatuh hingga mencapai kecepatan terminalnya. Rumus untuk Hukum Newton II sebagai berikut:

$$F = m \times a \tag{3.1}$$

Pada persamaan 3.1 gaya yang diterima oleh pesawat ketika jatuh ada dua, yaitu gaya berat pesawat (W) dan gaya hambat udara (D). Sehingga persamaan menjadi:

$$W - D = m \times a \tag{3.2}$$

Dengan memasukkan rumus dari gaya berat dan gaya hambat udara pada persamaan 3.2, akan didapat percepatan pesawat ketika jatuh

$$m \times g - \frac{Cd \times \rho \times A \times v^2}{2} = m \times a \tag{3.3}$$

$$a = g - \frac{Cd \times \rho \times A \times v^2}{2 \times m} \tag{3.4}$$

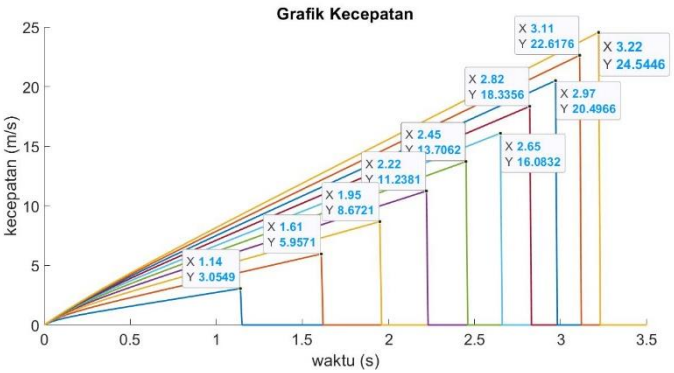
Dengan diketahui percepatan dan kecepatan, bisa diperkirakan juga jarak yang ditempuh pesawat ketika jatuh. Untuk memperkirakan jarak yang ditempuh digunakan persamaan Gerak Lurus Berubah Beraturan sebagai berikut:

$$a[n] = g - \frac{Cd \times \rho \times A \times v[n-1]^2}{2 \times m} \tag{3.5}$$

$$v[n] = v[n-1] + a[n] \times \Delta t \tag{3.6}$$

$$h[n] = v[n] \times \Delta t + \frac{1}{2} \times a[n] \times \Delta t^2 \tag{3.7}$$

$$htotal[n] = h[n] + h[n-1] \tag{3.8}$$



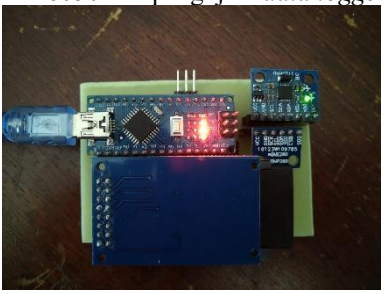
Gambar 3.14 Grafik Jarak untuk Mencapai Kecepatan Terminal

Gambar 3.14 adalah grafik jarak yang ditempuh pesawat mulai dari sesaat setelah parasut terbuka hingga pesawat mencapai kecepatan terminal ketika parasut terbuka. Semakin besar massa pesawat, semakin tinggi nilai kecepatan terminalnya sehingga semakin jauh jarak yang ditempuh untuk mencapai kecepatan terminal. Sehingga berakibat jika semakin berat massa pesawat, ketinggian minimal untuk membuka parasut semakin tinggi juga. Rumus 3.5 sampai 3.8 kemudian dimasukkan ke dalam program utama di Arduino nano.

.....Halaman ini sengaja dikosongkan.....

BAB 4
PENGUJIAN DAN ANALISIS

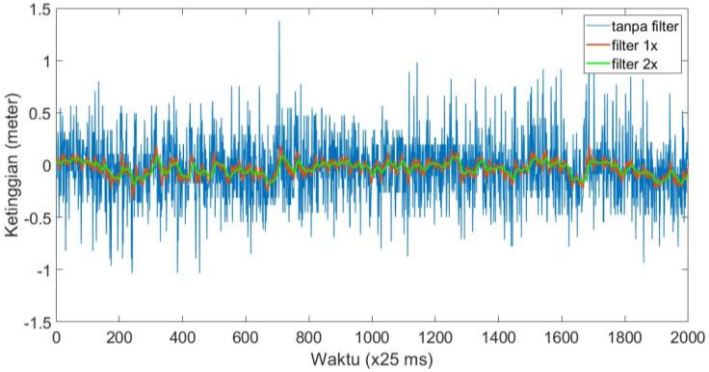
Pada bab ini akan dibahas mengenai pengujian dari sistem yang telah dirancang pada bab sebelumnya. Bab ini bertujuan untuk mendapatkan data yang kemudian dilakukan analisa pada masing-masing pengujian. Pengujian yang dilakukan meliputi pengujian sensor BMP280, pengujian sensor MPU6050 dan pengujian *data logger*.



Gambar 4.1 Realisasi Alat (Mikrokontroler, Sensor, dan Data Logger)

4.1. Pengujian Pembacaan Ketinggian

Pengujian sensor BMP280 dilakukan untuk mengetahui error pembacaan pada sensor. Didapat data mentah dari sensor memiliki *noise* yang sangat banyak, sehingga harus difilter terlebih dahulu agar hasilnya lebih bagus.



Gambar 4.2 Perbandingan Pembacaan Ketinggian Tanpa Filter dan Dengan Filter

Dari gambar 4.2 terlihat bahwa pembacaan keti nggian setelah difilter menjadi lebih stabil. Pembacaan ketinggian nanti akan digunakan untuk menghitung kecepatan vertikal dari pesawat, sehingga dibutuhkan hasil pembacaan yang stabil.

Kemudian untuk mengetahui error pembacaan, dilakukan pengujian di Gedung B Teknik Elektro ITS dari lantai bawah sampai lantai yang paling atas. Ketinggian dari lantai dasar sampai paling atas yaitu lantai 4 diperkirakan adalah 12 meter. Setiap kenaikan lantai memiliki ketinggian 4 meter. Pengujian ini dilakukan untuk mengetahui apakah pembacaan sensor sudah bekerja dengan baik. Seperti yang telah dijelaskan pada Bab 3, bahwa nilai tekanan udara yang dipakai pada penghitungan ketinggian adalah nilai tekanan udara di tempat sensor diletakkan saat itu. Pada pengujian, sensor melakukan kalibrasi di lantai 1, sehingga seharusnya ketinggian yang didapat adalah nol meter.

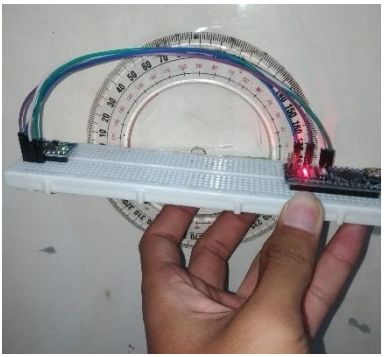
Tabel 4.1 Hasil pengujian BMP280 di Gedung B Teknik Elektro ITS terhadap lantai dasar Gedung

Lokasi	Ketinggian (meter)	Pembacaan sensor (meter)	Error (meter)
Lantai 1	0	-0,2821	0,2821
Lantai 2	4	3,6269	0,3731
Lantai 3	8	7,6493	0,3507
Lantai 4	12	11,84	0,16

Dari hasil pembacaan ketinggian pada tabel 4.1 dapat diketahui error rata – rata sebesar 0,2915 meter.

