BAB 3
METODE PENELITIAN

3.1 Blok Diagram dan Flowchart

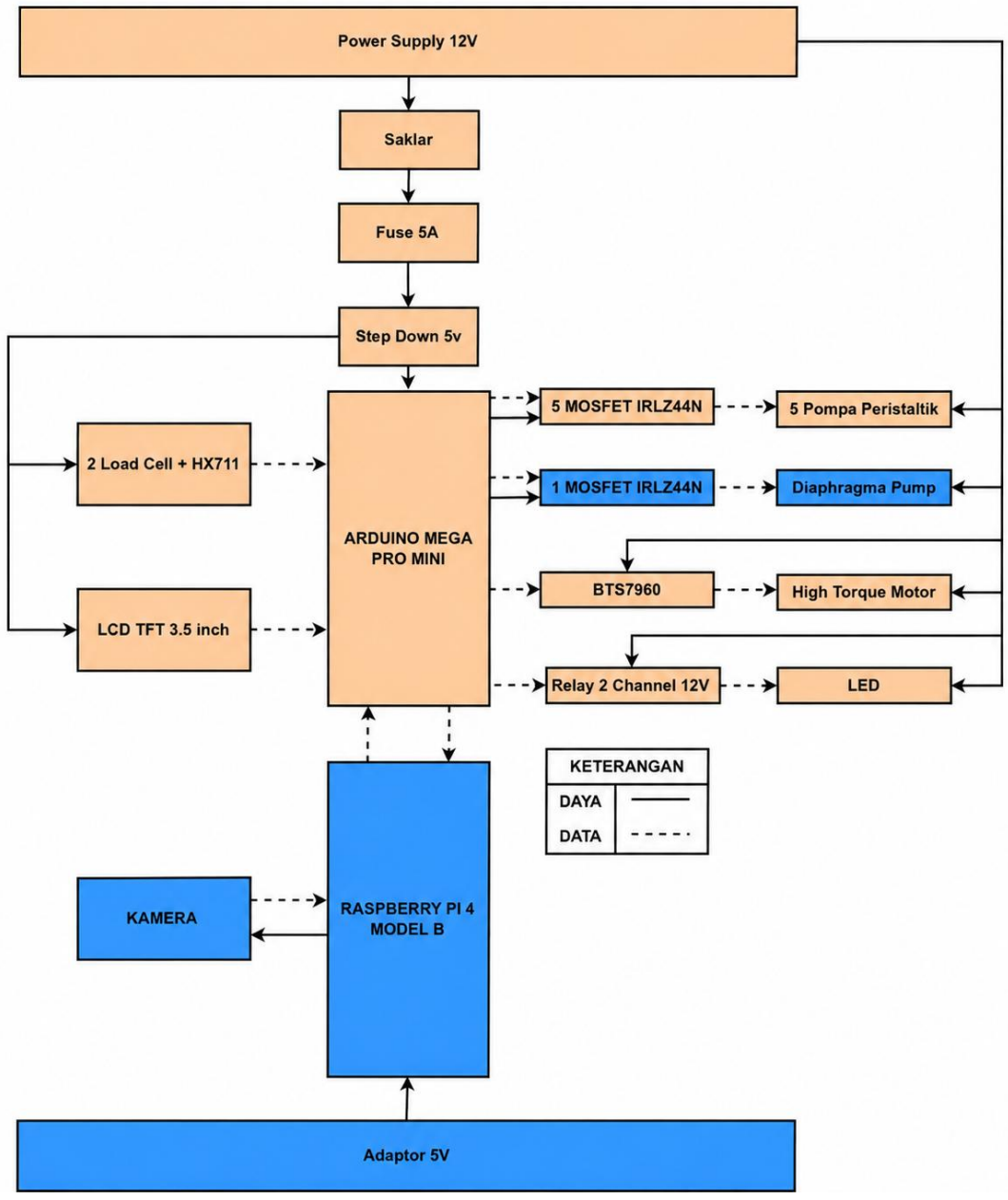
3.1.1 Blok Diagram

Blok diagram sistem merupakan sistem yang menggambarkan hubungan kerja antar komponen yang digunakan dalam proses koreksi warna cat secara otomatis yang telah dirancang. Sehingga diagram blok menjadi acuan dalam memahami cara kerja sistem sebelum dilakukan pembahasan lebih rinci pada setiap bagian.

Perancangan sistem koreksi warna cat otomatis ini merupakan sebuah integrasi teknologi *Computer Vision* dan sistem kendali tertutup yang bertujuan untuk menghasilkan campuran warna secara presisi tanpa intervensi manual. Inti dari sistem ini terletak pada pemrosesan citra digital menggunakan Raspberry Pi 4, di mana warna cat dideteksi dan dikonversi ke dalam ruang warna HSV untuk menjaga stabilitas pembacaan terhadap variasi intensitas cahaya. Selisih antara parameter warna target dengan hasil aktual dihitung menggunakan metode *Euclidean* Distance untuk menghasilkan nilai galat (ΔE). Nilai galat ini menjadi *input* utama bagi algoritma *Proportional* melalui persamaan matematis $u(\mathcal{K}) = Kp\ x\ \Delta E\ (\mathcal{K})$, yang bertugas menentukan *volume* pigmen tambahan yang diperlukan untuk mencapai target warna yang diingingkan secara bertahap atau iteratif.

Perancangan sistem koreksi warna cat otomatis ini merupakan sebuah integrasi teknologi *Computer Vision* dan sistem kendali tertutup yang bertujuan untuk menghasilkan campuran warna secara presisi tanpa intervensi manual. Inti dari sistem ini terletak pada pemrosesan citra digital menggunakan Raspberry Pi 4, di mana warna cat dideteksi dan dikonversi ke dalam ruang warna HSV untuk menjaga stabilitas pembacaan terhadap variasi intensitas cahaya. Selisih antara parameter warna target dengan hasil aktual dihitung menggunakan metode *Euclidean* Distance untuk menghasilkan nilai galat (ΔE). Nilai galat ini menjadi *input* utama bagi algoritma *Proportional* melalui persamaan matematis $u(\mathcal{K}) =$

$Kp\ x\ \Delta E\ (\mathcal{K})$, yang bertugas menentukan *volume* pigmen tambahan yang diperlukan untuk mencapai target warna yang diingingkan secara bertahap atau iteratif.



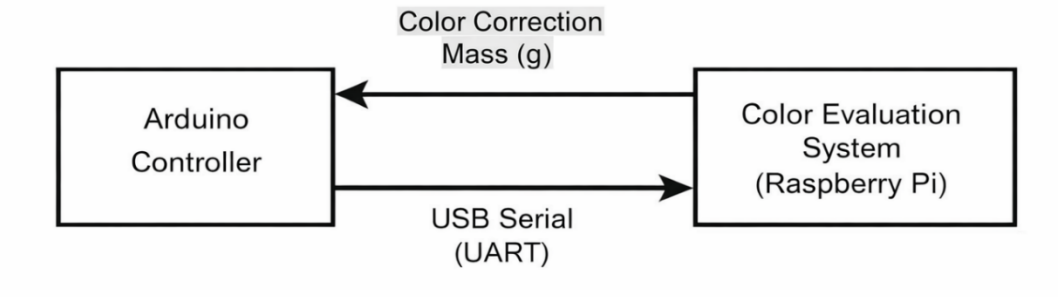
Gambar 3.1 Diagram Blok Sistem
(Dokumentasi Peneliti)

Perancangan sistem koreksi warna cat otomatis ini merupakan sebuah integrasi teknologi *Computer Vision* dan sistem kendali tertutup yang bertujuan

untuk menghasilkan campuran warna secara presisi tanpa intervensi manual. Inti dari sistem ini terletak pada pemrosesan citra digital menggunakan Raspberry Pi 4, di mana warna cat dideteksi dan dikonversi ke dalam ruang warna HSV untuk menjaga stabilitas pembacaan terhadap variasi intensitas cahaya. Selisih antara parameter warna target dengan hasil aktual dihitung menggunakan metode *Euclidean* Distance untuk menghasilkan nilai galat (ΔE). Nilai galat ini menjadi *input* utama bagi algoritma *Proportional* melalui persamaan matematis 2.5 yang bertugas menentukan *volume* pigmen tambahan yang diperlukan untuk mencapai target warna yang diingingkan secara bertahap atau iteratif.

Secara arsitektural, sistem ini menerapkan konsep *Cyber-Physical System* (CPS) dengan membagi tugas antara unit pemrosesan dan unit eksekusi. Raspberry Pi bertindak sebagai unit *Cyber* yang menangani algoritma visi dan kalkulasi kontrol, sementara Arduino Mega Pro Mini berfungsi sebagai unit *Physical* yang mengendalikan perangkat keras secara langsung melalui komunikasi serial. Sinergi ini memungkinkan kontrol aktuasi yang sangat presisi pada pompa diafragma melalui *driver* MOSFET IRLZ44N, dengan dukungan sensor *load cell* dan modul HX711 sebagai mekanisme umpan balik massa untuk memastikan berat pigmen yang dikeluarkan sesuai dengan perhitungan. Dengan demikian, sistem tidak hanya sekadar mendeteksi warna, tetapi secara aktif melakukan tindakan koreksi fungsional hingga konvergensi warna tercapai dalam batas toleransi yang ditentukan [43].

3.1.2 Diagram Blok Komunikasi



Gambar 3.2 Diagram Blok Komunikasi

(Dokumentasi Peneliti)

Sistem koreksi warna cat otomatis ini menerapkan arsitektur Master-Slave yang memisahkan lapisan komputasi visi (*Cyber Layer*) dengan lapisan eksekusi mekanik (*Physical Layer*). Dalam skema ini, Raspberry Pi 4 Model B bertindak sebagai unit Master yang memegang kendali penuh atas pengambilan keputusan. Proses dimulai dengan akuisisi citra melalui modul kamera, diikuti dengan konversi ruang warna ke format HSV untuk menjamin stabilitas deteksi terhadap variasi cahaya. Berdasarkan perhitungan jarak warna menggunakan metode *Euclidean* Distance, Raspberry Pi menjalankan algoritma *Iterative Proportional* untuk menentukan dosis koreksi berupa massa pigmen dalam satuan gram. Pembagian tugas ini bertujuan agar Raspberry Pi dapat fokus pada pemrosesan algoritma visi yang kompleks, sementara tugas real-time seperti pembacaan sensor massa dan kontrol aktuator diserahkan kepada Arduino Mega Pro Mini sebagai unit Slave.

Jembatan data antara kedua kontroler tersebut dikelola melalui protokol UART (*Universal Asynchronous Receiver-Transmitter*) yang dikoneksikan via media kabel USB. UART dipilih karena merupakan protokol komunikasi serial asinkron yang sangat stabil untuk pertukaran data *point-to-point* jarak pendek. Melalui jalur ini, Raspberry Pi mentransmisikan instruksi *"Color Correction Mass* (g)" ke Arduino, dan sebaliknya, Arduino mengirimkan balik sinyal status konfirmasi penuangan atau data berat aktual dari *load cell*. Penggunaan UART memastikan sinkronisasi data yang presisi antara instruksi digital dari Master

dengan aksi fisik yang dilakukan oleh Slave. Dengan integrasi ini, sistem mampu menjalankan siklus pengadukan, penimbangan, dan pengambilan citra ulang secara berulang *(Iterative)* hingga warna campuran cat mencapai titik konvergensi yang sesuai dengan target [44].

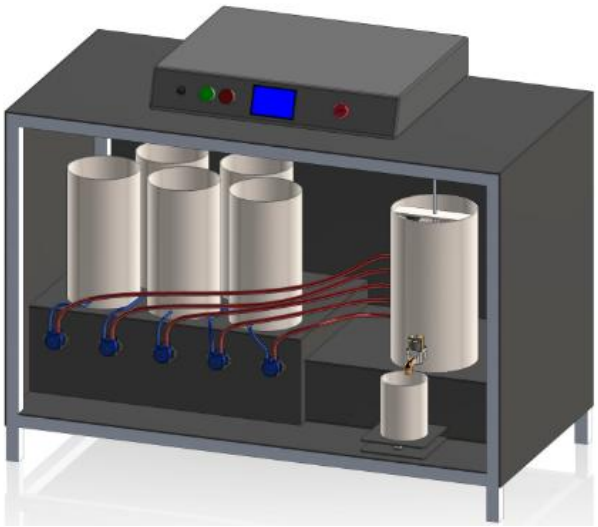
Sistem koreksi warna cat otomatis ini bekerja dengan mekanisme Nested Control Loop (lingkaran kendali bersarang) yang mengintegrasikan unit komputasi visi tingkat tinggi dengan unit eksekusi fisik tingkat rendah. Alur dimulai pada *Outer Visi Loop*, di mana Raspberry Pi berfungsi sebagai unit Master yang menerima input Target Warna (HSV) dan membandingkannya dengan warna aktual hasil tangkapan kamera. Selisih warna yang ditemukan dihitung menggunakan metode *Euclidean* Distance untuk menghasilkan nilai galat warna (ΔE). Berdasarkan galat tersebut, Raspberry Pi mengekseskusi algoritma *Proportional* melalui persamaan 2.5 untuk menentukan dosis koreksi berupa instruksi dosis massa pigman $u(\mathcal{K})$ dalam satuan gram. Pengguna Raspberry Pi sebagai *Master* sangat krusial agar proses pengolahan citra yang berat tidak terganggu oleh tugas aktuasi, sementara pengiriman data instruksi dilakukan secara stabil melalui protokol UART Serial via USB yang memiliki ketahanan tinggi terhadap gangguan elektromagnetik (*noise*) dari motor pengaduk.

Pada lapisan fisik atau *Inner Loop*, Arduino Mega Pro Mini bertindak sebagai unit Slave yang menerima instruksi massa dan mengubahnya menjadi sinyal kontrol (PWM/logika) untuk menggerakkan aktuator pompa peristaltik 12V melalui driver Mosfet IRLZ44N. Meskipun penuangan dikendalikan secara digital, sistem ini menyertakan Load Cell + HX711 sebagai sensor umpan balik massa yang sangat vital untuk menjamin akurasi. Penggunaan load cell memastikan bahwa penuangan pigmen didasarkan pada berat riil (gramasi), bukan sekadar durasi waktu pembukaan katup yang rentan berubah akibat perbedaan viskositas atau tekanan cairan. Setelah target berat tercapai dan penuangan selesai, arduino melaporkan balik status ke raspberry PI untuk memicu proses pengadukan hingga homogen. Kamera kemudian akan memotret ulang sebagai bentuk validasi visual, dan siklus iterasi ini akan terus berulang secara otomatis hingga nilai galat warna mencapai titik konvergensi di bawah batas toleransi yang telah ditetapkan [45].

3.2 Gambar 3D

3.2.1 Isometric

Visualisasi 3D isometrik merepresentasikan perancangan fisik sistem koreksi warna cat otomatis yang mengintegrasikan aspek mekanik, elektronik, dan fungsional dalam satu kesatuan unit. Struktur alat ini terbagi menjadi tiga area utama, dimulai dari deretan tangki penyimpanan pigmen warna dasar di sisi kiri yang terhubung dengan pompa peristaltik untuk distribusi cairan yang higienis dan presisi. Di sisi kanan, terdapat wadah pencampuran utama (*mixing tank*) yang dilengkapi dengan sistem pengaduk mekanik (*stirrer*) untuk menjamin homogenitas campuran cat, serta area penempatan sensor *load cell* di bagian bawah sebagai unit umpan balik massa riil. Seluruh komponen aktuator dan sensor tersebut diintegrasikan melalui panel kendali di bagian atas unit yang memuat LCD TFT 3.5 inci sebagai antarmuka pengguna, lampu indikator status operasional, serta kompartemen internal untuk menempatkan Raspberry Pi 4 dan Arduino Mega Pro Mini sebagai otak kendali sistem. Desain modular ini memastikan bahwa proses distribusi pigmen dari sisi (fisik) dapat berjalan selaras dengan instruksi perhitungan dari sisi *Master* (cyber) dalam satu kerangka rancang bangun yang ergonomis dan kokoh.



Gambar 3.3 Isometric

(Dokumentasi Peneliti)

Rancanngan perangkat keras pada sistem ini terdiri dari beberapa bagian utama yang bekerja secara terintegrasi. Pada bagian kiri dan Tengah unit, terdapat deretan tabung vertikal yang berfungsi sebagai tanki penyimpanan pigmen dasar. Untuk mengalirkan pigmen tersebut, pada bagian bawah setiap tangka dilengkapi dengan pompa peristaltic. Penggunaan pompa peristaltik yang dipadukan dengan selang fleksibel digunakan sebagai pencegahan terjadinya kontaminasi silang antar warna. Setiap pompa juga dikendalikan secara presisi oleh Arduino melalui modul MOSFET, dengan durasi dan pergerakannya merujuk pda instruksi massa yang dikirimkan oleh Raspberry Pi

Seetlah dipompa, pigmen warna akan dialirkan menuju bagian kanan unit, tempat wadah silinder besar yang berfungsi sebagai tangka pencampuran utama (*mixing tank)*. Dalam tahap ini, campuran cat harus dipastikan benar-benar homogen sebelum kamera melakukan proses pengambilan citra. Maka pada bagian atas wadah dilengkapi oleh sistem motor pengaduk (*stirrer)* dalam proses pencampuran. Akurasi takaran bahan juga dijaga melalui sensor *load cell* yang ditempatkan tepat di bawah wadah pencampuran. Sensor tersebut berfungsi sebagai umpan balik *(feedback)* data massa secara *real-time* kepada sistem kendali.

