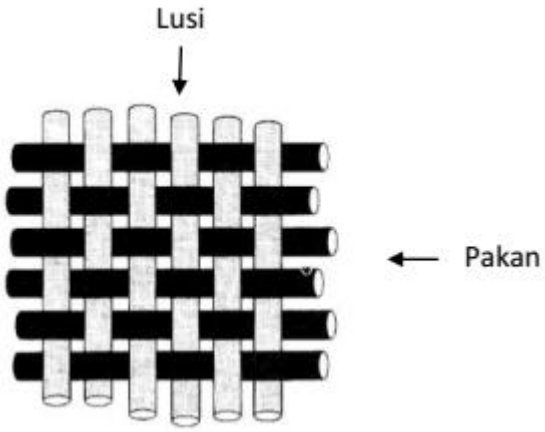
BAB II
DASAR TEORI

Pada bab ini dijelaskan mengenai teori-teori penunjang yang digunakan dalam Skripsi ini.

2.1 Kerusakan Utama pada Kain

Kain adalah bahan yang terbuat dari hasil tenunan benang. Struktur kain dibentuk dari anyaman dua jenis benang yaitu benang lusi (memanjang) dan benang pakan (melebar) dengan ukuran tertentu [6]. Jenis kain yang dijual di pasaran ada tiga macam kain, yaitu kain kapas (*cotton*), kain *polyester (synthetic)*, dan kain campuran kapas dan *polyester*. Kain yang dijual dapat berupa gulungan maupun potongan kain dengan ukuran tertentu.

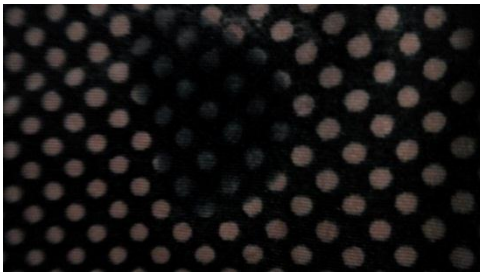


Gambar 2.1 Kain

Kerusakan utama adalah jenis cacat yang biasa ditemukan oleh pelanggan [11]. Ada banyak sekali jenis-jenis kerusakan utama, namun yang dibahas di Skripsi ini hanya enam. Jenis-jenis kerusakan tersebut yaitu:

a. **Noda (*stain*)**

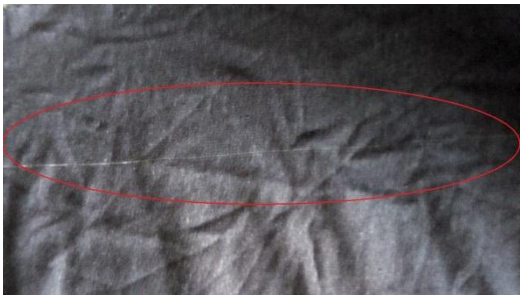
Dapat terjadi akibat mesin celup yang tidak bersih setelah melakukan banyak pencelupan, adanya oli pada mesin, kain menyentuh lantai atau tempat kotor lain selama proses pemindahan, atau kain disentuh dengan kondisi tangan yang kotor.



Gambar 2.2 Noda

b. **Benang putus (*broken/ missing end*)**

Dapat terjadi akibat keadaan mesin yang sudah rusak, bahan baku yang kurang baik, atau penyetelan mesin yang kurang tepat.



Gambar 2.3 Benang Putus

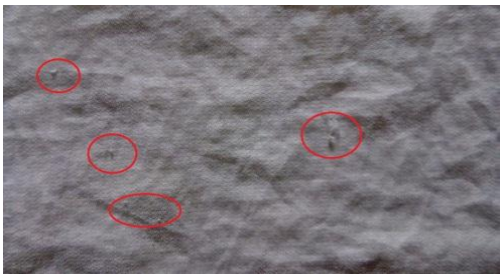
c. **Lubang (*holes*)**

Dapat terjadi akibat ketegangan benang yang terlalu tinggi, benang *overfeed/ underfeed*, rusaknya lusi/ pakan, atau adanya kesalahan pada celah antara *dial* dan *cylinder rings*.

Gambar 2.4 Lubang

d. ***Float***

Dapat terjadi akibat *warp* yang kendur atau kerusakan kartu pola.



