BAB IV
PERANCANGAN DAN IMPLEMENTASI

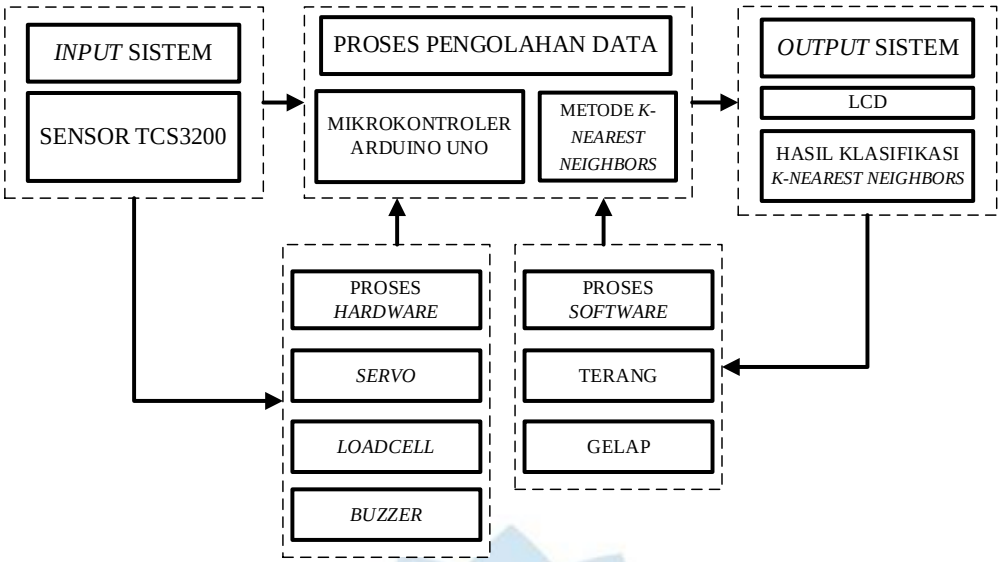
Perancangan dan implementasi adalah proses penting dalam penelitian. Perancangan adalah tahap merancang dan mendesain sistem yang baik sebelum implementasi. Implementasi adalah realisasi dari perancangan yang telah dirancang secara matang, mengaplikasikan setiap detail konsep awal ke dalam bentuk nyata, memastikan semua aspek teknis dan fungsional bekerja sesuai tujuan yang telah ditetapkan.

4.1 Perancangan

Tahap perancangan bertujuan untuk menentukan setiap komponen penyusun sistem secara rinci, meliputi aspek teknis, fungsional, dan estetika, agar hasil akhir memenuhi spesifikasi dan kebutuhan yang ditetapkan. Proses ini memastikan semua elemen teridentifikasi dan direncanakan dengan baik sehingga alat dapat berfungsi sesuai harapan. Subbab ini menjelaskan perancangan sistem, *hardware*, dan *software* dari prototipe konveyor pemilah limbah tekstil kain berdasarkan tingkatan warna dengan metode klasifikasi *K-Nearest Neighbors.*

4.1.1 Perancangan Sistem

Perancangan sistem rancang bangun prototipe konveyor pemilah limbah tekstil kain ini terdapat tiga blok sistem, yaitu sistem *input*, proses, dan *output*. Pada blok sistem *input*, digunakan sensor TCS3200 sebagai sensor untuk membaca nilai RGB dari limbah kain perca. Hasil dari pembacaan nilai RGB oleh sensor TCS3200 akan di proses oleh mikrokontroler, yang nantinya data nilai RGB tersebut juga akan menjadi data *training* untuk algoritma *K-Nearest Neighbors*. *Output* yang dihasilkan dari proses mikrokontroler dan metode *K-Nearest Neighbors* akan di tampilkan pada LCD 16x2. Adapun diagram blok dari perancangan sistem ini dapat dilihat pada Gambar 4.1.



Gambar 4. 1 Diagram blok sistem.

1) Bagian *Input*

   Bagian *input*, terdapat sensor TCS3200 yang digunakan untuk membaca nilai RGB dari limbah kain perca, nilai-nilai tersebut nantinya akan di proses oleh mikrokontroler dan juga menjadi data *training* untuk metode *K-Nearest Neighbors.*