4.2. Pengujian Pembacaan Roll dan Pitch

Pengujian MPU6050 dilakukan untuk mengetahui respon sensor MPU6050 terhadap perubahan sudutnya. Pengujian dilakukan dengan menguji kemiringan sensor berdasarkan sumbu X dan sumbu Y atau jika dalam pesawat dikenal dengan istilah roll dan pitch. Pengujian menggunakan busur derajat untuk mengetahui derajat kemiringan sensor.



Gambar 4.3 Pengujian Sensor MPU6050

4.2.1. Pengujian Roll

Tabel 4.2 Hasil Pengukuran Sudut Roll dan Error

Posisi (º)	Pembacaan (º)	Error (º)	Posisi (º)	Pembacaan (º)	Error (º)
0	0,71	0,71	-180	-179,33	0,67
10	11,91	1,91	-170	-170,25	0,25
20	23,12	3,12	-160	-161,46	1,46
30	34,23	4,23	-150	-152,52	2,52
40	45,15	5,15	-140	-143,53	3,53
50	55,79	5,79	-130	-134,53	4,53
60	66,27	6,27	-120	-125,10	5,10
70	76,79	6,79	-110	-115,70	5,70
80	87,04	7,04	-100	-105,85	5,85
90	96,77	6,77	-90	-96,12	6,12
100	106,80	6,80	-80	-86,15	6,15
110	116,32	6,32	-70	-75,66	5,66
120	126,01	6,01	-60	-65,50	5,50
130	135,49	5,49	-50	-54,39	4,39
140	144,49	4,49	-40	-43,62	3,62
150	153,71	3,71	-30	-32,77	2,77
160	162,76	2,76	-20	-21,67	1,67
170	171,90	1,90	-10	-10,49	0,49

Untuk pembacaan nilai roll memiliki rentang dari -180^{o} sampai 180^{o}. Dari hasil pembacaan terlihat bahwa semakin tegak sensor, semakin besar errornya. Kemudian didapat rata – rata error sebesar 4,2^{o}.

4.2.2. Pengujian Pitch

Untuk pengujian sudut pitch, terdapat dua tabel yang berbeda karena nilai dari sudut pitch memiliki rentang nilai dari -90 sampai 90. Sehingga akan dilihat juga pembacaan nilai rollnya juga.

Tabel 4.3 Pembacaan Nilai Pitch, Posisi 0^{o} Menghadap ke Atas

Posisi Pitch (-180^{o} s/d 180^{o})	Posisi Pitch (-90^{o} s/d 90^{o})	Pembacaan Roll (^{o})	Pembacaan Pitch (^{o})	Error Pitch (^{o})
-90	-90	123,54	-87,21	2,79
-80	-80	28,40	-85,45	5,45
-70	-70	9,17	-75,72	5,72
-60	-60	6,79	-65,86	5,86
-50	-50	5,91	-55,64	5,64
-40	-40	4,61	-45,24	5,24
-30	-30	5,45	-34,65	4,65
-20	-20	4,93	-23,81	3,81
-10	-10	4,76	-12,98	2,98
0	0	5,21	-1,23	1,23
10	10	3,65	9,87	0,13
20	20	-1,81	21,30	1,30
30	30	6,50	32,51	2,51
40	40	-1,94	43,91	3,91
50	50	8,10	54,79	4,79
60	60	7,80	66,17	6,17
70	70	10,90	77,01	7,01
80	80	39,21	87,05	7,05

Tabel 4.4 Pembacaan Nilai Pitch, Posisi 0^o Menghadap ke Bawah

Posisi Pitch (-180^o s/d 180^o)	Posisi Pitch (-90^o s/d 90^o)	Pembacaan Roll (^{o})	Pembacaan Pitch (^{o})	Error Pitch (^{o})
90	90	170,10	81,80	8,20
100	80	-160,48	72,85	7,15
110	70	-175,85	63,69	6,31
120	60	177,74	53,51	6,49
130	50	177,28	44,37	5,63
140	40	177,62	35,16	4,84
150	30	178,93	26,01	3,99
160	20	177,87	16,92	3,08
170	10	177,54	7,82	2,18
180	0	179,77	-1,11	1,11
-170	-10	177,23	-17,51	7,51
-160	-20	177,01	-26,47	6,47
-150	-30	176,97	-35,49	5,49
-140	-40	176,89	-44,45	4,45
-130	-50	176,54	-53,67	3,67
-120	-60	175,81	-62,71	2,71
-110	-70	172,23	-74,45	4,45
-100	-80	160,17	-83,65	3,65

Dari pembacaan nilai pitch dapat diketahui bahwa ada perubahan nilai roll juga. Untuk tabel 4.3 didapat nilai roll akan selalu kurang dari mutlak 90 atau mendekati 0, sedangkan untuk tabel 4.3 nilai roll akan selalu lebih dari mutlak 90 atau mendekati 180. Error pembacaan yang didapatkan sebesar 4,545^o.

4.3. Pengujian Pada Pesawat

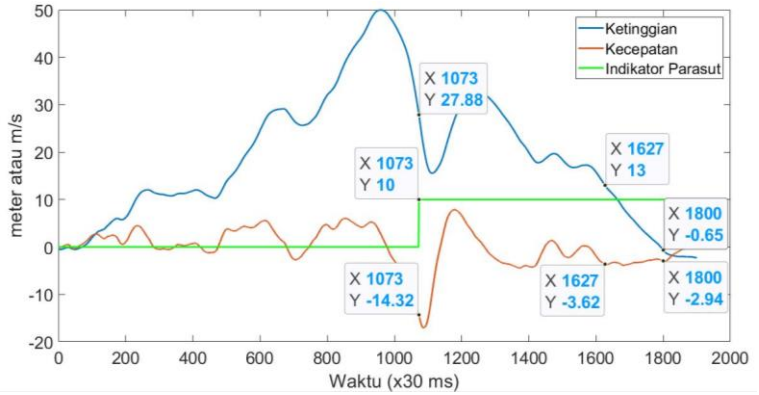
Pengujian dilakukan dilakukan dengan memposisikan pesawat pada kondisi darurat, yaitu jika pesawat miring roll atau pitch dan kecepatan vertikalnya besar. Dari beberapa pengujian ada beberapa yang berhasil untuk membuka parasut dan ada yang tidak. Yang tidak berhasil dikarenakan posisi pesawat masih belum dikenali dalam kondisi darurat. Kemudian yang berhasil akan diperhatikan berapa waktu yang dibutuhkan parasut saat mulai keluar dari pesawat sampai parasut sudah mengembang dengan sempurna. Akan dibandingkan juga antara perhitungan rumus dan data sebenarnya dan dicari errornya.