Seluruh rangkaian proses mekanis dan pembacaan sensor tersebut juga dipantau dan diatur melalui panel kendali yang diposisikan pada bagian atas unit. Panel ini mengintegrasikan seluruh komponen elektronik dan bertindak sebagai antarmuka pengguna. Pada bagian tengah panel, terdapat layar LCD TFT 3.5 Inch yang difungsikan untuk menampilkan status sistem, seperti progres penimbangan bahan dan status koneksi alat. Panel ini juga dilengkapi dengan lampu LED indikator berwarna merah dan hijau sebagai penanda operasional atau peringatan , beserta tombol fisik untuk kendali darurat (Emergency Stop) dan memulai proses. Di balik kompartemen panel tersebut, terdapat unit komputasi utama di mana Raspberry Pi 4 dan Arduino Mega Pro Mini diletakkan secara berdampingan. Kedua perangkat ini bekerja secara sinergis melakukan komunikasi data melalui jalur USB Serial (UART) untuk memastikan seluruh proses berjalan otomatis dan presisi.

3.2.2 Tampak Belakang

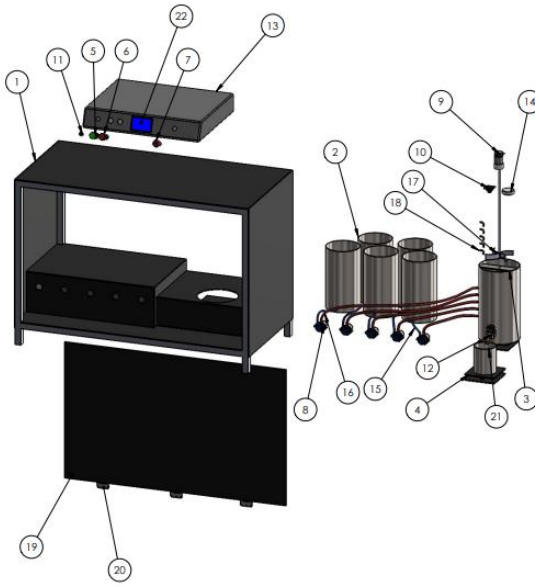
Gambar visualisasi tampak belakang prototipe sistem koreksi warna cat otomatis. Desain ini dirancang dengan mengutamakan prinsip aksesibilitas dan kemudahan pemeliharaan (*maintenance-friendly design*), mengingat sistem ini melibatkan penanganan cairan (cat pigmen).

Gambar 3.4 Tampak Belakang
Dokumentasi Peneliti

Tampak belakang unit ini didominasi oleh panel penutup utama yang berfungsi untuk melindungi mekanisme internal sistem dari debu dan potensi gangguan eksternal. Di bagian bawah panel penutup, terpasang tiga buah engsel pintu (Pintu Bawah) yang diposisikan di kiri, tengah, dan kanan. Engsel ini memungkinkan panel penutup untuk dibuka ke arah bawah atau ke atas (tergantung mekanisme penguncian), memberikan akses penuh ke kompartemen internal sistem. Akses langsung tersebut menjadi fasilitas dua kegiatan pemeliharaan utama. Desain pintu pada bagian belakang panel memungkinkan peengguna melakukan pengisian ulan pigmen pada tabung-tabung warna dasar dengan praktis. Juga bukaan panel tersebut memberi ruang pandang dan Gerak yang memadai bagi pengguna saat melakukan pemeliharaan aktuator.

3.2.3 Penamaan Komponen

Identifikasi komponen pada sistem pencampuran cat otomatis dilakukan melalui skema penomoran terstruktur yang merujuk pada table daftar gambar berikut:



Gambar 3.5 Penamaan Komponen

(Dokumentasi peneliti)

Tabel 3.1 Penamaan Komponen

ITEM NO	PART NUMBER	QTY
1	Rangka	1
2	Tabung Kecil	5
3	Tabung Besar	1
4	Load Cell	2
5	Pilot Lamp Hijau	1
6	Pilot Lamp Merah	1
7	*Emergency Stop Button*	1
8	*Adafruit Peristaltic Pump*	5

9	Motor Dc	1
10	*Camera Raspberry*	1
11	*Button On/OFF*	1
12	Electric Water Solenoid Valve 12 V DC	1
13	*Box Control*	1
14	Lampu	1
15	Selang In	1
16	Selang *Ouput*	1
17	As Pengaduk	1
18	Elbow	5
19	Pintu Belakang	1
20	Engsel	3
21	Kaleng Penampungan	1
22	LCD	1

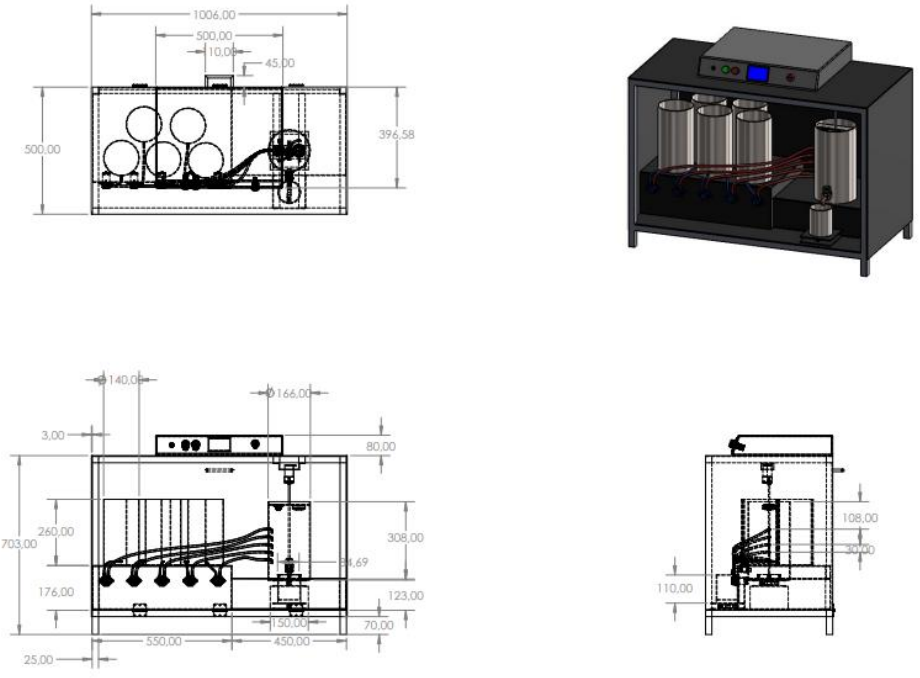
Proses penamaan ini mencakup elemen struktural seperti rangka alat, tangki penyimpanan pigmen, serta unit distribusi yang terdiri dari pompa peristaltik dan jaringan selang penghubung. Komponen fungsional utama untuk menjamin akurasi sistem meliputi *load cell* sebagai sensor massa, motor pengaduk (*stirrer*) untuk homogenitas campuran, dan katup solenoid untuk kontrol aliran. Seluruh integrasi perangkat keras tersebut dikendalikan oleh unit elektronik yang terpusat pada panel kontrol, mencakup Raspberry Pi dan Arduino sebagai otak sistem, serta antarmuka LCD, tombol operasional, dan lampu indikator. Pendekatan sistematis dalam penamaan ini bertujuan untuk mempermudah tahap perancangan, dokumentasi teknis, serta efisiensi dalam prosedur pemeliharaan perangkat di masa mendatang.

3.2.4 Ukuran Desain

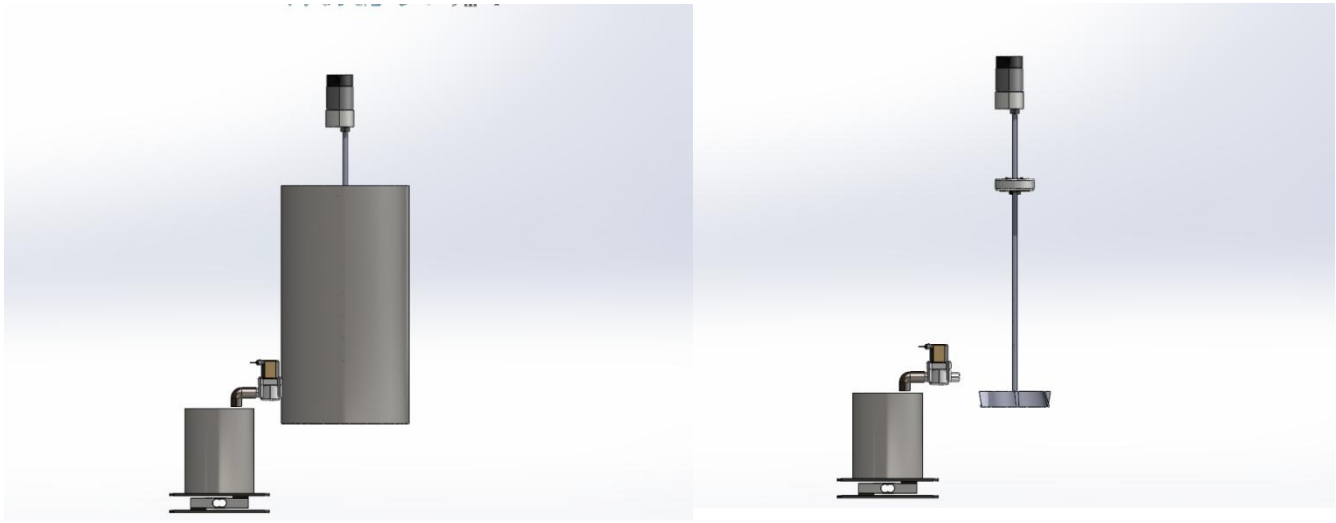
Dimensi perancangan digunakan sebagai petunjuk dari ukuran fisik sistem pencampuran cat otomatis yang dirancang. Melalui desain 3D ukuran Panjang,

lebar, dan tinggi dari setiap bagian alat dapat diketahui secara jelas sehingga memudahkan proses pembuatan rangka serta penempatan komponen pada sistem.

Pengukuran dan penentuan dimensi secara presisi menjadi faktor krusial dalam memastikan aspek ergonomi, stabilitas mekanis, serta kemudahan aksesibilitas saat pemeliharaan komponen. Melalui pendekatan desain 3D ini, tata letak (*layout*) antara ruang aktuasi mekanis—seperti dudukan motor *mixer* dan susunan pompa peristaltik—dengan modul pemrosesan optis (*camera box*) serta modul kendali elektrikal dapat diatur secara terstruktur. Hal ini bertujuan untuk meminimalkan potensi gangguan getaran mekanis terhadap pembacaan sensor, sekaligus mengoptimalkan volume ruang kerja agar seluruh rangkaian alat tetap kompak dan efisien. Rincian dimensi fisik serta proporsi dari masing-masing bagian sistem pencampuran cat otomatis ini ditunjukkan secara rinci pada gambar di bawah ini.



Gambar 3.6 Ukuran Desain
(Dokumentasi Peneliti)



Gambar 3.7 (a) Tampak Samping Tanpa Tabung Jarak Kamera Terhadap Mixing Tank, (b) Tampak Samping Dengan Tabung Jarak Kamera Terhadap Mixing Tank.

Rancangan fisik sistem ini memiliki dimensi keseluruhan dengan panjang 1006 mm, lebar 500 mm, dan tinggi rangka utama sebesar 703 mm. Penentuan dimensi tersebut didasarkan pada perhitungan kebutuhan volume ruang untuk mengintegrasikan seluruh komponen utama secara proporsional, termasuk lima tangki penyimpanan pigmen warna dasar berkapasitas masing-masing 2 kg serta satu unit *mixing tank* utama dengan kapasitas 5 kg. Dalam desain ini, komponen kamera diletakkan pada jarak 40 cm terhadap *mixing tank* sebagai parameter utama untuk pengambilan data visual secara *real-time.* Tata letak internal juga dirancang untuk mengakomodasi instalasi lima unit pompa peristaltik dan jaringan selang distribusi agar tetap rapi dan tidak saling tumpang tindih. Sementara itu, unit kontrol ditempatkan pada posisi ergonomis di bagian atas rangka guna mempermudah akses operasional serta pemantauan status sistem secara *real-time*.

Pemilihan dimensi yang spesifik ini bertujuan untuk menjamin stabilitas struktur mekanik saat proses pengadukan berlangsung, sekaligus memberikan ruang akses yang cukup bagi operator dalam melakukan prosedur perawatan maupun perapenulisn komponen. Dengan desain yang terukur ini, sistem tidak hanya menawarkan efisiensi ruang, tetapi juga memastikan keandalan operasional dalam jangka panjang.

3.3 Spesifikasi dan Fitur

3.3.1 Tabung Penyimpanan Warna

Kompartemen ini berfungsi sebagai wadah penampung pigmen warna dasar yang akan diolah dalam proses koreksi. Distribusi fluida dilakukan oleh pompa peristaltik yang terintegrasi dengan sistem kendali untuk menjamin akurasi volume penuangan sesuai formulasi. Pemilihan teknologi peristaltik didasarkan pada kemampuannya dalam menjaga sterilitas cairan serta meminimalisir risiko kontaminasi silang karena fluida hanya bersentuhan dengan dinding selang.

Spesifikasi utama penyimpanan cat:

Kapasitas Tabung	2 kg cat per tabung
Jumlah Tabung	5 tabung warna (Hitam, Putih, Merah, Kuning, Biru)
Material Tabung	Plastik
Aktuator Pengeluaran	Pompa peristaltic DC 12 V
Debit pompa	5.2 ~ 90ml/min Diameter selang : 3 × 5 mm

3.3.2 Tabung Pencampuran Cat

Mixing chamber dirancang sebagai wadah utama di mana seluruh pigmen disatukan hingga mencapai tingkat homogenitas yang diinginkan. Untuk menghasilkan adukan yang merata, sistem dilengkapi dengan motor pengaduk yang menciptakan aliran turbulensi di dalam tabung, sehingga konsistensi warna tetap terjaga di seluruh bagian cairan.

Spesifikasi utama

Kapasitas tabung	5 Kg
Material tabung	Stainless Steel

Motor pengaduk (Mixer)

Tipe Motor	DC Gear Motor 37GB31ZY
Tegangan Kerja	12 VDC
Kecepatan Putar	±30–60 RPM (gear motor)
Torsi	*high torque gear motor Driver motor : BTS7960 motor driver*

3.3.3 Sistem Pengukuran Massa

Akurasi komposisi cairan pada sistem ini dijamin melalui mekanisme pengukuran massa secara *real-time* menggunakan metode gravimetrik. Sensor *load cell* diimplementasikan untuk memberikan umpan balik (*feedback*) kontinu ke unit pemroses, yang secara otomatis akan menghentikan aktivitas penuangan saat target massa presisi telah terdeteksi.

Spesifikasi utama:

Sensor	Load Cell 5 kg
Metode pengukuran	*strain gauge*
Modul pembaca	HX711 24-bit ADC

3.3.4 Sistem Kontrol

Sistem kontrol pada alat ini mengadopsi arsitektur Master-Slave yang memisahkan beban komputasi visi tingkat tinggi dengan eksekusi perangkat keras secara *real-time*. Raspberry Pi 4 berperan sebagai unit *Master* yang bertugas melakukan akuisisi citra melalui kamera, mengonversi ruang warna ke format HSV, dan menghitung deviasi warna menggunakan algoritma *Euclidean* Distance. Berdasarkan hasil perhitungan tersebut, Raspberry Pi menjalankan algoritma *Iterative Proportional* untuk menentukan dosis koreksi massa pigmen yang kemudian ditransmisikan ke Arduino Mega Pro Mini sebagai unit melalui protokol UART Serial USB. Arduino selanjutnya mengaktifkan aktuator pompa peristaltik menggunakan modulasi PWM via MOSFET IRLZ44N, dengan pengawasan

kontinu dari sensor Load Cell melalui modul HX711 untuk menerapkan mekanisme Gravimetric Dosing. Proses ini ditutup dengan aktivasi motor pengaduk melalui *driver* BTS7960 untuk mencapai homogenitas warna, sebelum sistem kembali ke tahap evaluasi visual untuk memastikan nilai galat warna telah mencapai titik konvergensi yang diinginkan.

Spesifikasi utama:

Mikrokontroler	Arduino Mega Pro Mini
Pengendali pompa	MOSFET IRLZ44N
Driver motor mixer	BTS7960
Unit pemrosesan citra	Raspberry Pi 4
Sensor visual	Raspberry Pi Camera

3.4 Teknik Fabrikasi

3.4.1 Flowhcart Sistem Master – Slave

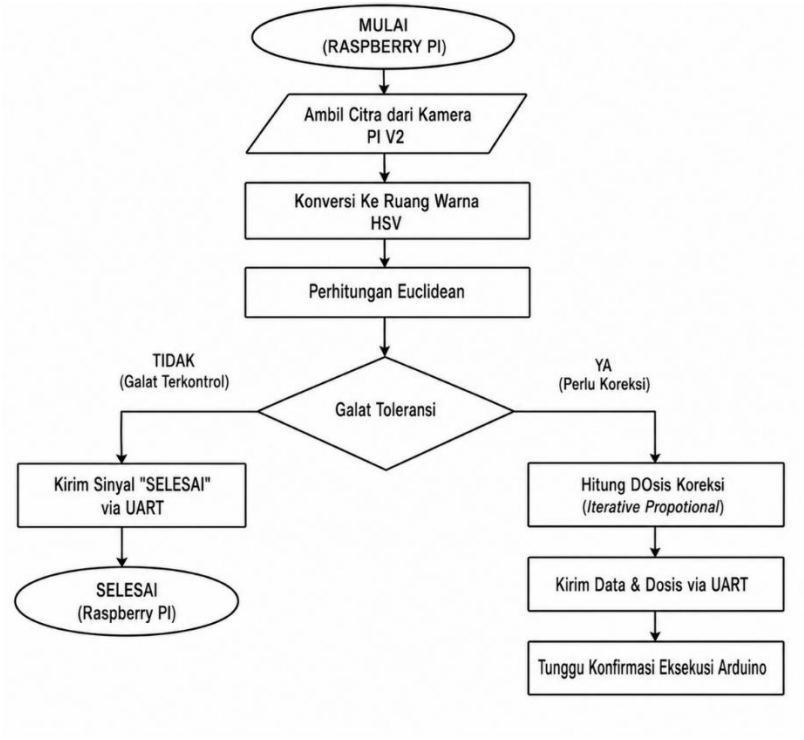
Dalam arsitektur ini, unit *Master* (Raspberry Pi 4) memegang kendali penuh atas pemrosesan citra digital, perhitungan galat warna menggunakan metode *Euclidean Distance*, serta penentuan nilai koreksi massa melalui algoritma *Proportional*. Sementara itu, mikrokontroler (Arduino) diposisikan sebagai unit yang bertugas menerjemahkan instruksi komputasi dari *Master* menjadi aksi perangkat keras, meliputi pembacaan umpan balik dari sensor *load cell* dan pengendalian aktuasi pompa peristaltik berbasis PWM. Pemisahan beban kerja ini dirancang untuk mencegah terjadinya *bottleneck* komputasi dan memastikan sistem kendali waktu nyata (*real-time*) tetap responsif.