2) Bagian Proses

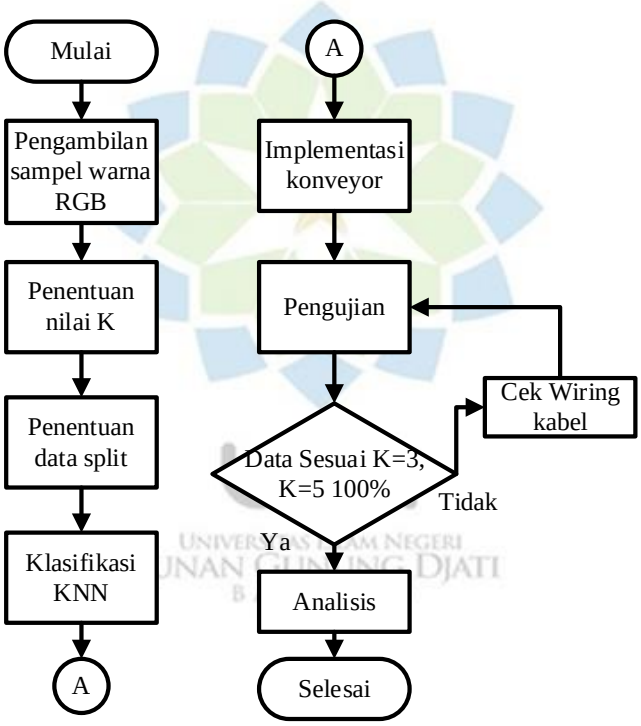
   Proses pengolahan data menggunakan mikrokontroler Arduino Uno dan metode *K-Nearest Neighbors*, Arduino Uno digunakan untuk mengolah data RGB. Data tersebut kemudian diproses menggunakan metode *K-Nearest Neighbors* pada Arduino Uno. Proses *hardware* terdiri dari *servo*, *loadcell* HX-711, dan *buzzer*. Semua komponen *hardware* tersebut akan diproses oleh Arduino. Pada bagian *software*, terdapat pembagian kelompok warna yaitu cerah dan gelap. Pembagian kelompok warna ini diproses menggunakan metode *K-Nearest Neighbors*.

3) Bagian *Output*

   Bagian *output,* hasil pemrosesan pada mikrokontroler dan metode *K-Nearest Neighbors* ditampilkan pada LCD.

Sistem pemilah limbah kain perca secara keseluruhan dimulai dari pengujian *hardware* dan pengujian *software*. Pada pengujian *hardware* terdiri dari

pengujian komponen-komponen yang digunakan seperti Arduino UNO, *servo*, motor DC *gearbox*, *loadcell* HX-711, LCD 16x2, dan *buzzer*. Semua komponen tersebut diintegrasikan menjadi sebuah sistem pemilah. Selanjutnya, pada pengujian *software* dimulai dengan memasukan metode *K-Nearest Neighbors* pada Arduino IDE kemudian dilakukan pengambilan data dari sensor TCS3200, pengambilan data tersebut menjadi hasil dari klasifikasi dengan metode *K-Nearest Neighbors,* yang dibagi menjadi dua kelas klasifikasi yaitu cerah dan gelap. Hasil dari klasifikasi tersebut nantinya akan ditampilkan pada *output* LCD 16x2. Sistem pemilah limbah kain perca dapat dilihat pada Gambar 4.2.

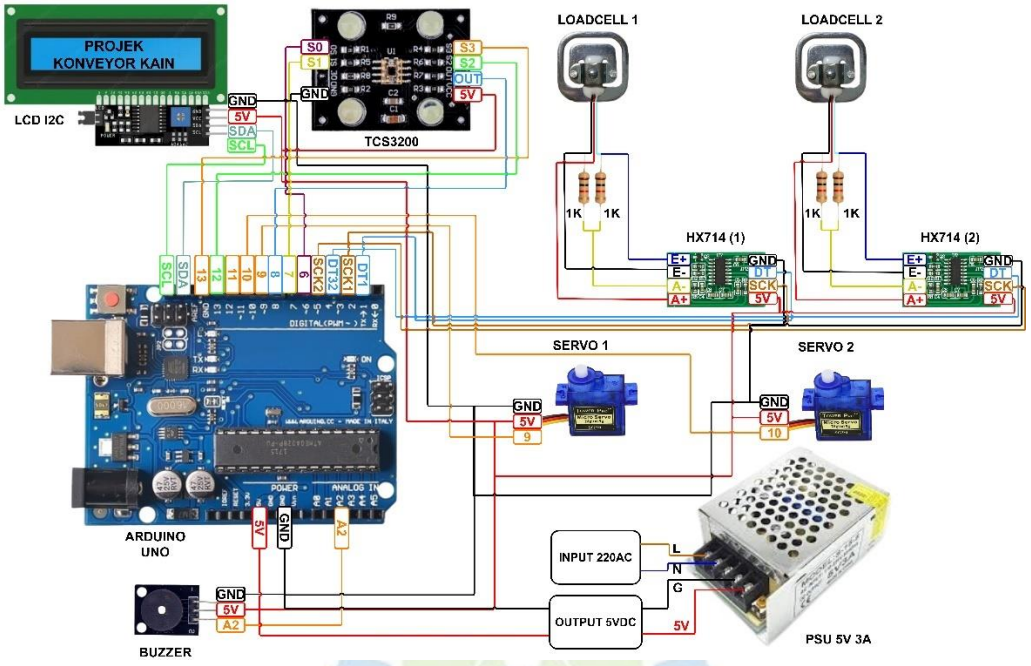


Gambar 4. 2 *Flowchart* sistem.