4.3.1. Pengujian Pertama

Percobaan pertama dilakukan di sekitar Apartemen Puncak Kertajaya. Pengujian kali ini untuk menguji saat kondisi pesawat jatuh bebas. Pada mulanya diatur terlebih dahulu kecepatan terminal ketika pesawat jatuh tanpa parasut dan pesawat jatuh dengan parasut mengembang. Kemudian dari penghitungan kecepatan terminal keduanya, didapatkan jarak minimal yang harus ditempuh dari kecepatan terminal pesawat jatuh tanpa pesawat sampai kecepatan terminal pesawat jatuh dengan parasut mengembang

Gambar 4.4 Percobaan Pertama, Parasut Terbuka

Hasilnya parasut berhasil terbuka seperti yang terlihat pada gambar 4.4. Indikator parasut terbuka tercatat pada data ke 1073 atau pada detik ke 32,19. Namun parasut baru mengembang pada data ke 1627 atau detik ke 48,81. Sehingga terjadi keterlambatan pembukaan parasut selama 16,62 detik. Kecepatan pesawat ketika jatuh sebesar 3,23 m/s. Berdasarkan perhitungan pada bab 3, didapat bahwa kecepatan terminalnya 2,546 m/s. Sehingga terdapat error sebesar 0,684 m/s antara perhitungan dengan data sebenarnya. Hal ini dapat disebabkan oleh angin yang menerpa pesawat dan parasut.

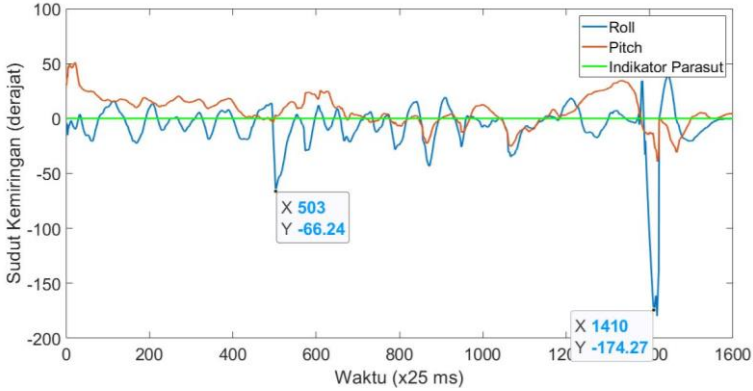


Gambar 4.5 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Pertama

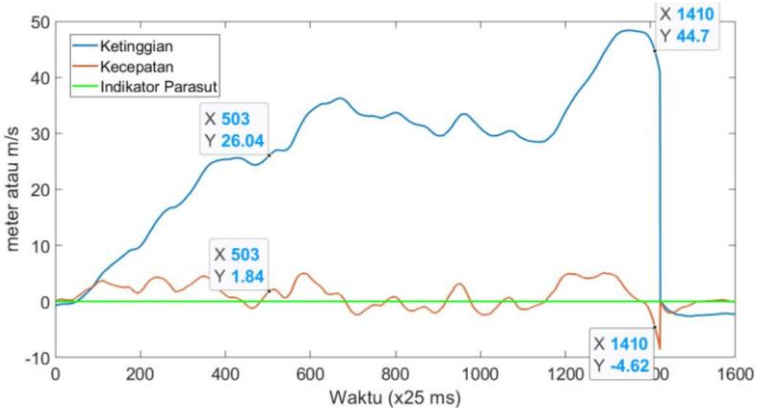
4.3.2. Pengujian Kedua

Pengujian kedua dilakukan di tempat yang sama seperti sebelumnya. Namun kali ini ada perubahan algoritma yang digunakan. Jika sebelumnya hanya bergantung pada kecepatan terminal pesawat saat jatuh dengan parasut tidak terbuka dan parasut terbuka, kali ini pengujian dilakukan dengan mendeteksi kemiringan roll dan pitch pesawat dahulu. Setelah indikasi kemiringan telah terlampaui, yakni lebih dari 75 derajat atau kurang dari -75 derajat, kemudian program akan melakukan penghitungan untuk memprediksi jarak ketinggian minimal untuk membuka parasut. Sudut yang akan dideteksi adalah sudut roll atau pitch.

Hasilnya pada pengujian ini parasut tidak berhasil terbuka karena dari data yang tercatat menunjukkan bahwa nilai dari roll atau pitch masih belum melewati batas nilai sehingga pesawat belum bisa dikatakan dalam kondisi darurat. Selain itu, tercatat juga bahwa nilai roll pesawat mencapai -174,27 derajat (gambar 4.6). Namun karena saat pada posisi tersebut ketinggian pesawat masih di ketinggian 44,7 meter (gambar 4.7), maka sensor tidak mendeteksi bahwa pesawat dalam kondisi darurat. Hal ini dapat terjadi karena posisi peletakan sensor tidak sejajar dengan posisi badan pesawat. Sehingga ketika pesawat terlihat sudah melebihi batas nilai roll atau pitchnya, pada pembacaan sensor masih belum mencapai batas nilai yang telah ditentukan.



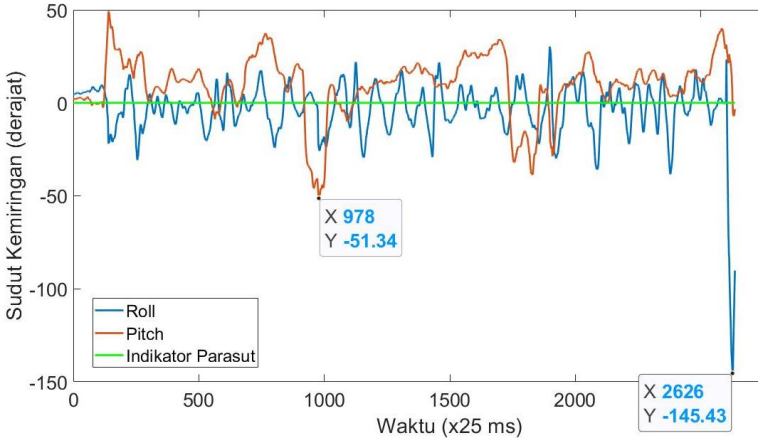
Gambar 4.6 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Kedua



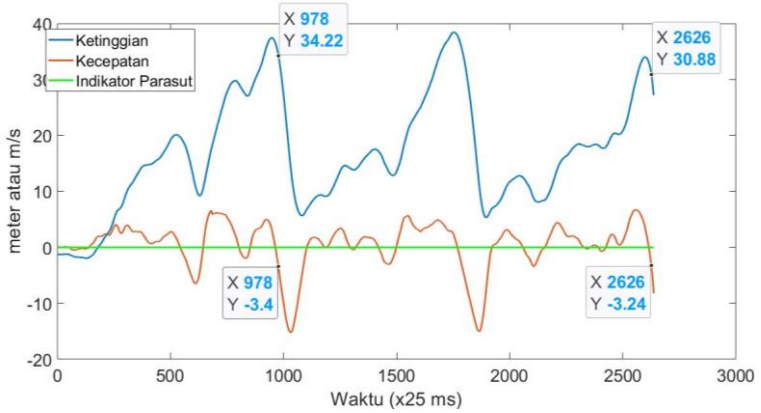
Gambar 4.7 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Kedua

4.3.3. Pengujian Ketiga

Pada pengujian ketiga, parasut tidak berhasil terbuka karena dari data yang tercatat menunjukkan bahwa pesawat masih belum dalam kondisi darurat, yaitu nilai roll atau pitch mencapai lebih dari 70 derajat. Dari grafik tercatat nilai roll ada yang mencapai -145,43 derajat (gambar 4.8). Namun karena ketinggian pesawat masih berada di ketinggian 30,88 meter (gambar 4.9) sensor masih belum mendeteksi bahwa posisi dalam kondisi darurat sehingga parasut tidak terbuka