Komunikasi data antar kedua unit ini dijembatani melalui protokol komunikasi serial (UART). Unit *Master* secara aktif mengirimkan *array* data berupa instruksi *setpoint* massa pigmen, sedangkan unit mengirimkan status aktuasi dan pembacaan sensor kembali ke *Master* sebagai umpan balik (*feedback*). Selain itu, arsitektur ini juga dilengkapi dengan mekanisme keamanan (*fail-safe*); sebagai contoh, apabila unit mendeteksi anomali pembacaan pada *load cell* saat

proses penakaran, mikrokontroler dapat secara mandiri menghentikan aktuasi pompa dan memicu sinyal untuk mencegah kegagalan sistem.

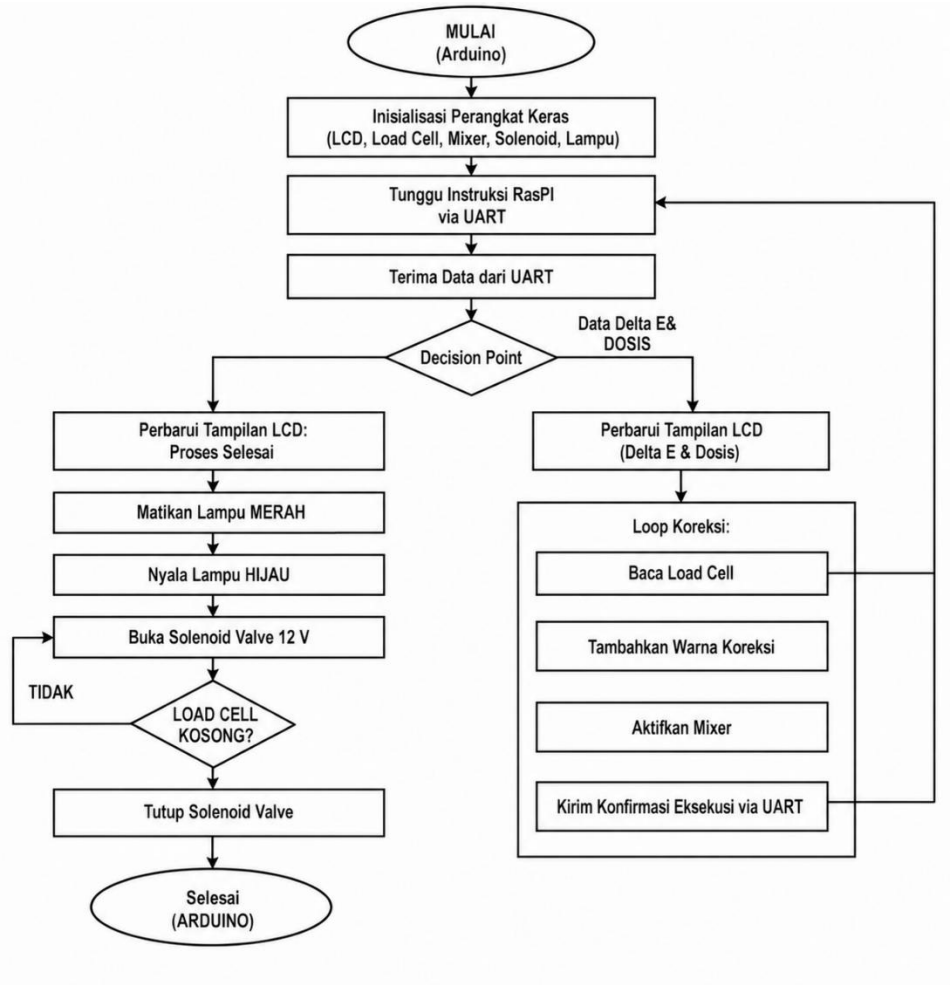
Integrasi antarnode ini tidak hanya memperjelas batas fungsi *hardware* dan *software*, namun juga mengoptimalkan efisiensi daya komputasi pada masing-masing unit pemroses. Dengan membebankan tugas-tugas berat seperti kalkulasi matriks piksel, analisis histogram HSV, dan manipulasi aljabar pada algoritma *Proportional* ke Raspberry Pi, *core* prosesor Arduino dapat beroperasi penuh pada tingkat penanganan waktu nyata (*real-time constraint*) tanpa terdistraksi oleh *overhead* grafis. Sebaliknya, keterisolasian tugas eksekusi mekanis pada unit memastikan bahwa pewaktuan sinyal PWM untuk pompa peristaltik dan pembacaan frekuensi *load cell* tetap berjalan secara konsisten tanpa mengalami fluktuasi *clock*.

Stabilitas arsitektur master-slave ini turut ditopang oleh standarisasi format paket data yang ditransmisikan melalui jalur serial UART. Penggunaan struktur *frame* data yang ringkas disertai mekanisme validasi (*checksum/parsing logic*) memungkinkan kedua perangkat untuk mempertahankan integritas informasi meskipun bekerja pada frekuensi sampling yang berbeda. Dengan demikian, sinkronisasi antara visualisasi *real-time* pada antarmuka *Master* dan respon aktuasi pada dapat terjalin secara berkelanjutan, membentuk sistem loop tertutup (*closed-loop system*) yang andal dalam menghadapi dinamika perubahan parameter warna fluida, serta gambar flowchart *Master* dan dapat dilihat pada gambar di bawah ini



Gambar 3.8 Flowchart *Master*

(Dokumentasi Peneliti)



Gambar 3.9 Flowchart *Slave*

(Dokumentasi Peneliti)

3.4.2 Alur Kerja Sistem (*Workflow)*

Proses awal kerja ssistem diawali pada tahap penginderaan, Rapsberry Pi sebagai sebagai *Master Controller* mengambil citra sampel cat menggunakan kamera, kemudian mengolah citra tersebut dengan mengonversi ruang warna RGB ke HSV. Nilai HSV hasil pembacaan selanjutnya dibandingkan dengan nilai HSV target menggunakan metode jarak *Euclidean* (ΔE) untuk menentukan besar galat warna (*color*). Nilai ΔE ini menjadi dasar dalam menentukan apakah warna campuran telah memenuhi target atau masih memerlukan proses koreksi.

Ketika nilai ΔE masih berada pada luar batas toleransi yang ditentukan, Raspberry Pi memasuki tahap untuk mengambil keputusan untuk menghitung nilai

besar massa pigmen koreksi menggunakan algoritma kendali proporsional. Nilai dosis yang dihasilkan kemudian dikirimkan ke Arduino Mega Pro Mini melalui komunikasi serial UART. Raspberry pi juga mengirimkan informasi nilai galat sebagai referensi proses koreksi. Setelah data diterima, Arduino sebagai *Controller* menampilkan status proses pada LCD dan mempersiapkan proses aktuator.

Pada tahap eksekusi fisik, Arduino mengendalikan pompa peristaltik untuk menambahkan pigmen sesuai dosis yang diterima dari Raspberry Pi. Selama penambahan dosis berlangsung, Load Cell yang terhubung dengan modul HX711 melakukan pengukuran massa secara kontinu sebagai umpan balik sehingga penambahan pigmen dapat dihentikan secar tepat ketika massa target telah tercapai. Setelah proses penakaran selesai, Arduino mengaktifkan motor pengaduk sehingga pigmen tercampur secara homogen, lalu mengirimkan sinyal konfirmasi kepada Raspberry Pi bahwa seluurh porses mekanis telah selesai dilaksanakan.

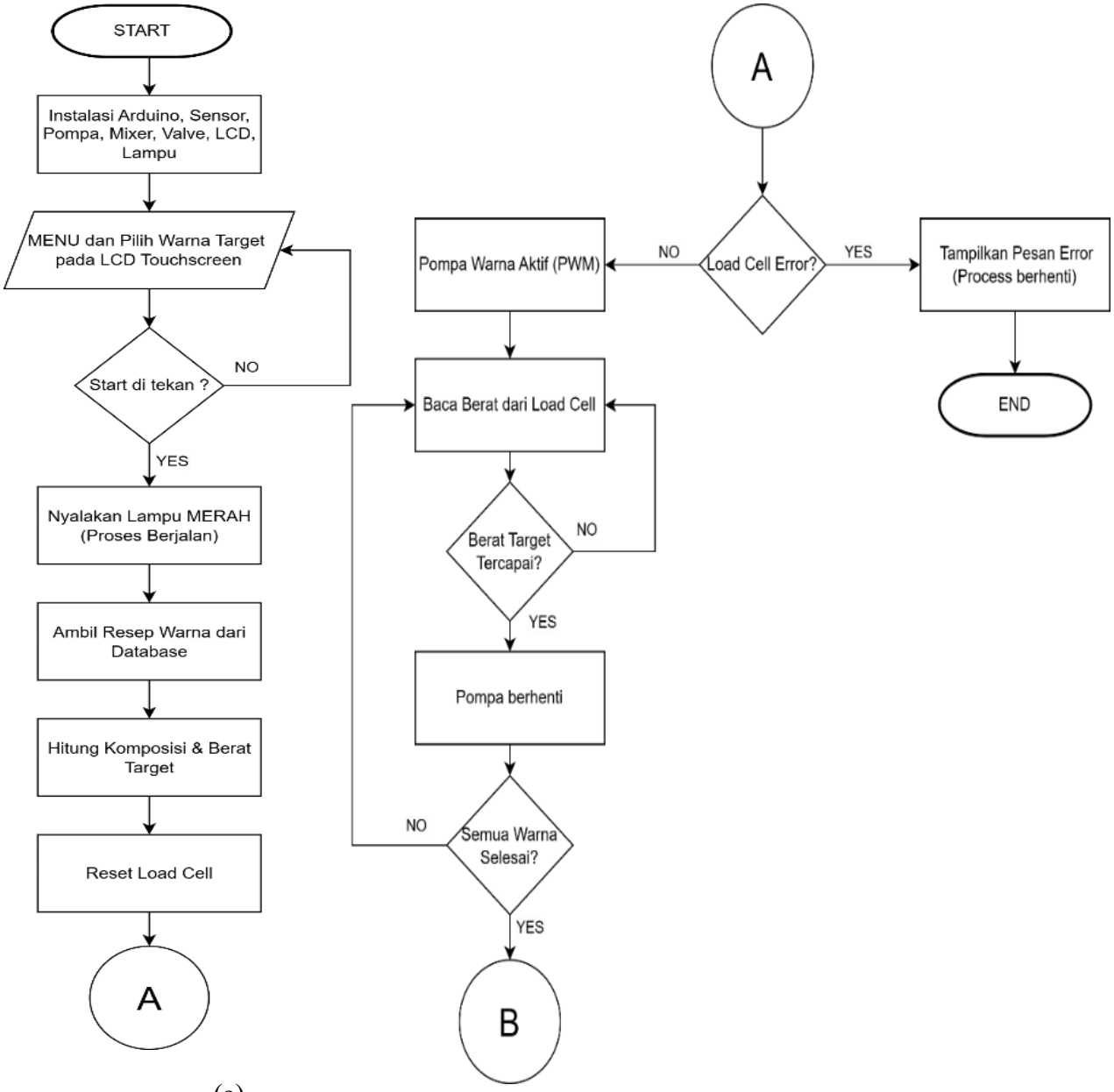
Setelah menerima sinyal konfirmasi tersebut, Raspberry melakukan proses akuisisi citra untuk mengevaluasi hasil pencampuran. Siklus tersebut berlangsung secara iterative hingga nilai galat warna (ΔE) berada di bawah ambang batas toleransi yang telah ditentukan. Ketika kondisi tersebut tercapai, Raspberry mengirimkan perintah “SELESAI” kepada Arduino. Sebagai respons, Arduino mengaktifkan lampu indikator hijau sebagai penanda bahwa proses koreksi telah berhasil diselesaikan, kemudian menyalakan pompa diafragma untuk mengeluarkan hasil campuran cat dari dalam tanki [46].

3.5 Metode

Operasional sistem pencampuran cat otomatis ini diawali dengan fase inisialisasi periferal pada mikrokontroler Arduino Mega Pro Mini sebagai unit dan Raspberry Pi 4 sebagai unit *master*, yang mencakup verifikasi konektivitas terhadap sensor Load Cell, modul HX711, *driver* motor BTS7960, serta sinkronisasi antarmuka LCD TFT Touchscreen. Setelah sistem dinyatakan siap, operator menentukan target warna melalui menu pada LCD yang secara simultan memicu akses ke basis data resep untuk mengekstraksi nilai *setpoint* massa masing-masing pigmen, diikuti dengan prosedur *zero-tracking* pada sensor massa guna mengompensasi berat statis wadah. Memasuki tahap pengisian yang dikelola

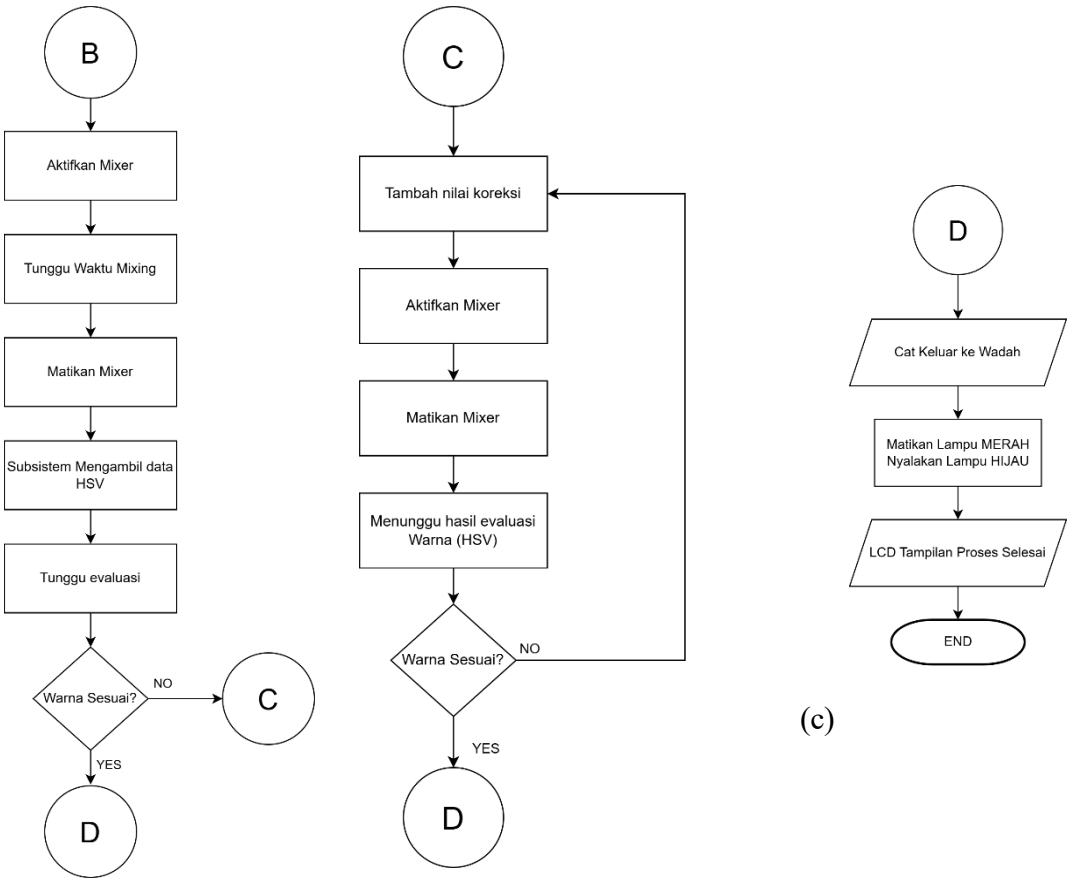
melalui mekanisme Sequential Control, pompa peristaltik diaktifkan secara berurutan dengan penerapan PWM (*Pulse Width Modulation*) Control untuk mengatur *duty cycle* motor secara dinamis, sehingga kecepatan aliran cairan berkurang secara bertahap saat mendekati target massa guna meminimalisir efek *Overshoot* dan menjamin akurasi Gravimetric Dosing. Setelah akumulasi massa tercapai, sistem mengaktifkan motor mixer untuk proses homogenisasi awal, yang kemudian disusul dengan pengiriman interupsi data melalui protokol komunikasi serial UART kepada Raspberry Pi 4 untuk memulai fase evaluasi visi. Pada unit *master*, Raspberry Pi Camera V2 melakukan akuisisi citra di bawah pencahayaan konstan modul LED, di mana data citra tersebut ditransformasikan ke ruang warna HSV (*Hue, Saturation, Value*) untuk mendapatkan nilai kromatisitas yang lebih stabil terhadap fluktuasi intensitas cahaya. Sistem koreksi bekerja dengan menghitung deviasi warna menggunakan metode *Euclidean* Distance; apabila nilai galat ΔE melampaui ambang batas toleransi, algoritma *Iterative Proportional* akan mengonversi selisih tersebut menjadi variabel dosis massa tambahan yang dikirim kembali ke Arduino untuk dieksekusi oleh pompa. Siklus iteratif yang meliputi penambahan pigmen, pengadukan ulang, dan evaluasi visi ini terus berlangsung secara *closed-loop* hingga tercapai titik konvergensi warna sesuai target, yang kemudian memicu instruksi terminasi kepada unit untuk mengubah indikator lampu menjadi hijau dan mengaktifkan pompa diafragma. Cairan cat yang telah terkoreksi secara sempurna dialirkan keluar melalui gaya gravitasi, sementara sistem terus memantau status beban pada *load cell* hingga tangki terdeteksi kosong, sehingga

katup ditutup kembali dan seluruh siklus operasional berakhir secara otomatis dengan pembaruan status pada layar LCD [47].