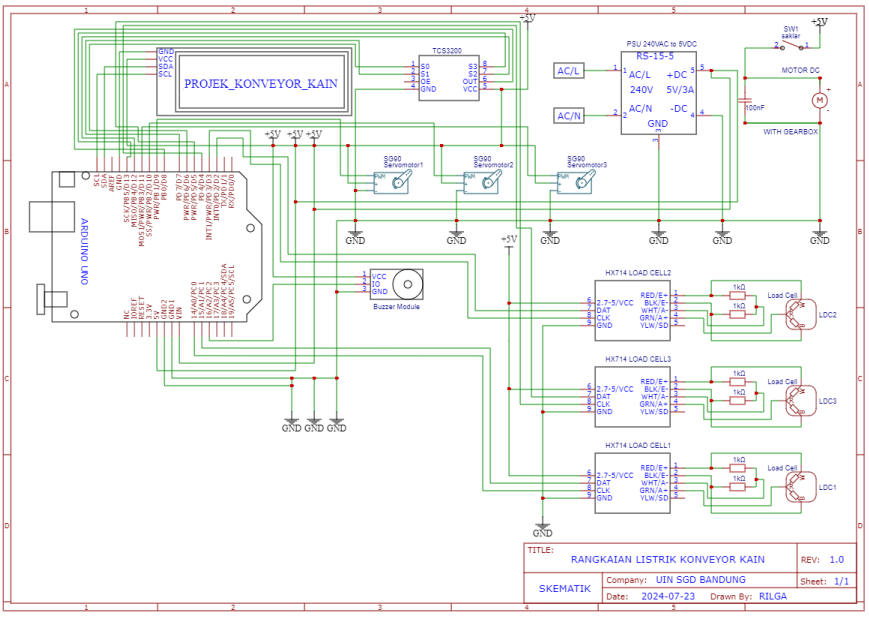
4.1.2 Perancangan *Hardware*

Pada tahap perancangan *hardware*, menggunakan berbagai jenis komponen elektronik maupun non-elektronik. Bagian perancangan *hardware* ini menjelaskan mengenai skematik sistem pemilah limbah kain perca, desain elektrik sistem, desain keseluruhan sistem dan *flowchart* sistem *hardware*. Perancangan yang pertama yaitu perancangan skematik sistem pemilah limbah kain perca. Perancangan

skematik sistem pemilah dan skematik rangkaian tersebut dapat dilihat pada Gambar 4.3. dan 4.4.



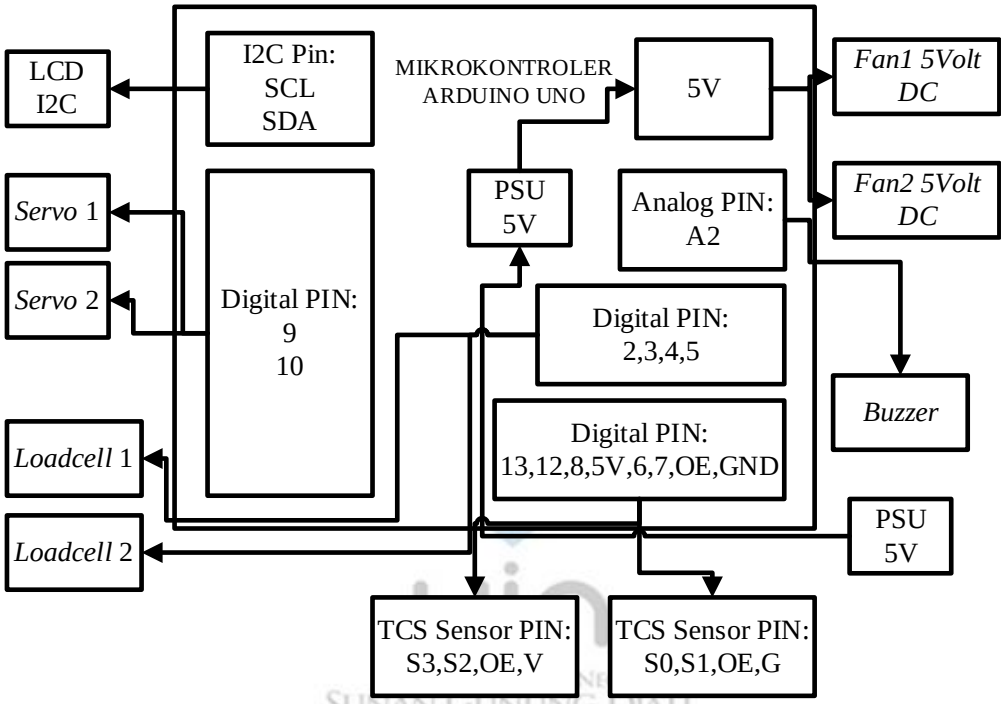
Gambar 4. 3 Skematik sistem



Gambar 4. 4 Skematik rangkaian.