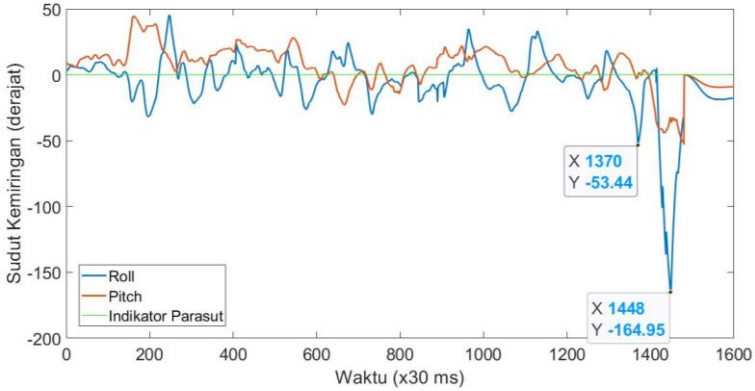
Gambar 4.8 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Ketiga



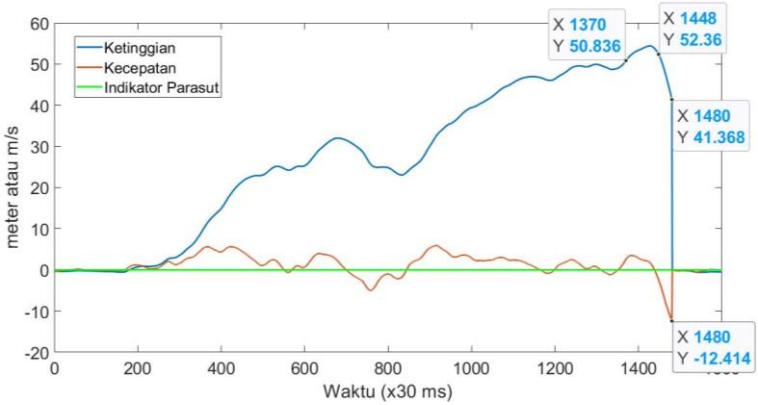
Gambar 4.9 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Percobaan Ketiga

4.3.4. Pengujian Keempat

Pada pengujian keempat, parasut tidak terbuka karena dari data yang tercatat pesawat belum mencapai kondisi darurat. Pada gambar 4.10 ketika roll pada kemiringan -164,95 derajat ketinggian pesawat 41,368 meter, terlihat pada gambar 4.11.



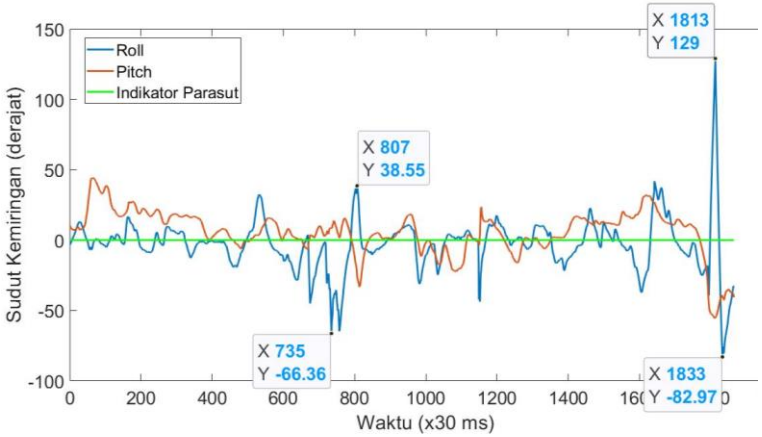
Gambar 4.10 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Keempat



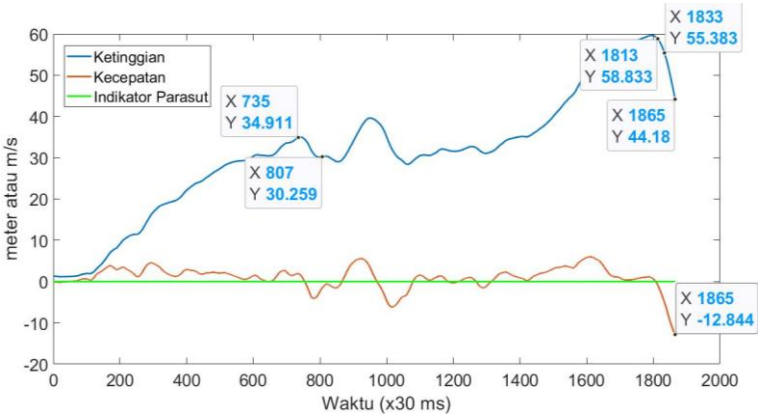
Gambar 4.11 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Keempat

4.3.5. Pengujian Kelima

Pada pengujian kelima, parasut tidak terbuka karena dari data yang tercatat pesawat belum mencapai kondisi darurat. Pada gambar 4.12 ketika roll sudah melebihi batas nilai kondisi darurat, ketinggian pesawat masih diatas 30 meter seperti terlihat pada gambar 4.13.



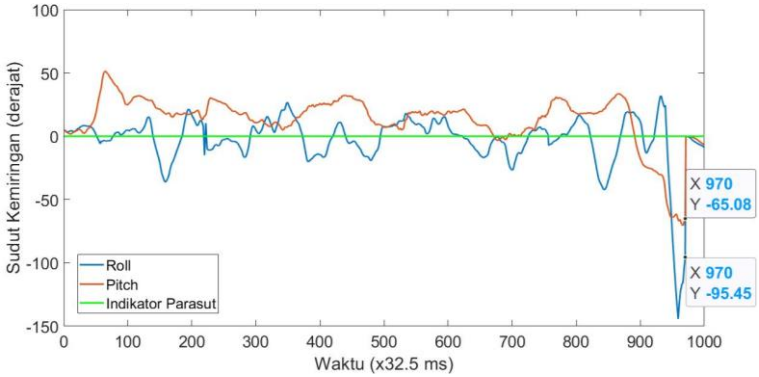
Gambar 4.12 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Kelima



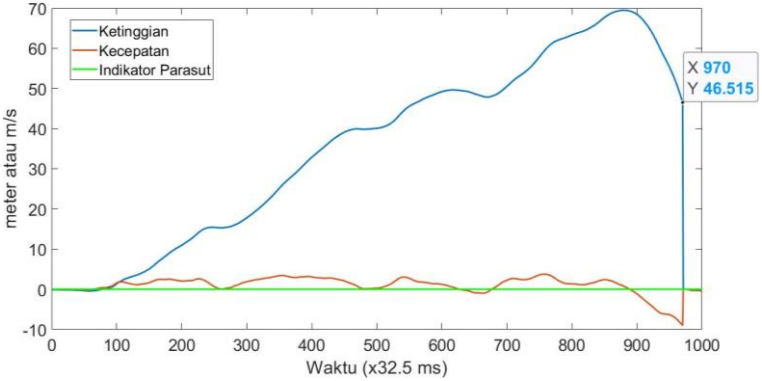
Gambar 4.13 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Kelima

4.3.6. Pengujian Keenam

Pada pengujian kelima, parasut tidak terbuka karena dari data yang tercatat pesawat belum mencapai kondisi darurat. Pada gambar 4.14 ketika roll sudah melebihi batas nilai kondisi darurat, ketinggian pesawat masih diatas 30 meter seperti terlihat pada gambar 4.15.