Gambar 3.11 (a) Persiapan Sistem (b) Dosing Berbasis Massa

(Dokumentasi Peneliti)



Gambar 3. 12 *(a) mixing* dan Evaluasi awal *(b)* Koreksi Warna *(c) Output* Akhir (Dokumentasi Peneliti)

Alur kerja sistem perncangan ini dirancang secara sistematis melalui lima tahapan utama yang saling berkesinambungan, yaitu inisialisasi, penakaran (*dosing),* pencampuran (*mixing)*, evaluasi berbasis *computer vision*, dan koreksi otomatis Proses diawali dengan tahap inisialisasi dan persiapan, di mana seluruh perangkat keras diaktifkan, meliputi mikrokontroler Arduino, sensor load cell, pompa peristaltik, motor mixer, pompa diafragma, serta antarmuka LCD touchscreen. Pengguna kemudian memilih warna target melalui menu pada LCD, dan sistem akan menunggu hingga tombol *start* ditekan. Begitu tombol ditekan, sistem secara otomatis menjalankan serangkaian instruksi awal, yaitu mengaktifkan lampu indikator merah sebagai penanda bahwa proses sedang berjalan, mengambil data resep warna berupa nilai setpoint HSV dari basis data, menghitung komposisi massa masing-masing pigmen yang dibutuhkan, serta melakukan kalkulasi berat target

sekaligus fungsi *tare (reset)* pada *load cell* agar pembacaan massa pigmen dapat dimulai dari titik nol.

Setelah tahap persiapan selesai, sistem memasuki tahap gravimetric dosing, yaitu fase penakaran pigmen warna dasar. Pada tahap ini, sistem terlebih dahulu melakukan validasi terhadap kondisi sensor load cell; apabila terdeteksi adanya malfungsi, sistem akan menghentikan proses dan menampilkan pesan peringatan. Namun, jika kondisi sensor dinyatakan normal, pompa warna akan diaktifkan menggunakan kontrol *PWM (Pulse Width Modulation*). Sistem kemudian bekerja dalam skema loop tertutup *(closed-loop*), di mana massa aktual pigmen dibaca secara real-time, dan pompa akan terus beroperasi hingga massa yang terukur mencapai berat target yang telah ditentukan. Proses penakaran ini dilakukan secara sekuensial untuk setiap pigmen warna dalam resep sebelum sistem melanjutkan ke tahap berikutnya.

Setelah seluruh pigmen terkumpul di dalam wadah pencampur, sistem memasuki tahap homogenisasi dan evaluasi visual. Motor mixer diaktifkan untuk mengaduk dan menghomogenkan campuran cat sesuai dengan durasi waktu pencampuran *(mixing time)* yang telah ditentukan yaitu selama 50 detik. Begitu proses homogenisasi selesai, kendali beralih ke subsistem *computer vision*, yang bekerja dengan menangkap citra permukaan cat menggunakan kamera, kemudian mengekstraksi data HSV aktual dari citra tersebut. Selanjutnya, sistem melakukan komputasi *Euclidean Distance* untuk membandingkan warna aktual dengan warna target. Apabila nilai jarak tersebut sudah berada di bawah ambang batas *(threshold)* yang ditetapkan, sistem akan langsung melanjutkan ke tahap akhir. Sebaliknya, jika hasil evaluasi menunjukkan bahwa warna belum sesuai, sistem akan masuk ke tahap koreksi.

Pada tahap koreksi, sistem menerapkan metode *Proportional*, di mana nilai galat (error) yang diperoleh dari hasil evaluasi digunakan sebagai input untuk menghitung massa pigmen tambahan yang diperlukan guna mendekatkan warna aktual terhadap warna target. Setelah pigmen tambahan tersebut dimasukkan, motor mixer kembali diaktifkan untuk melakukan homogenisasi ulang. Tahap ini bersifat iteratifsecara berkesinambungan hingga diperoleh nilai warna yang konvergen

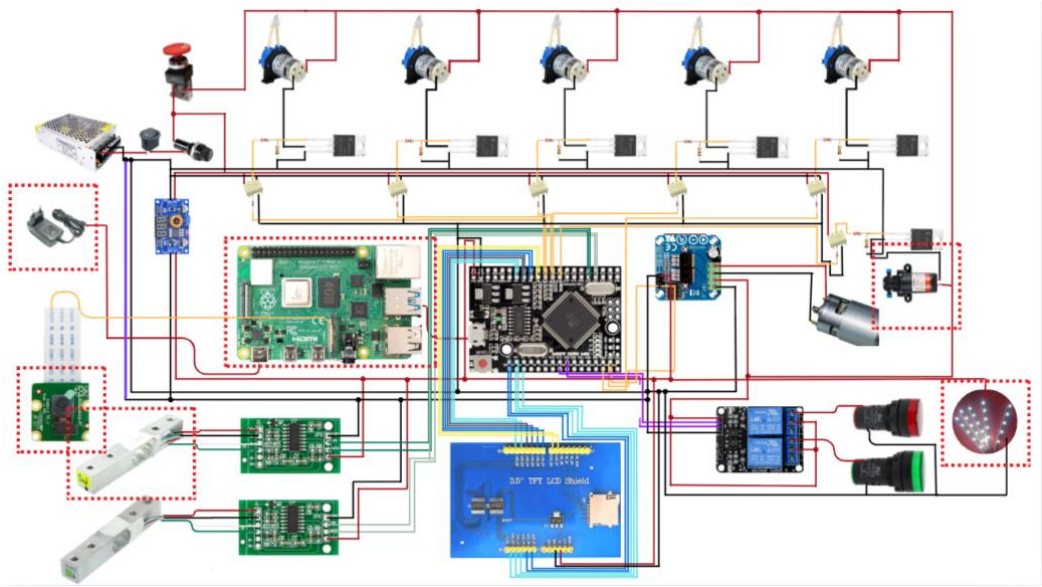
terhadap target, dengan tujuan menjaga akurasi warna yang presisi tanpa menimbulkan risiko *overshoot*.

Etika hasil evaluasi akhirnya menyatakan bahwa warna telah sesuai dengan target, sistem memasuki tahap output akhir. Cat yang telah tervalidasi kemudian dikeluarkan melalui pompa diafragma menuju wadah penampung akhir. Sebagai penanda berakhirnya proses, lampu indikator merah dipadamkan dan digantikan dengan menyalanya lampu indikator hijau, sementara layar LCD menampilkan informasi "Proses Selesai". Dengan demikian, seluruh rangkaian sistem mencapai kondisi *END*, yang menandakan bahwa proses perancangan dan koreksi warna cat secara otomatis telah selesai dilakukan.

3..6 Perancangan Elektrikal

3.6.1 Wiring Komponen Elektrikal

Pada bagian subbab ini menjelaskan terkait dengan perancangan wiring komponen elektrikal yang digunakan dalam sistem. Wiring diagram digunakan sebagai penghubung antar komponen elektronik sehingga seluurh bagian sistem dapat saling terintegrasi dan bekerja sesuai dengan fungsinya. Pada tahap ini dilakukan pengaturan jalur koneksi antara mikrokontroler, sensor, aktuator, serta sumber daya listrik yang digunakan.



Gambar 3.13 Elektrikal Wiring Sistem

(Dokumentasi Peneliti)

Tahap pemrosesan pada sistem ini dimulai setelah seluruh pigmen warna dasar berhasil dimasukkan ke dalam *mixing tank*, di mana Arduino Mega Pro Mini melakukan validasi massa menggunakan sensor Load Cell dan modul HX711 untuk memastikan berat aktual telah sesuai dengan instruksi dosis dari Raspberry Pi 4. Setelah massa terverifikasi secara presisi melalui mekanisme *gravimetric dosing*, sistem mengaktifkan Motor DC Gearbox 37GB31ZY melalui *driver* BTS7960 dengan kendali PWM (*Pulse Width Modulation*) untuk memulai proses pengadukan. Selama fase *mixing* ini, Raspberry Pi Camera V2 yang didukung oleh pencahayaan konstan dari modul Lampu LED melakukan evaluasi visual secara kontinu untuk memantau homogenitas campuran berdasarkan nilai kromatisitas warna. Jika target warna telah tercapai dan campuran dinyatakan homogen, Raspberry Pi mengirimkan sinyal akhir ke Arduino untuk menyalakan pompa diafragma 12V guna mengalirkan cat hasil koreksi ke wadah penampungan akhir. Seluruh urutan kerja ini, mulai dari pemantauan berat *real-time*, durasi pengadukan, hingga status pembukaan katup solenoid, diintegrasikan secara otomatis dan ditampilkan pada antarmuka LCD TFT 3.5 Inch untuk memastikan siklus produksi berjalan konsisten dan akurat [48]

3.6.2 Perancangan Kerangka Alat

Perancangan kerangka alat difokuskan pada penyediaan struktur yang stabil untuk mendukung akurasi evaluasi warna dan proses pengadukan (*mixing*). Rangka utama dirancang menggunakan profil logam yang kokoh dengan dimensi 1006 x 500 x 703 mm, yang dibagi menjadi area basah di bagian bawah dan panel kontrol kedap cairan di bagian atas. Fokus utama pada bagian ini adalah integrasi dudukan Load Cell yang diletakkan pada titik berat terpusat di bawah *mixing tank*. Struktur dudukan sensor ini dibuat independen dari getaran rangka utama agar pembacaan massa saat proses *dosing* dan *mixing* tetap presisi. Selain itu, bagian atas rangka dirancang memiliki kompartemen khusus untuk menempatkan Raspberry Pi Camera V2 dan modul Lampu LED dengan sudut pengambilan citra yang tegak lurus terhadap permukaan cat, guna menjamin konsistensi data visual yang diperoleh.

3.6.3 Perancangan Komponen Pada Alat

Setelah struktur utama terbentuk, tahap berikutnya adalah pemasangan komponen pengolah dan aktuator keluaran. Mixing Tank berkapasitas 5 kg diposisikan tepat di atas sensor *load cell* dengan sambungan yang fleksibel agar tidak menghambat pengukuran berat. Motor pengaduk DC Gearbox 37GB31ZY dipasang pada *bracket* atas dengan as agitator yang masuk ke dalam tabung, di mana dudukan motor dilengkapi dengan pe*red*am getaran untuk meminimalisir gangguan pada sensor massa saat motor beroperasi. Pada bagian dasar tabung pencampuran, dipasang pompa diafragma sebagai unit *outlet* otomatis. Pemasangan katup ini dilakukan dengan kemiringan tertentu agar cairan cat dapat keluar secara maksimal melalui gaya gravitasi. Unit elektronik seperti Arduino Mega Pro Mini, Raspberry Pi 4, dan *driver* BTS7960 ditempatkan di dalam *box control* atas untuk melindunginya dari korosi atau kebocoran cat, dengan penataan kabel yang memisahkan jalur daya tegangan tinggi (untuk motor dan pompa diafragma) dengan jalur sinyal sensorik [49].

3.6.4 Perancangan Perangkat Lunak

Perangkat lunak pada sistem ini dirancang dengan arsitektur Master-Slave untuk mengoordinasikan proses validasi massa, pengadukan, hingga evaluasi visi. Program pada Arduino Mega Pro Mini () bertanggung jawab memantau berat aktual dari *load cell* secara *real-time*. Setelah berat dinyatakan stabil dan sesuai target, Arduino mengeksekusi proses *mixing* dengan mengatur kecepatan motor melalui sinyal PWM pada *driver* BTS7960. Selama proses ini, Arduino berkomunikasi secara serial dengan Raspberry Pi 4 (*Master*). Program pada Raspberry Pi bertugas mengaktifkan Lampu LED dan kamera untuk mengambil data citra cat [50].

Citra tersebut diproses menggunakan algoritma analisis warna HSV dan *Euclidean* Distance untuk menentukan apakah campuran sudah homogen dan sesuai target warna. Jika evaluasi visual menyatakan warna sudah masuk dalam toleransi galat ($\triangle E$), Raspberry Pi mengirimkan perintah akhir kepada Arduino. Arduino kemudian merespons dengan menghentikan motor pengaduk dan membuka pompa diafragma selama durasi tertentu hingga tangki kosong kembali. Seluruh alur logika ini, mulai dari grafik kenaikan massa hingga status finalisasi

warna, ditampilkan secara interaktif pada LCD TFT 3.5 Inch untuk memberikan informasi yang transparan kepada operator mengenai tahapan produksi yang sedang berjalan [51].

3.7 Sistem Komunikasi

Dalam implementasi perangkat keras, Arduino Mega 2560 Pro Mini dikonfiguras sebagai *Controller* yang bertugas menerima instruksi dari Raspberry Pi 4 sebagai *Master Controller* melalui komunikasi serial UART. Komunikasi ini memanfaatkan jalur serial 0, yaitu pin D0 (RX) sebagai penerima data dan D1 (TX) sebagai pengirim data. Penggunaaan jalur serial 0 dipilih karena terhubung langsung dengan antarmuka komunikasi serial utama mikrokontroler, sehingga mampu memberikan kecepatan respons yang tinggi dan komunikasi yang stabil dalam menerima instruksi dosis massa pigmen yang dikirimkan oleh Raspberry Pi.

Data yang diterima melalui pin D0 (RX) diproses menggunakan mekanisme pembacaan serial (*polling*) dengan format data yang telah ditentukan. Raspberry Pi mengirimkan parameter dosis koreksi dalam bentuk string terstruktur yang kemudian diuraikan (*parsing*) oleh Arduino untuk menentukan jenis aktuator yang akan diaktifkan. Berdasarkan hasil parsing tersebut, Arduino mengendalikan pompa peristaltik melalui rangkaian MOSFET yang terhubung pada pin D10–D13 dan D44, serta mengendalikan motor pengaduk melalui modul driver BTS7960 yang menggunakan pin D45 dan D46 sebagai sinyal kendali PWM. Dengan mekanisme ini, setiap instruksi yang diterima dari Master dapat dieksekusi secara akurat sesuai kebutuhan proses koreksi warna.

Disisi lain, Arduino juga melakukan prosisi akuisisi data massa secara bersamaan melalui sensor Load Celll yang dihubungkan dengan modul HX711 pada pin D22-25. Data massa yang diperoleh digunakan sebagai umpan balik untuk memastikan bahwa jumlah pgimen yang ditambhakan telah sesuai dengan dosis yang diperintahkan oleh Raspberry Pi. Setelah massa target tercapai dan proses penakaran selesai, Arduino mengirimkan sinyal konfirmasi melalui komunikasi UART kepada Raspberry Pi sebagai penanda bahwa proses mekanis telah berhasil diselesaikan. Selanjutnya, Raspberry Pi melakukan akuisisi citra kembali untuk mengevaluasi hasil pencampuran warna. Dengan demikian, komunikasi UART

tidak hanya berfungsi sebagai media pengiriman instruksi dari Master ke Slave, tetapi juga sebagai jalur komunikasi dua arah yang mengintegrasikan proses komputasi, kendali aktuator, dan umpan balik sensor sehingga keseluruhan sistem kendali tertutup dapat bekerja secara sinkron dan berkesinambungan.

3.7.1 Algoritma Pemrosesan Citra Digital (VS Code Python)

Perancangan perangkat lunak pada unit *Master* menggunakan bahasa pemrograman python yang dikembangkan melalui VS Code. Perangkat lunak tersebut bertugas untuk melakukan akuisis citra, pemrosesan warna, analisis galat, hingga menentukan dosis pigmen koreksi yang akan dikirimkan kepada unit .Seluruh tahapan proses citra dirancang sehingga menghasilkan representasi warna yagn objektif dan konsisten sehingga mampu mendukung proses kendali terutup pada sistem pencampuran cat otomatis.

Tahapan pertama adalah penentuan *Region of Interest (ROI),* yaitu proses pembatasan area pengambilan sampel citra pada bagian tengah wadah pencampuran. Tujuan penggunaan ROI adalah menghilangkan objek-objek yang tidak berkaitan dengan warna cat, seperti dinding tangki, batang pengaduk, maupun komponen mekanis lainnya yang dapat memengaruhi hasil pembacaan warna. Dengan hanya mengambil area yang berisi permukaan cat, nilai warna yang diperoleh menjadi lebih representatif terhadap kondisi aktual campuran sehingga meningkatkan akurasi analisis warna [52].

Setelah area ROI ditentukan, citra diproses menggunakan *Gaussian Blur* sebagai filter spasial untuk me*red*uksi *noise* berfrekuensi tinggi yang muncul akibat pantulan cahaya lampu LED maupun tekstur permukaan cat. Filter Gaussian bekerja dengan menghaluskan distribusi intensitas piksel sehingga variasi lokal yang disebabkan oleh pantulan cahaya dapat diminimalkan. Hasilnya, tekstur warna menjadi lebih homogen dan nilai warna yang diperoleh lebih stabil sebagai masukan bagi algoritma pengolahan citra [53].