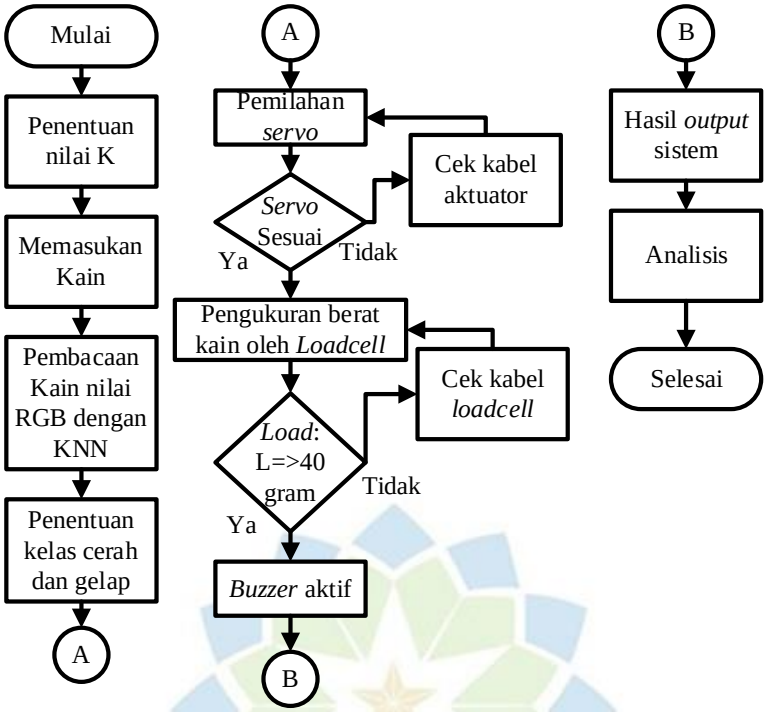
Gambar 4.3 dan 4.4 merupakan skematik untuk sistem pemilah limbah kain perca yang terdiri dari satu sensor TCS3200, dua *Loadcell* HX-711, dua *servo*, satu LCD 16x2, satu buah *buzzer*, dan satu buah Arduino Uno. Sensor TCS3200

berfungsi untuk membaca nilai RGB dari limbah kain. Nilai RGB tersebut nantinya akan di proses oleh mikrokontroler arduino uno dan juga menjadi data *training* untuk metode *K-Nearest Neighbors. Loadcell* HX-711 merupakan fitur tambahan dalam sistem ini yang digunakan untuk mengetahui berat kain hasil klasifikasi. Jika beratnya mencapai batas yang ditentukan, sinyal akan dikirim ke Arduino, yang kemudian memerintahkan *buzzer* untuk menyala. Selanjutnya yaitu desain elektrik sistem pemilah limbah kain perca. Desain elektrik dapat dilihat pada Gambar 4.5.



Gambar 4. 5 Desain elektrikal.

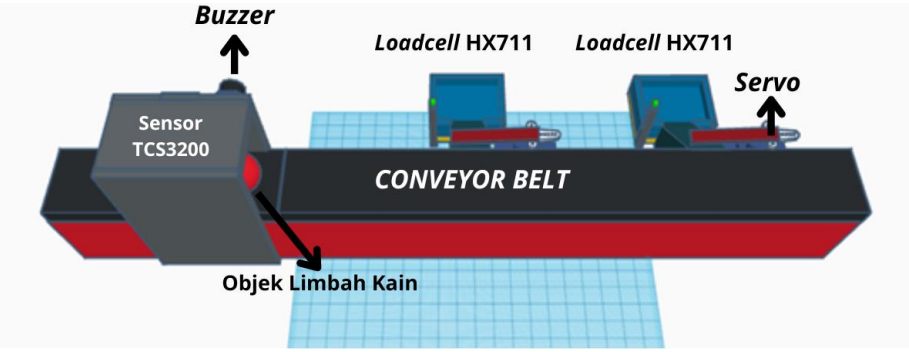
Gambar 4.5 merupakan desain elektrikal dari sistem pemilah limbah kain perca, desain elektrikal tersebut menunjukan penggunaan pin-pin yang digunakan dalam membangun sistem pemilah. Desain elektrikal tersebut sebagian besar memanfaatkan pin digital. *Flowchart* dari sistem *hardware* pemilah limbah kain perca dapat dilihat pada Gambar 4.6.



Gambar 4. 6 *flowchart* sistem *hardware*.

Gambar 4.6 merupakan *flowchart* sistem *hardware*, sistem dimulai dengan menentukan nilai K yang akan digunakan dalam algoritma *K-Nearest Neighbors* (KNN) untuk proses klasifikasi. Setelah nilai K ditetapkan, kain dimasukkan ke dalam sistem untuk dilakukan pembacaan nilai RGB dengan menggunakan algoritma KNN. Hasil dari pembacaan ini digunakan untuk menentukan kelas kain, apakah termasuk dalam kategori cerah atau gelap. Setelah penentuan kelas warna kain, langkah berikutnya adalah memeriksa kesesuaian kerja motor *servo*. Jika motor *servo* berfungsi dengan baik, proses dilanjutkan dengan mengaktifkan sensor *loadcell*. Sensor *loadcell* kemudian akan mengukur berat kain yang ada, dan jika beratnya melebihi atau sama dengan 40 gram, *buzzer* akan diaktifkan sebagai tanda. Apabila pada langkah sebelumnya ditemukan ketidaksesuaian dalam kerja motor *servo* atau sensor *loadcell*, maka akan dilakukan pengecekan kabel pada aktuator atau *loadcell* untuk memastikan tidak ada kesalahan koneksi. Setelah semua langkah tersebut berjalan sesuai prosedur, sistem akan menghasilkan *output* yang akan dianalisis lebih lanjut.