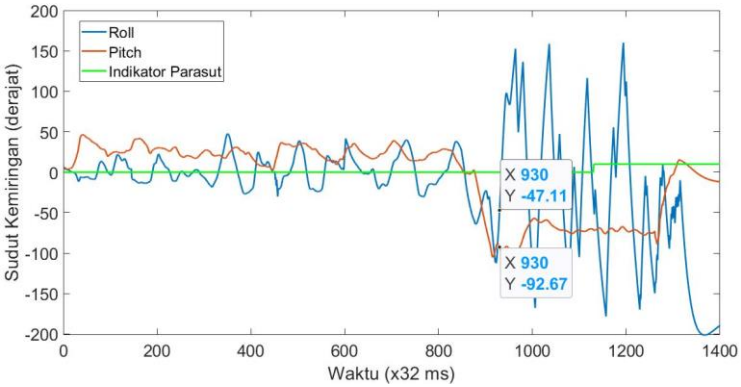
Gambar 4.14 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Keenam



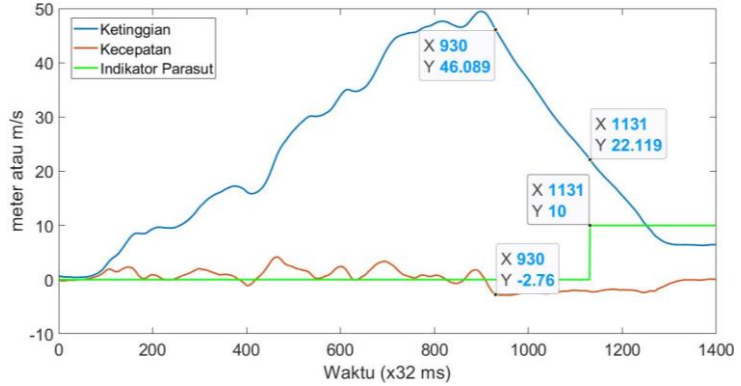
Gambar 4.15 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Keenam

4.3.7. Pengujian Ketujuh

Pada pengujian ketujuh, parasut berhasil terbuka pada detik ke 29,76 (data ke 930). Namun indikator parasut terbuka baru menyala pada detik ke 36,192 (data ke 1131). Hal ini dapat terjadi karena pengait untuk menahan parasut agak longgar, sehingga terlepas ketika terbang. Kemudian untuk kecepatan pesawat ketika jatuh dengan parasut terbuka yaitu 2,26 m/s, sedangkan berdasarkan perhitungan kecepatan terminalnya didapat 2,546 m/s. Sehingga terdapat error sebesar 0,286 m/s.



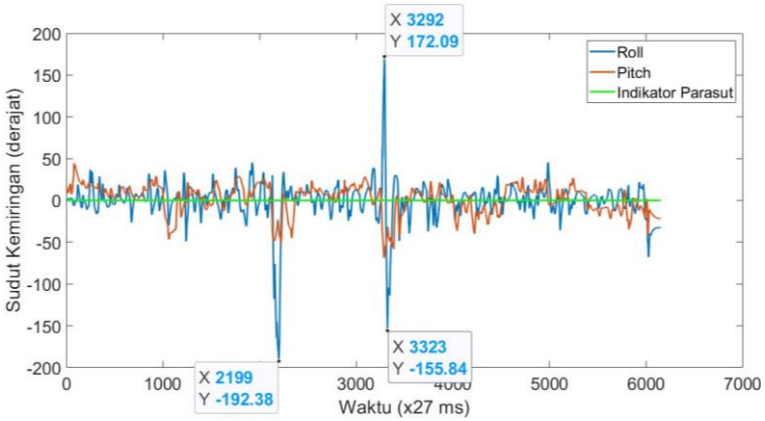
Gambar 4.16 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Ketujuh



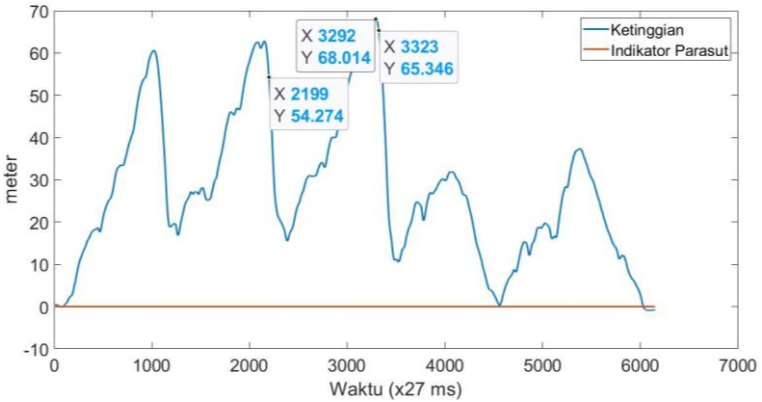
Gambar 4.17 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Ketujuh

4.3.8. Pengujian Kedelapan

Pada pengujian kelima, parasut tidak terbuka karena dari data yang tercatat pesawat belum mencapai kondisi darurat. Pada gambar 4.18 ketika roll sudah melebihi batas nilai kondisi darurat, ketinggian pesawat masih diatas 30 meter seperti terlihat pada gambar 4.19.



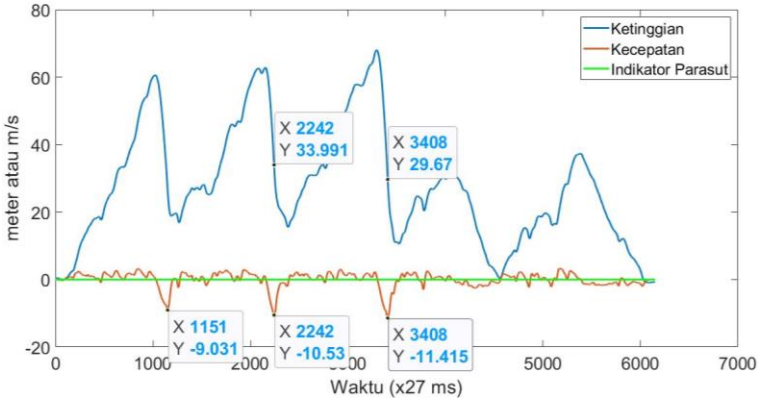
Gambar 4.18 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Kedelapan



Gambar 4.19 Grafik Ketinggian dan Indikator Parasut Pengujian Kedelapan

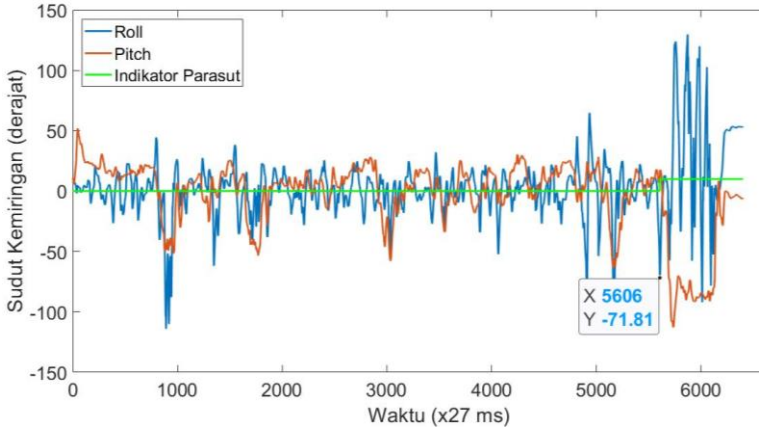
Kemudian jika dilihat berdasarkan kecepatan jatuhnya pada gambar 4.20, ketika kecepatannya lebih dari 10 m/s ketinggian pesawat masih diatas ketinggian minimalnya. Sehingga parasut tidak terbuka. Ketinggian minimalnya dihitung dapat dihitung dengan cara 10 detik dikali kecepatan terminal ketika parasut sudah terbuka, berikut perhitungannya