Tahapan berikutnya adalah ekstraksi nilai HSV, yaitu mengonversi citra dari ruang warna RGB ke ruang warna HSV (*Hue, Saturation, Value*). Setelah proses konversi selesai, sistem menghitung nilai rata-rata (*mean*) pada masing-masing kanal H, S, dan V untuk menghasilkan satu representasi numerik warna yang stabil.

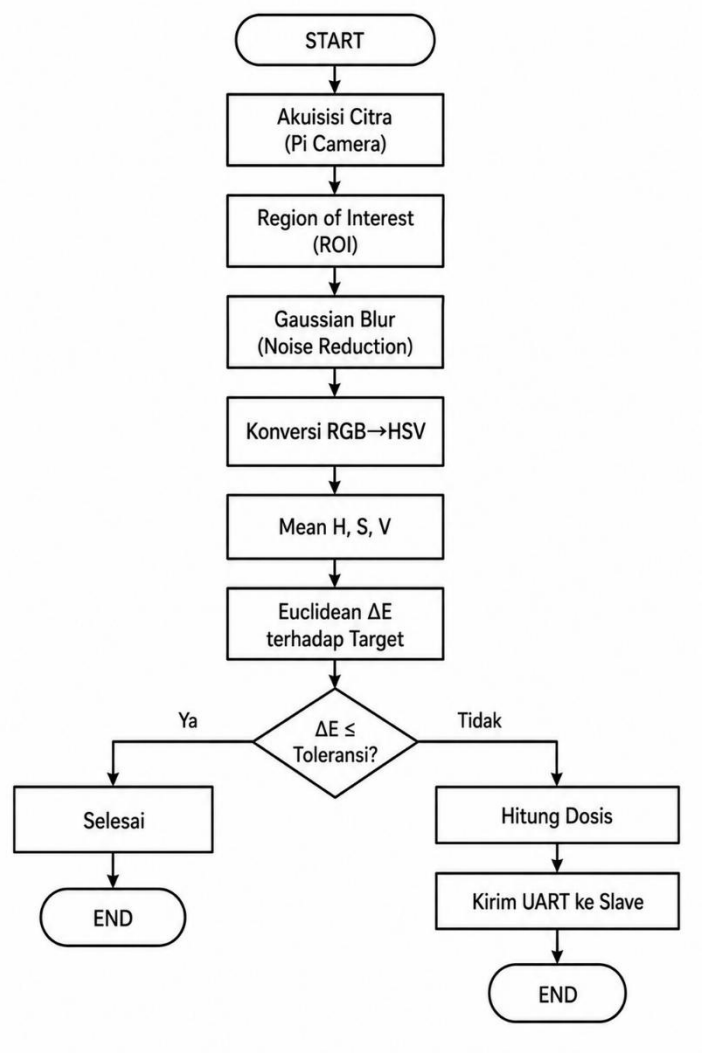
Nilai rata-rata tersebut kemudian digunakan sebagai masukan dalam proses perhitungan jarak *Euclidean* terhadap warna target sehingga dapat ditentukan besar galat warna yang harus dikoreksi.

Dalam implementasi *computer vision*, parameter pada setiap tahapan pemrosesan citra bersifat dinamis sehingga perubahan nilainya dapat memengaruhi akurasi pembacaan warna. Salah satu parameter yang paling berpengaruh adalah ukuran *Region of Interest (ROI).* Apabila ROI dibuat terlalu besar, area pengambilan citra akan mencakup objek lain seperti dinding tangki, bayangan batang pengaduk, maupun refleksi cahaya sehingga nilai rata-rata HSV tidak lagi merepresentasikan warna cat secara murni. Kondisi tersebut menyebabkan nilai jarak *Euclidean* (ΔE) menjadi kurang akurat. Sebaliknya, apabila ROI dibuat terlalu kecil, jumlah piksel yang digunakan sebagai sampel menjadi sangat sedikit sehingga pembacaan warna menjadi sensitif terhadap gangguan lokal, misalnya gelembung udara atau riak pada permukaan cat. Akibatnya, nilai HSV yang diperoleh akan berfluktuasi dan menyebabkan proses koreksi warna menjadi kurang stabil [54].

Parameter lain yang dapat mempengaruhi performa sistem Adalah ukuran kernel *Gaussian Blur.* Kernel merupakan matriks konvulusi berukuran ganjil dan ketika penggunaan kernel berukuran kecil sering kali belum mampu menghilangkan gangguan cahaya secara optimal sehingga *Value* masih menunjukkan nilai yang sangat tinggi pada titik tertentu, serta distribusi intensitas cahaya tidak merata, sehingga ukuran kernel dapat terlebih dahulu untuk diubah menjadi lebih besar sehingga nilai HSV yang dihasilkan juga lebih stabil dan mendekati kondisi warna sebenenarnya [55].

Disamping ukuran ROI dan kernel *Gaussian,* stabilitas nilai pada setiap kanal HSV juga menjadi factor penting dalam sistem. Perubahan intensitas pencahayaan memberikan faktor terbesar pada kanal *value*, sedangkan kanal Hue tetap konstan selama warna cat tidak mengalami perubahan. Oleh karena itu, stabilitas kalan Hue menjadi indikator utama keberhasilan sistem pengolahan citra. Apabila dalam pengujian ditemukan nilai Hue yang mengalami fluktuasi, meskipun warna cat tetap, maka diperlukan evaluasi terhadap kondisi pencahayaan, seperti mengurangi

pengaruh cahaya luar atau menambahkan *enclosure* pada tangki pencampuran agar lingkungan pengambilan citra menjadi lebih terkendali. Dengan demikian, sistem mampu menghasilkan data warna yang konsisten dan meningkatkan akurasi proses koreksi warna secara keseluruhan [56].



Gambar 3.14 Alur Pengolahan Citra

3.7.2 Perancangan Metode *Euclidean Distance*

Untuk menjalankan fungsi perhitungan ini, sistem membutuhkan dua set data koordinat spasial warna sebagai nilai masukan (*input*). Data pertama adalah Nilai Target (H1, S1, V1), yang merupakan nilai konstanta dari basis data resep standar warna yang ingin dicapai. Data kedua adalah Nilai Aktual (H2, S2, V2), yang

diperoleh secara dinamis dari hasil pembacaan sensor visual. Nilai aktual ini merupakan hasil akhir dari tahapan pemrosesan citra, di mana sistem mengambil nilai rata-rata (mean) dari matriks *Region of Interest* (ROI) setelah citra melalui proses *Gaussian Blur* dan diekstraksi ke dalam ruang warna HSV [57].

Di dalam skrip program, kalkulasi jarak absolut antara kedua titik warna tersebut dieksekusi memanfaatkan *library* komputasi matematis bawaan dengan menghitung akar kuadrat dari total selisih kuadrat masing-masing komponen HSV. Hasil keluaran dari fungsi perhitungan ini akan menghasilkan variabel numerik baru yang merepresentasikan besaran deviasi atau persentase galat warna, hal tersebut dapat dilihat pada skrip di bawah ini

```
import math

def hitung_euclidean_hsv(hsv_target, hsv_aktual):
    # Ekstraksi komponen nilai HSV
    H1, S1, V1 = hsv_target
    H2, S2, V2 = hsv_aktual

    # Implementasi persamaan Euclidean Distance
    delta_e = math.sqrt((H1 - H2)**2 + (S1 - S2)**2 + (V1 - V2)**2)

    # Kalkulasi persentase error (opsional, disesuaikan dengan programmu)
    error_persen = (delta_e / 255.0) * 100

    return delta_e, error_persen
```

Variabel galat tersebut kemudiann diolah lebih lanjut oleh sistem sebagai parameter pengambil keputusan menggunakan logika percabangan kondisional. Raspberry akan membandingan nilai galat yang dihasilkan terhadap batas ambang toleransi yang telah dikonfigurasi sebelumnya. pabila nilai galat dievaluasi melebihi batas toleransi, program secara otomatis mengidentifikasi bahwa warna belum konvergen dan akan memanggil fungsi algoritma *Iterative Proportional*

untuk mengkalkulasi kebutuhan dosis pigmen tambahan. Sebaliknya, jika nilai galat perhitungan berada di bawah atau sama dengan batas toleransi, status warna dinyatakan terkontrol, dan unit pemroses pusat akan langsung mentransmisikan sinyal penyelesaian proses via komunikasi serial UART kepada mikrokontroler aktuator [58].

3.8 Permisalan dan Simulasi Analisa Data

Sebagai ilustrasi pengujian, diasumsikan sistem mengambil 5 sampel data nilai kanal *Hue* (dengan rentang array 0-179 pada OpenCV) pada sampel warna biru, dan didapatkan sekumpulan data berturut-turut: 100, 102, 101, 100, 102. Berdasarkan perhitungan menggunakan persamaan standar deviasi, diperoleh nilai simpangan baku *(standard deviation)* sebesar 1,00. Nilai tersebut menunjukkan bahwa penyebaran data terhadap nilai rata-ratanya sangat kecil, sehingga variasi antarhasil pengukuran dapat dikatakan rendah. Maka setiap proses akuisisi citra menghasilkan nilai Hue yang hamper sama meskipun dilakukan secara berulang.

Jika dibandingkan dengan rentang maksimum kanal Hue pada OpenCV yang bernilai 179, simpangan baku sebesar 1,00 hanya merepresentasikan fluktuasi sepenulisr 0,56% dari keseluruhan rentang pengukuran. Besarnya simpangan yang relatif kecil tersebut menunjukkan bahwa sistem memiliki tingkat presisi yang tinggi serta mampu mempertahankan konsistensi pembacaan warna terhadap gangguan (*noise*) visual maupun variasi kecil pada permukaan sampel. Kondisi ini mengindikasikan bahwa tahapan pemrosesan citra, seperti penentuan Region of Interest (ROI) dan penerapan Gaussian Blur, telah bekerja secara efektif dalam menghasilkan data warna yang stabil [59].

Sebaliknya, ketika hasil pengujian menunjukkan nilai standar deviasi yang jauh lebih besar, maka hal tersebut mengindikasikan bahwa pembacaan warna masih mengalami fluktuasi yang tinggi. Kondisi tersebut disebabkan oleh beberapa factor, seperti ukuran kernel *Gaussian Blur* yang belum optimal, perubahan intensitas pencahayan, pantulan chaaya pada permukaan cat, atau ketidakstabilan posisi kamera selama proses akuisisi citra. Sehingga, diperlukan proses kalibrasi ulang terhadap parameter pemrosesan citra umaupun evaluasi terhadap sistem

pencahayaan dan perangkat keras sehingga nilai Hue yang diperoleh tetap konsisten serta mampu meningkatkan akurasi sistem koreksi warna secara keseluruhan.

3.9 Skenario Uji Validasi Koreksi Warna

Skenario ini bertujuan untuk memvalidasi kinerja algoritma koreksi secara keseluruhan. Pengujian dilakukan dengan sengaja memberikan input warna awal yang salah (meleset dari target). Sistem kemudian diabaikan melakukan koreksi secara otomatis. Parameter yang dianalisis adalah Laju Penurunan Galat ΔEterhadap jumlah iterasi. Keberhasilan pengujian ditandai dengan kemampuan sistem mencapai target warna dengan jumlah iterasi yang efisien tanpa terjadi kondisi *Overshoot* warna [60].

Pengujian ini mengadopsi prinsip *closed-loop* control berbasis umpan balik (feedback), di mana sistem secara kontinu melakukan akuisisi data warna aktual, menghitung galat terhadap target, dan menghasilkan aksi koreksi yang diperbarui pada setiap iterasi. Mekanisme ini memungkinkan sistem untuk melakukan penyesuaian secara adaptif hingga mencapai kondisi konvergensi yang stabil. Dalam konteks pengendalian warna berbasis visi, pendekatan *closed-loop* telah terbukti efektif dalam meningkatkan akurasi reproduksi warna melalui proses koreksi bertahap yang dipandu oleh nilai error sebagai parameter utama pengendali.

Selain itu, evaluasi kinerja sistem tidak hanya ditinjau dari kemampuan mencapai nilai galat minimum, tetapi juga dari karakteristik dinamika respon sistem, seperti kecepatan konvergensi, kestabilan terhadap gangguan, serta kemampuan menghindari kondisi overshoot yang dapat menyebabkan deviasi warna melewati target. Oleh karena itu, analisis terhadap kurva penurunan ΔE terhadap jumlah iterasi menjadi indikator penting dalam menilai performa algoritma koreksi yang diimplementasikan. Pendekatan ini sejalan dengan penelitian pada sistem pengendalian warna real-time berbasis sensor, yang menunjukkan bahwa integrasi mekanisme umpan balik dalam proses koreksi mampu menghasilkan performa yang lebih presisi dan stabil dalam mencapai target warna yang diinginkan.

1. **Perhitungan Manual (5 Spektrum Warna**

   Untuk memvalidasi data yang penulis miliki yaitu spektrum warna, dapat mensimulasikan sistem *closed-loop.* Penulis tentukan Target dan Kondisi Awal (salah). Tujuan validasi Adalah melihat ΔE mengecil di setiap iterasi. Dengan permisalan $\Delta E < 5$.

Tabel 3.2 Spektrum Warna HSV

Spektrum	Target (HSV)	Awal Terbaca (HSV)	Perhitungan ΔE Awal
Merah	(0,255,255)	(10,200,200)	ΔE $= \sqrt{(0-10)^2 + (255-200)^2 + (25$ ≈ 78.4
Kuning	(30, 255, 255)	(20,220,230)	ΔE $= \sqrt{(30-20)^2 + (255-220)^2 + (2$ ≈ 37.7
Biru	(120, 255,255)	(110,200,180)	ΔE $= \sqrt{(120-110)^2 + (255-200)^2 +$ ≈ 91.2
Hitam	(0,0,0)	(0,20,50)	ΔE $= \sqrt{(0-0)^2 + (0-20)^2 + (0-50)^2}$ ≈ 53.8
Putih	(0,0,255)	(0,15,200)	ΔE $= \sqrt{(0-0)^2 + (0-15)^2 + (255-2$ $\approx 57.$

Contoh Proses Iterasi (Warna Merah):

1. Iterasi 1: ΔE = 78.4. Sistem menambah dosis pigmen merah.
2. Iterasi 2: Warna berubah jadi (5, 230, 230). ΔE baru≈35.7.

3. Iterasi 3: Warna berubah jadi (2, 250, 250) ΔE baru ≈ 7.3.
4. Iterasi 4: Warna berubah jadi (0, 254, 254) ΔE baru ≈ 1.4. (CONVERGENCE/STOP)

2. Pengembangan Coding Python Rekognisi Citra

Dalam pengujian warna untuk validasi dalam phyton penulis perlu melakukan pembacaan kamera Raspberry Cam V2 dengan logika validasi yaitu:

```
import math
import time

# --- 1. INISIALISASI PARAMETER ---
# Target Warna (Merah)
target_hsv = (0, 255, 255)

# Kondisi Awal (Warna Salah)
aktual_hsv = [10, 200, 200]

# Parameter Kontrol
Kp = 0.2          # Konstanta Proportional
threshold = 5.0   # Batas toleransi berhenti
max_iterasi = 10

def hitung_delta_e(target, aktual):
    # Rumus Euclidean Distance: sqrt((H1-H2)^2 +
(S1-S2)^2 + (V1-V2)^2)
    return math.sqrt(
        (target[0] - aktual[0])**2 +
        (target[1] - aktual[1])**2 +
        (target[2] - aktual[2])**2
    )

print("---    SIMULASI    KONTROL    ITERATIVE
PROPORTIONAL ---")
print(f"Target Warna: {target_hsv}")
```

```
print(f"Kondisi Awal: {aktual_hsv}\n")

# --- 2. LOOPING KONTROL (ITERATIVE) ---
for i in range(1, max_iterasi + 1):
    # A. Hitung Galat (e)
    error = hitung_delta_e(target_hsv,
aktual_hsv)

    print(f"Iterasi ke-{i}:")
    print(f"  > Warna Saat Ini : {[round(x, 2)
for x in aktual_hsv]}")
    print(f"  > Galat (Delta E): {error:.2f}")

    # B. Cek Kondisi Berhenti (Validasi)
    if error <= threshold:
        print("  > STATUS: STABIL (Konvergen ke
Target)")
        break

    # C. Hitung Output Kontrol (u)
    # u(k) = Kp * e(k)
    massa_koreksi = Kp * error
    print(f"  > Aksi: Tambahkan
{massa_koreksi:.2f} gram pigmen")

    # D. SIMULASI PERUBAHAN WARNA (Feedback)
    # Di alat asli, bagian ini digantikan oleh
adukan cat yang difoto kamera.
    # Di sini penulis simulasikan warna aktual
mendekati target secara matematis.
    aktual_hsv[0] -= (aktual_hsv[0] -
target_hsv[0]) * Kp
    aktual_hsv[1] += (target_hsv[1] -
aktual_hsv[1]) * Kp
```

```
        aktual_hsv[2]    +=    (target_hsv[2]    -
    aktual_hsv[2]) * Kp

        print("-" * 40)
        time.sleep(1)

    print("\n--- PENGUJIAN VALIDASI SELESAI ---")
```

Fungsi hitung_delta_e digunakan untuk menghitung nilai galat () antara warna target dan warna aktual menggunakan metode *Euclidean* Distance pada ruang warna HSV. Nilai hasil perhitungan disimpan pada variabel error yang menjadi dasar evaluasi apakah warna telah memenuhi batas toleransi. Selanjutnya, variabel massa_koreksi dihitung menggunakan persamaan 2.5, sehingga semakin besar nilai galat maka semakin besar pula massa pigmen yang ditambahkan. Setelah proses koreksi dilakukan, nilai aktual_hsv diperbarui agar semakin mendekati warna target sebagai simulasi mekanisme *feedback*. Proses ini berlangsung secara iteratif hingga nilai galat berada di bawah ambang toleransi atau mencapai jumlah iterasi maksimum.