Gambar 2.5 *Float*

e. **Cacat warna (*Color bleeding*)**

Dapat terjadi karena benang yang digunakan dalam proses penenunan kain merupakan benang abnormal.

Gambar 2.6 Cacat Warna

f. **Cacat pola (*Broken pattern*)**

Dapat terjadi karena kartu pola/ kisi tidak diatur dengan benar atau mesin tenun gagal menyesuaikan rangkaian pola.

Gambar 2.7 Cacat Pola

2.2 GLCM

Gray Level Co-occurence Matrix (GLCM) atau nama lainnya yaitu *Gray-Tone Spatial-Dependence Matrices* adalah tabulasi mengenai seberapa sering perbedaan kombinasi dari nilai kecerahan piksel (*grey levels*) muncul pada suatu citra. Dalam metode GLCM terdapat 14 ekstraksi ciri, beberapa di antaranya yaitu [3]:

1. **Kontras (*Contrast*)**

   Menunjukkan ukuran penyebaran elemen-elemen matriks citra. Nilai kontras akan semakin besar jika letaknya semakin jauh dari diagonal utama.

$$Kontras = \sum_{i,j=0}^{N-1} P_{i,j}(i-j)^2 \quad \text{.....(2.1)}$$

   Keterangan:

   i adalah nomor baris dan j adalah nomor kolom.

   $P_{i,j}$ adalah nilai probabilitas yang dicatat untuk sel i,j.

   N adalah nomor baris atau kolom.

2. **Homogenitas (*Homogeneity*)**

Menunjukkan kehomogenan citra berderajat keabuan yang sejenis. Citra yang homogen memiliki nilai homogenitas yang besar.

$$Homogenitas = \sum_{i,j=0}^{N-1} \frac{P_{i,j}}{1+(i-j)^2} \quad \text{.....(2.2)}$$

Keterangan:

i adalah nomor baris dan j adalah nomor kolom.

$P_{i,j}$ adalah nilai probabilitas yang dicatat untuk sel i,j.

N adalah nomor baris atau kolom.

3. **Energi (*Energy*)**

Menunjukkan ukuran konsentrasi pasangan dengan intensitas keabuan tertentu pada matriks.

$$Energi = \sum_{i,j=0}^{N-1} (P_{i,j})^2 \quad \text{.....(2.3)}$$

Keterangan:

i adalah nomor baris dan j adalah nomor kolom.

$P_{i,j}$ adalah nilai probabilitas yang dicatat untuk sel i,j.

N adalah nomor baris atau kolom.

4. **Korelasi (*Correlation*)**

Menunjukkan korelasi suatu piksel dengan piksel tetangganya pada seluruh citra.

$$Korelasi = \sum_{i,j=0}^{N-1} P_{i,j} \left[\frac{(i-\mu_i)(j-\mu_j)}{\sqrt{(\sigma_i^2)(\sigma_j^2)}} \right] \quad \text{.....(2.4)}$$

Di mana:

$$\mu_i = \sum_{i,j=0}^{N-1} i(P_{i,j})$$

$$\mu_j = \sum_{i,j=0}^{N-1} j(P_{i,j})$$

$$\sigma_i^2 = \sum_{i,j=0}^{N-1} P_{i,j}(i - \mu_i)^2$$

$$\sigma_j^2 = \sum_{i,j=0}^{N-1} P_{i,j}\left(j - \mu_j\right)^2$$

Keterangan:

i adalah nomor baris dan j adalah nomor kolom.

$P_{i,j}$ adalah nilai probabilitas yang dicatat untuk sel i,j.

N adalah nomor baris atau kolom.

μ adalah *mean*.

σ^2 adalah *variance*.

Langkah-langkah kerja metode GLCM adalah sebagai berikut [2]:

1. Kuantisasi

   Kuantisasi adalah proses mengubah nilai *grayscale* ke dalam rentang nilai tertentu. Hal ini bertujuan untuk meringankan proses komputasi dan untuk mengurangi angka perhitungan.

2. *Co-occurence*

   Pada tahapan ini, sebuah *co-ocurence matrix* dibuat berdasarkan *framework* yang telah ditentukan nilainya. Nilai *framework* ini berasal dari nilai derajat keabuan yang dihasilkan dari proses kuantisasi.