$$10\ detik\ \times 2{,}546\ {}^{m}/_{s} = 25{,}46\ m$$

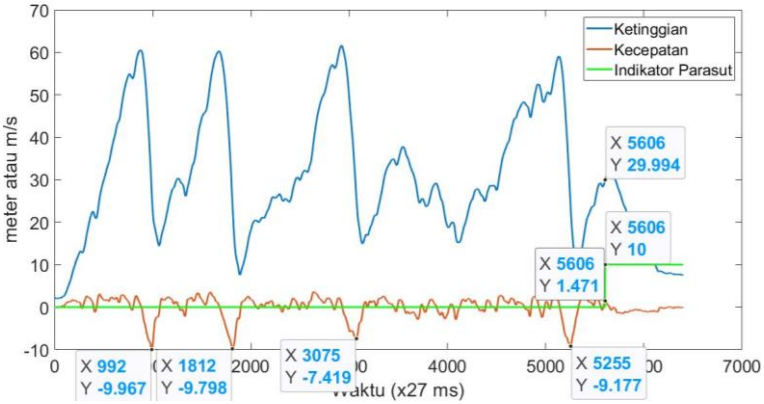


Gambar 4.20 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Kedelapan

4.3.9. Pengujian Kesembilan



Gambar 4.21 Grafik Roll, Pitch, dan Indikator Parasut Pengujian Kesembilan



Gambar 4.22 Grafik Ketinggian, Kecepatan, dan Indikator Parasut Pengujian Kesembilan

Pada pengujian kesembilan parasut berhasil terbuka. Indikator parasut menyala pada detik ke 151,362 dan parasut mengembang keseluruhan pada detik ke 154,71. Sehingga ada selisih waktu selama 3,348 detik antara parasut keluar dari pesawat dan parasut mengembang. Kemudian kecepatan pesawat jatuh ketika parasut sudah terbuka sebesar 0,98 m/s. Jika dibandingkan dengan perhitungan yaitu sebesar 2,546 m/s, maka errornya sebesar 1,566 m/s. Hal ini mungkin terjadi karena pesawat sempat terkena angin ketika akan jatuh.

4.4. Analisa Keseluruhan Sistem

Secara keseluruhan alat ini mampu bekerja dengan baik. Penyimpanan data sensor ke dalam SD card berjalan dengan baik dalam format .csv. Sensor barometer dapat mengetahui ketinggian pesawat ketika terbang, namun ketika di udara karena angina yang terlalu kencang, ketinggian yang terbaca bisa menjadi lebih tinggi dari ketinggian sebenarnya. Untuk sensor akselerometer dan giroskop bisa membaca kemiringan pesawat dengan baik.

Untuk pendeteksian kondisi darurat pesawat, sudah bisa mendeteksi kondisi darurat pesawat. Namun karena kecepatan mikrokontroler untuk membaca data sensor masih kurang cepat, sistem masih meleset untuk menetukan pesawat dalam kondisi darurat atau tidak.

BAB 5
PENUTUP

5.1. Kesimpulan

Berdasarkan percobaan yang telah dilakukan pada pelaksanaan tugas akhir ini didapat beberapa kesimpulan sebagai berikut:

1. Nilai error rata – rata pada pembacaan ketinggian sebesar 0,2915 meter.
2. Nilai error rata – rata pada pembacaan roll dan pitch masing – masing sebesar 4,2 derajat dan 4,545 derajat.
3. Error kecepatan terminal sebenarnya dibandingkan dengan perhitungan sebesar 0,845 m/s.
4. Selisih waktu antara parasut keluar dari pesawat dengan parasut mengembang dengan sempurna adalah 3,348 detik.

5.2. Saran

Sebagai sarana pengembangan Parasut Otomatis untuk Kondisi Darurat UAV ini, maka penulis memberikan saran

1. Menambahkan indikator untuk mendeteksi kondisi darurat UAV.
2. Menggunakan metode lain seperti *neural network* untuk melakukan pendeteksian
3. Perlu memperhitungkan desain untuk meletakkan parasut pada pesawat.

.....Halaman ini sengaja dikosongkan.....

DAFTAR PUSTAKA

[1] "Home," *Mars Parachute*. [Daring]. Tersedia pada: https://www.marsparachutes.com/. [Diakses: 02-Jun-2019].
[2] "Home Page," *SkyFallX*. [Daring]. Tersedia pada: http://www.skyfallx.com/. [Diakses: 02-Jun-2019].
[3] "GUIDE," *Skycat.pro Parachute Launchers*. [Daring]. Tersedia pada: https://www.skycat.pro/tech-specs. [Diakses: 16-Jun-2019].
[4] "Our Solutions," *ParaZero*. .
[5] "Fixed Wing Bundle : Fruity Chutes!, Parachutes Manufacturer Drones, Rockets, UAV, Research." [Daring]. Tersedia pada: https://fruitychutes.com/buyachute/drone-and-uav-parachute-recovery-c-21/fixed-wing-bundle-c-21_22/. [Diakses: 02-Jun-2019].
[6] "FLIRT," *Abris Pilotless Tools*, 28-Des-2017. .
[7] "The UAV - Unmanned Aerial Vehicle." [Daring]. Tersedia pada: https://www.theuav.com/. [Diakses: 21-Mei-2019].
[8] A. S. Al Ayyubi, A. B. Cahyono, dan H. Hidayat, "Pemetaan Foto Udara Menggunakan Wahana Fix Wing UAV (Studi Kasus : Kampus ITS Sukolilo," *Jurnal Teknik ITS*, vol. 6, no. 2, Sep 2017.
[9] F. Cain, "What Is Terminal Velocity?," *Universe Today*, 14-Sep-2010. .
[10] "Terminal Velocity." [Daring]. Tersedia pada: https://www.grc.nasa.gov/www/k-12/airplane/termv.html. [Diakses: 05-Mei-2019].
[11] "Shape Effects on Drag." [Daring]. Tersedia pada: https://www.grc.nasa.gov/www/k-12/airplane/shaped.html. [Diakses: 22-Jun-2019].
[12] "BMP280." [Daring]. Tersedia pada: https://www.bosch-sensortec.com/bst/products/all_products/bmp280. [Diakses: 26-Mar-2019].
[13] "MPU-6050 | TDK." .
[14] H. Mayditia, "PERANCANGAN DAN IMPLEMENTASI PENGENDALIAN SISTEM ATTITUDE INERSIAL SATU SUMBU MENGGUNAKAN REACTION WHEEL DAN GIROSKOP MEMS," hlm. 105, 2011.
[15] "What is MATLAB?" [Daring]. Tersedia pada: https://www.mathworks.com/discovery/what-is-matlab.html. [Diakses: 21-Mei-2019].