Untuk menggambarkan cara kerja *Iterative Proportional* Control, dilakukan perhitungan manual menggunakan persamaan 2.5 dengan nilai konstanta proporsional $K_p = 0{,}2$. Nilai ini menunjukkan bahwa setiap kenaikan 1 poin galat warna (ΔE) akan menghasilkan penambahan 0,2 gram pigmen. Pada kondisi awal, diperoleh nilai ΔE = 78,4 sehingga massa pigmen yang harus ditambahkan sebesar 15,68 gram. Setelah pigmen dituang dan campuran diaduk hingga homogen, sistem kembali melakukan pembacaan warna dan diperoleh nilai ΔE = 35,7. Karena galat telah berkurang, massa pigmen yang dihitung juga menurun menjadi 7,14 gram. Pada iterasi berikutnya, nilai galat semakin kecil, yaitu ΔE = 7,3, sehingga massa pigmen yang ditambahkan hanya 1,46 gram sebagai proses fine-tuning untuk menyempurnakan warna. Dari perhitungan tersebut terlihat bahwa semakin kecil nilai galat warna yang diperoleh, semakin

kecil pula massa pigmen koreksi yang diberikan. Proses ini berlangsung secara iteratif hingga nilai ΔE berada di bawah ambang toleransi yang telah ditentukan, sehingga warna hasil pencampuran dinyatakan stabil dan sesuai dengan warna target [61].

Ketika penulis sudah mendapatkan perhitungan secara manual penulis dapat menentukan variabel Kpnya dalam bentuk kode di Phtyon sehingga penulis mendapat perbandingan antara perhitungan manual dengan codingan untuk penentuan kestabilan warna dalam proses mixing

```
# Simulasi Kontrol Proporsional Pencampuran Cat (Tanpa
Kamera/CV2)

# --- PARAMETER KONTROL ---
Kp = 0.2
threshold = 5.0
error_saat_ini = 78.4    # Asumsi Galat (Delta E) awal
sesuai data manualmu

# --- PARAMETER SISTEM TANGKI (SIMULASI) ---
# Asumsi: Setiap 1 gram pigmen yang dituang dan diaduk,
error akan turun sebesar 2.7 poin
faktor_reaksi_tangki = 2.7

def hitung_massa_pigmen(galat):
    return Kp * galat

# --- LOOPING KOREKSI SIMULASI ---
iterasi = 1
print(f"=== SIMULASI DIMULAI (Kp = {Kp}) ===")

while True:
    print(f"\n--- Iterasi ke-{iterasi} ---")
          print(f"Galat    saat    ini    (Delta    E):
{error_saat_ini:.2f}")
```

```
    # 1. EVALUASI: Cek apakah warna sudah masuk toleransi
    if error_saat_ini <= threshold:
        # Jika error negatif, artinya warna terlalu pekat
(Overshoot)
        if error_saat_ini < 0:
            print(">>> STATUS: GAGAL! WARNA TERLALU PEKAT
(OVERSHOOT). Kp TERLALU BESAR!")
        else:
             print(">>> STATUS: WARNA STABIL & AKURAT.
PROSES SELESAI.")
        break

    # 2. AKSI: Hitung massa pigmen yang harus dituang
    massa_tambahan = hitung_massa_pigmen(error_saat_ini)
     print(f"Aksi: Menuangkan {massa_tambahan:.2f} gram
pigmen.")

     # 3. SIMULASI FISIK: Update nilai error setelah
dituang (Pengganti Kamera)
     # Di dunia nyata, ini adalah saat kamera membaca
ulang warna setelah diaduk
           penurunan_error    =    massa_tambahan    *
faktor_reaksi_tangki
    error_saat_ini = error_saat_ini - penurunan_error

    iterasi += 1

    # Safety break
    if iterasi > 15:
         print(">>> STATUS: GAGAL! Sistem terlalu lambat
(Kp terlalu kecil) atau tidak stabil.")
        break
```

Pada simulasi ini, sistem kendali proporsional diterapkan untuk menggambarkan proses koreksi warna tanpa menggunakan kamera atau *computer vision*. Parameter utama yang digunakan meliputi Kp, threshold, dan faktor_reaksi_tangki. Nilai Kp berfungsi sebagai konstanta proporsional yang menentukan besarnya massa pigmen yang harus ditambahkan berdasarkan nilai galat (*Delta E*), sedangkan threshold merupakan batas toleransi yang digunakan sebagai kondisi penghentian ketika warna telah dianggap sesuai dengan target. Sementara itu, faktor_reaksi_tangki merepresentasikan karakteristik fisik proses pencampuran, yaitu besarnya penurunan nilai galat yang dihasilkan setiap penambahan satu gram pigmen.

Perhitungan massa pigmen dilakukan melalui fungsi hitung_massa_pigmen**()**, yang mengimplementasikan persamaan 2.5 Fungsi ini menerima nilai galat sebagai masukan, kemudian menghasilkan massa pigmen yang harus ditambahkan pada setiap iterasi. Dengan demikian, semakin besar nilai galat yang terukur maka semakin besar pula massa pigmen yang dihitung, sedangkan ketika galat semakin kecil, massa pigmen yang diberikan juga akan berkurang secara proporsional [62].

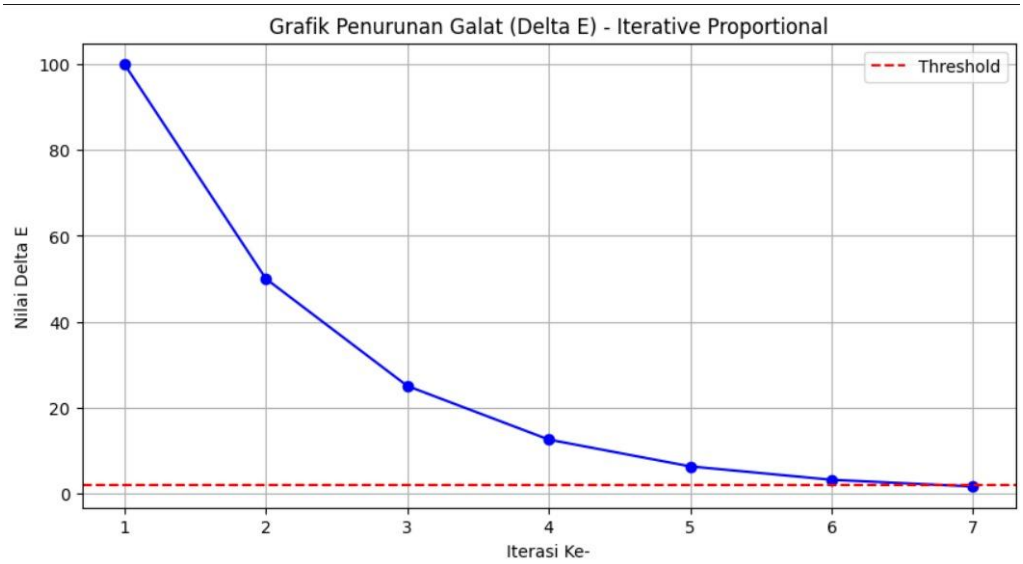
Proses simulasi dijalankan secara berulang menggunakan *looping* yang terdiri dari tahap evaluasi, aksi, serta pembaruan kondisi sistem. Pada tahap evaluasi, program memeriksa apakah nilai galat telah berada di bawah *threshold*. Ketika kondisi tersebut sudah sesuai, proses dihentikan karena warna telah dianggap stabil. Namun ketika galat masih melebihi batas toleransi, maka tem menghitung massa pigmen yang harus ditambahkan, kemudian mensimulasikan penurunan nilai galat menggunakan faktor_reaksi_tangki sebagai pengganti proses pembacaan ulang oleh kamera. Nilai galat yang telah diperbarui selanjutnya digunakan sebagai masukan pada iterasi berikutnya sehingga proses koreksi berlangsung secara bertahap hingga mencapai kondisi konvergen.

Program juga dilengkapi dengan mekanisme *safety break* yang membatasi jumlah iterasi maksimum sebanyak 15 kali. Pembatasan tersebut bertujuan mencegah *looping* tanpa akhir ketika nilai Kp terlalu kecil sehingga sistem gagal mencapai toleransi yang diinginkan. Simulasi tersebut menunjukkan bahwa

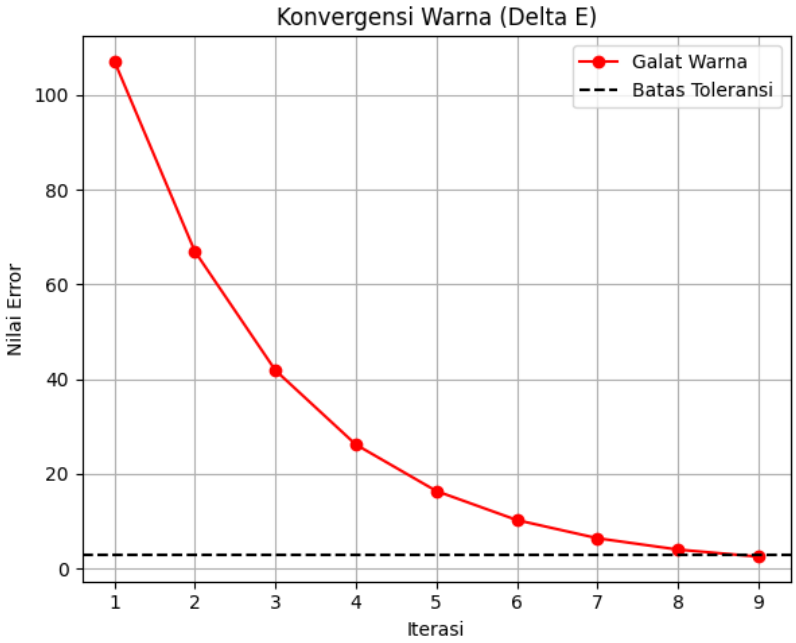
penggunaan metode *Proportional* mampu menghasilkan proses koreksi warna yagn lebih stabil karena galat dikurangi secara bertahap, mengurangi Risiko *overshoot* akibat penambahan pigman yang berlebihan, serta memungkinkan sistem beradaptasi terhadap perubahan nilai galat yang terjadi pada proses pencampuran.

3.10 Perancangan Awal Sistem Dengan Tahap Validasi Algoritma

Guna memvalidasi keandalan sistem, dirancang sebuah skrip simulasi dasar yang bertujuan untuk memvalidasi algoritma *Euclidean Distance* dalam menghitung galat warna (*Delta E*) antara nilai acuan (*setpoint*) ruang warna target dengan warna aktual. Selain perhitungan jarak warna tersebut, tahap ini juga menguji logika kendali proporsional yang menjadi dasar sistem.



Gambar 3.15 (Hasil Simulasi Perhitungan Galat *Euclidean Distance dan Proportional)*



Gambar 3.16 (Hasil Simulasi Perhitungan Galat *Proportional)*

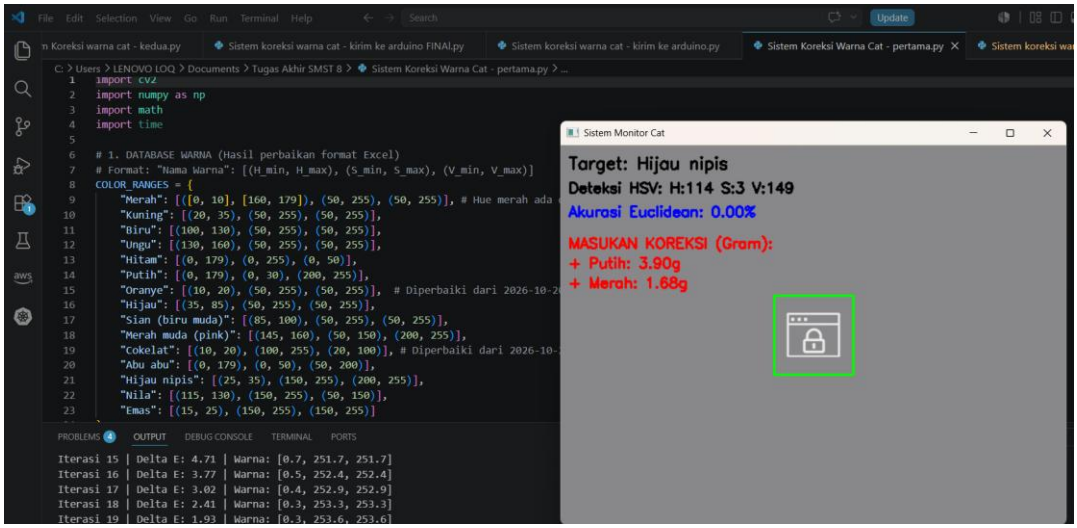
Melalui pemodelan ini, dibuktikan secara teoretis dan komputasional bahwa persamaan metode *Proportional* yang dirancang mampu mengonversi selisih warna menjadi estimasi massa, sekaligus menurunkan nilai galat secara konvergen menuju batas toleransi yang diizinkan.

Algoritma *Euclidean Distance* pada penelitian ini bekerja dengan mengukur jarak numerik antara dua titik koordinat pada ruang warna HSV, yaitu titik warna target dan titik warna aktual, sehingga nilai galat (*Delta E*) yang dihasilkan dapat merepresentasikan tingkat kemiripan warna secara kuantitatif. Nilai galat inilah yang kemudian diolah oleh logika kendali proporsional untuk menentukan besar estimasi massa pigmen yang perlu ditambahkan, sehingga terdapat keterkaitan langsung antara akurasi pengukuran warna dan ketepatan hasil kompensasi massa pada tahap simulasi ini [63].

3.11 Perancangan *Computer Vision* dan Antarmuka

Perancangan awal sistem visi komputer (*computer vision*) dikembangkan menggunakan pustaka VsCode istem diinisialisasi dengan mendefinisikan basis data memori yang memuat rentang nilai ruang warna HSV (*Hue, Saturation, Value*) untuk berbagai spektrum pigmen cat dasar. Program dirancang agar mampu

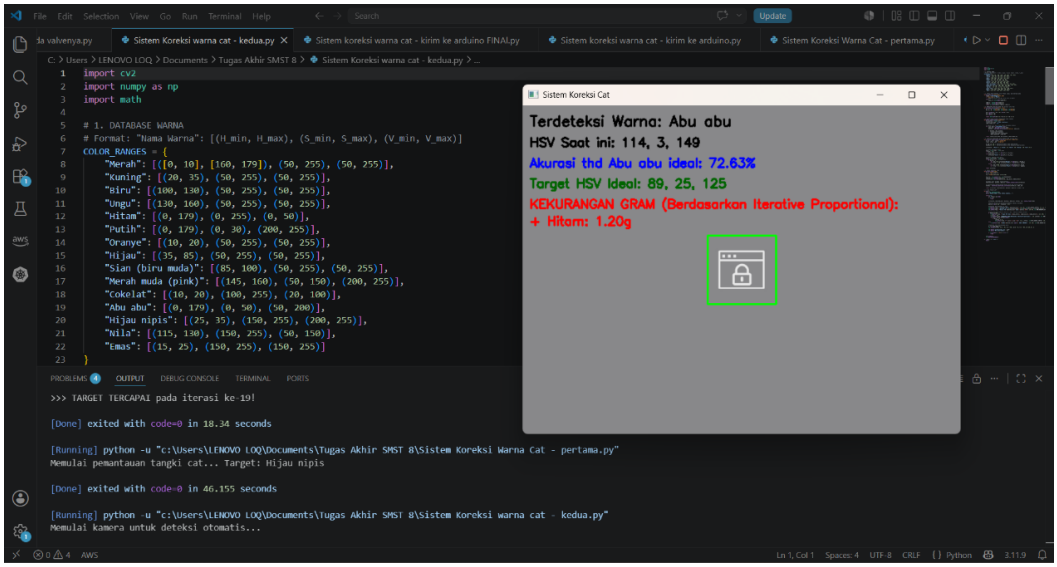
mengakuisisi piksel warna secara *real-time* dari tangkapan kamera, melakukan konversi ruang warna, kemudian mendeteksi warna dominan berdasarkan tingkat kedekatannya terhadap nilai-nilai pada basis data tersebut.



Gambar 3.17 (Hasil Percobaan Awal akuisisi Citra dan Deteksi Warna Awal)

Secara umum, ruang warna HSV dipilih dalam bidang visi komputer karena representasinya memisahkan komponen rona (*hue*), saturasi, dan intensitas cahaya (*value*) secara terpisah, sehingga proses deteksi warna dominan menjadi lebih tahan terhadap variasi pencahayaan dibandingkan ruang warna RGB secara langsung. Basis data HSV yang telah didefinisikan berfungsi sebagai acuan pembanding bagi setiap piksel citra yang diakuisisi kamera, sehingga sistem dapat secara otomatis mengklasifikasikan warna dominan yang paling mendekati salah satu kelas pigmen dasar yang telah didaftarkan.