3. Matriks Simetris

   Matriks simetris adalah hasil penjumlahan dari matriks *co-occurence* dengan matriks hasil transposenya sendiri.

4. Normalisasi

   Proses normalisasi adalah proses membagi jumlah kemunculan ketetanggaan piksel dengan jumlah seluruh ketetangan piksel yang mungkin muncul.

5. Ekstraksi ciri

   Ekstraksi ciri adalah proses mengekstraksi nilai ciri dari citra yang akan dikenali oleh sistem. Ciri yang diekstraksi kemudian akan

digunakan sebagai pembeda antara satu citra dengan citra yang lainnya.

2.3 SVM

Support Vector Machine (SVM) adalah sebuah metode klasifikasi biner menggunakan model pembelajaran terawasi. Tujuan dari metode ini adalah untuk menemukan bidang pemisah (*hyperplane*) yang memisahkan data pelatihan dan untuk memaksimalkan jarak di antara dua kelas [1].

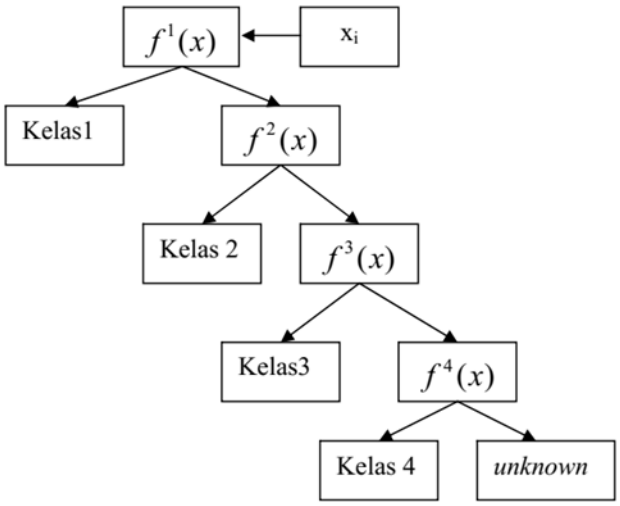
Support Vector Machine bekerja dengan cara memetakan data ke *feature space* berdimensi tinggi sehingga titik data dapat dikelompokkan. Suatu *hyperplane* yang berada di antara kelas-kelas akan ditemukan. Kemudian data ditransformasikan sehingga *hyperplane* dapat digambarkan dengan jelas. Dengan menggunakan hasil tersebut, karakteristik dari data baru dapat digunakan untuk menentukan di kelas mana data tersebut seharusnya berada [10].

Pada awalnya, konsep SVM hanya ditujukan untuk mengklasifikasinya dua kelas saja. Untuk mengatasi masalah klasifikasi yang terdiri dari banyak kelas, maka diusulkan dua metode baru dengan menggabungakn semua data yang terdiri dari beberapa kelas atau dengan mengkombinasikan beberapa klasifikasi biner. Salah satu *multiclass* SVM yang dapat dipakai yaitu metode "*One-vs-all*".

Pada metode ini, dibuat *k* buah model SVM biner di mana *k* adalah jumlah kelas. Misalnya, terdapat pemasalahan dengan 4 buah kelas. Empat buah SVM biner yang digunakan untuk pelatihan dan penggunaannya dalam mengklasifikasikan data baru dapat dilihat pada tabel 2.1 dan gambar 2.8 [9].

Tabel 2.1 Klasifikasi 4 kelas menggunakan metode *multiclass* SVM *"One-vs-All"*

***y i* = 1**	***y i* = -1**	**Hipotesis**
Kelas 1	Bukan kelas 1	$f^1(x) = (w^1)x + b^1$
Kelas 2	Bukan kelas 2	$f^2(x) = (w^2)x + b^2$
Kelas 3	Bukan kelas 3	$f^3(x) = (w^3)x + b^3$
Kelas 4	Bukan kelas 4	$f^4(x) = (w^4)x + b^4$



Gambar 2.8 Contoh klasifikasi memakai *multiclass* SVM "*One-vs-all*"