[16] Y. Chen dan X. Shi, “An automatic parachute separation system for aerial dispersal miniature reconnaissance robot,” dalam *2016 Chinese Control and Decision Conference (CCDC)*, Yinchuan, China, 2016, hlm. 1538–1545.
[17] C. Wang, Institute of Electrical and Electronics Engineers, dan Hubei-Gongye-Daxue, Ed., *2009 International Workshop on Intelligent Systems and Applications: (ISA) ; Wuhan, China, 23 - 24 May 2009*. Piscataway, NJ: IEEE, 2009.
[18] Keisoku-jidō-seigyō-gakkai, Akita Daigaku, dan IEEE Control Systems Society, Ed., *2012 proceedings of SICE annual conference (SICE 2012): Akita, Japan, 20 - 23 August 2012*. Piscataway, NJ: IEEE, 2012.
[19] Y. Chen, C. Zhang, Q. Zhang, dan X. Hu, “UAV fault detection based on GA-BP neural network,” dalam *2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, Hefei, China, 2017, hlm. 806–811.
[20] D. Novak dan P. Cermak, “Advantages of using multi-agent principles in FAIL - SAFE UAV system design,” dalam *2011 16th International Conference on Methods & Models in Automation & Robotics*, Miedzyzdroje, Poland, 2011, hlm. 162–167.
[21] P. Vivekanandan, G. Garcia, H. Yun, dan S. Keshmiri, “A Simplex Architecture for Intelligent and Safe Unmanned Aerial Vehicles,” dalam *2016 IEEE 22nd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, Daegu, South Korea, 2016, hlm. 69–75.
[22] “BMP280 Pressure Sensor Module - Wiki.” [Daring]. Tersedia pada: http://wiki.sunfounder.cc/index.php?title=BMP280_Pressure_Sensor_Module. [Diakses: 03-Jun-2019].
[23] “Interfacing MPU-6050 GY-521 Accelerometer Gyrometer with Arduino Uno,” *electroSome*, 16-Des-2018. .
[24] “SD/TF Card Reader Module - Wiki.” [Daring]. Tersedia pada: http://wiki.sunfounder.cc/index.php?title=SD/TF_Card_Reader_Module. [Diakses: 03-Jun-2019].
[25] “Velocity During Recovery.” [Daring]. Tersedia pada: https://www.grc.nasa.gov/WWW/K-12/VirtualAero/BottleRocket/airplane/rktvrecv.html. [Diakses: 21-Mei-2019].

LAMPIRAN A

Program Pada Arduino

```
#include <Kalman.h>
#include <Servo.h>
#include <SD.h>
#include <SPI.h>
#include <Wire.h>
#include <BME280I2C.h>

#define SAMPLE_RATE 50
#define I2C_ADDRESS 0x68
#define JML_KALIBRASI 1000

#define M 1.2 //massa
#define G 9.80665 //percepatan gravitasi
#define Cd 1.75 //koefisien gesek
#define RHO 1.2 //massa jenis udara
#define A 1.722 //luas permukaan
#define TS 0.1

Kalman kalmanX;
Kalman kalmanY;
Servo servo;
File data;
BME280I2C bmp;

uint32_t mulai, selesai, data_timer;
int parasut = 0, a = 0;

//variabel mpu6050
double accelX, accelY, accelZ, gyroX, gyroY, gyroZ;
double gyro_x_kalibrasi, gyro_y_kalibrasi, gyro_z_kalibrasi;
double roll, pitch, yaw;
double X, Y;
```

```
uint32_t timer1;

//variabel bmp280
double seaLevelhPa, tekanan;
double ketinggian, alt_filter, alt_before, alt_filter2, alt_before2;
double kecepatan, kecepatan_filter, v_sebelum, dt2, x1, x2;
uint32_t last;
double v_after, v_before, accel, h, h_total;
double v_terminal;

void setup() {
  pinMode(5, OUTPUT);
  digitalWrite(5, HIGH);
  Serial.begin(115200);
  Wire.begin();
  servo.attach(3);
  servo.write(45);

  v_terminal = sqrt((2*M*G) / (Cd*RHO*A));

  //bmp280
  if (!bmp.begin()) Serial.println("bmp tidak terbaca");
  kalibrasi_bmp();

  //sd card
  if (!SD.begin(4)) Serial.println("initialization failed!");

  //mpu6050
  setupMPU();
  kalibrasi_gyro();
  baca_accel();
  X = atan2(accelY, accelZ)*180/PI;
  Y = -atan2(accelX, sqrt(pow(accelY, 2) + pow(accelZ, 2)))*180/PI;
  kalmanX.setAngle(roll);
  kalmanY.setAngle(pitch);
  timer1 = micros();
```

```
  digitalWrite(5, LOW);
}

void loop() {
//---------------------------------MPU6050--------------------------------
  baca_accel();
  baca_gyro();
  gyroX -= gyro_x_kalibrasi;
  gyroY -= gyro_y_kalibrasi;

  X = atan2(accelY, accelZ)*180/PI;
  Y = -atan2(accelX, sqrt(pow(accelY, 2) + pow(accelZ, 2)))*180/PI;

  double dt1 = (double)(micros() - timer1) / 1000000;
  timer1 = micros();

  roll = kalmanX.getAngle(X, gyroX, dt1); //kalman filter
  pitch = kalmanY.getAngle(Y, gyroY, dt1);

  Serial.print("\tkalman X: "); Serial.print(roll);
  Serial.print("\tkalman Y: "); Serial.print(pitch);

//---------------------------------BMP280---------------------------------------
  baca_tekanan();
  ketinggian = tinggi(seaLevelhPa, tekanan);
  alt_filter = expFilter(ketinggian, alt_before, 0.1);
  alt_before = alt_filter;
  alt_filter2 = expFilter(alt_filter, alt_before2, 0.1);
  alt_before2 = alt_filter2;

  if(millis()- last >= SAMPLE_RATE){
    last = millis();
    dt2 = SAMPLE_RATE/1000.0;
    x2 = alt_filter2;
    kecepatan = (x2 - x1)/dt2;
    kecepatan_filter = expFilter(kecepatan, v_sebelum, 0.1);
```

```
    v_sebelum = kecepatan_filter;
    x1 = x2;
   }

//  Serial.print("\t"); Serial.print(alt_filter2);
//  Serial.print("\t"); Serial.print(kecepatan_filter);

   if (roll > 75 || roll < -75 || pitch > 75 || pitch < -75){
    v_before = -1*kecepatan_filter;
    accel = G;
    while (accel > 0.001 || accel < -0.001) {
     accel = G - ((pow(v_before,2)*RHO*A*Cd) / (2*M));
     v_after = v_before + accel*TS;
     h = v_after*TS + 0.5*accel*pow(TS,2);
     h_total = h_total + h;
     v_before = v_after;
//     Serial.print("\t"); Serial.print(v_before,3);
//     Serial.print("\t"); Serial.print(accel,3);
//     Serial.print("\t"); Serial.println(h_total,3);
//     Serial.print("\t"); Serial.println(4*v_terminal + h_total);
    }
    if (alt_filter2 < 10*v_terminal + h_total){
     servo.write(145);
     parasut = 10;
     digitalWrite(5, HIGH);
    }
//    h_total = 0;
   }
   else if (kecepatan_filter < -13){
    v_before = -1*kecepatan_filter;
    accel = G;
    while (accel > 0.001 || accel < -0.001) {
     accel = G - ((pow(v_before,2)*RHO*A*Cd) / (2*M));
     v_after = v_before + accel*TS;
     h = v_after*TS + 0.5*accel*pow(TS,2);
     h_total = h_total + h;
```

```
      v_before = v_after;
    }
    if (alt_filter2 < 10*v_terminal + h_total){
      servo.write(145);
      parasut = 20;
      digitalWrite(5, HIGH);
    }
//    h_total = 0;
  }
  Serial.print("\t"); Serial.print(v_before,3);
  Serial.print("\t"); Serial.print(h_total,3);

//--------------------Penyimpanan Data-------------------
  data = SD.open("23_Juni.csv", FILE_WRITE);
  if (data) {
    data.print(roll); data.print("\t");
    data.print(pitch); data.print("\t");
    data.print(alt_filter2,3); data.print("\t");
    data.print(kecepatan_filter,3); data.print("\t" );
    data.print(parasut); data.print("\t");
    data.print(h_total,3); data.println();
    data.close();
  } else {
     Serial.println("error opening data.csv");
  }
  h_total = 0;
  Serial.println();
}

float expFilter(float input, float smoothBefore, float alpha){
  return smoothBefore + alpha*(input-smoothBefore);
}

void setupMPU(){
  Wire.beginTransmission(I2C_ADDRESS); //This is the I2C address of
the MPU (b1101000/b1101001 for AC0 low/high datasheet sec. 9.2)
```

```
  Wire.write(0x6B);                 //Accessing the register 6B - Power
Management (Sec. 4.28)
  Wire.write(0b00000000);           //Setting SLEEP register to 0.
(Required; see Note on p. 9)
  Wire.endTransmission();

  Wire.beginTransmission(I2C_ADDRESS); //I2C address of the MPU
  Wire.write(0x1B);                 //Accessing the register 1B - Gyroscope
Configuration (Sec. 4.4)
  Wire.write(0x08);                 //Setting the gyro to full scale +/-
500deg./s
  Wire.endTransmission();

  Wire.beginTransmission(I2C_ADDRESS); //I2C address of the MPU
  Wire.write(0x1C);                 //Accessing the register 1C -
Acccelerometer Configuration (Sec. 4.5)
  Wire.write(0x10);                 //Setting the accel to +/- 8g
  Wire.endTransmission();

  Wire.beginTransmission(I2C_ADDRESS); //Start communication with
the address found during search
  Wire.write(0x1A);                  //We want to write to the CONFIG
register (1A hex)
  Wire.write(0x03);                 //Set the register bits as 00000011 (Set
Digital Low Pass Filter to ~43Hz)
  Wire.endTransmission();           //End the transmission with the gyro
}

void kalibrasi_gyro() {
  Serial.println("kalibrasi gyroscope");
  for(int i=0; i<JML_KALIBRASI; i++)
  {
   baca_gyro();
   gyro_x_kalibrasi += gyroX;
   gyro_y_kalibrasi += gyroY;
   gyro_z_kalibrasi += gyroZ;
```

```
  }
  gyro_x_kalibrasi /= JML_KALIBRASI;
  gyro_y_kalibrasi /= JML_KALIBRASI;
  gyro_z_kalibrasi /= JML_KALIBRASI;
}

void baca_accel() {
  Wire.beginTransmission(I2C_ADDRESS); //I2C address of the MPU
  Wire.write(0x3B); //Starting register for Accel Readings
  Wire.endTransmission();
  Wire.requestFrom(I2C_ADDRESS,6); //Request Accel Registers (3B -
40)
  while(Wire.available() < 6);
  accelX = Wire.read()<<8|Wire.read();
  accelY = Wire.read()<<8|Wire.read();
  accelZ = Wire.read()<<8|Wire.read();
  accelX /= 4096;
  accelY /= 4096;
  accelZ /= 4096;
}

void baca_gyro() {
  Wire.beginTransmission(I2C_ADDRESS); //I2C address of the MPU
  Wire.write(0x43); //Starting register for Accel Readings
  Wire.endTransmission();
  Wire.requestFrom(I2C_ADDRESS,6); //Request Accel Registers (43 -
48)
  while(Wire.available() < 6);
  gyroX = Wire.read()<<8|Wire.read(); //Store first two bytes into accelX
  gyroY = Wire.read()<<8|Wire.read(); //Store middle two bytes into
accelY
  gyroZ = Wire.read()<<8|Wire.read(); //Store last two bytes into accelZ
  gyroX /= 65.5;
  gyroY /= 65.5;
  gyroZ /= 65.5;
}
```

```
void baca_tekanan() {
  BME280::PresUnit presUnit(BME280::PresUnit_hPa);
  tekanan = bmp.pres(presUnit);
}

double tinggi(double x, double sekarang){
  return 44330 * (1.0 - pow(sekarang / x, 0.1903));
}

void kalibrasi_bmp() {
  Serial.println("kalibrasi tekanan");
  for (int i=0; i<JML_KALIBRASI; i++) {
    baca_tekanan();
    seaLevelhPa += tekanan;
  }
  seaLevelhPa /= JML_KALIBRASI;
}
```