Konsep dasar perancangan prototipe alat ini didasarkan pada kebutuhan untuk dapat memilah limbah kain hasil klasifikasi. Alat ini dirancang untuk dapat memilah kain berdasarkan dua tingkatan warna yaitu, cerah dan gelap. Prototipe ini memiliki dimensi dengan panjang 80 cm, tinggi 10 cm, dan lebar 10 cm. Material yang digunakan dalam perancangan prototipe ini menggunakan bahan papan PCV dengan kombinasi ketebalan 3 mm dan 2 mm. Prototipe ini memiliki dua sensor utama yaitu sensor warna TCS3200 dan sensor *Loadcell* HX-711, sensor warna TCS3200 digunakan untuk menyerap warna dan mendeteksi warna RGB, sensor ini digunakan sebagai sensor yang masuk dalam sistem klasifikasi penerapkan algoritma *K-Nearest Neighbors.* Sensor *Loadcell* HX-711 digunakan sebagai sensor tambahan yang berfungsi untuk mengetahui berat dari masing-masing kain hasil pemilahan. Pemilahan limbah kain menggunakan dua *servo* untuk membantu kain masuk ke kotak hasil pemilahan yang sudah delengkapi dengan Sensor *Loadcell* HX-711. Perancangan alat menggunakan *software Tinkercad* untuk mempermudah dalam pembuatan desain 3D. Desain 3D dapat dilihat pada Gambar 4.7 Perancangan *hardware.*



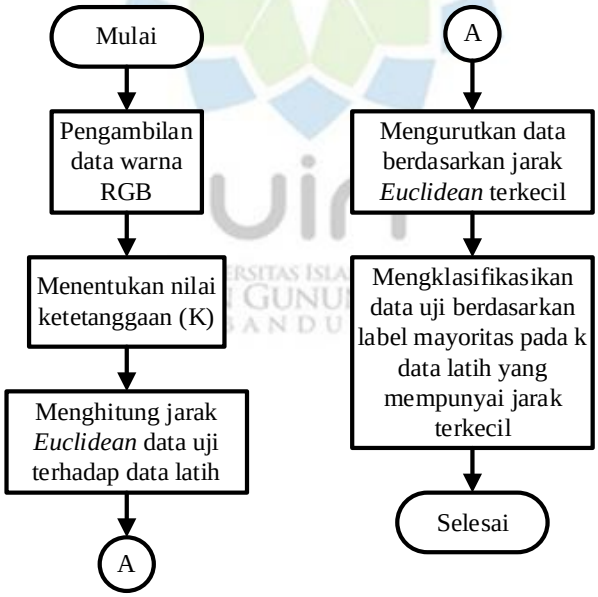
Gambar 4. 7 Prototipe 3D konveyor.

4.1.3 Perancangan *Software*

Perancangan perangkat lunak (*software*) dalam penelitian ini disesuaikan dengan kebutuhan untuk mendukung tahapan metodologi. Perancangan tersebut mencakup kebutuhan perangkat lunak yang akan digunakan dalam penelitian, termasuk perangkat lunak pemrograman dan perangkat lunak dalam pembuatan skematik sistem. Perangkat lunak yang dipilih harus sesuai dan mendukung

perangkat keras yang telah dirancang. Adapun perangkat lunak yang digunakan yaitu:

1. *Arduino IDE Arduino IDE* merupakan salah satu *software* yang dapat digunakan untuk membuat, memverifikasi, dan mengunggah kode program ke mikrokontroler. *Software* ini menggunakan pemrograman bahasa C atau C++. Perangkat lunak ini digunakan untuk mengolah masukan nilai RGB dari sensor TCS3200, masukan tersebut nantinya diolah kembali dalam menerapkan metode *K-Nearest Neighbors. Arduino IDE* ini juga memprogram untuk *servo*, serta komponen-komponen lainnya menjadi suatu sistem yang menjadi satu kesatuan.
2. *Draw IO software draw IO* merupakan perangkat lunak diagram yang digunakan untuk membuat berbagai jenis diagram visual. Perangkat lunak ini digunakan dalam membuat skematik-skematik sistem yang digunakan dalam pembuatan sistem pemilah limbah kain perca.



Gambar 4. 8 Tahapan proses metode KNN.

Gambar 4.8 menjelaskan alur tahapan metode *K-Nearest Neighbors* (KNN) yang dimulai dengan pengambilan data warna RGB dari kain melalui sensor yang telah dirancang khusus untuk menangkap nilai-nilai warna tersebut. Sebelumnya, dilakukan proses pelabelan oleh *expert* dari industri tekstil. *Expert* ini memiliki

peran penting dalam mengkategorikan kain ke dalam kelas tertentu, seperti "cerah" atau "gelap," berdasarkan penilaian mereka. Proses pelabelan ini bertujuan untuk memberikan label yang akan digunakan sebagai data latih dalam model KNN. Setelah proses pelabelan selesai, data yang telah dilabeli oleh *expert* kemudian melalui tahap *pre-processing*. Pada tahap ini, semua data yang telah dilabeli oleh *expert* diambil sepenuhnya, tanpa ada yang dihilangkan atau ditambahkan. Pelabelan data tersebut memang sudah dilakukan sejak awal oleh *expert* dari industri tekstil, sehingga proses *pre-processing* hanya fokus pada memastikan bahwa data yang dimasukkan ke sistem sudah terklasifikasi dengan benar. Setelah proses pelabelan oleh *expert* selesai, kain-kain tersebut dibaca oleh sensor untuk mendapatkan nilai RGB-nya.