Kalkulasi Kompensasi dan Antarmuka dilakukan guna mempermudah pemantauan proses komputasi, perancangan sistem dilanjutkan dengan pengembangan antarmuka pengguna grafis atau *GUI (Graphical User Interface)*

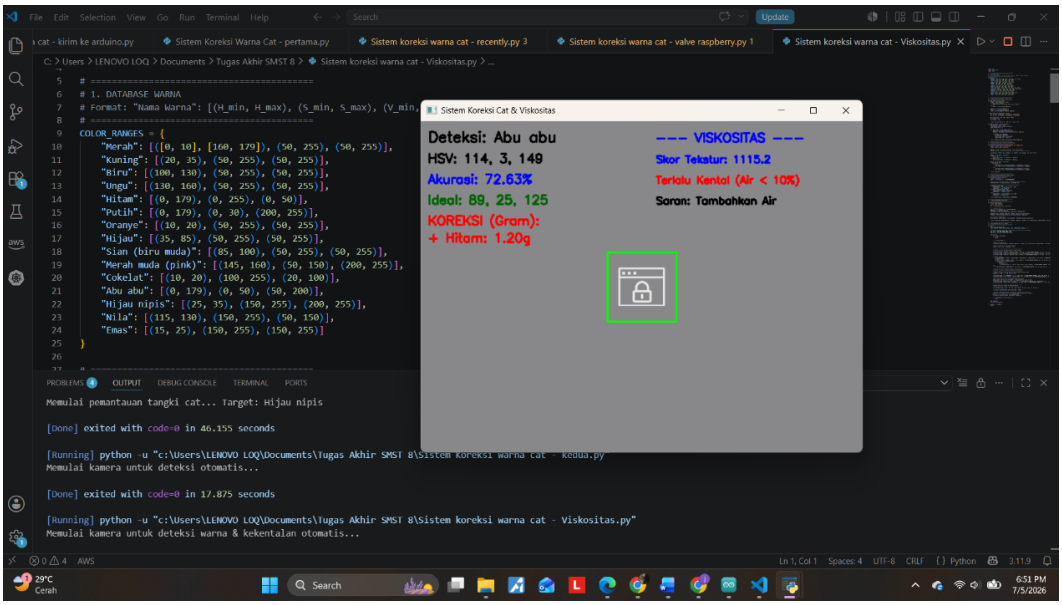


Gambar 3.18 (Hasil Percobaan Kalkulasi Kompensasi dan Antarmuka)

Sistem ini memproyeksikan hasil pengolahan citra secara *overlay* pada layar tampilan. Luaran matematis dari kontroler proporsional pada tahap ini tidak lagi terbatas pada nilai persentase akurasi, tetapi sistem juga mulai mampu mengestimasi dan menampilkan defisit massa (dalam satuan gram) yang dibutuhkan apabila akurasi kecocokan warna masih berada di bawah target 90%.

Penambahan antarmuka *GUI* pada tahap ini berperan penting dari sisi kebergunaan (*usability*), yaitu memberikan visualisasi langsung kepada pengguna mengenai status pencocokan warna tanpa perlu membaca data mentah pada konsol pemrograman. Ambang batas akurasi 90% yang ditetapkan berperan sebagai indikator keberhasilan (*acceptance criteria*) yang menjadi rujukan bagi sistem untuk menentukan apakah proses pencampuran warna dapat dihentikan atau perlu dilanjutkan dengan penambahan massa pigmen sesuai estimasi defisit yang ditampilkan.

Ekstraksi Ciri Fisik Cat Selain mendeteksi kroma warna, parameter analitik sistem diperluas untuk mendeteksi karakteristik fisik cairan cat.



Gambar 3.19 (Hasil Percobaan Ekstraksi Ciri Fisik Cat)

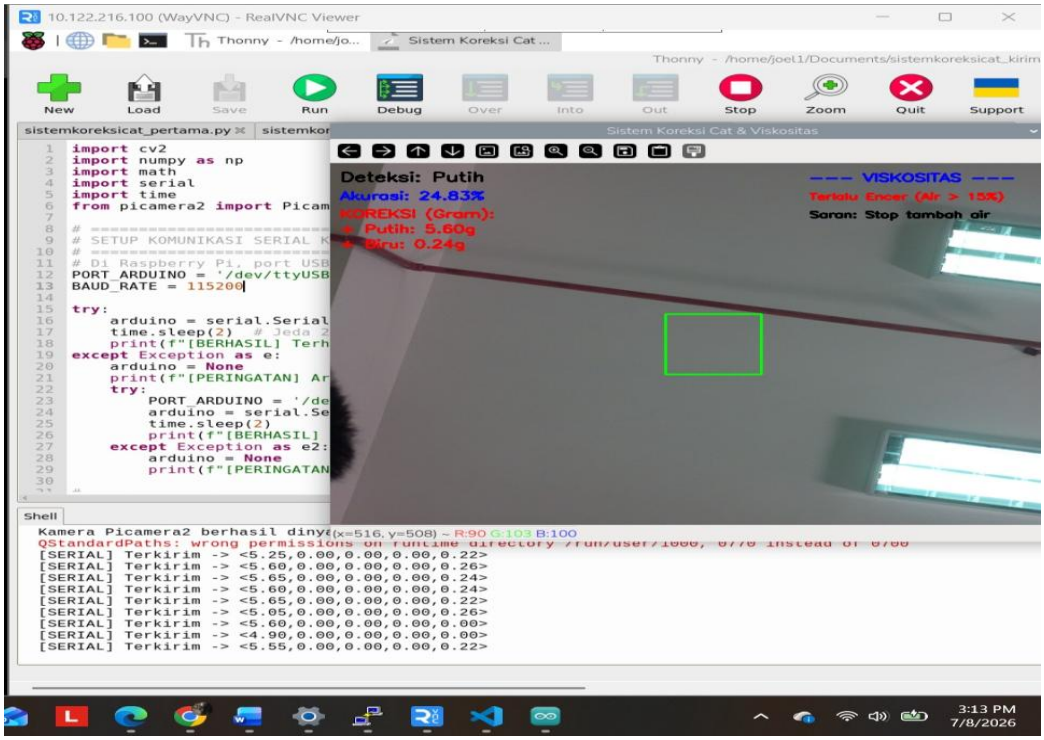
Ekstraksi fitur spasial ditambahkan menggunakan metode variansi Laplacian pada citra *grayscale* untuk menghasilkan luaran berupa Skor Tekstur. Algoritma ini berfungsi khusus untuk mendeteksi tingkat kekentalan (viskositas) cat, sehingga sistem dapat memberikan peringatan operasional dan merekomendasikan penambahan pelarut (air) apabila kondisi fisik cat dinilai belum ideal untuk proses pencampuran.

Metode variansi *Laplacian* yang digunakan pada tahap ini bekerja dengan mengukur tingkat perubahan intensitas piksel pada citra *grayscale*; semakin tinggi variansi yang dihasilkan, semakin tinggi pula tingkat ketajaman tekstur permukaan cairan cat yang terekam kamera. Nilai Skor Tekstur tersebut kemudian dijadikan indikator tidak langsung dari tingkat viskositas cat, karena cairan yang terlalu kental umumnya menghasilkan pola permukaan yang berbeda dibandingkan cairan dengan viskositas normal, sehingga sistem dapat mengambil keputusan otomatis untuk merekomendasikan penambahan pelarut

Protokol komunikasi UART digunakan sebagai jembatan antara pertukaran data dua arah antara Rapsberry Pi dan Arduino Mega Pro Mini. Pengujian tersebut digunakan guna memastikan siklus kompensasi massa, eksekusi perangkat keras, dan juga *feeback* berjalan sinkron membentuk kesatuan sistem *closed-loop*.

1. **Pengiriman Instruksi dari Raspberry Pi**

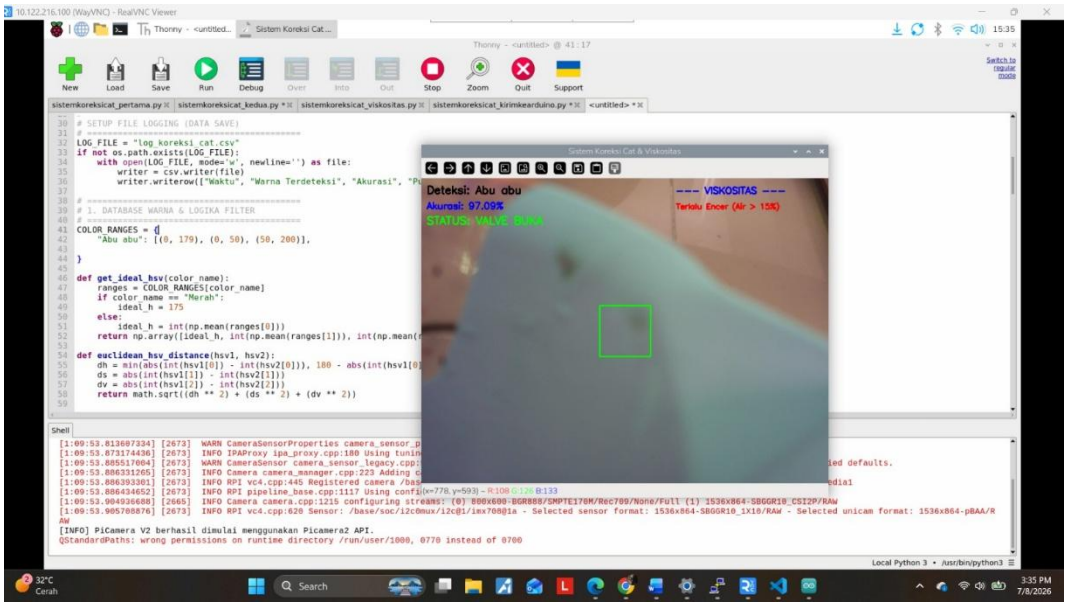
Komputer utama menggunakan *library serial* pada Python dengan *baud rate* 115200 untuk mengirimkan data hasil kalkulasi *computer vision*. Data kompensasi massa yang didapat dari metode Iterative Proportional dikirimkan menjadi satu data string terstruktur. Data tersebut menggunakan karakter "<" sebagai penanda awal dan ">" sebagai penanda akhir. Hal tersebut menunjukkan setiap nilai gramasi dipisahkan oleh tanda koma. Serta berdasarkan log sistem pada pengujian Gambar 3.20, Raspberry Pi mengirimkan data dengan format <5.60,0.00,0.00,0.00,0.00,0.24>, yang berarti sistem menginstruksikan aktuasi penambahan 5.60 gram untuk warna Putih dan 0.24 gram untuk warna Biru.



Gambar 3.20 (Pengujian Komunikasi Serial)

2. **Pemrosesan dan *Parsing Data* oleh Arduino Mega Pro Mini**

Sebagai penerima, Arduino Mega Pro Mini secara berkala memantau *buffer serial.* Saat mendeteksi adanya instruksi, mikrokontroler menjalankan algoritma *serial parsing*.



Gambar 3.21 (Hasil Simulasi Perekaman Data Historis)

Sistem memanfaatkan fitur *data logging* untuk merekam setiap iterasi pengujian secara mandiri. Parameter krusial seperti penanda waktu (*timestamp*), jenis warna terdeteksi, tingkat akurasi *Euclidean*, hingga takaran massa direkam dalam format dokumen *CSV* (*Comma-Separated Values*), yang juga diiringi dengan perancangan logika status kontrol buka-tutup katup (*valve*).

Pemilihan format *CSV* pada sistem perekaman data ini relevan karena sifatnya yang ringan, terstruktur, serta mudah diolah lebih lanjut menggunakan perangkat lunak pengolah data seperti Microsoft Excel maupun pustaka pemrograman seperti *pandas* pada Python. Data historis yang terekam secara otomatis pada tahap ini nantinya dapat digunakan sebagai bahan evaluasi kinerja sistem secara kuantitatif, misalnya untuk menganalisis tren akurasi *Euclidean* maupun konsistensi takaran massa pada setiap iterasi pengujian yang telah dilakukan.

Sinkronisasi komunikasi aktuator untuk kestabilan transfer instruksi pada arsitektur sistem terdistribusi ini disempurnakan melalui sinkronisasi temporal.

Program Python bertugas membuka jalur komunikasi, memformat hasil kalkulasi iterasi warna, mengirimkannya, dan *berhenti sejenak* (*blocking*) untuk menunggu konfirmasi dari mikrokontroler sebelum melanjutkan pengambilan gambar (*frame*) berikutnya.

```
import serial
import time

# Inisialisasi Komunikasi Serial (Diletakkan di awal program)
PORT_ARDUINO = '/dev/ttyUSB0' # Sesuaikan dengan port yang terbaca
BAUD_RATE = 115200

try:
    arduino = serial.Serial(PORT_ARDUINO, BAUD_RATE, timeout=1)
    time.sleep(2) # Memberikan jeda agar Arduino Mega Pro Mini selesai reset otomatis saat koneksi terbuka
    print("[BERHASIL] Komunikasi Serial Terhubung")
except Exception as e:
    arduino = None
    print(f"[PERINGATAN] Arduino tidak terdeteksi: {e}")

# Fungsi Pengiriman Data
def kirim_instruksi_massa(putih, hitam, merah, kuning, biru, valve):
    if arduino is not None:
        # Memformat float menjadi string dengan 2 angka di belakang koma
        # Menggunakan marker '<' sebagai awal dan '>' sebagai akhir paket data
        paket_data = f"<{putih:.2f},{hitam:.2f},{merah:.2f},{kuning:.2f},{biru:.2f},{valve:.2f}>"

        # Mengirimkan data via UART (wajib diubah ke format byte/utf-8)
        arduino.write(paket_data.encode('utf-8'))
```

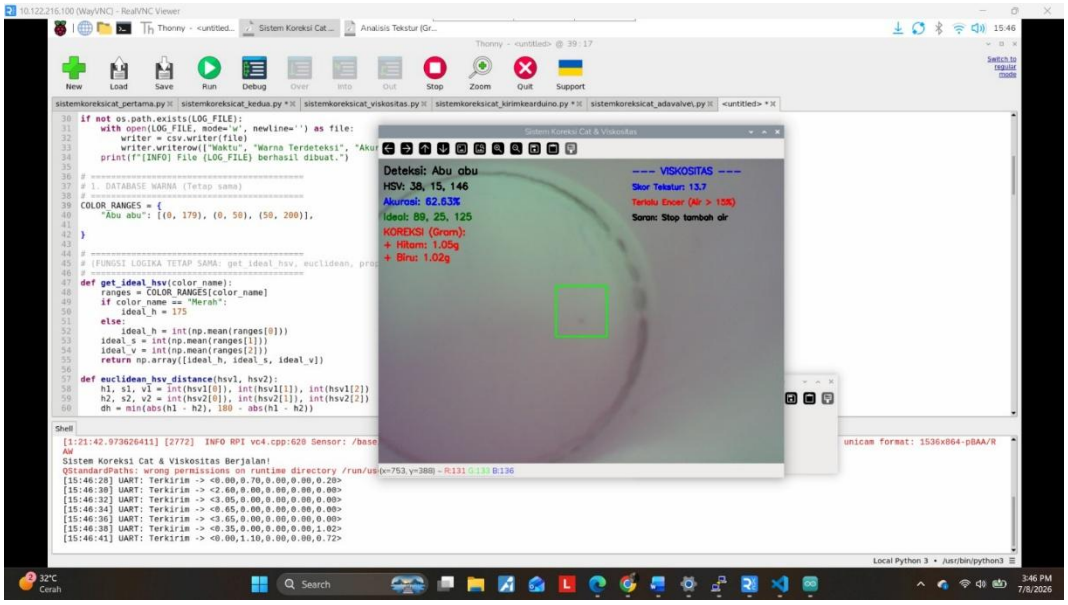
```
        print(f"[SERIAL] Terkirim -> {paket_data}")

# Fungsi Menunggu Feedback (Sistem Closed-Loop)
def tunggu_feedback_selesai():
    if arduino is not None:
        print("[SISTEM]    Menunggu    proses    penuangan
selesai...")
        while True:
            if arduino.in_waiting > 0:
                # Membaca balasan dari Arduino
                feedback = arduino.readline().decode('utf-
8').strip()

                # Cek jika Arduino mengirim sinyal selesai
                if feedback == "SELESAI":
                    print("[SISTEM]    Aktuasi    selesai,
melanjutkan iterasi kamera.")
                    break
            time.sleep(0.1) # Mencegah beban CPU 100% pada
loop
```

Agar Raspberry Pi dan Arduino Mega Pro Mini memiliki komunikasi yang berkesinambungan, Arduino melakukan *parsing string* yang panjang tersebut kembali menjadi sekumpulan angka *float* sebagai target aktuasi, dan