LAMPIRAN B

Program Pada Matlab

```
clc;
clear;

% Define constant
Ts = 0.01; %time sampling
g = 9.80665; %percepatan gravitasi
rho = 1.2; %massa jenis udara
A = 1.728; %luas permukaan benda
C = 1.75; %koefisien gesek

% Variables
t = 0:Ts:3.5;
n = length(t);
dm = 10; %banyak data
m = linspace(1,10,dm); %linspace(mulai, akhir,
banyak data)

% Initial Condition and Compute
v = 10*ones(dm,n);
a = zeros(dm,n);
h = zeros(dm,n);
htotal = zeros(dm,n);

figure(1);
hold on;
for i = 1:dm
    for j = 2:n
        %if (j < n/2)
        %   A = 0.7;
        %else
        %   A = 1.76;
        %end
        a(i,j) = g - (v(i,j-
1)^2*rho*A*C)/(2*m(i));
```

```
        v(i,j) = v(i,j-1) + a(i,j)*Ts;
        if (a(i,j) > -0.001)
            v(i,j) = 0;
        end
        if (a(i,j) < -0.001)
            h(i,j) = v(i,j)*Ts +
0.5*a(i,j)*Ts^2;
            htotal(i,j) = htotal(i,j-1) +
h(i,j);
        end
    end
    plot(t,v(i,:));
    %plot(t,htotal(i,:));
    xlabel('waktu (s)'); ylabel('kecepatan
(m/s)');
    title('Grafik Kecepatan');
end
%Plotting time
figure(2);
[x,y] = meshgrid(t,m);
s = surf(x,y,v);
%s = surf(x,y,htotal);
s.EdgeColor = 'none';
colorbar;
xlabel('waktu (s)'); ylabel('massa (kg)');
zlabel('kecepatan (m/s)');
```

LAMPIRAN C

.....Halaman ini sengaja dikosongkan.....

BIODATA PENULIS

Andre Fajar Nugroho atau sering disapa Andre merupakan anak ke dua dari tiga bersaudara. Penulis lahir di Kediri pada tanggal 7 Juni 1996. Tempat tinggal penulis sekarang berada di Desa Ngunut, Kecamatan Ngunut, Kabupaten Tulungagung, Jawa Timur. Penulis telah menyelesaikan pendidikannya di SDN 6 Ngunut, SMPN 1 Ngunut, serta SMAN 1 Blitar. Kemudian sekarang sedang melanjutkan pendidikannya di Departemen Teknik Elektro Institut Teknologi Sepuluh Nopember. Penulis mengambil konsentrasi bidang Elektronika. Selama kuliah, penulis aktif di dalam berbagai macam kepanitiaan.

Email : andrenugroho395@gmail.com
Line : andre_fajar_nugroho