Tahap berikutnya adalah penentuan nilai K, yaitu jumlah tetangga terdekat yang akan dipertimbangkan dalam proses klasifikasi. Nilai K ini sangat penting karena akan mempengaruhi akurasi hasil klasifikasi. Setelah nilai K ditetapkan, sistem akan menghitung jarak *Euclidean* antara data uji (data baru yang belum diketahui kelasnya) dan data latih (data yang telah dilabeli oleh *expert*). Jarak *Euclidean* ini digunakan untuk menentukan seberapa mirip data uji dengan setiap data latih.

Setelah jarak *Euclidean* dihitung, data uji diurutkan berdasarkan jarak terkecil hingga terbesar. Hal ini berarti data uji akan dibandingkan dengan sejumlah K data latih yang memiliki jarak terdekat. Langkah terakhir adalah melakukan klasifikasi data uji dengan melihat mayoritas label dari K data latih yang paling dekat. Label yang paling sering muncul dari K data latih tersebut akan menjadi prediksi kelas untuk data uji. Proses ini menghasilkan klasifikasi akhir, yang memungkinkan sistem untuk menentukan kategori kain yang paling sesuai berdasarkan warna yang terdeteksi.

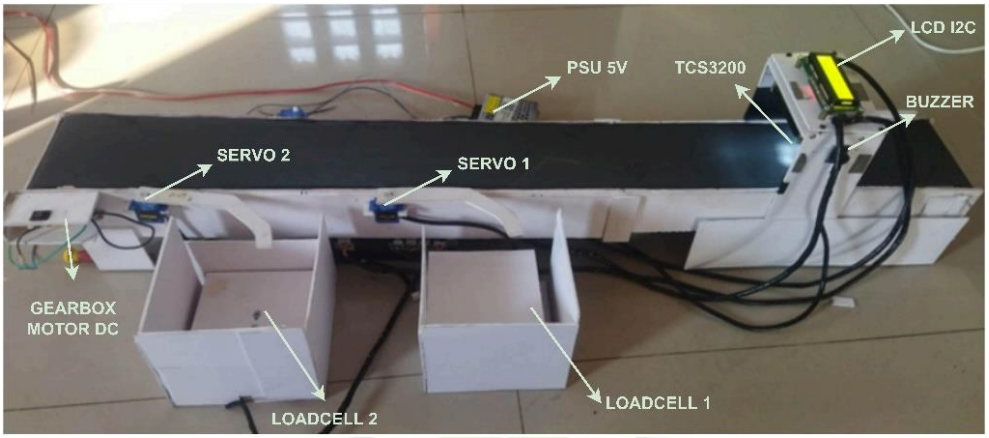
4.2 Implementasi

Setelah melewati tahapan perancangan dalam membuat desain *hardware*, skematik, maupun *software* secara keseluruhan, tahap selanjutnya adalah implementasi sistem pemilah limbah kain perca. Pada tahap implementasi ini akan

direalisasikan semua yang telah di buat saat perancangan, meliputi implementasi *hardware,* skematik, serta *software* untuk sistem pemilah limbah kain perca.

4.2.1 Implementasi *Hardware*

Implementasi *hardware* merupakan realisasi *hardware* yang telah dirancang sebelumnya. Bagian implementasi *hardware* ini mengimplementasikan desain desain 3D konveyor. Gambar 4.9 menunjukan hasil dari perancangan desain 3D sistem pemilah limbah kain perca.



Gambar 4. 9 Implementasi Desain 3D.

Gambar 4.9 komponen yang digunakan adalah satu buah mikrokontroler Arduino UNO, satu buah sensor TCS3200, dua buah *loadxell* HX-711, dua buah *servo*, satu buah LCD 16x2, dan satu buah *buzzer.*

4.2.2 Implementasi *Software*

Implementasi *software* yang digunakan mencakup Arduino IDE yang digunakan untuk pembuatan program pada mikrokontroler Arduino UNO. Metode *K-Nearest Neighbors* juga diimplementasikan sebagai bagian dari program yang dikembangkan menggunakan Arduino IDE. Selain Arduino IDE, dalam rancangan pembuatan sistem pemilah juga menggunakan *Draw IO* yang digunakan untuk merancang skematik sistem pemilah. Program sistem pemilah menggunakan *library* yang ditunjukan pada Tabel 4.1.

Tabel 4. 1 *Library* sistem pemilah.

Nama *Library*	Fungsi
Wire.h	Digunakan untuk mengatur komunikasi I2C *(Inter-Integrated Circuit)* antara mikrokontr1oler Arduino dan perangkat lain yang mendukung protokol I2C.
LiquidCrystal_I2C.h	Digunakan pada Arduino untuk mengontrol layar LCD karakter yang terhubung melalui *interface* I2C.
Servo.h	Digunakan pada Arduino untuk mengontrol motor *servo*.
HX711.h	Digunakan pada Arduino untuk menghubungkan dan mengontrol sensor timbangan *loadcell*.