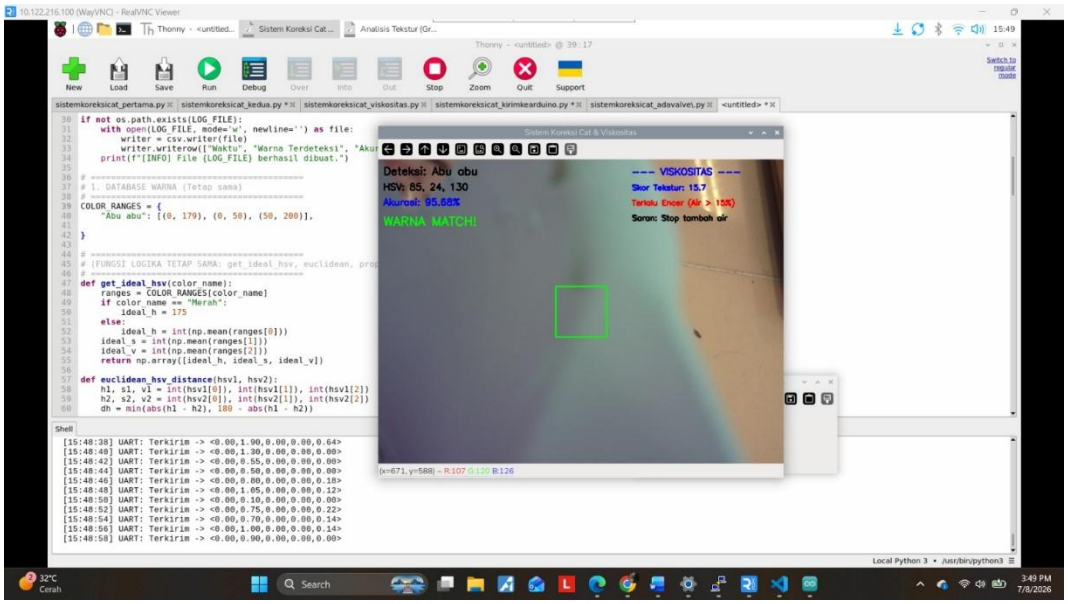
```
String dataMasuk = "";
bool paketLengkap = false;

// Variabel target penambahan massa
float targetPutih, targetHitam, targetMerah, targetKuning,
targetBiru, targetValve;

void setup() {
  Serial.begin(115200); // Harus sama dengan baud rate di
Python
}
```

```
void loop() {
  // 1. Blok Pembacaan Data Masuk (Asynchronous)
  while (Serial.available() > 0 && !paketLengkap) {
    char karakter = (char)Serial.read();

    if (karakter == '<') {
      dataMasuk = ""; // Kosongkan buffer jika mendeteksi
awal paket
    }
    else if (karakter == '>') {
      paketLengkap = true; // Tandai bahwa 1 paket data
telah diterima utuh
    }
    else {
      dataMasuk += karakter; // Rangkai karakter menjadi
kalimat
    }
  }

  // 2. Blok Eksekusi dan Parsing Data
  if (paketLengkap) {
    // Fungsi parsing ini memisahkan string "5.60,0.00,..."
berdasarkan tanda koma
    targetPutih = getValue(dataMasuk, ',', 0).toFloat();
    targetHitam = getValue(dataMasuk, ',', 1).toFloat();
    targetMerah = getValue(dataMasuk, ',', 2).toFloat();
    targetKuning = getValue(dataMasuk, ',', 3).toFloat();
    targetBiru = getValue(dataMasuk, ',', 4).toFloat();
    targetValve = getValue(dataMasuk, ',', 5).toFloat();

    // ------------------------------------------------
------
    // DI SINI PROSES AKTUASI HARDWARE BERJALAN
```

```
    // Motor pompa menyala, Load Cell membaca berat secara
real-time
    // sampai pembacaan sensor sama dengan target variabel
di atas
    // ------------------------------------------------
------

    // 3. Mengirimkan Feedback Balik ke Raspberry Pi
    // Setelah semua aktuasi pompa sesuai gramasi selesai,
kirim konfirmasi:
    Serial.println("SELESAI");

    // Reset status agar siap menerima paket data instruksi
baru
    dataMasuk = "";
    paketLengkap = false;
  }
}

// Fungsi bantuan untuk memecah string berdasarkan karakter
pemisah (koma)
String getValue(String data, char separator, int index) {
  int found = 0;
  int strIndex[] = {0, -1};
  int maxIndex = data.length()-1;
  for(int i = 0; i <= maxIndex && found <= index; i++){
    if(data.charAt(i) == separator || i == maxIndex){
        found++;
        strIndex[0] = strIndex[1] + 1;
        strIndex[1] = (i == maxIndex) ? i+1 : i;
    }}
  return  found  >  index  ?  data.substring(strIndex[0],
strIndex[1]) : "";
}
```



Gambar 3.22 (Hasil Simulasi Sinkroni Komunikasi Aktuator Sebelum Warna Match)



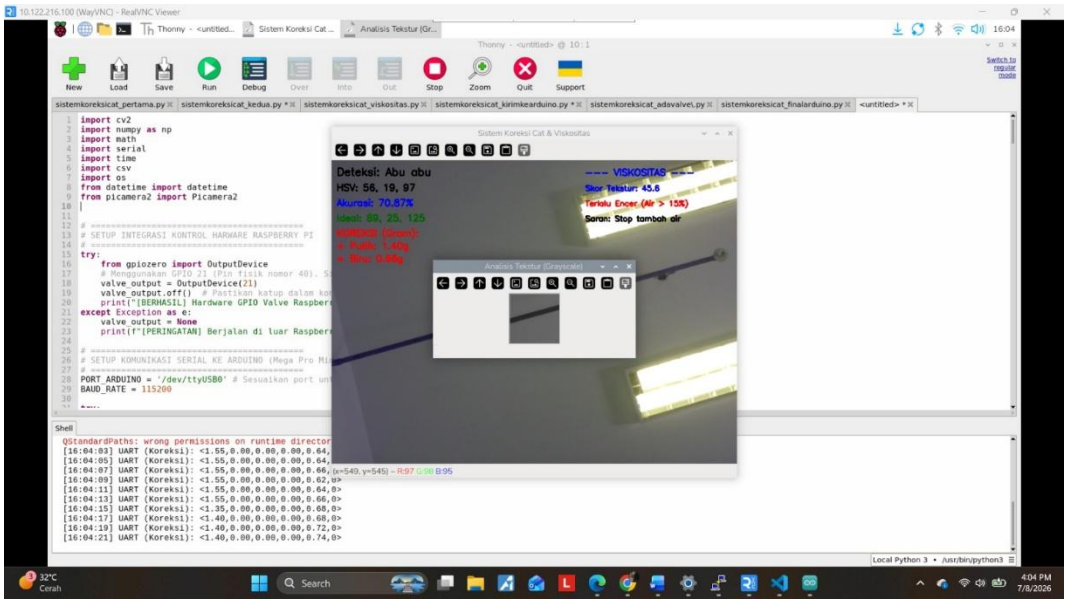
Gambar 3.23 (Hasil Simulasi Sinkronisasi Komunikasi Aktuator Sesudah Warna Match)

Optimalisasi dilakukan dengan mengatur algoritma jeda transmisi (*delay*) data serial secara periodik. Langkah ini dirancang khusus untuk mencegah terjadinya penumpukan data (*bottleneck*) saat mikrokontroler merespons dan mengeksekusi instruksi aktuasi beban mekanis.

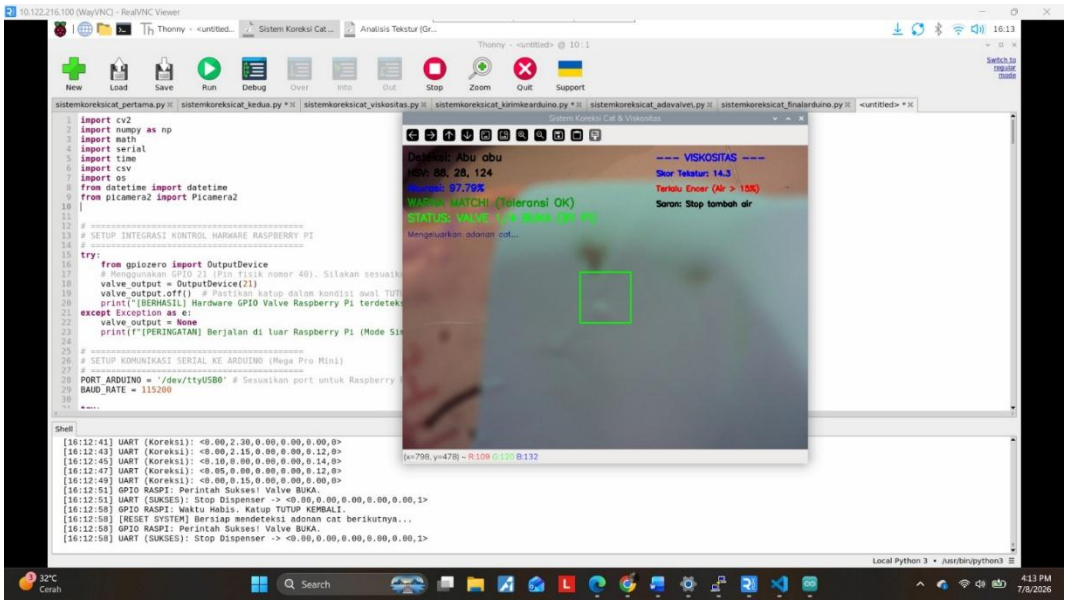
Permasalahan *bottleneck* pada komunikasi serial umumnya timbul akibat perbedaan kecepatan pemrosesan antara komputer utama dan mikrokontroler, terutama ketika mikrokontroler masih menjalankan instruksi aktuasi mekanis sementara komputer utama telah mengirimkan data instruksi berikutnya. Melalui pengaturan jeda transmisi yang tersinkronisasi, sistem dapat memastikan setiap instruksi diterima dan dieksekusi mikrokontroler secara berurutan, sehingga risiko tumpang tindih perintah yang dapat menyebabkan kegagalan aktuasi dapat diminimalkan [64].

3.12 Perancangan Kendali *Closed-Loop* Terintegrasi Untuk Pemindahan Kendali *Input/Ouput*

Arsitektur perangkat lunak dipusatkan dengan memigrasikan lingkungan operasional secara penuh.



Gambar 3.24 (Hasil Simulasi Pemindahan Kendali *Input/Output* Sebelum Menyentuh Warna Match)



Gambar 3.25 (Hasil Simulasi Pemindahan Kendali *Input/Ouput* Setelah Menyentuh Warna Match)

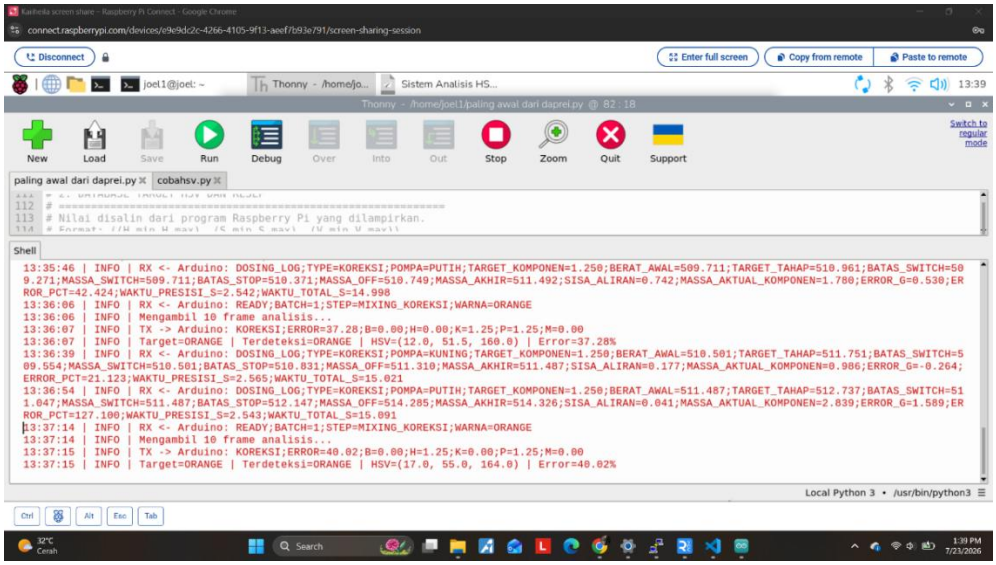
Kendali perangkat keras yang semula bertumpu pada komunikasi dengan mikrokontroler eksternal kini dialihkan langsung ke pin *General Purpose Input/Output* (*GPIO*) Raspberry Pi. Program mengendalikan aktuator *valve* secara *direct drive* menggunakan pustaka khusus guna meminimalkan latensi instruksi perangkat keras.

Pengalihan kendali aktuator dari mikrokontroler eksternal menuju pin GPIO Raspberry Pi secara langsung ini menghilangkan satu tahap komunikasi serial yang sebelumnya diperlukan pada arsitektur terdistribusi. Dengan pendekatan *direct drive* tersebut, instruksi aktuasi dapat dieksekusi tanpa melalui proses pengiriman maupun penerjemahan data antarperangkat, sehingga latensi sistem secara keseluruhan dapat ditekan dan respons aktuator valve terhadap keputusan sistem menjadi lebih cepat [65].

3.13 Perancangan Mekanisme Kompensasi Fluktuasi Ilminasi Pada Kamera

Fluktuasi intensitas cahaya dalam *Mixing Tank* berpotensi menyebabkan pembacaan nilai warna HSV menjadi tidak stabil sehingga memengaruhi hasil perhitungan galat warna dan memicu pemberian dosis pigmen yang tidak sesuai. Untuk meminimalkan pengaruh tersebut, pada perancangan alat ini dilakukan mekanisme toleransi iluminasi adaptif yang tidak menggunakan satu citra sebagai

dasar pengambilan keputusan *(single fram decision)*. Nilai HSV dari setiap citra juga diproses dan nilai median dari 10 fram digunakan sebagai representasi warna aktual. Hal tersebut dipilih karena nilai median lebih tahan terhadap *outlier* yang disebabkan oleh pantulan cahaya (*specula reflection)* maupun *noise* selama proses akuisisi citra [66].

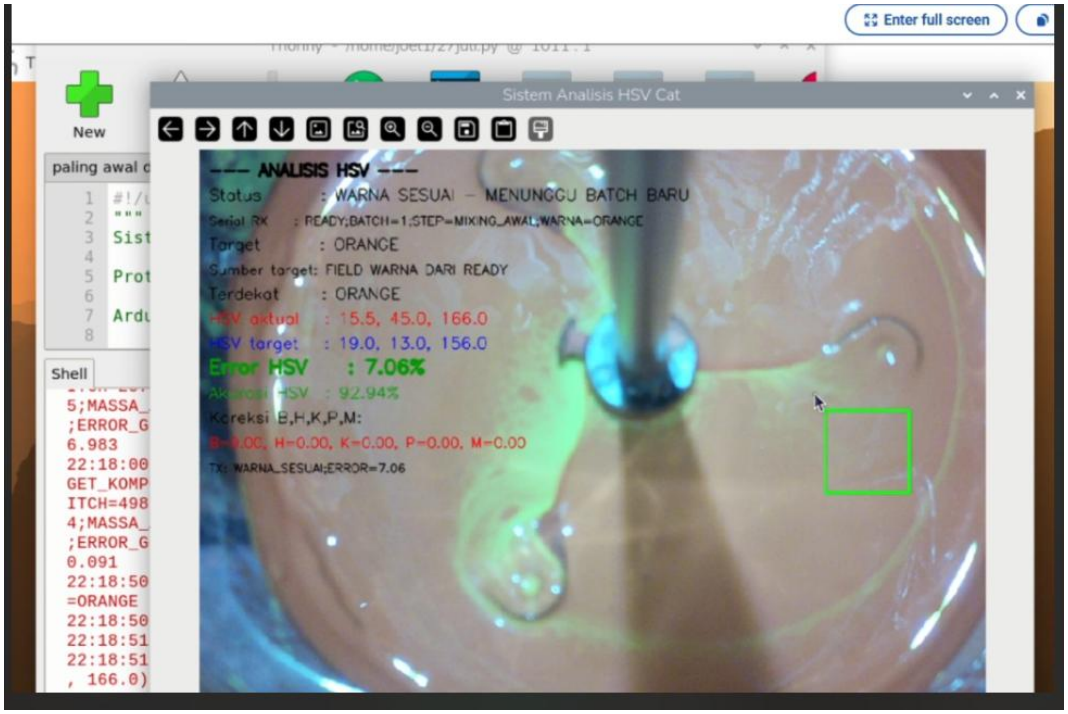


Gambar 3.26 *Frame Analysis*

3.14 Perancangan Ambang Batas (*Threshold)* Toleransi HSV dan *Hysteresis Threshold*

Pada penelitian ini dirancang mekanisme penghentian proses koreksi warna menggunakan *threshold* sebagai kriteria konvergensi pada sistem *close-loop*. Nilai *Eulidean Distance* pada ruang warna HSV ditetapkan sebesar 10% sebagai batas toleransi penerimaan warna. Ketika nilai galat berada di bawah batas tersebut, algoritma *Proportional* tidak lagi menghitung dosis koreksi dan Raspberry Pi akan mengirimkan sinyal *Interlock* kepada Arduino untuk melanjutkan proses pengeluaran cat. Penetapan nilai toleransi dilakukan karena hasil ekstraksi HSV masih dipengaruhi oleh variasi intensitas pencahayaan, refleksi ermukaan cat, serta *noise* kamera, sehingga kesesuaian warna tidak dapat ditentukan berdasarkan nilai HSV yang identik secara absolut. Oleh karena itu, sistem dirancang menggunakan rentang toleransi tertentu agar keputusan penghentian proses koreksi lebih realistis dan sesuai dengan kondisi implementasi.

Selain menggunakan *threshold* sebagai batas konvergensi, sistem juga dirancang menerapkan mekanisme *Hysteresis Threshold* untuk meningkatkan stabilitas keputusan ketika nilai galat berada di sepenulisr batas toleransi. Berdasarkan penelitian oleh Åström dan Murray (2021**)**, mekanisme hysteresis pada sistem kendali umpan balik digunakan untuk mencegah terjadinya fenomena *hunting*, yaitu perubahan status kontrol secara berulang akibat fluktuasi sinyal di sepenulisr nilai ambang. Berdasarkan konsep tersebut, penelitian ini mempertahankan status konvergen ketika perubahan nilai HSV masih berada di dalam rentang hysteresis yang telah ditentukan, sehingga sistem tidak kembali melakukan koreksi akibat fluktuasi kecil yang berasal dari *noise* akuisisi citra maupun perubahan iluminasi sesaat. Dengan perancangan tersebut, proses koreksi warna diharapkan berlangsung lebih stabil, mengurangi jumlah iterasi yang tidak diperlukan, serta meminimalkan kemungkinan terjadinya *over-correction* akibat penambahan pigmen yang berlebihan [67].



Gambar 3.27 Penggunaan *Hysterisis Threshold*