2.4 Teori Pengolahan Citra

Citra digital adalah gambar dua dimensi yang disimpan dalam bentuk digital. Pengolahan citra adalah pemrosesan citra (biasanya menggunakan komputer) agar kualitasnya menjadi lebih baik daripada kondisi sebelumnya. Pengolahan citra bertujuan untuk memperbaiki kualitas citra agar mudah diterjemahkan oleh manusia maupun komputer. Operasi-operasi pada pengolahan citra di antaranya [5]:

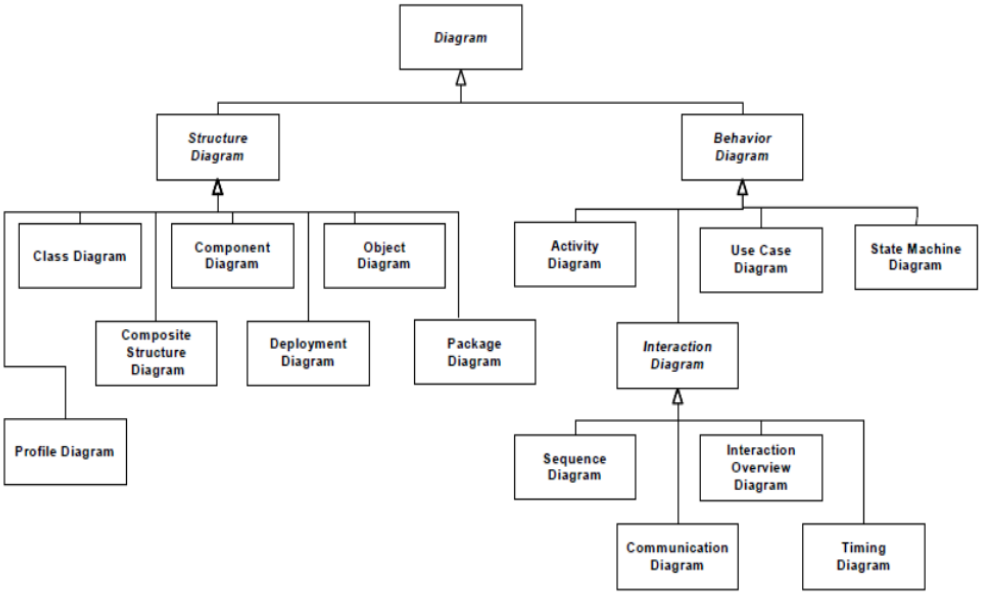
a. Perbaikan kualitas citra (*image enchancement*)
b. Pemugaran citra (*image restoration*)
c. Pemampatan citra (*image compression*)
d. Segmentasi citra (*image segmentation*)
e. Analisis citra (*image analysis*)
f. Rekonstruksi citra (*image reconstruction*)

Analisis citra (*image analysis*) adalah suatu operasi untuk mengektrasi informasi yang terdapat pada suatu citra dengan menggunakan teknik-teknik pengolahan citra. Teknik-teknik tersebut di antaranya:

a. Pengenalan objek 2D dan 3D
b. Deteksi tepi (*edge detection*)
c. Analisis daerah (*region analysis*)
d. *Motion detection*
e. *Video tracking*

2.5 UML

Unified Modelling Language (UML) adalah sebuah bahasa standar untuk penulisan cetak biru perangkat lunak. Perancang perangkat lunak membuat diagram UML untuk membantu pengembang dalam membangun suatu perangkat lunak. Versi UML yang digunakan untuk saat ini adalah UML 2.5.1. UML 2.5.1 menyediakan 14 jenis diagram yang dapat digunakan untuk pemodelan perangkat lunak [7].



Gambar 2.9 Diagram UML

Ke-14 diagram tersebut secara garis besar dibagi menjadi dua yaitu diagram struktural (*structure diagram*) dan diagram perilaku (*behavior diagram*). Diagram struktural menggambarkan segala sesuatu yang harus ada di dalam sistem sedangkan diagram perilaku menggambarkan segala sesuatu yang harus terjadi di dalam sistem.

UML meyediakan banyak fitur yang dapat digunakan untuk mengekspresikan semua aspek-aspek penting yang ada pada sebuah sistem. Perancang perangkat lunak juga diberikan kebebasan dalam menentukan diagram apa saja yang tidak ingin digunakan dalam pemodelan perangkat lunaknya untuk menghindari kekacauan akibat rincian informasi yang tidak berhubungan dengan sistem yang akan dibangun [8].