Penelitian ini, beberapa *library* Arduino digunakan untuk mengimplementasikan fungsi-fungsi penting dalam sistem pemilah limbah kain perca. *Library Wire.h* digunakan untuk mengatur komunikasi I2C (*Inter-Integrated Circuit*) antara mikrokontroler Arduino dan perangkat lain yang mendukung protokol I2C, memungkinkan transfer data yang baik antara komponen-komponen sistem. *Library LiquidCrystal_I2C.h* diaplikasikan untuk mengontrol layar LCD karakter yang terhubung melalui *interface* I2C, sehingga hasil klasifikasi dapat ditampilkan secara jelas. Selanjutnya, *library Servo.h* digunakan untuk mengontrol motor *servo* yang bertugas mengarahkan kain perca ke wadah penampungan sesuai dengan hasil klasifikasi warna. Terakhir, *library HX711.h* digunakan untuk menghubungkan dan mengontrol sensor timbangan *loadcell*, yang berfungsi untuk menimbang berat kain di setiap wadah penampungan. Jika berat melebihi batas yang ditentukan, *buzzer* akan berbunyi sebagai tanda peringatan.

Pemrograman arduino dilakukan proses perubahan nilai ADC yang mengkonversi kedalam bentuk RGB, yang di paparkan pada program 1.
Konversi nilai ADC ke RGB pada program di jelaskan pada program 1

Code 1

```
f_red = pulseIn(OutputSensor, LOW);
```

```
red_value = map(f_red, 0, 1023, 0, 255);
f_green = pulseIn(OutputSensor, LOW);
green_value = map(f_green, 0, 1023, 0, 255);
f_blue = pulseIn(OutputSensor, LOW);
blue_value = map(f_blue, 0, 1023, 0, 255);
```
(1)

Kodingan ini, konversi dari nilai ADC ke nilai RGB dilakukan menggunakan fungsi `*map()*`. Pertama, nilai yang diperoleh dari sensor warna diambil melalui fungsi `*pulseIn()*`, yang mengukur durasi pulsa pada pin `*OutputSensor*`. Durasi ini merepresentasikan intensitas warna tertentu (merah, hijau, atau biru) yang terdeteksi oleh sensor. Setelah itu, nilai durasi ini dikonversi ke dalam skala 0-255 menggunakan fungsi `*map()*`, di mana skala tersebut merupakan standar representasi untuk nilai RGB. Fungsi `*map()*` mengonversi nilai dari rentang 0-1023, yang merupakan rentang nilai dari ADC, menjadi nilai yang setara pada rentang 0-255. Dengan demikian, konversi dari nilai ADC ke nilai RGB tercapai melalui proses ini.

Sistem ini menggunakan metode klasifikasi *K-Nearest Neighbors*, yang dijelaskan pada program 2.

Jarak *Euclidean* untuk setiap data latih.

Code 2

```
for (int i = 0; i < trainSize; i++) {
    int dr = red - trainData[i][0];
    int dg = green - trainData[i][1];
    int db = blue - trainData[i][2];
    distances[i] = sqrt(dr * dr + dg * dg + db * db);
  }
```
(2)

Bagian kode ini menghitung jarak antara data yang akan diklasifikasikan dengan setiap data dalam dataset menggunakan algoritma *K-Nearest Neighbors* (KNN). Prosesnya dimulai dengan menghitung selisih nilai RGB antara data yang diukur dan data latih, yang kemudian digunakan dalam rumus jarak *Euclidean*.

Rumus ini, `*sqrt(dr * dr + dg * dg + db * db)*`, mengukur jarak dalam ruang tiga dimensi dengan mengkuadratkan perbedaan setiap komponen warna (merah, hijau, biru), menjumlahkannya, dan kemudian mengambil akar kuadratnya. Hasilnya adalah jarak *Euclidean*, yang menunjukkan seberapa dekat data yang akan diklasifikasikan dengan data latih. Jarak-jarak ini disimpan dalam *array `distances`* dan digunakan untuk menentukan tetangga terdekat yang akan menentukan kelas dari data yang diukur.

4.2.3 Fase *Training* Dengan Algoritma *K-Nearest Neighbors*

Sampel limbah kain perca akan diklasifikasi menjadi dua kelas, yaitu cerah dan gelap, berdasarkan fitur warna menggunakan algoritma *K-Nearest Neighbors.* Proses yang dilakukan terdiri dari fase pelatihan dan fase pengujian atau fase klasifikasi. Pada fase pelatihan, algoritma KNN tidak melakukan proses belajar melainkan hanya menyimpan atau mengingat data latih. Pada tahap ini, data latih yang berupa fitur warna RGB dari sampel-sampel limbah kain perca yang telah dipersiapkan akan disimpan. Sampel-sampel ini sebelumnya telah dipilah dan dilabeli oleh seorang *expert* di bidang tekstil, memastikan bahwa klasifikasi cerah dan gelap sesuai dengan standar yang berlaku. Standar ini mencakup warna cerah yang memiliki tingkat kecerahan tinggi, seperti kuning cerah, merah muda, biru langit, hijau muda, oranye cerah, dan merah terang. Sementara itu, warna gelap mencakup warna-warna dengan tingkat kecerahan rendah, seperti coklat tua, ungu tua, hijau daun gelap, biru laut dalam, merah marun, hitam, dan abu-abu gelap. Setelah dilabeli sesuai dengan standar tersebut, data dimasukkan ke dalam sistem, dan sensor TCS3200 digunakan untuk mengambil nilai RGB dari masing-masing sampel. Nilai RGB ini kemudian digunakan sebagai data latih dalam algoritma KNN. Sampel-sampel tersebut diletakkan satu per satu di depan sensor TCS3200, dimulai dengan sampel kain perca yang cerah, diikuti oleh sampel kain perca yang gelap. Untuk menentukan jumlah sampel, dilakukan split validation dengan rasio pengujian, seperti 22 sampel untuk rasio 90:10, 20 sampel untuk rasio 80:20, dan 18 sampel untuk rasio 70:30.

Berikut merupakan tabel rekapan *output* nilai RGB pada tiap rasio data latih yang digunakan untuk melakukan klasifikasi limbah kain pada saat melakukan *training* data.

1. Data latih untuk rasio 90:10

   Split validation 90:10, jumlah sampel data latih adalah 22 data, yang dibagi menjadi 2 kelas secara rinci yaitu kelas cerah, dan gelap, masing-masing sebanyak 11 data. Hasil data latih dengan algoritma KNN berdasarkan frekuensi warna RGB-nya ditunjukkan pada Tabel 4.2 berikut:

Tabel 4. 2 Data latih untuk rasio 90:10.

No Sampel	Frekuensi Warna RGB			Kelas
	Merah (R)	Hijau (G)	Biru(B)	
1	25	25	30	Cerah
2	35	40	38	Cerah
3	45	50	55	Cerah
4	60	60	65	Cerah
5	50	55	53	Cerah
6	38	42	40	Cerah
7	55	57	60	Cerah
8	48	50	45	Cerah
9	62	65	63	Cerah
10	52	58	55	Cerah
11	43	45	47	Cerah
12	65	68	70	Gelap
13	75	80	78	Gelap
14	85	85	90	Gelap
15	70	72	75	Gelap
16	80	82	85	Gelap
17	65	70	68	Gelap
18	80	80	79	Gelap
19	29	32	68	Gelap
20	26	67	26	Gelap
21	73	11	12	Gelap
22	60	63	67	Gelap

2. Data latih untuk rasio 80:20

Pada split validasi dengan rasio 80:20, jumlah total sampel yang digunakan untuk data latih adalah sebanyak 20 data, yang secara spesifik dibagi menjadi dua kelas, yaitu kelas cerah dan kelas gelap. Masing-masing kelas tersebut terdiri dari 10 data, sehingga perbandingan antara kelas cerah dan gelap tetap seimbang. Data latih ini ditampilkan pada Tabel 4.3.

Tabel 4. 3 Data latih untuk rasio 80:20.

No Sampel	Frekuensi Warna RGB			Kelas
	Merah (R)	Hijau (G)	Biru(B)	
1	25	25	30	Cerah
2	35	40	38	Cerah
3	45	50	55	Cerah
4	60	60	65	Cerah
5	50	55	53	Cerah
6	38	42	40	Cerah
7	55	57	60	Cerah
8	48	50	45	Cerah
9	62	65	63	Cerah
10	52	58	55	Cerah
11	65	68	70	Gelap
12	75	80	78	Gelap
13	85	85	90	Gelap
14	70	72	75	Gelap
15	80	82	85	Gelap
16	65	70	68	Gelap
17	80	80	79	Gelap
18	29	32	68	Gelap
19	26	67	26	Gelap
20	73	11	12	Gelap

3. Data latih untuk rasio 70:30

Split validation 70:30, jumlah sampel data latih adalah 18 data, yang dibagi menjadi 2 kelas secara rinci yaitu kelas cerah, dan gelap, masing-masing sebanyak 8 data. Hasil data latih dengan algoritma KNN berdasarkan frekuensi warna RGB-nya ditunjukkan pada Tabel 4.4.

Tabel 4. 4 Data latih untuk rasio 70:30.

No Sampel	Frekuensi Warna RGB			Kelas
	Merah (R)	Hijau (G)	Biru(B)	
1	25	25	30	Cerah
2	35	40	38	Cerah
3	45	50	55	Cerah
4	60	60	65	Cerah
5	50	55	53	Cerah
6	38	42	40	Cerah
7	55	57	60	Cerah
8	48	50	45	Cerah
9	62	65	63	Cerah
10	65	68	70	Gelap
11	75	80	78	Gelap
12	85	85	90	Gelap
13	70	72	75	Gelap
14	80	82	85	Gelap
15	80	80	79	Gelap
16	29	32	68	Gelap
17	26	67	26	Gelap
18	73	11	12	Gelap

Tabel 4.2, 4.3, dan 4.4 menunjukkan proses penyimpanan data sampel (*training*) pada setiap rasio yang diuji. Sampel kain perca diletakkan di bawah sensor selama 1 detik untuk mengambil frekuensi warna merah (R), hijau (G), dan biru (B). Nilai warna ini disimpan sebagai data latih untuk klasifikasi KNN. Proses pengambilan sampel warna dapat dilihat pada Gambar 4.9.

Gambar 4. 10 Proses pengambilan sampel warna.